8-1-1998

Topic Segmentation: Algorithms And Applications

Jeffrey C. Reynar
University of Pennsylvania

University of Pennsylvania Institute for Research in Cognitive Science Technical Report No. IRCS-98-21.

Topic Segmentation: Algorithms And Applications

Abstract
Most documents are *about*more than one subject, but the majority of natural language processing algorithms and information retrieval techniques implicitly assume that every document has just one topic. The work described herein is about clues which mark shifts to new topics, algorithms for identifying topic boundaries and the uses of such boundaries once identified.

A number of topic shift indicators have been proposed in the literature. We review these features, suggest several new ones and test most of them in implemented topic segmentation algorithms. Hints about topic boundaries include repetitions of character sequences, patterns of word and word *n*-gram repetition, word frequency, the presence of cue words and phrases and the use of synonyms.

The algorithms we present use cues singly or in combination to identify topic shifts in several kinds of documents. One algorithm tracks compression performance, which is an indicator of topic shift because self-similarity within topic segments should be greater than between-segment similarity. Another technique relies on word repetition and places boundaries by minimizing word repetitions across segment boundaries. A third method compares the performance of a language model with and without knowledge of the contents of preceding sentences to determine whether a topic shift has occurred. We use the output of this algorithm in a statistical model which incorporates synonymy, bigram repetition and other features for topic segmentation.

We benchmark our algorithms and compare them to algorithms from the literature using concatenations of documents, and then perform further evaluation of our techniques using a collection of news broadcasts transcribed both by annotators and using a speech recognition system. We also test the effectiveness of our algorithms for identifying both chapter boundaries in works of literature and story boundaries in Spanish news broadcasts.

We suggest ways to improve information retrieval, language modeling and various natural language processing algorithms by exploiting the topic segmentation.

Disciplines
Databases and Information Systems

Comments
University of Pennsylvania Institute for Research in Cognitive Science Technical Report No. IRCS-98-21.

TOPIC SEGMENTATION: ALGORITHMS AND APPLICATIONS

Jeffrey C. Reynar

A DISSERTATION

in

Computer and Information Science

Presented to the Faculties of the University of Pennsylvania

in Partial Fulfillment of the Requirements for the Degree of Doctor of Philosophy

1998

Mitchell P. Marcus
Adviser

Mark Steedman
Graduate Group Chair

In memory of Mom and Dad

Acknowledgements

Like all research, the work described in this dissertation was not conducted in isolation. It is impossible to thank everyone who has in some way contributed to it and equally impossible to thank individuals for all of their contributions. I am indebted for technical insights, encouragement and friendship to many members of the vibrant research community at the University of Pennsylvania, as well as those I had the privilege of working with at the Summer Institute of Linguistics, IBM's T.J. Watson Research Center, Fuji Xerox and Infonautics.

I first want to thank my advisor, Mitch Marcus, for providing me with many good ideas, inspiring others in me and helping me refine and develop my own. Thanks also to Mitch for allowing me the freedom to pursue the MUC-6 and Lexis-Nexis projects and to spend several summers working in industry.

Thanks to the members of my thesis committee, Julia Hirschberg, Aravind Joshi, Mark Liberman and Lyle Ungar for providing suggestions which improved the quality of this work. Additional thanks to Aravind, Mark and Mitch for serving on my WPE-2 committee.

I've benefited from conversations with and used natural language processing tools developed by the members of the University of Pennsylvania MUC-6 team: Breck Baldwin, Mike Collins, Jason Eisner, Adwait Ratnaparkhi, Joseph Rosenzweig, Anoop Sarkar and B. Srinivas. Adwait deserves special thanks for permitting me to use his maximum entropy modeling tools. Thanks also to Mickey Chandrasekar, Christy Doran, Beth Ann Hockey, Al Kim, Dan Melamed, Tom Morton, Michael Niv, Martha Palmer, Mike Schultz,

Matthew Stone and David Yarowsky for helpful discussions and feedback. My time at Penn was made more enjoyable by the friendship of each of these people as well.

I also greatly profited from discussions with researchers outside of Penn including Ken Church, Lance Ramshaw, Salim Roukos, Penni Sibun, Larry Spitz and Mark Wasson. Thanks also to Kathy McCoy, my de facto undergraduate advisor, for introducing me to computational linguistics.

Thanks to Betsy Norman and Amy Dunn for regularly squeezing me into Mitch's schedule on short notice and for good conversation while waiting to see him. I'm grateful to Mike Felker for help getting over the administrative hurdles I faced as a graduate student and to the staff of the CIS business office for helping me navigate the complexities of the University of Pennsylvania's payroll and billing systems.

Finally, I owe the most gratitude to my parents, who always encouraged me to pursue my dreams and strive to do my best, and to my wife, Angela, for picking up where they left off.

Abstract

TOPIC SEGMENTATION: ALGORITHMS AND APPLICATIONS

Jeffrey C. Reynar

Supervisor: Mitchell P. Marcus

Most documents are *about* more than one subject, but the majority of natural language processing algorithms and information retrieval techniques implicitly assume that every document has just one topic. The work described herein is about clues which mark shifts to new topics, algorithms for identifying topic boundaries and the uses of such boundaries once identified.

A number of topic shift indicators have been proposed in the literature. We review these features, suggest several new ones and test most of them in implemented topic segmentation algorithms. Hints about topic boundaries include repetitions of character sequences, patterns of word and word n-gram repetition, word frequency, the presence of cue words and phrases and the use of synonyms.

The algorithms we present use cues singly or in combination to identify topic shifts in several kinds of documents. One algorithm tracks compression performance, which is an indicator of topic shift because self-similarity within topic segments should be greater than between-segment similarity. Another technique relies on word repetition and places boundaries by minimizing word repetitions across segment boundaries. A third method compares the performance of a language model with and without knowledge of the contents of preceding sentences to determine whether a topic shift has occurred. We use the output

of this algorithm in a statistical model which incorporates synonymy, bigram repetition and other features for topic segmentation.

We benchmark our algorithms and compare them to algorithms from the literature using concatenations of documents, and then perform further evaluation of our techniques using a collection of news broadcasts transcribed both by annotators and using a speech recognition system. We also test the effectiveness of our algorithms for identifying both chapter boundaries in works of literature and story boundaries in Spanish news broadcasts.

We suggest ways to improve information retrieval, language modeling and various natural language processing algorithms by exploiting the topic segmentation.

Contents

List of Tables

List of Figures

Chapter 1

Introduction

At the end of the twentieth century, people contend with an ever-increasing quantity of information in various forms from numerous sources. Radio, television, the internet, consumer electronic devices and other people bombard us daily with information about many subjects in many ways. Much of this information is contained within what we will call *documents*, for lack of a better term. By a document we mean a repository for a snippet of natural language in any medium which can be accessed, frequently using a computer, after it is created. Recorded radio broadcasts and television programs, books, papers stored on a computer, handwritten notes, scanned reports, web pages, voice mails and faxes are all examples of documents. Unrecorded conversations, for example, would not be considered documents according to our definition because they lack persistence.

The number of documents in existence today is staggering. The database service Lexis-Nexis alone indexes more than 1 billion of them and their collection grows by 9.5 million per week [Lexis-Nexis, 1998]. The number of pages on the world-wide web (WWW) is difficult to count, but users of the popular search engine *Alta Vista* can search on the order of a terabyte of data and *Alta Vista* does not index the entire web [Digital Equipment Corporation, 1997]. The Library of Congress collection numbers over 108 million items. Even though many of these items, such as photographs, are not documents according to our definition, this is undoubtedly one of the largest collections of documents ever assembled [Library of Congress, 1997].

Documents are generated at a far greater rate today than ever before. Despite the claims about the impending paperless office made in the 1970s and earlier [Lancaster, 1978], the increased pace is not due exclusively to the growth of broadcast industries or the rise of the internet. The rate of production of paper documents is growing, as evidenced by the fact that world-wide paper production doubled between 1961 and 1981 [Food and Agriculture Organization of the United Nations, 1995]. In fact, instead of producing fewer paper documents today because of information technology, we are producing more because it is easier to do so—high quality printers are inexpensive and word processing and desktop publishing packages are ubiquitous.

The proliferation of documents due primarily to increased access to information technology is unlikely to cease, and will most likely become an even larger burden on the average consumer of information until the technology to access a desired piece of information catches up with the technology to create new documents [Fox et al., 1995]. Since we are unlikely to stem the tide of documents any time soon, we need better ways to cope with them—that is, to store them, index them, access them, convert them from one form into another more easily used form, and so forth. These needs are partly responsible for interest in research areas as seemingly diverse as digital libraries, optical character recognition (OCR), speech recognition, information extraction and information retrieval (IR).

The common thread running through these lines of research is access to needed information. This is obviously the purpose of both information retrieval and information extraction. The goal of information retrieval is to permit document databases to be searched in arbitrary ways. Information extraction focuses on identifying predetermined types of information as new documents are entered into a document collection. Digital libraries provide enhanced access to collections by making electronic documents accessible at a distance and, at least in theory, permitting more documents to be stored in a single repository. They also allow users to search in ways traditional libraries do not. OCR and speech recognition both enhance access by converting documents from one form which is difficult to manipulate with today's access tools into another form, namely text, the medium required by most natural language processing, information retrieval and information extraction systems.

1.1 Motivation

This dissertation is about techniques for improving access to information and improving the underlying language technology that helps provide access by dividing lengthy documents into topically-coherent sections. Such technology is necessary because people who search for information are interested in finding it in conveniently-sized packages. Clearly, it is unhelpful to learn that the answer to a question can be found in the local library, or for that matter on the internet or somewhere in a Dow Jones database. It is marginally more useful to discover that the answer lies in the section of books whose call numbers are in the range from 100 to 199, but even at the average community library looking through all these titles, let alone reading through each book, would require too much time. Even finding out that the needed information was in a particular book might not be sufficiently specific if the information was needed quickly or, in the event that more time were available, if reading through the less immediately relevant material in the book would not be beneficial.

Generally the best possible solution to a request for information is to give the requester the answer, if there is one. Often times, however, there is not, strictly speaking, an answer. Many queries posed to IR systems are open-ended enough that even if technology were much more advanced than it is today, the best response would still consist of a set of documents. Consider the query "Who shot John F. Kennedy?" The answer "Lee Harvey Oswald" would suffice in some cases, but alternate theories abound. Ideally the user of an IR system should be made aware of some of them.

Technology is not yet advanced enough to provide a simple answer, like "Lee Harvey Oswald," for many queries. The answer to this particular query may be found in a database about the presidents, but many information retrieval systems search large text databases and extracting answers to arbitrary natural language questions from text is an open research area. A good compromise given the state of technology is to limit the size of the documents presented to information seekers to be just large enough to satisfy their information needs, but no larger. No one should be expected to read the Warren Commission report in its entirety to learn that Oswald was almost certainly J.F.K.'s assassin.

Another potential approach to satisfying demands for information is to synthesize material from selected passages within a document, or even across multiple documents, thereby

creating new documents on-demand which present the requested information in summary form. There is currently a variety of work being done in this area, but we will touch on this topic only briefly. (See, for instance, [Aone et al., 1997]).

There are important constraints on what "just large enough" should mean when responding to a demand for information. It is crucial that the returned material be interpretable. There should be few unresolved pronominal references, sufficient context to allow ambiguous words and phrases to be understood and enough information to respond to the original query. If these constraints were unimportant, then arbitrary passages, possibly beginning and ending mid-sentence, could be presented to users. IR systems could index all document fragments consisting of contiguous sequences of words and return them to users in response to their queries. Although this can place an unbearable storage burden on systems which index large collections, IR performance does improve using this technique. (See [Callan, 1994, Kaszkiel and Zobel, 1997], among others.)

A modest improvement to this approach is to limit the indexed segments to those beginning at sentence boundaries. No response presented to a user would then begin with an uninterpretable sentence fragment. This would also reduce the amount of storage space required by the IR system. However, problems still remain. Pronouns may not be resolvable and there may not be enough context to understand the meaning of a text fragment.

A better solution would be to divide documents into sections, with each section limited to the extent of a particular topic and the boundaries between topic segments aligned with sentence boundaries for clarity. This would result in a minimal increase in the storage space required by an IR system and would solve many of the problems stemming from lack of context. Ideally, pronominal references to entities outside the section would be resolved so that sections were as intelligible on their own as possible.

The difficulty of making these sections stand on their own varies greatly and depends on the type of document being partitioned. Some types of documents are created with partitions in place. Newspapers are divided into many articles and each article is labeled with a title. Readers can choose to read or not to read articles based on their titles. Documents without any labels or segment boundaries, such as transcripts of interviews,

telephone conversations and other spontaneous communications lie at the opposite end of the continuum. It might seem that no additional partitioning of labeled documents is necessary, but the level of subdivision is rarely fine enough. Short newspaper articles may be restricted to a single topic, but feature articles and articles from magazines and journals may range over a number of related topics. Documents in other media may have segments marked implicitly, but recovering the boundaries between the segments may be difficult.

This difficulty pertains to segmenting television and radio news broadcasts which are produced with the independence of stories in mind. It is obvious to viewers of news programs where the boundaries between stories occur. But, it can be challenging to find the point where one story ends and another begins using only a transcript.

1.2 Segmentation Cues

In this dissertation we focus on segmenting documents using cues present in text and transcribed speech. There are, however, two additional sources of cues for segmenting video broadcasts, one of which applies equally well to audio data. The video portion of programs can be a rich source of information about transitions. For example, all black frames sometimes appear between stories and shifts from on-location reports back to the anchor in the studio often result in more dramatic alterations of image content than are caused by camera movement or shifting the focus of attention from one anchor to another. Cues like these have been used to divide video documents into scenes (e.g. [Yeo and Liu, 1995]).

The second source of cues is the speech stream itself. Some linguistic cues present in speech, such as intonation and the use of pauses, are not generally transcribed. Hirschberg and Grosz studied the relationship between intonation and discourse structure and found correlations between intonational features and annotator's labeling of discourse structure [Hirschberg and Grosz, 1992, Grosz and Hirschberg, 1992]. Hirschberg and Nakatani showed that annotators more consistently segment discourse using speech in conjunction with transcripts than using text alone and that annotators can reliably segment both spontaneous and planned speech [Hirschberg and Nakatani, 1996]. Passoneau and Litman

studied interannotator consistency and the correlation between linguistic cues and segmentation. They proposed segmentation algorithms based on cue words, referential noun phrases and the presence of pauses. The performance of their algorithms was encouraging, but not as good as human performance [Passoneau and Litman, 1996].

Ultimately textual, spoken and video cues should be combined and incorporated into a video-on-demand system, like the Informedia Digital Video Library, which permits video documents to be searched and browsed in the same ways that information retrieval systems do for textual documents [Christel et al., 1995, Hauptmann and Witbrock, 1997]. We leave the combination of textual cues with those from other sources to future work.

1.3 The Noisy Channel Model

If we assume there is a canonical topic segmentation, we can treat the creation of documents without explicitly marked topic boundaries as an instance of the noisy channel model [Cover and Thomas, 1991]. Use of this model has resulted in advances in statistical techniques for speech recognition [Bahl et al., 1983] and for many natural language processing tasks [Brown et al., 1990a, Brown et al., 1990b].

We will use the production of a news broadcast as an example of how the noisy channel model applies. The producer of a news program begins with a collection of independent stories. The set of stories she intends to convey corresponds to the Original Message in Figure 1.1. The content of the stories passes through an Encoder, namely the journalists who translate the content of the stories into words. At this stage, assuming each story is written independently, the boundaries between stories are still clear. We can consider the written stories which will ultimately form the news broadcast to be the Encoded Message in the figure. This message passes through a Noisy Channel which, according to a stochastic process, blurs or removes the boundaries between stories. There is a simple explanation for this in the production process. The goal of making the broadcast flow smoothly motivates journalists to relate stories to one another and provide natural transitions between them.[1]

[1] If every story began with a set phrase such as *In this story*, detecting boundaries between stories would be trivial if this phrase was used nowhere else, but news broadcasts would be less captivating than they currently are.

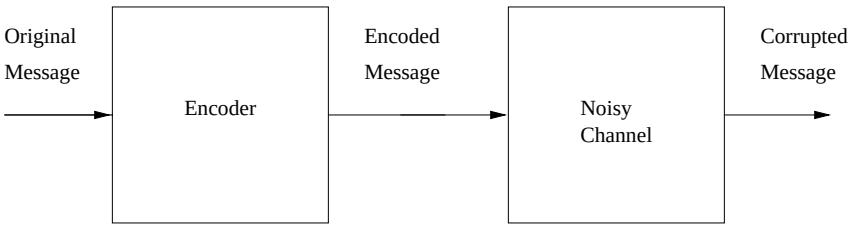

Figure 1.1: The noisy channel model.

These connections and transitions obscure the boundaries between stories. Once they are in place, the broadcast is in its final form, which corresponds to the Corrupted Message in the figure. In broadcast form, some topic boundaries may still be marked by phrases such as *Coming up*, intonational cues or changes in image content. In fact, all of the boundaries may still be detectable without determining the meaning of the program, but the markings will no longer be as explicit as they were before the stories passed through the noisy channel.

The above example dealt only with the boundaries between stories. A similar process could obscure the topic boundaries within individual stories from news broadcasts and documents of other types as well. In fact, authors focus on making boundaries subtle, since readability is partly a function of the smoothness of the transitions between topics. This can be said of participants in conversations as well, but we focus on structuring primarily monologues.

1.4 Goals

We will describe methods for recognizing the boundaries between topic segments within documents. The best of our methods is applicable to a wide variety of document types in various media. However, the contents of the documents must be converted to text for these techniques to be used.

Until recently no topic-boundary annotated corpora were available. As a result, we will first present our earliest results for simulations using concatenated documents. In the rest of this dissertation, however, we will demonstrate the effectiveness of our methods on

newspaper articles, both transcribed and speech-recognized television and radio broadcasts, and works of literature. We will also show that our methods apply to languages other than English using transcripts of Spanish television and radio broadcasts.

Evaluating the segmentations produced by text structuring[2] algorithms presents several problems. Therefore, we will discuss the merits of various performance measures and, in order to demonstrate the utility of the algorithms we present, we will also describe a number of applications which would benefit from using topic-segmented documents. For example, performance on various natural language processing tasks, such as coreference resolution and word-sense disambiguation is likely to be improved. We will also describe information retrieval experiments using both the segmentations produced by human annotators and those identified algorithmically.

1.5 Topic Segmentation, not Discourse Segmentation

The algorithms we propose for text segmentation are intended to be useful for language engineering applications. As a result, they identify boundaries which demarcate units of text that are appropriately sized for these particular tasks. The segments the boundaries delimit range in length from several sentences to several paragraphs. Most theories of discourse structure [Grosz and Sidner, 1986, Grosz et al., 1995, Mann and Thompson, 1988] focus on relating much smaller units of discourse, namely utterances. As a result, empirical work on discourse segmentation has focussed primarily on identifying relations between these units as well [Hirschberg and Grosz, 1992, Passoneau and Litman, 1993]. Research regarding the relationships between utterances has numerous implications for language processing tasks, but is outside the scope of this work.

The coarse-grained segmentation we are pursuing may be a subset of the more detailed analysis that is the goal of discourse segmentation, but we do not believe the techniques we propose are applicable to discourse segmentation. The features we use for topic segmentation do not have enough resolving power to be of much use on a finer scale.

[2]For variety, we will freely interchange the phrases *text structuring*, *text segmentation* and *topic segmentation*. We reserve *discourse segmentation* for predominantly hierarchical analyses that are finer-grained than those produced by the algorithms we present.

1.6 Outline

In Chapter 2 we will briefly summarize theories of discourse structure that relate to text structuring. Knowledge of some of these theories is necessary to understand the algorithms that we review in Chapter 4, while others have influenced the design of the algorithms we describe in Chapter 5. We will then discuss features useful for identifying topic structure in Chapter 3. Some of these clues have been used in previous approaches to text structuring while others are novel. Chapter 4 reviews previous computational approaches and in Chapter 5 we will outline our own algorithms. In Chapter 6 we will describe methods of evaluating text structuring techniques and will measure the performance of our algorithms using some of them. In Chapter 7 we will describe applications which benefit from text structuring and demonstrate the utility of text structuring techniques for a subset of these applications. Finally, we will summarize and discuss our findings and outline some future work in Chapter 8.

Chapter 2

Theories of Discourse Structure

There is a large body of literature on the structure of discourse. This work addresses a wide range of issues, including: How large are the units whose relationships with one another should be described by a theory of discourse structure [Harris, 1952, Longacre, 1979, Stark, 1988]? What is the nature of the relationships between individual phrases in written texts [Mann and Thompson, 1988]? What is the relationship between coherence, attention and the selection of referring expressions [Grosz and Sidner, 1986, Grosz et al., 1995]? Is discourse structured hierarchically, linearly or otherwise [Hurtig, 1977, Hinds, 1979, Skorochod'ko, 1972, Webber, 1991, Sibun, 1992]? What are the relationships between the utterances in multi-party discourse [Hobbs, 1983, Walker and Whittaker, 1990]? How can large-scale shifts in narrative texts be identified [Grimes, 1975, Youmans, 1990]?

Since we cannot review the entire discourse structure literature, we will summarize only the articles most pertinent to our work here. The algorithms we present in Chapter 5 structure text in a theory-neutral way. However, they depend implicitly on particular answers to the above questions.

The algorithms we propose divide texts into sections which range from several sentences to several paragraphs in length. Much of the literature about discourse structure, however, focuses on the relations between utterances and is only tangentially related to analyses involving much larger fragments of discourse.

Although there is ample evidence suggesting that discourse is hierarchically structured [Grosz and Sidner, 1986, Webber, 1991], the algorithms we propose partition text linearly.

The segments our methods identify contain utterances whose inter-relations may best be described by a hierarchical structure. Identifying that structure is in the provenance of discourse segmentation research, however. It may even be that the segments our algorithms locate fit naturally into a hierarchical structure. Segmenting documents linearly also facilitates comparison with algorithms from the literature on topic segmentation and permits evaluation using available linearly-segmented corpora.

Some of the corpora we use to evaluate text segmentation algorithms contain brief passages of dialogue. Many news broadcasts are hosted by several anchors who occasionally discuss the news with one another. However, the majority of such documents is monologue. As a result, we do not survey work on dialogue, and reserve the application of our techniques to more typical, conversational text for future work.

Below we review Halliday and Hasan's theory of the types of relations between textual elements which provide coherence to a document. Their work influenced the design of several text segmentation algorithms, including our own. We review Grosz and Sidner's theory of discourse structure because understanding one of the algorithms we will review requires knowledge of it. Finally, we summarize Skorochod'ko's theory of text organization, which underpinned Hearst's work on dividing long documents into subtopic sections.

2.1 Halliday and Hasan

In their book *Cohesion in English*, Halliday and Hassan describe *texture* as a property possessed by a text, but which an arbitrary collection of sentences does not have. Readers can frequently tell whether or not a series of sentences exhibits texture. The sentences in Example 2.1 do exhibit it, while those in Example 2.2 do not [Halliday and Hasan, 1976].

(2.1) Wash and core six cooking apples. Put them into a fireproof dish.

(2.2) Wash and core six cooking apples. The prices of computers drop regularly.

Cohesion is one of the elements of a discourse which contributes to its texture. Cohesion is present when an element in a text is best interpreted in light of a previous (or, less frequently, following) element of the same text. Halliday and Hasan identify five cohesive relations which contribute texture to a document.

Reference References are like pointers. Rather than repeat a phrase in the text, a writer or speaker may use a pointer to the entity selected by a phrase instead. Halliday and Hasan distinguish two main types of reference. Exophoric references are to entities in the world of the discourse and endophoric references are to portions of the text itself. The word *he* in Example 2.3 is an exophoric reference and *so* is an endophoric reference in Example 2.4.

(2.3) John likes apples, but he loves pears.

(2.4) For he's a jolly good fellow. And so say all of us.

Substitution Substitution and reference are similar, but differ in that substitution occurs prior to semantic interpretation while reference occurs after interpretation. That is, a substitute acts merely as a pointer to a region of text which refers to an entity in the world of the discourse, while a reference refers directly to an entity without the mediation of the original referring phrase. In Example 2.5 *does* substitutes for the phrase *like apples*.

(2.5) Do you like apples? Everybody does.

Ellipsis Ellipsis is similar to substitution. It can be viewed as substitution by a zero. In Example 2.6, *bought* has been replaced by a null phrase in *Mary some flowers*.

(2.6) John bought some chocolates and Mary some flowers.

Conjunction Conjunction is more difficult to define than the previous three relations. It holds between elements of a text when they are ordered temporally, one causes the other, when they describe a contrast or when one elaborates on the other. Examples from *Cohesion in English* will demonstrate these relations. Each of the sentences (a) through (d) should be read immediately following the first sentence in Example 2.7.

(2.7) For the whole day he climbed up the mountainside, almost without stopping.

(a) Then, as dusk fell, he sat down to rest. (Temporal order)

(b) So by night time the valley was far below him. (Causation)

(c) Yet he was hardly aware of being tired. (Contrast)

(d) And in all this time he met no one. (Elaboration)

Lexical cohesion Lexical cohesion holds between two tokens in a text which are either of the same type or are semantically related in a particular way. There are five semantic relations that constitute lexical cohesion.

1. *Reiteration with identity of reference* occurs when a particular entity previously referred to in a discourse is referred to again.

   (2.8) John saw a dog.

   (2.9) The dog was a retriever.

   Example 2.8 refers to a particular dog and Example 2.9 refers to the same dog again.

2. *Reiteration without identity of reference* occurs when reference is made to the entire class to which an entity previously referred to in a discourse belongs.

   (2.10) John saw a small retriever.

   (2.11) Retrievers are usually large.

   Example 2.10 refers to one particular member of the set of dogs identified as retrievers while Example 2.11 refers to the entire class of retrievers.

3. *Reiteration by means of superordinate* occurs when reference is made to a superclass of the class to which a previously mentioned entity belongs.

   (2.12) John saw the retriever.

   (2.13) Dogs are his favorite animals.

   Example 2.12 refers to a retriever, which is a type of dog, while Example 2.13 refers to dogs in general.

4. A *systematic semantic relation* holds when a word, or group of words, has a clearly definable relationship with a previously used word or phrase. For example, both could refer to members of the same set.

   (2.14) John likes retrievers.

(2.15) He doesn't like collies.

Example 2.14 refers to retrievers and Example 2.15 mentions collies, both of which are subsets of the species of dogs. In this case the relationship can be classified as membership in a particular class.

5. A *nonsystematic semantic relation* holds between two words or phrases in a discourse when they pertain to a particular theme or topic, but the nature of their relationship is difficult to specify. Recognizing this category in a computational system would be more difficult than recognizing the other categories.

   (2.16) John spent the afternoon studying in his dormitory room.

   (2.17) He loves attending college.

   A semantic connection exists between the word *dormitory* in Example 2.16 and *college* in Example 2.17, but it is hard to classify and unlikely that all such relations, or even the preponderance of them, could be found in a knowledge source in the way that many synonymy relations can be identified using a thesaurus.

Halliday and Hasan's categories overlap to some degree. For example, it can be difficult to distinguish instances of substitution from endophoric reference. Substitution is subtly different in that it relates words of the text, is not a semantic relation and requires the substituted phrase to have the same role as the phrase it substitutes for. This is not the case with reference. Nonetheless, Halliday and Hasan acknowledge that there are instances where more than one category applies equally well.

Halliday and Hasan explain that texts frequently exhibit varying degrees of cohesion in different sections. Obviously, the start of a text cannot be cohesive with preceding sections, nor can the end exhibit cohesion with later sections. In the middle of a text, however, the quantity of cohesion can vary greatly. Some authors, Halliday and Hasan suggest, prefer to alternate between high and low degrees of cohesion.

Texture—which is more frequently called *coherence*—and cohesion are often confused,[1] but differ significantly. Cohesion relates elements of a text and can generally be identified

[1] See [Hobbs, 1979, Hobbs, 1983] for an extended discussion of the difference between the two.

out of context. Texture, however, is a property that applies to an entire text. It is more difficult to define, but can be recognized upon reading a text in its entirety.

2.2 Grosz & Sidner

In their influential paper on the organization of discourse [Grosz and Sidner, 1986], Grosz and Sidner present a tripartite theory. Their theory addresses the relationships between attentional state, linguistic structure and intentional structure.

Attentional state pertains to conversants' focus of attention and the accessibility and salience of discourse entities. Grosz and Sidner track attentional state using a stack, and the standard stack operators, *push* and *pop*. The elements of the stack are data structures called *focus spaces* which contain lists of available entities and the *discourse segment purpose*, which is an element of the intentional structure discussed below. The top element of the stack, the focus space most recently added, and some of the elements lower in the stack are available to the hearer or reader of an utterance. Aspects of the linguistic structure determine how focus spaces are added to and removed from the stack.

The linguistic structure captures the relationships between successive utterances and divides a text into *discourse segments*. These segments form a hierarchical structure. The linguistic structure constrains changes in attentional state. The focus space stack is updated when transitioning from one segment to another. When leaving a segment and entering a sister segment, the stack is popped prior to pushing the focus space associated with the new segment onto it. When entering an embedded segment, the focus information associated with that segment is simply pushed onto the stack.

The intentional structure models the goals and subgoals of the discussants. The discourse segment purpose (DSP) is the intention associated with a discourse segment. Grosz and Sidner call the intention of the entire discourse the *discourse purpose* (DP). They classify the relationships between intentions, which can be identified from the linguistic structure, as either dominance or satisfaction precedence. Dominance means that satisfying the dominated intention contributes to the satisfaction of the dominating intention. Satisfaction precedence indicates that one intention cannot be satisfied until another is

itself satisfied. These relations mirror relations in the linguistic structure. When one intention dominates another in the intentional structure, then the dominated intention corresponds to a discourse segment in the linguistic structure which is a descendant of the discourse segment related to the dominating intention. A satisfaction-precedence relationship between two intentions indicates that the corresponding discourse segments are sisters in the linguistic structure.

An example, similar to one from Grosz and Sidner's paper, will illustrate the relationships between the components of this theory. Imagine a text consisting of only four sentences, each of which constitutes a separate discourse segment. The first sentence is about a particular topic which the second and third sentences support. The final sentence is about a different subject. Figure 2.1 shows the linguistic structure in terms of discourse segments, the intentional structure as encoded by the list of dominance relations and the attentional structures represented by a stack containing focus spaces. Part 1 of the figure represents the state of the model after the first two sentences have been processed. Once the third sentence of the discourse has been processed, the structures would be updated to correspond to those shown in part 2 of the figure. After the final sentence has been processed, the structures would be as shown in part 3.

In part 1, the first sentence, labeled discourse segment 1 (DS1), has contributed the bottom element to the focus space stack, while the second sentence (DS2) has contributed the top element. The DSP associated with the first segment (DSP 1) dominates the DSP associated with the second segment (DSP 2). When the third sentence is processed, the hearer recognizes that it is in a separate discourse segment, DS3, which is a sister to DS2. This fact is reflected in the linguistic structure: both DS2 and DS3 are dominated by DS1 which is indicated on the figure by indenting the dominated segments. The model of attentional state is updated by pushing the focus space associated with DS3 onto the stack. Processing the fourth sentence does not change the dominance hierarchy, but does affect the attentional state. The focus spaces associated with both DS3 and DS1 would be popped from the stack and the focus space associated with DS4 would be pushed.

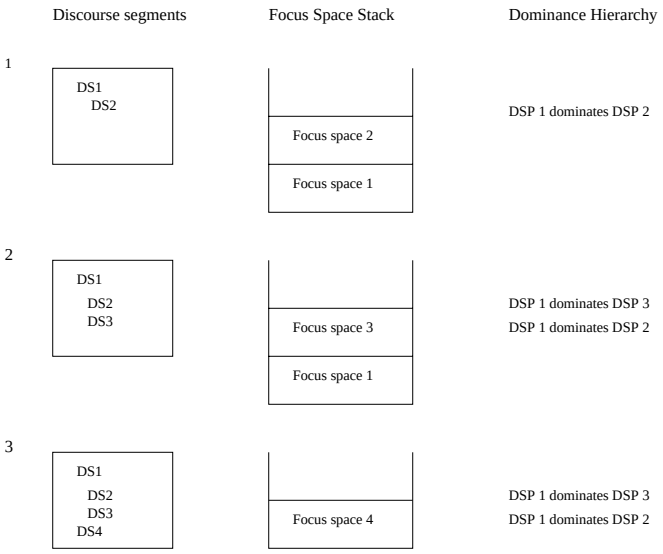

Figure 2.1: Changes in the attentional, intentional and linguistic structure when processing a sample text.

2.3 Skorochod'ko

Skorochod'ko discussed methods of automatically generating abstracts for documents. He also outlined a typology of texts based on the presence of semantic relationships between textual elements—either sentences or paragraphs—in a document, which he suggested could be identified using a semantic network [Skorochod'ko, 1972]. He proposed four types of text:

Chained Only neighboring elements are strongly related.

Ringed When the relationships between segments are represented as a graph, this type of text has a single cycle, which encompasses all textual elements. A newspaper article with a leading summary, some explanatory text and a conclusion is likely to possess this type of structure, since the first and last segments would be related and, presumably, neighboring segments throughout the document would be related as well.

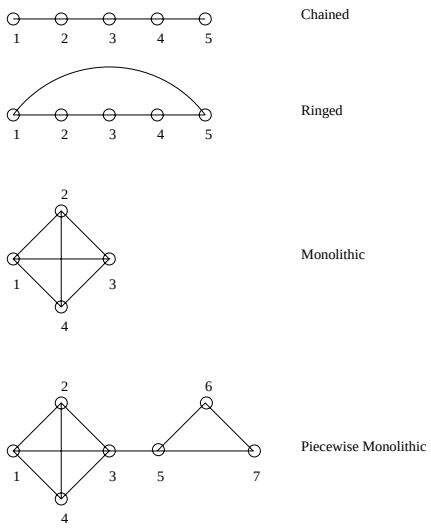

Figure 2.2: Skorochod'ko's four text types.

Monolithic All of the segments of the text are related. The graph for a text with this kind of structure is a clique.

Piecewise Monolithic Portions of the text are themselves monolithic, but there are few connections between the monolithic portions.

Figure 2.2 contains diagrams representing the four types of structure Skorochod'ko identified. Circles represent textual elements and lines indicate the presence of semantic relations between them. The numbers associated with textual elements indicate their order in the text. The figure shows that documents having the chained structure could be easily segmented. Those exhibiting the ringed structure could be segmented as well. If the relationship between the first and final units is ignored, a document with ringed structure is transformed into one with chained structure. Piecewise monolithic documents could also be segmented straightforwardly. In fact, these texts could be decomposed into segments about individual topics which are themselves monolithic in nature. Monolithic texts pose a problem for text segmentation techniques, since all their elements are related.

Chapter 3

Structuring Clues

Text structuring algorithms from the literature employ a wide variety of features. In this chapter we will describe most of these features as well as those used by our algorithms. We will describe features used by only a single system from the literature in Chapter 4 in conjunction with the algorithms that employ them. We will first summarize some interesting clues suggested in the literature which cannot be used in computational systems given current natural language processing technology. These cues highlight the fact that people may use different cues to discover topic shifts than computers are currently capable of using. At the end of this chapter we will list some additional indicators of topic structure which could be used by future topic segmentation systems.

3.1 Currently Uncomputable Features

Documents exhibit a number of features which have been described as being strong indicators of topic shifts. However, some of these features cannot easily be recognized using current computational techniques. We survey the most interesting of these features below.

3.1.1 Grimes

Grimes suggested four clues about the presence of topic boundaries. The first three indicators are found only in narratives. Shifts to new segments may be marked by changes to the:

1. Scene
2. Participant orientation
3. Chronology
4. Theme

These indicators do not always mark topic shifts and a particular boundary may have multiple markers. Further explanation of these indicators is in order. First, a change in the setting of the action, as is frequently found in novels and plays, often indicates a topic shift. Next, changes in participant orientation—the importance ranking of characters in a story—can affect the story's structure. For instance, a new topic segment often begins when a new character is introduced and the focus of attention shifts away from previously important characters. A shift from one time period to another—for example from the evening of one day to the morning of the next without any description of the intervening time—most likely indicates the start of a new segment. Finally, shifts in theme, which can be conveyed through dialogue and which frequently occur independent of changes in setting and time period, may signal the beginning of a new topic segment [Grimes, 1975].

3.1.2 Nakhimovsky

Nakhimovsky identified four types of shift from one text segment to another in narratives. These are:

1. Topic shifts
2. Shifts in space and time
3. Discontinuities of figure and ground
4. Changes in narrative perspective

Nakhimovsky also devised heuristics for segmenting narrative discourse, but most of these would be difficult to implement [Nakhimovsky, 1988]. His heuristics recognize the four types of shift he identified. A change in topic may be signaled when there are no

anaphoric relations between a new segment and an in-focus segment and there are no obvious inferences which relate the new segment to an in-focus segment. Nakhimovsky himself acknowledged that this was vague unless the inferences were defined with reference to the abilities of a particular computational system. Temporal discontinuities are marked by tense shifts or flashbacks to earlier scenes. Shifts in figure and ground are accompanied by aspectual changes or changes in the discourse focus from temporal to spatial information. A change in narrative perspective may be accompanied by, among other things, a shift in story teller. Shifting from an uninvolved narrator to a particular character who relates the action in the first person is one example of changing narrative perspective.

3.1.3 Summary

The techniques proposed by Grimes and Nakhimovsky should be straightforward for people to use when processing discourse. However, they are currently difficult to implement because they require a level of natural language understanding which cannot be attained using current technology. For example, how could flashbacks be recognized? The features in the next section require much shallower processing. Although they would be more tedious for people to use to manually identify topic boundaries, they can be implemented in an automatic text segmentation system.

3.2 Currently Computable Features

In this section we describe the features used by topic segmentation systems from the literature and those used by the algorithms we describe in Chapter 5. We present examples of these features using sample texts from the HUB-4 Broadcast News Corpus collected by the Linguistic Data Consortium (LDC) for the spoken document retrieval portion of the 1997 TREC Text Retrieval Conference [HUB-4 Program Committee, 1996]. Boundaries between topics in these examples are generally indicated with vertical white space. The text in these examples was transcribed from speech and lacks proper capitalization, punctuation, sentence breaks and paragraph boundaries. We produced the figures in which words and phrases are linked using SRA's Discourse Tagging Tool.

actually	also	although	and	basically
because	but	essentially	except	finally
first	further	generally	however	indeed
like	look	next	no	now
ok	or	otherwise	right	say
second	see	similarly	since	so
then	therefore	well	yes	

Table 3.1: Cue words from [Hirschberg and Litman, 1993].

3.2.1 Cue Words & Phrases

Grosz and Sidner explained that some words in discourse are used to indicate changes in the discourse structure rather than to convey information about the subject matter being discussed. Their example is *Incidentally Jane swims every day.* Since it is improbable that Jane unintentionally happens to swim daily, the most likely interpretation of *incidentally* is that it conveys information about the relationship of the utterance *Jane swims every day* to the current discourse. That is, *incidentally* marks a brief diversion from the main topic and signals the start of a new discourse segment dominated by the current segment [Grosz and Sidner, 1986].

Other researchers have examined the relationship between particular words and phrases and discourse structure as well [Reichman, 1981, Dahlgren, 1996, DiEugenio et al., 1997]. Cue words play an important role in the discourse segmentation work of Hirschberg and Litman, among others. They present a number of cue words that indicate changes in discourse structure which they gleaned from various sources [Hirschberg and Litman, 1993]. They used these cue words, which are shown in Table 3.1, to study the correspondence between intonation and cue word usage in speech. These cue words are relatively domain independent and may mark topic shifts in many genres.

Domain Cues

The cue words and phrases we intend to use differ from those Hirschberg and Litman used in two important ways. First, they are highly domain-specific. Domain-specificity means

that new lists must be created before documents from new sources can be segmented. This is a drawback because manually creating the lists is time-consuming and automatically creating them requires an annotated corpus. However, the advantage is that genre-specific conventions are often reliable indicators of topic shifts. For example, in the domain of news broadcasts, cue phrases involving greetings such as *good evening*, *good night* and *good morning* occur almost exclusively at the beginning and end of broadcasts in brief segments which begin or end each show. This makes them good indicators of shifts from one topic segment to the next. The cue phrase *good evening* is highly domain specific. In the Penn *Wall Street Journal* Treebank [Marcus et al., 1993], it does not occur at all in 1.3 million words of text. Even if it is found in other *Wall Street Journal* articles, there is no guarantee that it will mark a topic shift.

Second, some of our cue phrases contain sequences of words of particular types, such as person or place names. The presence of word sequences of particular types makes the cue phrases dynamic. Another example from news broadcasts will illustrate this. Reporters often conclude on-location reports with the phrase *reporting from* followed by the name of a place—a country or city, for example. Since these reports are often followed by new topic segments, this type of phrase is a good indicator of a topic boundary. Employing mildly productive cue phrases reduces the number of misidentifications. For example, rather than label only instances of *I'm* followed by a reporter's name, we could identify all instances of *I'm* and treat them as cue phrases. Although simpler, we would mislabel many non-cue phrases this way. We would mislabel *I'm* in the phrase *I'm late*, for example, as a cue phrase.

To prevent confusion with more conventional notions of what a cue phrase is, we refer to the cue words and phrases we use as *domain cues*. We manually identified our domain cues from the broadcast news domain and separated them into several categories. These categories are new person cues, greeting cues, introductory cues, pointers to upcoming stories, shifts to other broadcasters, returns from commercials and signing-off cues. New person cues occur when guests join the broadcasters or reporters are introduced. Greetings usually mark the beginning or end of a broadcast. Introductory cues frequently accompany the beginning of news stories. Pointers to upcoming stories are used to keep the

audience interested because of the suggested importance of what will be discussed later in the broadcast. The LDC eliminated commercials from the corpus we used, but phrases indicating that a commercial just ended remained and are useful indicators of topic shifts. The final category, sign-off cues, contains all of the productive phrases we used and signals the transition from an on-location report back to the anchor in the studio.

Table 3.2 lists the specific domain cues we used. PERSON is shorthand for a person's name, PLACE refers to a location and STATION indicates the call letters of a station or network—for example, *C. N. N.* Figure 3.1 is an example from the HUB-4 broadcast news corpus which contains one of these cue phrases.

We divided the domain cues into categories because not all cues indicate upcoming topic shifts. Greetings most often precede new topic segments, but introductory phrases more frequently follow topic boundaries. Our text segmentation algorithms use only the category of each phrase because individual phrases occurred too rarely for them to be used reliably in a statistical model. With more training data, we could explore the appropriateness of the categories we established and test the value of individual cue phrases as well. In Section 6.3.3, we show that with a larger corpus we can induce a list of useful domain cues.

Most of the domain cue categories are self-explanatory. However, the greeting category contains some phrases unlikely to be commonly regarded as greetings. These are phrases, such as *brought to you by* and words like *transcript* which often occur at the start or end of a broadcast, and which rarely occur elsewhere. As a result, they behave like the other members of the greeting category to indicate upcoming topic shifts, but are not obviously greetings in the way that *good morning* is.

Computing Domain Cues

We identified television station and network names using regular expressions. To label words with the categories PERSON and PLACE we built a maximum entropy model using the modeling tools designed and implemented by Adwait Ratnaparkhi which have been used for part-of-speech tagging, parsing and end-of-sentence detection, among other things [Ratnaparkhi, 1996, Ratnaparkhi, 1997a, Reynar and Ratnaparkhi, 1997].[1]

[1]See [Berger et al., 1996] for an introduction to maximum entropy modeling for natural language processing and [Ratnaparkhi, 1997b] for information about Ratnaparkhi's implementation in particular.

Domain Cue	Category
joining us	New person
good night	Greeting
good evening	Greeting
good morning	Greeting
hello	Greeting
that's the news	Greeting
tomorrow on	Greeting
brought to you by	Greeting
transcript	Greeting
top story	Introductory
top stories	Introductory
in the news	Introductory
let's begin	Introductory
this just in	Introductory
and finally	Introductory
more now on	Introductory
stay with us	Pointer
still ahead	Pointer
when we come back	Pointer
i'll be (right) back	Pointer
when we return	Pointer
coming up	Pointer
we'll come back	Pointer
still to come	Pointer
in the next half hour	Pointer
after this	Pointer
in a moment	Pointer
we'll be (right) back	Pointer
will continue	Pointer
thanks	Passing
thank you	Passing
welcome back	Return from commercial
and we're back	Return from commercial
i'm PERSON	Sign off
PERSON STATION	Sign off
STATION's PERSON	Sign off
reporting from PLACE	Sign off
this is PERSON	Sign off
live from PLACE	Sign off

Table 3.2: Domain cues we identified by hand using 36 documents from the HUB-4 broadcast news corpus.

the u. n. says it's observers will stay in liberia only as long as west african peacekeepers do but west african states are threatening to pull out of the force unless liberia's militia leaders stop violating last year's peace accord after seven weeks of chaos in the capital monrovia relative calm returned this week as peace troops redeployed fighters stashed their guns as faction heads claimed another truce but peacekeeping officials warn they can't sustain a cease-fire without more troops and equipment and for that they need more western aid meanwhile the security council friday also urged member countries to enforce a nineteen ninety two arms embargo against liberia peacekeeping officials complain of constant violations human rights groups cite peace troops as among those smuggling the arms **i'm jennifer ludden** reporting

whitewater prosecution witness david hale began serving a twenty eight month prison sentence today the arkansas judge and banker pleaded guilty two years ago to defrauding the small business administration hale was the main witness in the whitewater related trial that led to the convictions of arkansas governor jim guy tucker and james and susan mcdougall hale initially said that then governor bill clinton pressured him to make a three hundred thousand dollar loan to susan mcdougall in nineteen eighty six

Figure 3.1: Example of a domain cue marking the boundary between topic segments in a fragment of a transcript of an episode of National Public Radio's show *All Things Considered*. The phrase in bold is a domain-specific cue phrase of the form: *I'm* PERSON. The gap separates two different news stories.

We trained our model using the annotated training data from the MUC-7 Named Entity Task, but we identified only a subset of the types participants in the MUC competition were expected to label [Chinchor, 1997]. Our subset did not contain some of the simpler types which could be identified using regular expressions, including monetary amounts, (such as $1,000.00), percentages, (e.g. 15%) and dates (e.g. May 23, 1972).

The maximum entropy model predicts the most likely label—PERSON, PLACE, COMPANY or none of these—for each word in a document using features present in the word's surrounding context. The model uses these features:

- Is the word on the list of corporate designators shown in Figure 3.3?
- Is the previous word on the list of corporate designators shown in Figure 3.3?
- Is the next word on the list of corporate designators shown in Figure 3.3?
- Is the word on a list of places taken from the MUC-6 gazetteer?
- The identity of the two preceding words together.
- The identity of the preceding word.
- The identity of the two following words together.
- The identity of the following word.
- The probability that the word was capitalized when used in the middle of a sentence in the Treebank *Wall Street Journal* corpus.

All of the features except the final one are self-explanatory. The last one is a rough indicator of how often a word is part of a proper noun and is useful for identifying person and company names and place names not in the gazetteer.

For example, the model would use the features shown in Figure 3.2 to predict the most likely label for the word *apple* in Example 3.1.

(3.1) because of the heavy selling of widely held stocks such as oracle systems **apple** corporation and lotus development the nasdaq composite index slumped 4.74 or 1 percent to 458.15

AB	A.B.	Aktiobolog	AE	A.E.
AENP	AG	AG&COKG	AL	A/L
AMBA	A.M.B.A.	AO	A.O.	APS
A&P	AS	AS	A. S.	A/S
AY	BA	B. A.	BHD	BM
B.M.	BSC	BV	BVBA	B.V.B.A.
BVCV	B.V./C.V.	CA	C.A.	CDERL
CV	C.V.	Company	CO	CO.
Corporation	CORP	CORP.	CPORA	CPT
EC	E.C.	EG	EGMBH	EPE
E.P.E.	GMBH	Ges.m.b.H	GBR	GGMBH
GMK	GM.K	GMBH&COKG	GP	G.P.
GSK	HF	H.F.	HMIG	H.Mij
H.Mig	HVER	H.Ver.	Incorporated	INC
Inc.	IS	I/S	KB	KG
KGAA	K.G.a.A	KK	KS	K/S
KY	LDA	LTD	LTDAPS	LLC
L.L.C.	LP	L.P.	LTDA	MIJ
NL	N.L.	NPL	N.P.L.	NV
N.V.	OHG	OE	O.E.	OY
OYAB	PERJAN	PERSERO	PERUM	PLC
PN	PP	PT	PVBA	SA
S.A.	SAC	S.A.C.	SACA	S.A.C.A.
SACC	SACCPA	SACEI	SACIF	S.A.C.I.F.
SADECV	SAIC	SAICA	S.A.I.C.A.	SALC
S.A.L.C.	SANV	S.A.N.V.	SARL	S.A.R.L.
SAS	S.a.s.	SCI	S.C.I.	SCl
S.C.L.	SCP	S.C.P.	SCPA	S.C.p.A.
SCPDEG	SCRL	SDERL	SDERLDECV	SENC
SENCPORA	SEND	Send.	SICI	S.I.C.I.
SL	S.L.	SMA	S.M.A.	SMCP
S.M.C.P.	SNC	S.N.C.	SPA	S.P.A.
SPRL	S.P.R.L.	SRL	S.R.L.	SV
SZRL	S.Z.R.L.	TAS	T.A.S.	UPA
U.p.a.	VN	WLL	W.L.L.	

Table 3.3: List of corporate designators used for named entity recognition.

```
Word=apple
ProbabilityUpperCase=.61
PreviousWord=systems
SecondPreviousWord=oracle
PreviousTwoWords='oracle systems'
NextWord=corporation
SecondNextWord=and
NextTwoWords='corporation and'
NextWord=CorporateDesignator
```

Figure 3.2: Features used by our named entity recognizer when determining the label for the word *apple* in Example 3.1.

The fact that *apple* precedes a corporate designator and has substantial probability of being capitalized in training data should be sufficient to enable the model to identify *apple* as part of a company name.

We designed this named entity recognizer for speech-recognized or transcribed text which lacks punctuation and contains only lowercase letters. It had labeling accuracy of 96.0 percent on the MUC-6 named entity test corpus. That is, it labeled each token in the text as either a person, place, company or not a named entity with only 4.0 percent error. A simple baseline algorithm which posited that every token was not a named entity achieved 91.8 percent accuracy.

The MUC Named Entity data came from the *New York Times*. In order to build a model for speech-recognized data, we preprocessed the training and test data to remove capitalization and punctuation. As a result, we could not directly compare the performance of our model to other systems, such as the entries in the MUC-6 competition (e.g. [Krupka, 1995]) or the Nymble system [Bikel et al., 1997], since they used punctuation and capitalization to help identify named entities. Our focus on only a subset of the categories labeled in the MUC competition also complicated comparing our technique with others from the literature.

3.2.2 First Uses

Youmans suggested that first uses of words within documents often accompany topic shifts [Youmans, 1990]. New people, places and events are often discussed using words not found in the preceding text when a new topic segment begins. At the start of a document, when the majority of words are used for the first time, there will obviously be a higher proportion of first uses than in later portions of a document. The preponderance of first uses at the start of a document is partly due to the use of words which occur independent of topic. The word *the*, for instance, is likely to be used in a text about any subject, and will probably be used in the first few sentences. This observation applies to most frequent function words.

In long documents, the number of first uses associated with new topics will decline as the document progresses because authors have finite vocabularies. Despite these complications, an unusually large number of first uses is likely to be a good indicator of the start of a new topic. Figure 3.3 presents an example, again from the HUB-4 corpus, that shows the number of first uses within a new topic segment. Note the large number of first uses immediately following the vertical white space in the figure, which marks the shift to a new topic. Twelve of the first thirteen words in the segment about Whitewater occurred for the first time in the document.

Considering only first uses of content words reduces the severity of the bias toward first uses at the beginning of documents. Ignoring first uses of function words is also beneficial because their usage is less dependent on topic than content words. The sample NPR transcript is shown again in Figure 3.4, this time with only first uses of non-function words highlighted.

3.2.3 Word Repetition

Halliday and Hasan's work on lexical cohesion [Halliday and Hasan, 1976] pointed out that the repetition of words and phrases provides coherence to a text. They also observed that the degree of lexical cohesion within a topic segment should be greater than across a topic boundary. This forms the basis of Morris and Hirst's lexical chaining algorithm [Morris and Hirst, 1991]. Their technique required some hand annotation, since not all lexical cohesion relationships in text can be reliably identified computationally. Hearst's work

the u. n. says it's **observers** will **stay** in liberia **only** as **long** as west **african peacekeepers do** but west african **states are threatening** to **pull out** of the **force unless liberia's militia** leaders **stop violating last year's** peace **accord after seven weeks** of **chaos** in the **capital monrovia relative calm returned** this **week** as peace troops **redeployed fighters stashed** their **guns** as **faction heads claimed another truce** but **peacekeeping officials warn** they **can't sustain** a **cease-fire** without **more** troops and **equipment** and for that they need more **western aid meanwhile** the security council friday also **urged member countries** to **enforce** a **nineteen ninety two arms embargo against** liberia peacekeeping officials **complain** of **constant violations human rights groups cite** peace troops as **among those smuggling** the arms i'm jennifer ludden **reporting**

whitewater prosecution witness david hale began serving a **twenty eight month prison sentence** today the **arkansas judge** and **banker pleaded guilty** two **years ago** to **defrauding** the **small business** administration hale **was** the **main** witness in the whitewater **related trial** that **led** to the **convictions** of arkansas **governor jim guy tucker** and **james** and **susan mcdougall** hale **initially** said that **then** governor **bill** clinton **pressured him** to **make** a **three hundred thousand dollar loan** to susan mcdougall in nineteen **eighty six**

Figure 3.3: Example of a large number of first uses of words marking a new segment. This is a fragment of a transcript of an episode of National Public Radio's show *All Things Considered*. Words in bold are used for the first time. The gap is between two news items.

the u. n. says it's **observers** will **stay** in liberia **only** as **long** as west **african peacekeepers** do but west african **states** are **threatening** to **pull** out of the **force** unless **liberia's militia** leaders **stop violating last year's** peace **accord** after **seven weeks** of **chaos** in the **capital monrovia relative calm returned** this **week** as peace troops **redeployed fighters stashed** their **guns** as **faction heads claimed another truce** but **peacekeeping officials warn** they can't **sustain** a **cease-fire** without more troops and **equipment** and for that they need more **western aid meanwhile** the security council friday also **urged member countries** to **enforce** a **nineteen ninety two arms embargo** against liberia peacekeeping officials **complain** of **constant violations human rights groups cite** peace troops as among those **smuggling** the arms i'm jennifer ludden **reporting**

whitewater prosecution witness david hale began serving a **twenty eight month prison sentence** today the **arkansas judge** and **banker pleaded guilty** two **years** ago to **defrauding** the **small business** administration hale was the **main** witness in the whitewater **related trial** that **led** to the **convictions** of arkansas **governor jim guy tucker** and **james** and **susan mcdougall** hale **initially** said that then governor **bill** clinton **pressured** him to **make** a **three hundred thousand dollar loan** to susan mcdougall in nineteen **eighty six**

Figure 3.4: Example of a large number of first uses of open-class words marking a new segment. This is a fragment of a transcript of an episode of National Public Radio's show *All Things Considered*. Words in bold are used for the first time. The gap separates two news stories.

using the vector space model [Hearst, 1994b], our optimization algorithm [Reynar, 1994] and a number of other algorithms in the literature approximate the identification of simple lexical cohesion relationships by looking at patterns of word repetition.

Figure 3.5 shows the number of word repetitions within an excerpt of NPR's program *All Things Considered*. The number of repetitions within each topic segment is greater than the number which cross the topic boundary. This anecdotally demonstrates that the existence of few repetitions spanning a potential topic boundary is a good indicator that a boundary is present.

The use of function words does not depend heavily on the topic being discussed. As we noted before, the word *the* appears in almost all documents. Also, function words account for a large percentage of the words in most documents. In the Penn Treebank *Wall Street Journal* corpus, open class words account for only 57.7 percent of all tokens. For these two reasons, focusing on word repetition of only open-class words or lemmas should be beneficial for segmentation accuracy and speed. Figure 3.6 shows the same excerpt of *All Things Considered* as the previous figure with only repetitions of open-class words linked together. Ignoring function words reduces the number of repetitions across the topic boundary to one, while the number within each topic segment is at least four.

Multiple occurrences of identically spelled words do not necessarily contribute to cohesion. Texts may contain homographs, words which are spelled the same but have different meanings. For example, *lie* can mean either prevaricate or recline. The two meanings come from different root words: *lie* and *lay*, respectively. But there are also cases where the roots are the same, but the meanings differ. This eliminates the possibility of relying on morphology normalization. The goal of word-sense disambiguation algorithms is to identify the intended meaning in both cases. (e.g. [Yarowsky, 1992]) Part-of-speech tagging is sufficient to determine the appropriate sense in some cases, but in others, more sophisticated techniques are needed. However, we ignore these problems and assume that repetitions of identical word forms contribute to cohesion. Justification for this decision comes from work which suggests that generally only one meaning is associated with each word type in a discourse [Gale et al., 1992].

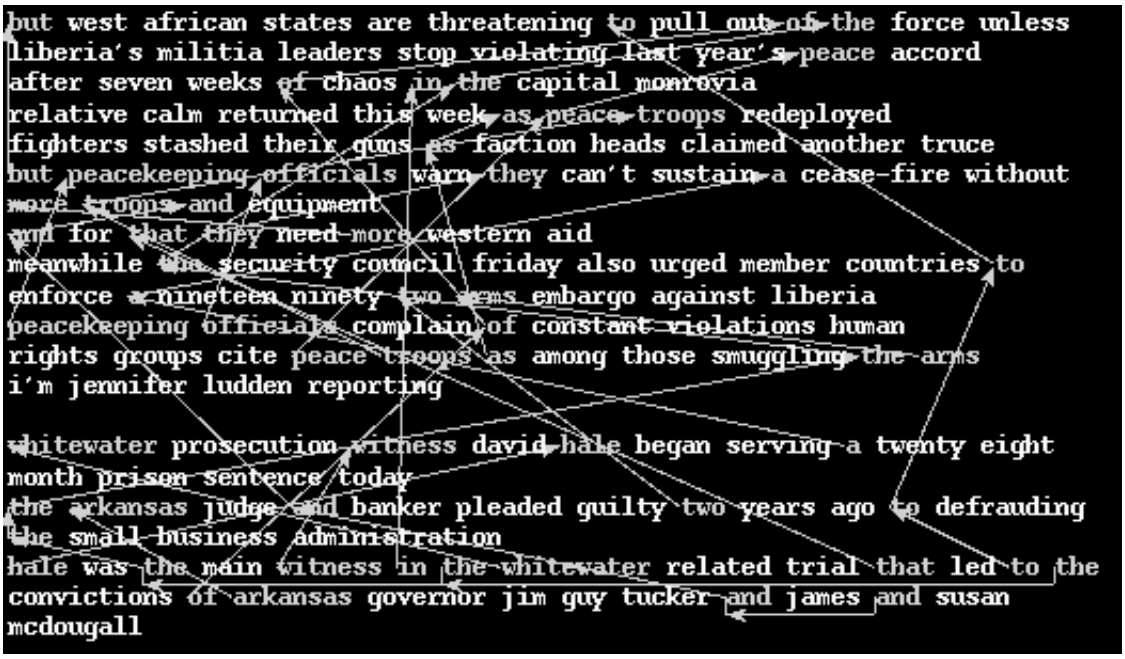

Figure 3.5: Example showing the degree of word repetition within topic segments. The data is from the National Public Radio program *All Things Considered*.

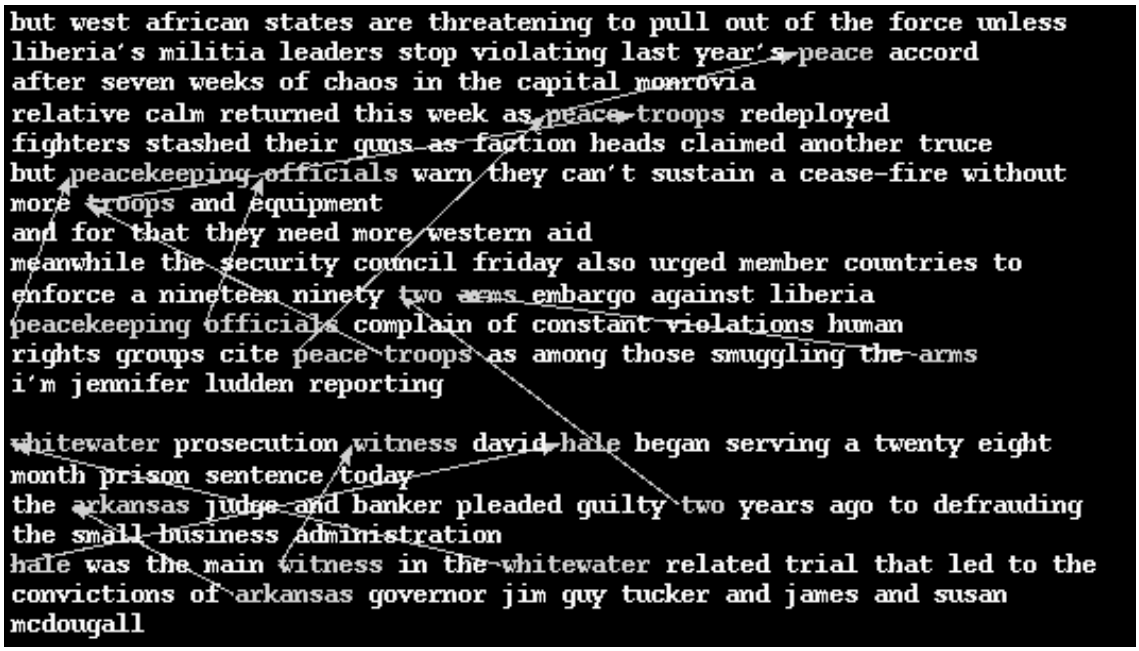

Figure 3.6: Example demonstrating the quantity of content word repetition within topic segments. The text is from the National Public Radio program *All Things Considered*.

3.2.4 Word n-gram Repetition

A natural extension of the word repetition techniques from the previous section is to count repetitions of word n-grams. Looking for repetitions of multi-word phrases has one primary advantage. Phrases are less likely than words to occur independently in unrelated topic segments. One reason for this is that phrases exhibit fewer sense ambiguities than words. The number of phrases likely to incidentally overlap in discourses about different topics should be much smaller than the number of incidentally overlapping words, thus making phrases better indicators of topic segmentation than words.

None of the topic segmentation techniques in the literature directly use word n-grams. The approach proposed by Beeferman *et al.* uses n-grams within a language model, but does not explicitly track the frequency of multiple word phrases [Beeferman et al., 1997b].

The number of repeated bigrams in text is smaller than the number of repeated words and the number of trigrams which repeat within documents is smaller than the number of bigrams. This trend applies even more strongly to longer n-grams. As a result, our algorithms use only repetitions of bigrams, since trigrams will provide little additional information. In the Penn Treebank *Wall Street Journal* corpus, there are 105294 bigrams which occur more than once, but only 34472 trigrams and merely 12469 4-grams. Figure 3.7 demonstrates the usefulness of tracking repetitions of bigrams for identifying shifts in topic. The presence of multiple instances of a particular bigram suggests that the regions containing those bigrams most likely are within the same topic segment.

As is the case with word repetition, some repeated phrases are more informative than others. In particular, bigrams of function words, such as *of the* are frequent and should be given little weight as hints about topic structure. Bigrams consisting of a content word and a function word—for example, *the book*—convey little additional information beyond the repetition of the content word alone. Therefore, in our algorithms we restrict the bigrams that contribute evidence about segmentation to be those containing two content words. The statistics regarding n-gram frequency are even more skewed for n-grams of only content words. The Treebank *Wall Street Journal* corpus contains 19089 content bigram repetitions within documents, but only 2644 and 253 repetitions of trigrams and 4-grams,

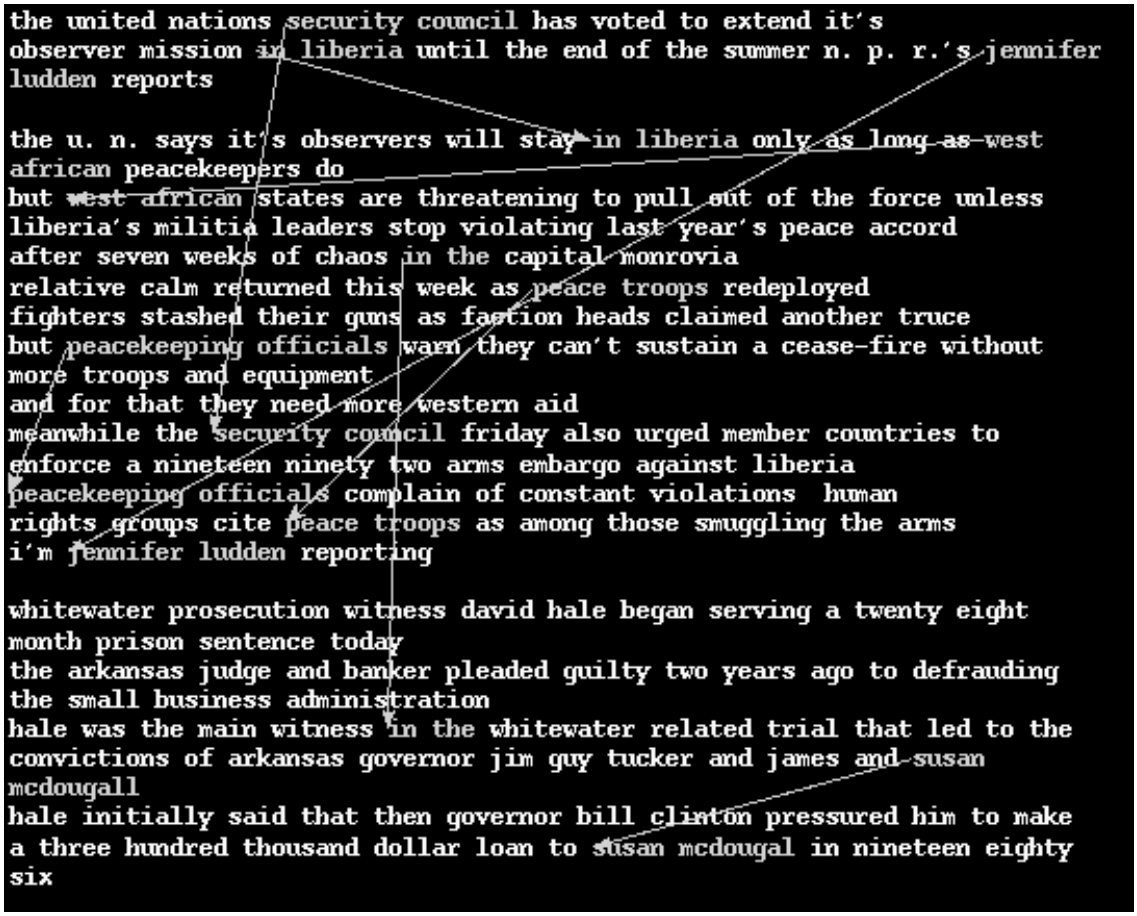

Figure 3.7: Example indicating the usefulness of tracking the repetition of word bigrams for topic segmentation. The data is from the National Public Radio program *All Things Considered*.

respectively. Figure 3.8 shows a portion of a document with repetitions of contentful bigrams annotated.

A more sophisticated approach than this might consider only the repetition of terminology. Example terms are *modal dialog box*, *junk bond*, *New York*, *Alan Greenspan* and *American Telephone and Telegraph*. The last example points out one advantage of identifying terminology: phrases containing function words are permitted but only when they are likely to be informative. We could use a system that identifies terminology, such as the one Justeson and Katz propose [Justeson and Katz, 1995], to identify interesting multiple

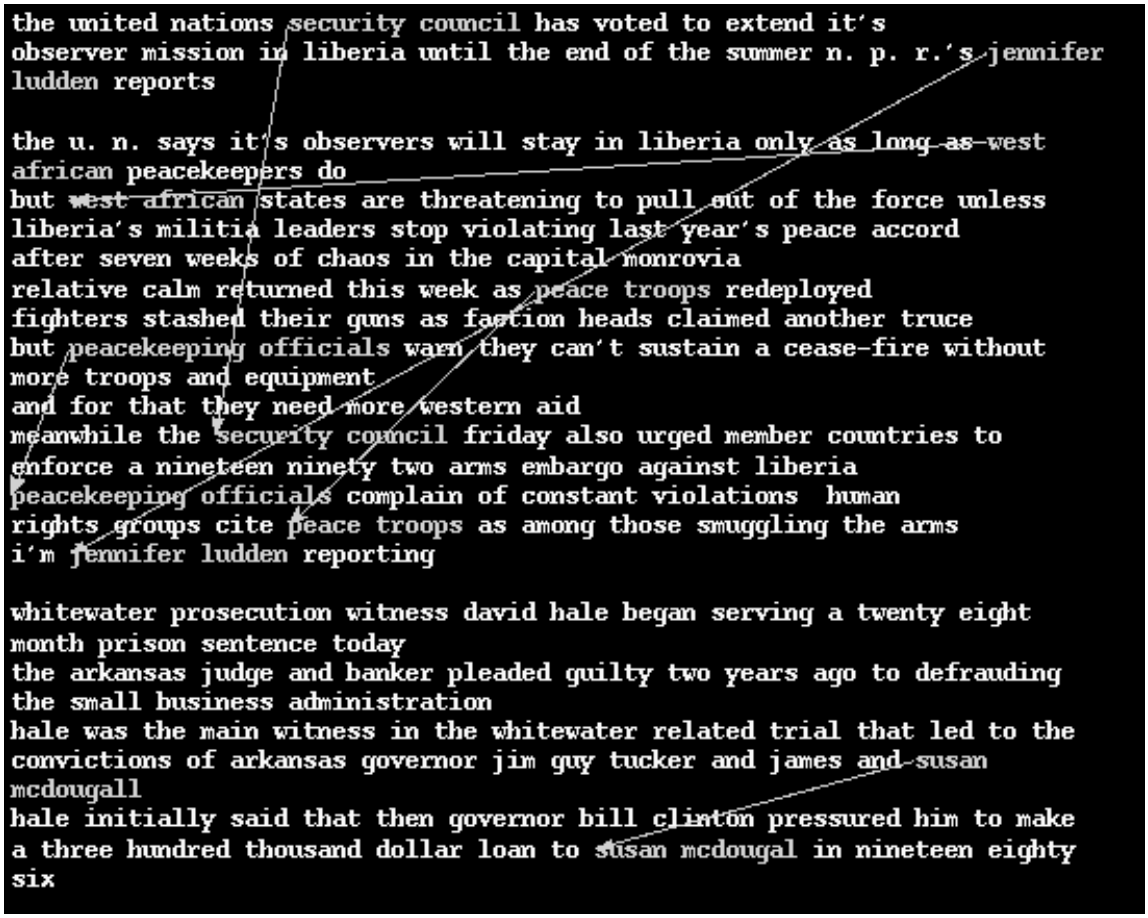

Figure 3.8: Example showing the usefulness of tracking the repetition of content word bigrams for topic segmentation. The text is transcribed from the National Public Radio program *All Things Considered*.

word phrases. We could then track repetitions of these phrases rather than contentful bigrams. This approach would limit the applicability of segmentation algorithms to domains for which large training corpora exist. It would also prevent bigrams seen for the first time in test data from contributing information about segmentation. As a result we will use repetitions of bigrams containing two content words as an indicator of topic shift.

3.2.5 Word Frequency

Using word frequency for topic segmentation differs from using models of word or bigram repetition in that word frequency assumes prior knowledge about how often individual words occur in a corpus. Models which predict the frequency of occurrence of words are called language models. They are most often used in speech recognition (for instance [Lau et al., 1993]) but have also been applied to optical character recognition [Hull, 1992] and a wide variety of problems in natural language processing including author identification [Mosteller and Wallace, 1964], language identification [Dunning, 1994], part-of-speech tagging [Church, 1988], text compression [Suen, 1979] and spelling correction [Kukich, 1992].

One advantage of using word frequency to detect shifts in topic over merely counting the number of word repetitions is that repetitions of words can be weighted differently depending on their probability. Simply because the word *and* occurs in two neighboring sentences does not imply that the sentences are within the same topic segment. In fact, it hardly says anything about the likelihood that they are in the same topic segment. However, if the word *onomatopoeia* occurs in neighboring sentences, even without reading the rest of the text, we could guess with high confidence that the two sentences were in the same segment. This is because *and* is frequent and can occur in the discussion of any topic, while *onomatopoeia* is rather rare and is consequently unlikely to be used in discussing most subjects.

To address this problem, we could modify a word repetition algorithm to ignore frequent words, but even among content words, there is a continuum of importance. Repetitions of light verbs like *make* might be more informative than repetitions of function words such as *and*, but are still far less helpful for segmentation than repetitions of rare content

words. Using word frequency provides a useful way to weight words: repetitions of higher frequency words should contribute less information about segmentation than repetitions of lower frequency ones. Closed class words, if not simply ignored, will consequently be paid little heed. In the *Wall Street Journal*, *and* occurs more often than *make* which occurs more often than *onomatopoeia*. Repetitions of *onomatopoeia* will be weighted more highly than those of *make* which in turn will count more than repeat occurrences of *and*.

Another advantage of tracking word frequency is that occurrences of one word type can be weighted differently depending on their probability. For example, a model of word frequency might assign the first occurrence of a word a different probability than the second occurrence, which in turn might have a different probability than all additional occurrences. Word repetition algorithms, on the other hand, tend to treat all occurrences identically. See Section 5.7 for a discussion of burstiness—the phenomenon that content words are more likely to repeat once they have been used once.

There is a cost associated with using the additional information exploited by a word frequency algorithm. We can track word repetition without a statistical model of language. In fact, assuming no preprocessing is done, we could segment text in any language which is not highly agglutinative using our optimization algorithm [Reynar, 1994] or the version of Hearst's *TextTiling* which does not normalize for term frequency [Hearst, 1994b]. Algorithms which rely on word frequency, such as the language modeling technique developed by Beeferman *et al.* [Beeferman et al., 1997b], require knowing the language of the text and make assumptions about its content as well. Such assumptions are necessary because the frequency of occurrence of words crucially depends on the subject matter being discussed. The word *million* is quite frequent in the Penn Treebank *Wall Street Journal* corpus. It occurs 4.5 times in every 1000 words. In the Brown corpus, however, it only occurs 0.2 times in 1000 words [Kučera and Francis, 1967]. Even though much of the Brown corpus is newspaper text, *million* is 22 times more likely to occur in a sample of financial news than a sample from that corpus. This variation in word frequency suggests that the utility of a technique based on word frequency will depend on accurate knowledge of the source of the text.

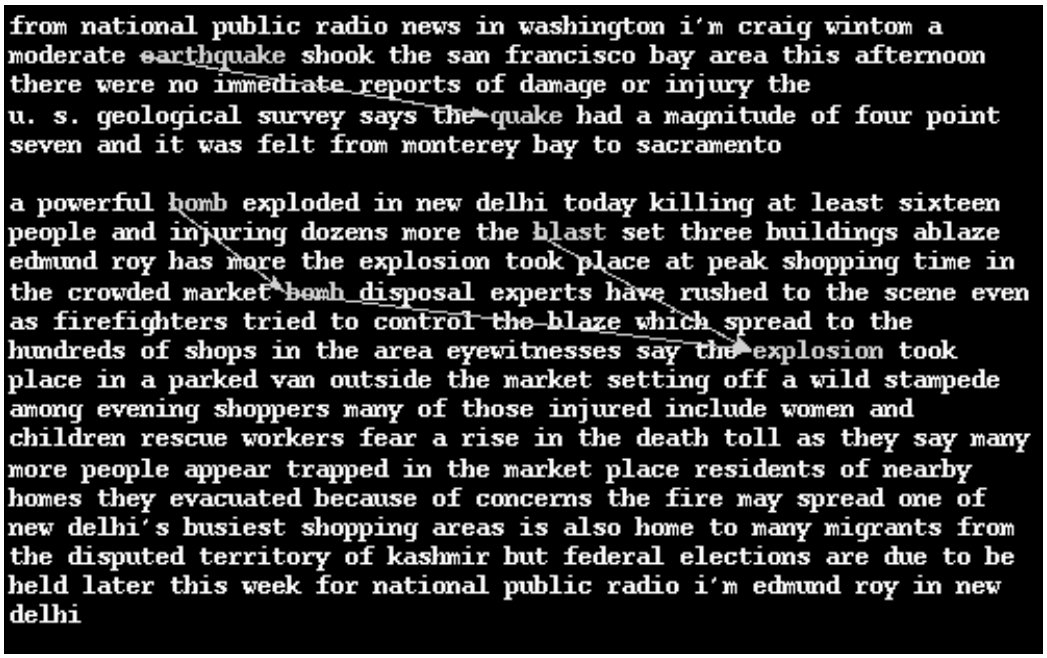

Figure 3.9: Example indicating the utility of tracking synonyms for identifying topic boundaries. The data is from the National Public Radio program *All Things Considered*.

3.2.6 Synonymy

A subset of Halliday and Hasan's lexical cohesion relationships can also be captured by identifying pairs of words which are synonyms. Figure 3.9 is an example from NPR's *All Things Considered* which shows the relations between synonyms.

Identifying synonyms computationally using a knowledge source which encodes synonymy, such as WordNet [Miller et al., 1990] or Roget's 1911 Thesaurus [Roget, 1911], can be problematic. Both WordNet and Roget's Thesaurus were designed for broad coverage, which means that they contain synonymous relations between words in a wide variety of contexts. This is advantageous for human users, however, it causes spurious synonymous relations to be identified algorithmically. For example, WordNet considers *man* to be synonymous with *operate* because both words have the sense of "work." These words are legitimate synonyms, but not in all contexts. Part-of-speech tagging could prevent

these two words from being labeled as synonyms, but the nouns *plant* and *factory*, for example, are not always synonymous. Without part-of-speech tagging and good word-sense disambiguation techniques, naive use of a thesaurus can result in the overgeneration of synonymous relations between words.

Another problem with these knowledge sources is coverage. Synonyms restricted to particular domains, such as technical or legal text, are unlikely to be present. For instance, the word *keyboard* is not in Roget's 1911 Thesaurus at all. That thesaurus predates the computer by many years, but language evolves and new jargon is constantly created, as evidenced by WordNet's lack of an entry for *trackball*.

3.2.7 Named Entities

The indicators of topic shift described in the previous sections all address Halliday and Hasan's category of lexical cohesion. Relations from the reference category also pertain to topic shifts. Many referential links involve pronouns and can therefore only be identified using pronoun resolution techniques. Figure 3.10 shows a portion of a transcript of an episode of NPR's *All Things Considered* annotated with links indicating which phrases refer to the same entities.

Unfortunately, pronoun resolution remains an unsolved problem and most computational systems today address only a subset of it [Baldwin, 1997]. However, references made using repetitions of people's names, the names of companies or organizations, and place names can be easily detected. In the same way that identical word n-grams are unlikely to arise independently in different topic segments, repetitions of proper names are also unlikely to occur by chance in neighboring topic segments. As a result, they too are good indicators that portions of text are within the same topic segment. Figure 3.11 shows the same transcript as the previous figure, but this time only coreference links involving proper names are indicated.

We can identify coreference links involving proper names, so-called Named Entities [Chinchor, 1997], using the statistical model we built to identify proper nouns which occur within dynamic cue phrases. We do not attempt to resolve pronouns, but instead identify only references involving names and portions of names. For instance, we would identify

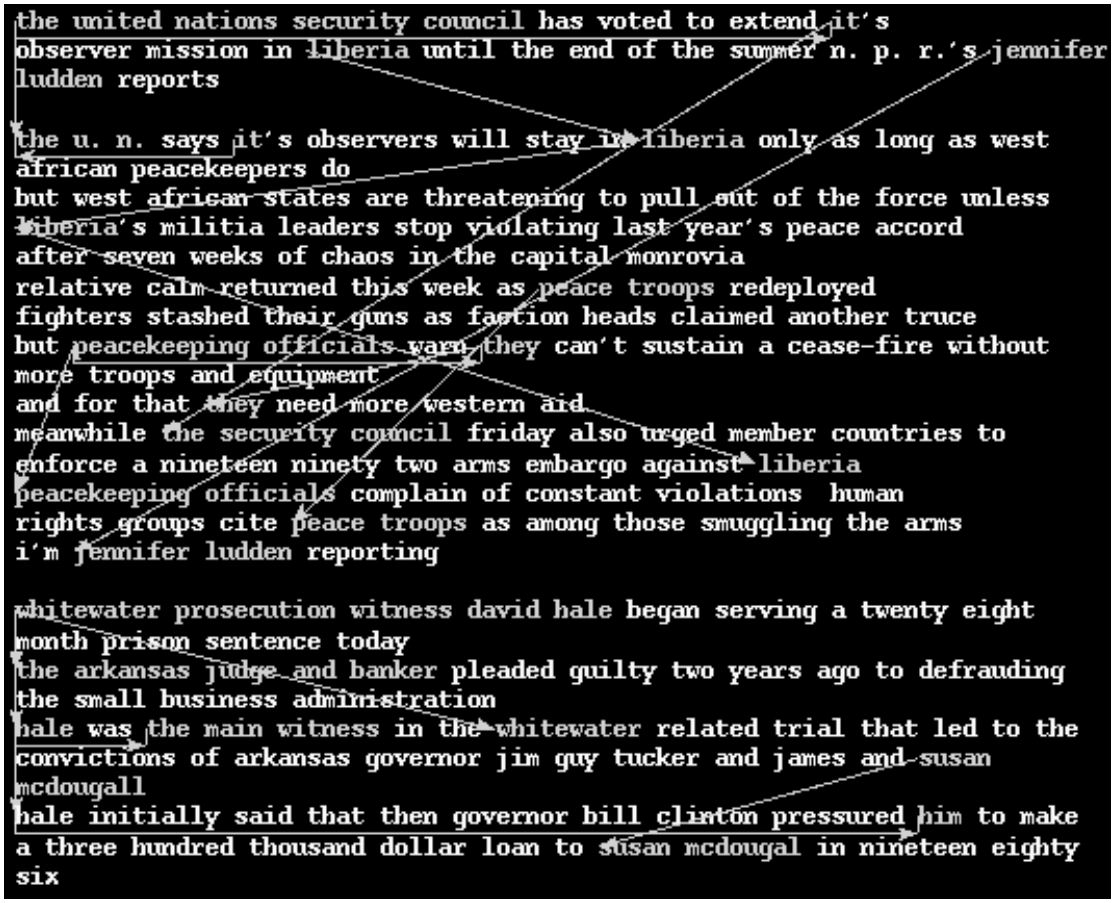

Figure 3.10: Example transcript indicating the usefulness of coreference for topic segmentation. The text is transcribed from the National Public Radio program *All Things Considered*.

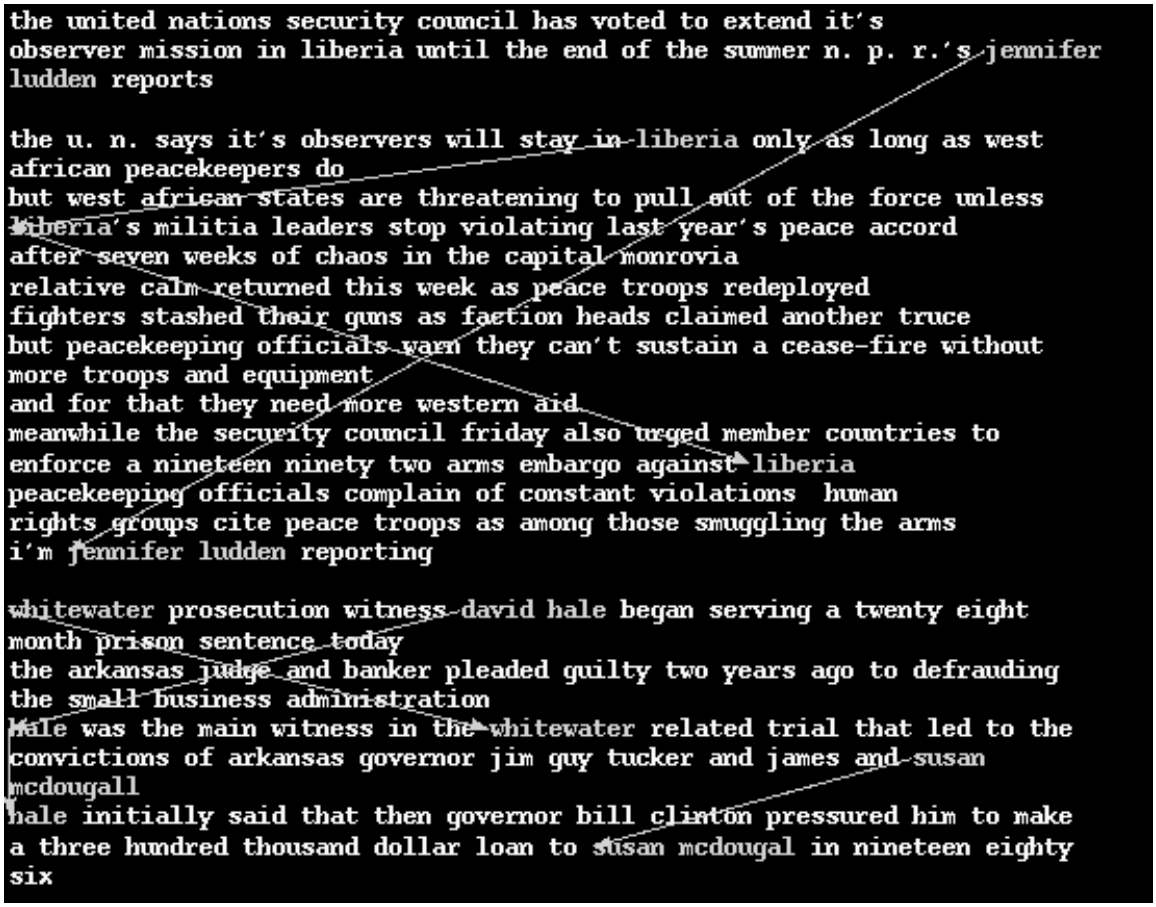

Figure 3.11: Example showing the usefulness of limited coreference between named entities. The text is a transcript of the National Public Radio program *All Things Considered*.

the relationship between *International Business Machines Corporation* and *International Business Machines*, but would miss a reference to *IBM* as *the company* or *it*.

Most of the repetitions of Named Entities we will identify can be found using string matching techniques, and are consequently subsumed by approaches which identify repetitions of n-grams. Named Entities merit special attention, however, because at least in the domain of news broadcasts, they are particularly informative clues. Most news items are about the doings of particular people, companies or nations and most of these entities figure in few news stories at once. As a result, distinguishing repetitions of these entities from other word n-gram repetitions and weighting them more highly should improve topic segmentation performance.

3.2.8 Pronoun Usage

In her dissertation, Levy described a study of the impact of the type of referring expressions used, the location of first mentions of people and the gestures speakers make upon the cohesiveness of discourse [Levy, 1984]. She found a strong correlation between the types of referring expressions people used, in particular how explicit they were, and the degree of cohesiveness with the preceding context. Less cohesive utterances generally contained more explicit referring expressions, such as definite noun phrases or phrases consisting of a possessive followed by a noun, while more cohesive utterances more frequently contained zeroes and pronouns.

We will use the converse of Levy's observation about pronouns to gauge the likelihood of a topic shift. Since Levy generally found pronouns in utterances which exhibited a high degree of cohesion with the prior context, we investigate the hypothesis that the use of a pronoun contraindicates the presence of a topic boundary. Thus, the presence of a pronoun among the first words immediately following a putative topic boundary provides some evidence that no topic boundary actually exists there.

Figure 3.12 presents an example of this phenomenon from the HUB-4 Broadcast News corpus. In the example text, there is no topic boundary before the sentence beginning *he said finally.*

twenty-nine year old by the name of todd reeche who is from hot springs montana an upset winner in the javelin he grew up on uh the flat head katuni indian reservation for eighteen years and came down here a little heralded but uh there's something called the native american sports council and john egaldey from the sports council got together and held a religious ceremony with todd uh the night before the race which he credits in sort of giving him a certain extra spirit uh he won the event on his very first throw and he said it was the biggest thrill since he was in high school he went to hot springs high school graduating class of twenty four his senior year he won the state track and field championships by himself when he won the hundred meters two hundred four hundred three hundred hurdles and of course the javelin
wow
he said finally he had matched his performance of high school
goodness sakes and and finally give us one event just one key event for wednesday in atlanta

Figure 3.12: Example of a text region beginning with a pronoun which contraindicates the existence of a preceding segment boundary. This is a fragment of a transcript of an episode of National Public Radio's show *All Things Considered*. The crucial pronoun is shown in bold. There is no topic boundary before the sentence beginning with *he said finally*.

It seems unlikely in general that a new segment would begin with a pronoun. Of course there are exceptions, including instances of cataphora and uses of pronouns in metaphorical statements. Nonetheless, the presence of a pronoun at the beginning of a region of text which follows a putative boundary is a good indicator that no topic boundary is present.

3.2.9 Character n-gram Repetition

One of the complications of using either word repetition or word frequency to track topic shift is that word types often occur in different inflected forms within text. It is desirable to note the relationship between singular and plural forms of a particular noun and different inflected forms of verbs. We can identify these relations by using a morphological analyzer, such as the XTAG morphology system [Karp et al., 1992], to convert words to their roots prior to looking for repetition. Unfortunately, such systems are not available for all languages. Also, without first part-of-speech tagging the text, there will be many ambiguities. Take the word *said* for example. According to the XTAG morphology software there are two roots: *say* and *sayyid*. The first lemma is more frequently correct, but

the second equally valid lemma is for the noun form of *said*, which is an alternate form of *sayyid*, an Arabic word, meaning "Lord" or "Sir," which has crept into English. If we mistakenly determine that the lemma of the verb form *said* is *sayyid* then we will miss the relationship between *said* and *say*.

The previous example demonstrates that difficulties may arise from relying on naive lemmatizers to discover relations between words. This suggests that more sophisticated lemmatizers should be used, but such tools generally rely on part-of-speech tagging, which is difficult when standard cues such as capitalization, punctuation and sentence boundaries are not present, as is the case with speech-recognized text.

Instead of lemmatizing text and identifying word repetitions, we could ignore lemmatization and rely on n-grams of characters. For example, without morphology normalization, the words *said* and *say* have the character sequence *sa* in common. Figure 3.13 shows character n-gram overlap within a transcript of NPR's *All Things Considered*. This figure only highlights repetitions of characters within words with common roots. Indicating all instances of multiple character repetition would render the text of the figure unreadable. One particularly interesting relation shown in the figure is between the name *McDougall* and a misspelling of it which actually occurred in the transcript of that story.

There are drawbacks to using character n-grams. Some common words are spelled using character sequences that frequently occur in longer words. For instance, the most common word in many genres, *the*, is a substring of *there* and *other*. An algorithm which identifies similarities between regions of text using character n-grams will recognize these spurious similarities just as readily as it will observe legitimate ones involving, for instance, singular and plural forms of the same noun.

Removing function words from the data reduces the severity of this problem, but does not eliminate it. For instance, the open class word *dent* is a substring of the unrelated word *identify*. Also, unrelated words share features of inflectional and derivational morphology: the verb forms *takes* and *faxes* share the ending *es* but do not have a common root. It is unclear whether this sort of overlap will be a help or a hindrance. It could prove useful for segmentation by permitting the recognition of similarities in verb tense and writing style. Alternatively, it might cause many unhelpful substring repetitions to be found.

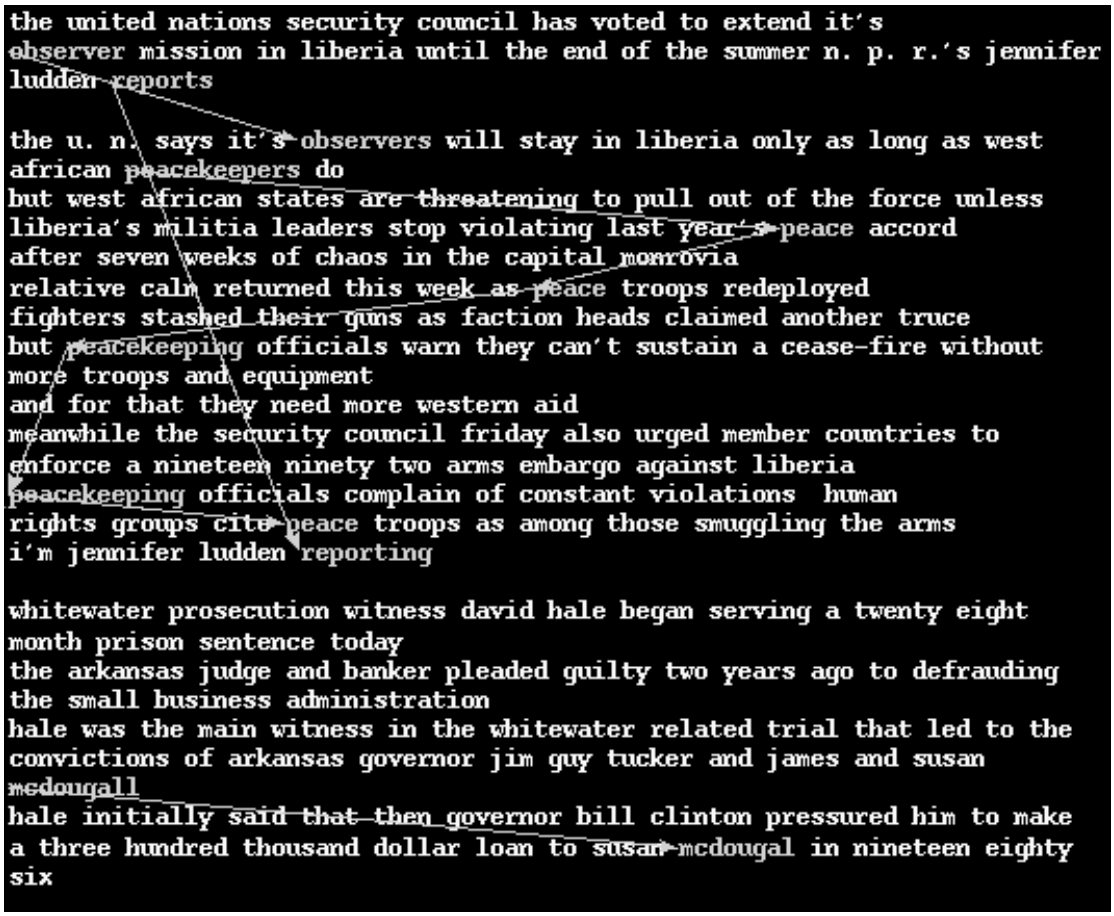

Figure 3.13: Example showing the utility of tracking character n-gram repetition for identifying topic boundaries. This is a transcript of the National Public Radio program *All Things Considered*.

Character strings have been used for text processing applications in the past. The literature on discourse structure does not mention tracking repeated character strings, but both Church's parallel text alignment technique [Church, 1993] and Helfman's work on text analysis depend on character string repetition [Helfman, 1994].

3.2.10 Additional Indicators

Our algorithms employ the indicators listed in the previous sections. However, there are other indicators one might use to identify topic shifts. For instance, structural parallelism is used to provide continuity to texts. Conversely, neighboring sentences with similar structure are therefore unlikely to have a topic boundary between them. Parallelism could be identified by processing the output of a parser such as the XTAG parser [XTAG Group, 1995] or Ratnaparkhi's statistical parser [Ratnaparkhi, 1997a].

Authors often number related points for clarity. The use of such enumerations generally signifies that sentences are within the same topic segment. It would be straightforward to identify sentences beginning with *first*, *second* and so forth using regular expressions.

Finally, major shifts in topic may be accompanied by changes in the complexity of the text. Text complexity could be measured using one of the so-called reading level indicators that are commonly found in word processing packages, such as the Fog index [Gunning, 1952]. Reading level according to this metric is a function of word length and sentence length.

All three of these suggestions are left to future work because they are likely to be relevant in a small proportion of documents. Also, parsing is computationally expensive and good performance generally necessitates annotating training data from the domain being parsed. This limits the applicability of the parallelism technique. More importantly, topic segmentation is meant to be applied at the early stages of language processing. Full parsing would most likely come later—especially if topic segmentation is used to improve preliminary processing steps, such as word sense disambiguation.

Chapter 4

Previous Text Segmentation Methods

Until recently publicly available topic-segmented corpora did not exist. As a result, comparing the performance of topic segmentation algorithms from the literature is difficult since researchers have evaluated their techniques a number of different ways. As a result, the simplest dimension on which to compare various algorithms is what features they use. Table 4.1 lists the features used by each of the algorithms described in this section. We will present a similar table in the next chapter for our own algorithms. The use of word repetition dominates previous work on segmenting text by topic. More than half of the techniques described below depend on word repetition. We will first summarize these techniques, then move on to methods which use only other features.

4.1 Word Repetition

The most frequently used indicator of topic shift is word repetition. Researchers have used word repetition alone to structure text in various ways and have also used it in conjunction with other features.

Algorithm	Cue Words & Phrases	Pronoun Usage	First Uses	Word Co-occurrence	Word Repetition	Word Frequency	Word n-grams	Named Entities	Synonymy	Character n-grams	Semantic Similarity
Morris & Hirst					⋆				⋆		
Hearst & Plaunt[a]					⋆	⋆					
Richmond *et al.*					⋆						
Yaari					⋆						
Van der Eijk					⋆						
Nomoto & Nitta					⋆						
Manabu & Takeo					⋆				⋆		
Berber Sardinha					⋆						
Beeferman *et al.*						⋆					
Phillips				⋆							
Youmans			⋆								
Kozima											⋆
Ponte & Croft											⋆

[a] *TextTiling* can be used both with and without $tf \cdot idf$ weighting. With $tf \cdot idf$, it is a word frequency algorithm and without it, it is a word repetition algorithm.

Table 4.1: The types of clues used by various text structuring algorithms.

4.1.1 Morris & Hirst

Morris and Hirst described a discourse segmentation algorithm [Morris and Hirst, 1991, Morris, 1988] based on lexical cohesion relations [Halliday and Hasan, 1976]. Since one of their goals was to provide support for Grosz and Sidner's theory of discourse structure [Grosz and Sidner, 1986], their algorithm divides texts into segments which form a hierarchical structure.

The first step in Morris and Hirst's algorithm is to link sequences of related words from a document to form *lexical chains*. Two words initially form a lexical chain when they are related by lexical cohesion. Each additional word added to an existing lexical chain must participate in a lexical cohesion relation with at least one word already in the chain. Morris and Hirst used Roget's thesaurus [Roget, 1977] to determine whether a pair of words satisfies one of these relations. They were forced to identify lexical chains by hand because Roget's 1977 thesaurus was not available in machine-readable form.

They used the thesaurus to determine whether there were lexical cohesion relations between words they thought likely to be related to the topic of a text, namely open class words which did not occur overly frequently. Understanding their algorithm requires knowledge of the organization of the knowledge source they used. Roget's 1977 thesaurus is divided into categories and has an index that indicates in which categories words appear. The categories themselves are paired and paired categories are labeled with words which are usually antonyms. Categories are also grouped into semantically related sets. Categories may contain both words and cross-references to other categories.

Roget's thesaurus is typically used to find synonyms for a particular word. One identifies synonyms by locating a word in the index and identifying the most pertinent category based on the word's meaning in context. From the words in that category, one selects the best synonym. For example, by looking up *text* in the index one finds that if the meaning is *part of writing*, the relevant category is number 55 which is labeled PART. This category contains a number of related words: *section*, *article*, and *page*, among others.

Morris and Hirst used the thesaurus much differently to identify lexical chains. They decided whether pairs of words satisfied Halliday and Hasan's lexical cohesion relation by

checking the index entries for the two words. Using their technique, two words are deemed to be related enough to be in the same lexical chain if any of the following are true:

1. They share a common category
2. One word is found in a category that contains a pointer to a category containing the second word
3. One word is a label of a category containing the other word
4. Each word is in a category containing a reference to a common category
5. The words are in the same group of categories

Examples of these relations will clarify things. *Text* and *article* are in the same category, number 55.2. They would be placed in a lexical chain for the first reason listed above. *Topic* and *essence* are related and would form a lexical chain for reason 2. Category 484.1 contains the word *topic* and a pointer to category 672.6 which contains the word *essence*. *Text* and *topic* would be in the same lexical chain for the third reason: TOPIC is the label of category 484, which has *text* as a member. *Document* can be found in category 602.10 which contains a pointer to category 570. *Record* is in category 608.4, which also points to category 570. *Document* and *record* would be placed in a lexical chain as a result of this common category. Finally, *text* and *composition* would be put in the same lexical chain because *text* is in category 55, *composition* is in category 58, and both these categories are in the group of categories labeled WHOLENESS.

In addition to excluding overly frequent words and closed-class words from participating in lexical chains, Morris and Hirst did not identify relations between pairs of words that were widely separated in the text. They handled relations between distant words which, if closer together, would have been in the same lexical chain by permitting lexical chains to be related to one another. After they identified all the lexical chains in a document, they compared the elements of chains to determine whether later chains were continuations of earlier ones. They labeled later chains that were related to earlier ones *chain-returns* because they revisited the topic of an established chain.

Morris and Hirst analyzed a small number of texts using their algorithm, but did not quantitatively evaluate its performance. Instead, they compared its output to the structure they identified for each discourse according to Grosz and Sidner's theory. Morris and Hirst found that the structures identified by their lexical chaining algorithm were similar to the structures they identified by hand.

Hearst later automated the lexical chaining process using an earlier version of Roget's thesaurus [Roget, 1911]. She found that the automatic algorithm did not label boundaries as well as Morris and Hirst's hand execution. Performance suffered because the 1911 version of the thesaurus was inferior to the 1977 version and because the automation of lexical chaining introduced errors. When building lexical chains manually, Morris and Hirst missed relations which Hearst's implementation of their algorithm found. Many of the relations found algorithmically were spurious and arose because of word sense ambiguities [Hearst, 1993].

4.1.2 Hearst

Hearst developed a technique to automatically divide long expository texts into segments several paragraphs in length, each of which was about a single subtopic. She chose to linearly segment text partly because of Skorochod'ko's work on the structure of texts and because of the difficulty of eliciting hierarchical segmentations from annotators [Rotondo, 1984, Passoneau and Litman, 1993].

Hearst's algorithm, *TextTiling*, is based on the vector space model, which determines the similarity between two texts by assuming documents can be represented as word vectors in a high dimensional space. In most information retrieval applications the two texts compared are the user's query and a document. *TextTiling*, however, uses the vector space model to determine the similarity of neighboring groups of sentences and places subtopic boundaries between dissimilar neighboring blocks.

The Vector Space Model

The simplest form of the vector space model treats documents as vectors whose values correspond to the number of occurrences of words in a document. For example, the phrase

give me liberty or give me death could be represented by the vector $< 2, 2, 1, 1, 1 >$. In this case, the first element of the vector is the number of occurrences of *give*, the second is the number of repetitions of *me* and so forth.

There are a number of ways to compute the similarity between two documents with the vector space model. The one most frequently used is the cosine distance, which determines the angle between the vectors associated with each document. The vectors that represent similar documents have a smaller angle between them than those that represent dissimilar documents.

If we refer to the two documents as D_i and D_j then we can write their similarity as $\texttt{sim}(D_i, D_j)$. If we label the word vector for the $i^{\texttt{th}}$ document W_i then the equation for the cosine distance measure is:

$$\texttt{sim}(D_i, D_j) = \cos(D_i, D_j) = \frac{W_i \cdot W_j}{|W_i||W_j|} \tag{4.1}$$

The formula for the cosine distance can also be written as shown below. n is the dimensionality of the word vectors and represents the number of different word types present.

$$\cos(D_i, D_j) = \frac{\sum_{k=1}^{n} W_{ik} W_{jk}}{\sqrt{\sum_{k=1}^{n} W_{ik}^2 \sum_{k=1}^{n} W_{jk}^2}} \tag{4.2}$$

In the second version of the formula, the summation over all word types makes it clearer that the number of common words and the number of times they appear in each document determines the similarity score.

Processing in *TextTiling*

Hearst describes in detail the processing steps used by *TextTiling* [Hearst, 1994a]. Her algorithm tokenizes a document and then removes common words, most of which are closed class, that are found on a stop-list. These words are eliminated because they are unlikely to be helpful for identifying subtopic sections.[1] *TextTiling* next reduces the remaining words to their morphological root using WordNet. Next, it divides the root

[1] That is the case unless issues of style or authorship are believed to affect the structure of a document. See [Mosteller and Wallace, 1964] for a discussion of the role of closed-class words in determining authorship and [Biber, 1989, Biber, 1990] for a discussion of the relationship between function words and register.

words into groups called pseudo-sentences. It then groups pseudo-sentences into fixed-size blocks. Finally, it computes similarity scores for adjacent blocks using the cosine distance measure. Computing similarity scores between blocks of pseudo-sentences, rather than paragraphs, eliminates difficulties with the vector space model that stem from length variation.

From the similarity scores, *TextTiling* then computes depth scores, which quantify the similarity between a block and the blocks in its vicinity. In terms of a graph of similarity scores, a depth score can be thought of as the sum of the differences between the top of the "peak" immediately to the left and right of a "valley." The computation of depth scores proceeds as follows: Start at a particular gap between two blocks and record the similarity score associated with the blocks on either side of that gap. Check the similarity score of the preceding gap. If it is higher, continue by examining the similarity score at the previous gap. Continue in this way until a score lower than a score already examined is found. Then, subtract the similarity score of the initial gap from the maximum similarity score encountered. Repeat this procedure for gaps between blocks following the first gap. Finally, sum the two differences computed. This value is the depth score for the first gap examined. Depth scores need only be computed for gaps which are local minima of the similarity function.

TextTiling next selects gaps with the highest depth scores as the sites of subtopic boundaries. The algorithm adjusts the identified locations to ensure that they correspond to paragraph boundaries. It also discards boundaries that lie too close to previously identified boundaries. An unsmoothed depth score graph is shown in Figure 4.1. Figure 4.2 shows the depth scores for the same data after one round of smoothing with a moving-average smoothing algorithm.

Hearst compared the segmentation produced by *TextTiling* to reader judgments of the locations of topic boundaries in thirteen magazine articles. She measured performance using the information retrieval metrics precision and recall. Precision is the ratio of the number of correct guesses to the total number of guesses and recall is the ratio of the number of correct guesses to the total number of answers in the scoring key. Values for precision and recall range from 0 to 1, with 1 indicating perfect performance. *TextTiling*

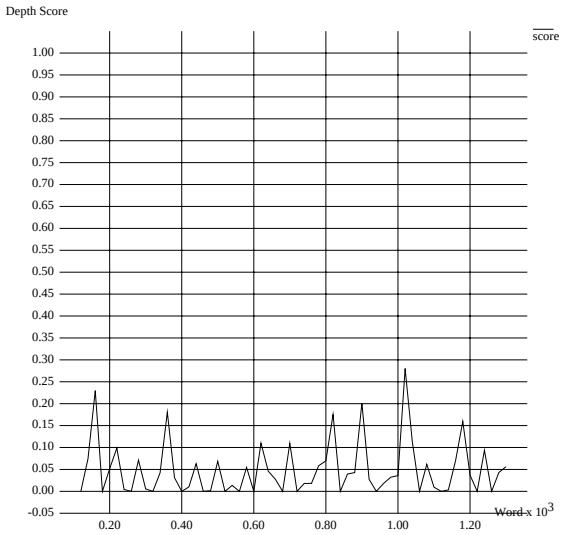

Figure 4.1: Unsmoothed depth score graph of two concatenated *Wall Street Journal* articles.

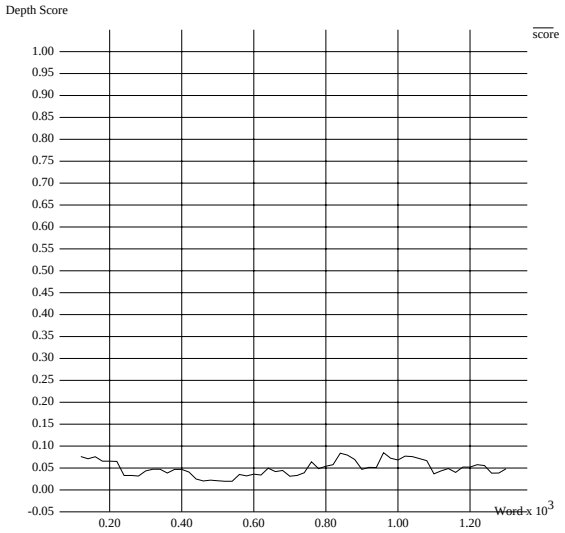

Figure 4.2: Smoothed depth score graph of two concatenated *Wall Street Journal* articles.

scored precision of 0.66 and recall of 0.61 when Hearst compared the segmentation *TextTiling* produced to the consensus labelings generated by a group of human judges. Hearst also evaluated her automated version of Morris and Hirst's lexical chaining algorithm. That algorithm performed at 0.64 precision and 0.58 recall.

The performance of both algorithms was better than two baseline techniques, which scored 0.44 precision and 0.37 recall and 0.43 precision and 0.42 recall by randomly guessing boundaries. The judges, however, were more consistent among themselves than any of the algorithms were compared to their collective judgments. The judges averaged 0.81 precision and 0.71 recall compared to the consensus of their individual annotations [Hearst, 1994b].

4.1.3 Richmond, Smith & Amitay

Richmond, Smith and Amitay also describe a technique for locating topic boundaries [Richmond et al., 1997]. Their method weights the importance of words based on their frequency within a document and the distance between repetitions. They determine the similarity between neighboring regions of text by summing the weights of the words which occur in both regions and then subtracting the summed weights of words which occur only in one segment. They normalize this figure by dividing by the number of words in each section.

Their algorithm has five steps. First, some basic preprocessing is done. Next, they calculate the weight of each word, which they call its significance. They compute these values using the formula below. Significance scores for different instances of the same word type may differ depending on context.

$$\texttt{significance}(x) = \frac{1}{n} \cdot \sum_{i=1}^{n} \texttt{arctan}(\frac{D_{x,i}}{\frac{W}{\omega}}) \tag{4.3}$$

x represents a particular word token. W is the number of word tokens in the document. ω is the number of occurrences of words of the same type as word x. $D_{x,i}$ is the distance between word x and the $i^{\texttt{th}}$ nearest repetition of that word. n is the number of nearest neighbors deemed useful for the significance computation and is determined by the formula below.

$$n = (\frac{8}{1 + e^{-200 \cdot (\frac{\omega}{W} - 0.02)}}) + 2 \tag{4.4}$$

The values of n range from two to ten. The `significance`(x) ranges initially from 0 to $\frac{\pi}{2}$ and is normalized to lie between 0 and 1, with 0 indicating minimum significance.

Richmond *et al.* use the significance values for each word to compute the similarity between two regions of a document. They determined the optimal size of the regions they compared to be fifteen sentences. The formula for the similarity between regions, what they call Correspondence, is presented below.

$$\texttt{Correspondence} = \frac{\frac{|A'| - |A''|}{|A|} + \frac{|B'| - |B''|}{|B|}}{2} \tag{4.5}$$

In the above formula A is the bag of words contained in the first region and B is the bag found in the second region. A word type appears in one of these bags more than once if it occurs more than once in the associated region of text. A' and B' are the portions of A and B, respectively, containing words of types that occur in both A and B. A'' and B'' are the portions of A and B, respectively, which contain words of types that occur only in A or B. The notation $|A|$ indicates summing the significance scores of the words in A.

Richmond *et al.* smooth the correspondence scores using a form of weighted average smoothing. Finally, they place topic boundaries where the correspondence scores are lowest.

They applied their algorithm to the text of articles from the front page of a newspaper and a psychology paper. Their results suggested that the algorithm performed well, but they did not perform a systematic evaluation using a corpus.

4.1.4 Yaari

Yaari proposed that expository texts could be segmented using hierarchical agglomerative clustering (HAC) [Yaari, 1997]. HAC initially places each element of a set in a class by itself and recursively merges the most similar classes until all items are in one class. HAC can be used to produce a dendogram that depicts the relationships between elements based on the order of the merges between classes. Yaari modified HAC for text segmentation to

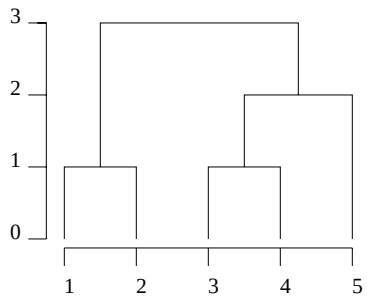

Figure 4.3: Sample dendogram of the type produced by Yaari's hierarchical clustering algorithm. The x-axis is the sentence number and the y-axis is the level of the merge.

permit only merges between neighboring segments. Figure 4.3 shows a dendogram similar to one produced by HAC.

Yaari removed closed class words from documents and then used Porter's algorithm to reduce the remaining words to their stems [Porter, 1980]. He then computed the similarity of paragraphs using the cosine measure with inverse document frequency (IDF) weighting, which weights rare words more highly than common ones. HAC used these similarity scores to group similar neighboring segments. After HAC clustered the paragraphs, Yaari created a dendogram showing the order of the merges, and then applied rules to convert the hierarchical clustering into a linear segmentation. He did this to facilitate comparing HAC's output to the linear segmentation produced by *TextTiling*.

He tested HAC's performance on a single article, the *Stargazers* article from Discover magazine which was in the collection Hearst used to evaluate *TextTiling*. He found that his algorithm better replicated the annotation produced by Hearst's judges than *TextTiling* did.

4.1.5 Other Approaches

Van der Eijk used *TextTiling* to compute the similarity between translations of documents in multiple languages [van der Eijk, 1994]. Nomoto and Nitta extended *TextTiling* to be used on Japanese texts [Nomoto and Nitta, 1994]. Manabu and Takeo developed a word-sense disambiguation algorithm which could also be used to perform text segmentation

[Manabu and Takeo, 1994]. Both Nomoto and Nitta's and Manabu and Takeo's algorithms were only semi-automatic and required some hand annotation. Berber Sardinha designed a number of techniques based on word repetition for manually segmenting text (e.g. [Berber Sardinha, 1993b, Berber Sardinha, 1993a].

4.2 Other Features

4.2.1 Beeferman, Berger & Lafferty

Beeferman, Berger and Lafferty describe a technique for identifying document boundaries using statistical techniques [Beeferman et al., 1997b]. At the heart of their method is a statistical framework called feature induction for random fields and exponential models. They built statistical models within this framework which incorporated a number of cues about the presence of story boundaries. These hints included:

- Do particular words appear up to several sentences prior to or following a potential boundary?
- Do particular cue words begin the preceding sentences?
- How well does a trigger-based language model [Beeferman et al., 1997a] predict the text compared to a static trigram language model?

The last feature provides a measure of topicality. Their trigger-based language model boosts the probability of a particular word based on the presence of other words in the preceding context which often occured near that word in a training corpus. If the trigger-based language model performs poorly relative to the static language model it may be because the preceding context is topically dissimilar to the current text.

Beeferman *et al.* measured performance segmenting a news feed containing concatenated *Wall Street Journal* articles to be 0.56 precision and 0.54 recall. These figures are significantly higher than those achieved by guessing randomly, placing a boundary at every possible point or locating no boundaries. They also proposed a probabilistic performance measure which we will discuss in Chapter 6 [Beeferman et al., 1997b].

4.2.2 Phillips

Phillips examined the relationship between word collocations and topic and described the features of a semiautomatic system [Phillips, 1985]. His system first preprocessed text to discard high and low frequency words, and then converted the remaining words to their root forms. Next, it counted collocations between a particular word and other words within a window that extended four words to the left and four to the right of that word. For example, from the start of the Gettysburg address *Four score and seven years ago our fathers brought ...*, Phillips would identify 8 bigrams involving the word *years*. Four of the bigrams would pair *years* with words to its left, and four with words to its right.

Phillips used the resulting collocational frequency statistics and cluster analysis to identify lexical networks in chapters of science textbooks. He showed that these clusters corresponded to the subtopic structure of the chapters identified by each book's author. He also proposed a method to identify global topic structure. First, he extracted nuclear words, those considered to be most central to a text, from the word clusters that he used to identify subtopic structure. Then, he compared sets of nuclear words from different chapters, and if they were sufficiently similar, noted relations between the chapters. Phillips suggested that these relations could be used for hypertext linking.

4.2.3 Youmans

Youmans described a text analysis technique with several applications in the study of literature [Youmans, 1990]. His technique is based on the observation that shifts in topic are likely to be accompanied by changes in word usage. In particular, when a new topic is introduced, words related to that topic will be used for the first time. To quantify this observation, he graphed the number of word types present in a document as a function of the number of word tokens. Figure 4.4 presents a sample of this type of graph. Obviously, at the beginning of a document, many tokens will be first instances of a type. As a result, the slope of the first portion of the type-token graph will be greater than the slope of later portions of the graph since after the start of the document most tokens will have been used at least once.

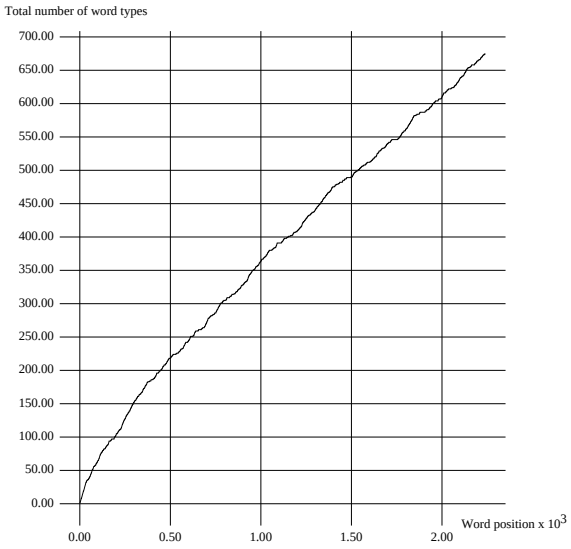

Figure 4.4: Type-token plot of Brown corpus file cd13.

In addition to suggesting the usefulness of this technique for tracking topic shift, Youmans proposed that the type-token curve would be useful for manually estimating the size of an author's vocabulary and that conclusions about authorship could be drawn from comparing the curves associated with different works. However, he did not present methods for automating any of these tasks.

Vocabulary Management Profiles

Youmans subsequently improved upon his type-token curve technique [Youmans, 1991]. One drawback of that technique was that topic boundaries were difficult to identify using the graphs because changes in topic resulted in the first use of only a few words. To address this, Youmans suggested counting the number of first uses within a fixed-size window of text, usually defined to be 35 words. He proposed that the number of first occurrences be graphed as a function of word position within the document. This type of graph, which Youmans called a Vocabulary Management Profile (VMP), represents an approximation of the derivative of the type-token curve. Figure 4.5 is the VMP of one document from the Brown corpus.

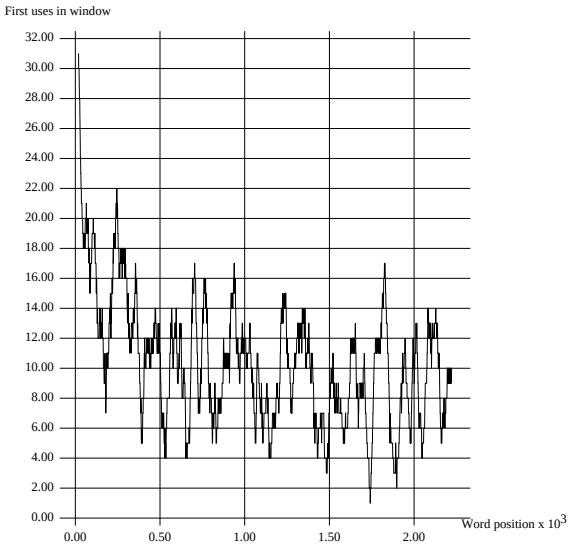

Figure 4.5: Vocabulary Management Profile of Brown corpus file cd13.

Youmans proposed that discourse boundaries could be identified by examining the VMP for sharp upturns after deep valleys. He defined a discourse boundary to be one common to the set of boundaries identified by the theories described in [Polanyi, 1988], [Chafe, 1974], [Longacre, 1983] and [Grimes, 1975]. His goal in identifying the boundaries common to these theories was "to place boundaries where trained readers of English literature are most likely to perceive them." Youmans did not describe any techniques to automatically identify discourse boundaries from VMPs. He did, however, perform several hand-analyses and concluded that the structures suggested by the VMP for a James Joyce novel and a George Orwell essay corresponded to the structures he perceived when reading.

4.2.4 Kozima

Kozima defined a measure of lexical cohesion, called the Lexical Cohesion Profile (LCP) [Kozima, 1993, Kozima and Furugori, 1994]. The LCP is computed using a similarity metric derived "by spreading activation on a semantic network which is systematically constructed from an English dictionary."[2] The LCP score for a particular word is the sum of

[2]See [Kozima and Furugori, 1993] for more details.

the semantic similarity scores which arise from comparing that word with each word in a window of preceding words. Kozima postulated that the local minima of these similarity scores would correspond to the positions of topic boundaries in text. He compared the boundaries identified using the LCP to the segmentation identified by 16 subjects who labeled a text whose paragraph boundaries had been removed. He did not quantitatively evaluate his algorithm's performance, but did state that the segmentation produced by the LCP resembled the one his human annotators generated.

4.2.5 Ponte & Croft

Ponte and Croft presented a topic segmentation technique [Ponte and Croft, 1997] which models the length of topic segments and uses Local Content Analysis (LCA), a query expansion technique used by IR systems [Xu and Croft, 1996]. Their goal was to identify the boundaries between short topic segments, such as those found in "What's News" articles from the *Wall Street Journal*, which contain brief summaries of news items discussed in greater detail elsewhere in that newspaper. Each summary is several sentences long and the average article contains two or three summaries.

They used a dynamic programming algorithm to identify the best partitioning of an article into segments each of which is about a single news item. They used LCA because short topic segments frequently contain few repeated words, which suggested to them that approaches based on word repetition would be of little use. LCA is generally used to identify related passages from a document database for IR. Ponte and Croft, however, used LCA to identify key concepts from the returned passages and used the concepts as surrogates for each of the sentences in a text. They then computed similarity scores for neighboring sentences by counting the number of concepts in common between the sets of concepts they identified for each sentence. They then employed these scores in their dynamic programming algorithm in conjunction with a model of length derived empirically from training data.

On first glance their results seem stellar: 0.89 recall and 0.83 precision on one test set. However, using only the words in the original sentences, which they claimed would be of little use because few occurred more than once, they achieved 0.70 recall and 0.63

precision. Moreover, their analysis of the contribution of LCA revealed that performance without length modeling is reduced greatly to 0.73 recall and 0.76 precision. This suggests that length modeling alone might account for more of a performance improvement than LCA. Also, the baseline performance on this task should be quite good. The test corpus on which they achieved these results contained only 228 sentences in 88 topic segments. Naively assuming a topic boundary between every sentence would achieve perfect recall and precision of 0.39.

4.3 Discussion

To permit comparisons between our algorithms and those from the literature, in the next chapter we will apply some of them to our evaluation data. Hearst showed that *TextTiling* outperformed Morris and Hirst's lexical chaining technique. Richmond, Smith and Amitay did not present evidence that their algorithm worked on more than a handful of texts. The same can be said of Yaari's hierarchical clustering method, which he tested on only a single document. Van der Eijk's multilingual technique was based on Hearst's and offered no new method of segmenting documents. The algorithms attributable to Nomoto and Nitta and Manabu and Takeo were both designed to structure Japanese text. None of Berber Sardinha's methods were automatic. As a result, we will compare the performance of our algorithms on English data to the segmentations produced by *TextTiling*, the best of the word repetition algorithms for English from the literature.

The model described by Beeferman *et al.* would be difficult to replicate, since it is based on their statistical modeling framework, uses their language model and was trained on a vast amount of data. Also, we will compare the performance of our algorithms to their performance because they tested on the TDT corpus, which we use to evaluate our algorithms in Chapter 6. Phillips' algorithm and Kozima's technique would also be difficult to replicate. Kozima's method requires the semantic network he used to compute the LCP scores, which is not publicly available. Also, Phillips' method was only semiautomatic, and he presented no quantitative results to suggest that it performed well. Ponte and Croft's algorithm addressed a different task than we do. Their task lies somewhere

between discourse segmentation and text segmentation in that the segment size is smaller than the one most text segmentation techniques identify but larger than those identified by discourse segmentation algorithms. This leaves only Youmans' technique from the collection of methods which structure text using features other than word repetition. So, we will compare our algorithms to our algorithmic implementation of Youmans' VMP method in the next chapter as well.

Chapter 5

Algorithms

A good text segmentation algorithm should possess certain attributes. We will discuss these in the next section, then describe four novel algorithms which do possess them. Table 5.1 shows the features used by each of these algorithms. Before describing the algorithms, we will explain the preprocessing we perform on documents and discuss the document segmentation simulation we use to compare the performance of our algorithms to two from the literature: Hearst's *TextTiling* algorithm[1] [Hearst, 1994b] and our automated implementation of Youman's Vocabulary Management Profile [Youmans, 1991]. We also present the results of novel algorithms based on the vector space model, which are similar to *TextTiling*.

5.1 Desiderata

An ideal text segmentation algorithm should have the following properties:

1. Fully automatic
2. Computationally efficient
3. Robust and applicable to a variety of document types

The algorithms we propose below satisfy the first criterion because they are completely computational. The only manual step required to use any of our algorithms was the

[1] *TextTiling* is available via ftp from elib.cs.berkeley.edu/src/texttiles.

Algorithm	Cue Words & Phrases	Pronoun Usage	First Uses	Word Co-occurrence	Word Repetition	Word Frequency	Word n-grams	Named Entities	Synonymy	Character n-grams	Semantic Similarity
Compression										⋆	
Optimization					⋆						
Word Frequency						⋆					
Max. Ent. Model	⋆	⋆	⋆			⋆	⋆	⋆	⋆		

Table 5.1: The features used by our text structuring algorithms.

identification of cue phrases we discussed in Section 3.2.1. We discuss a simple way to automate that step in the next chapter. Most of the algorithms described in Chapter 4 were automatic, but some required human intervention.

Efficiency is important because our primary goal is to build tools which can be used to address real NLP problems. An extremely accurate, but slow system, would be theoretically interesting but would not allow practical natural language processing to be improved. Real-time performance may not be necessary, but most of the NLP algorithms we discuss in the next chapter involve large, potentially fast-growing text collections.

Robustness is crucial so that text structuring algorithms can be applied to a wide variety of domains. The last two algorithms we present rely on word frequency statistics collected from a training corpus, and are therefore less robust than our first two, since they require no such statistics. The robustness of all of the algorithms we propose, as well as those in the literature, is diminished by their reliance on language-specific preprocessing.

5.2 Text Normalization

The four steps shown below constitute text normalization. We normalize documents by performing at least the first two steps prior to segmenting them. We discuss each step

in detail below since normalization is important for most natural language processing applications, but rarely receives much attention.

1. Tokenization
2. Conversion to lower case
3. Lemmatization (optional)
4. Removal of common words (optional)

5.2.1 Tokenization

Tokenization prevents word tokens and neighboring punctuation marks from jointly being misidentified as a single token. Consider Example 5.1. Tokenizing these sentences produces the text shown in Example 5.2. Without tokenization, NLP algorithms would miss the repetition of the word *stocks* since one instance is followed by a comma and the other is not. Depending on the source of the data, we will tokenize text either using a maximum entropy model trained on *Wall Street Journal* text or a simple set of rules designed to separate punctuation from words.

(5.1) Today, stocks closed higher on heavy trading. Many stocks, despite early losses, reached all time highs.

(5.2) Today , stocks closed higher on heavy trading . Many stocks , despite early losses , reached all time highs .

We built the maximum entropy model for tokenization using Ratnaparkhi's software [Ratnaparkhi, 1997b]. Our model determines whether a space should be inserted between neighboring characters in a document using features similar to those used by the named entity recognizer described earlier. The set of features we used for tokenization includes:

- The identity of the two characters preceding the spot where a space might be added.
- The identity of the preceding character.

- The identity of the two characters following that spot.
- The identity of the next character.
- Rudimentary class information about the preceding and following two characters.

We measured performance on a hand-corrected 100 file subset of the Penn *Wall Street Journal* Treebank to be 99.5 percent precision and 99.1 percent recall. This was considerably better than the hand-built tokenizer used for the University of Pennsylvania MUC-6 system, which scored 99.4 percent precision, but only 96.6 recall on the same data [Baldwin et al., 1995].

5.2.2 Conversion to Lowercase

The second step of text normalization is to convert all letters to lower case. This step eliminates problems caused by sentence initial capitalization. If we did not normalize text in this way, *Stock* and *stock* would be treated as different words in Example 5.3. Of course, this process alters useful capitalization as well. For example, the two occurrences of *international* in Example 5.4 differ in that the first is part of a proper noun and the second is not. This distinction is more difficult to observe when capitalization is eliminated.

(5.3) Stock prices edged higher today in light trading . Most stock markets will be closed tomorrow for the holiday .

(5.4) international business machines announced worldwide layoffs today , citing a need to reduce labor costs to better compete in the increasingly international personal computer market .

5.2.3 Lemmatization

The optional third component of normalization is replacing inflected forms by their roots. Though not perfect, lemmatization allows useful regularities in text to be identified. Example 5.5 shows the text from Example 5.1 after tokenization and morphology normalization using the XTAG morphology software [Karp et al., 1992].

a	aboard	about	above	across
after	against	ah	alas	albeit
all	along	alongside	although	among
amongst	an	and	another	any
anyone	anything	around	as	at

Table 5.2: Closed class words beginning with the letter *a* .

(5.5) Today , stock close high on heavy trade . Many stock , despite early loss , reach all time high .

5.2.4 Identifying Closed-Class Words

Finally, we sometimes flag frequent closed-class words found on a stop-list because doing so improves some text segmentation techniques. Function words generally contribute little information regarding topic segmentation, and ignoring them can both increase processing speed and improve performance. Table 5.2 shows the words beginning with the letter *a* from the list of closed-class words we used. The list we used was based on one available from the Summer Institute of Linguistics.[2]

5.3 Document Concatenations

We used an artificial task to refine and benchmark our text segmentation algorithms. The goal of the task was to identify the boundaries between pairs of concatenated *Wall Street Journal* articles drawn at random[3] from the Penn Treebank.

It is worthwhile to briefly consider our artificial task in light of the analogy between the noisy channel model and the process of obscuring topic boundaries presented in Chapter 1. Unlike the data on which actual topic segmentation will be performed, the document concatenations have not passed through the noisy channel. Thus, our simulation approximates the task of identifying boundaries which have not yet been obscured to provide the

[2]Available from ftp://www.sil.org.

[3]One restriction was placed on these articles. They were not permitted to be "NewsBytes," which are brief articles from the first page of the *Wall Street Journal* that summarize sets of articles that appear later in the paper.

Baseline Method	# within x words					
	0	20	40	60	80	100
Middle Sentence	5	7	14	19	25	36
Middle Paragraph	13	14	26	27	33	41
Random Sentence	4.6	7	14.2	19.4	24	31
Random Paragraph	8.4	10.2	16.2	22.2	25	31.2

Table 5.3: Accuracy of several baseline segmentation algorithms on 100 pairs of concatenated *WSJ* articles. The last two rows are average results for five iterations of the random selection algorithm.

reader or hearer with the appropriate level of continuity between topics. As such it should be easier than actual topic segmentation, but still serves as a useful benchmark.

This task is simpler than topic segmentation in part because the number of segment boundaries to identify is known in advance. Also, the concatenated documents may address entirely different subjects. Nonetheless, the boundaries between some pairs of concatenated articles will be difficult to identify because the domain of the *Wall Street Journal* is constrained. The task is also challenging because article length varies widely. A short article may be only 25 words long while feature stories may be several thousand words, or more, in length. Finally, some words in the *Wall Street Journal*, such as *million* and *company*, occur in most articles independent of their topic. Repeat occurrences of these words may mislead algorithms which use word repetition.

Admittedly, the task is artificial, but evaluating algorithms with it has several advantages. First, there is a definitive correct answer for each concatenation. Also, no human annotators need to be trained to locate boundaries and there is no need to identify a consensus segmentation from the pool of annotations they produce. Since the Treebank contains many articles, we can easily create new evaluation or training corpora. Another advantage is that we can measure baseline performance in several ways and compare the performance of more sophisticated algorithms to these baseline scores. One simple baseline is to randomly select a single boundary. Another trivial algorithm selects the middle sentence or paragraph of each concatenation. Table 5.3 presents the performance of these two techniques on our test corpus, which consists of 100 pairs of *Wall Street Journal* articles.

The 100 concatenations of *Wall Street Journal* articles we used for testing contained an average of 20.4 paragraphs, 43.3 sentences and 1064 words. We evaluated algorithms by counting the number of exact matches between guesses and article boundaries and the number of guesses lying within 20, 40, 60, 80 and 100 words of each boundary. Counting only exact matches would be overly harsh because it is desirable for an algorithm to score better if it places boundaries close to their actual location. When evaluating most algorithms, we will present results achieved with and without the optional normalization steps described above: lemmatizing words and removing words found on a stop list.

In theory a topic boundary could occur in the middle of a sentence. In this simulation, however, we know that the boundaries between documents occur between paragraphs. Both paragraph and sentence boundaries are labeled in the Penn Treebank. As a result, we will measure the performance of algorithms when guessed boundaries are constrained to occur between sentences and, more restrictively, only between paragraphs.

Restricting guesses to paragraph boundaries may boost performance, since there are fewer possible boundary sites to choose from. Most texts, however, are not annotated with paragraph boundaries and, if they are not initially marked, it is difficult to recover their bounds automatically. In some domains, even sentence boundaries are unlikely to be labeled, but systems exist to accurately disambiguate potential sentence boundaries using statistical techniques. (For example, [Reynar and Ratnaparkhi, 1997].)

5.4 Performance of Algorithms from the Literature

In order to permit comparisons between our algorithms and those from the literature, we tested Hearst's *TextTiling* algorithm and our implementation of Youmans' *VMP* technique on the 100 concatenations of *Wall Street Journal* articles. We present the performance of these techniques below. We will also describe and evaluate several novel text segmentation algorithms based on the vector space model which are similar to *TextTiling*.

Unit of Analysis (Sent|Para)	Normalization (Y|N)	# within x words					
		0	20	40	60	80	100
Sent	N	3	3	10	15	19	21
Para	N	7	**9**	12	14	21	23
Sent	Y	3	3	10	15	19	21
Para	Y	**8**	**9**	**17**	**21**	**25**	**30**

Table 5.4: *TextTiling* applied to document concatenations. *TextTiling* did not label any boundaries for 11 of the 100 concatenations because they were too short.

5.4.1 *TextTiling*

We tested *TextTiling* on the 100 document concatenations by permitting it one guess as to the location of the boundary between the two documents in each concatenation. Table 5.4 shows its performance with and without the optional normalization steps described above. The first column indicates whether boundaries were constrained to lie between sentences or paragraphs. The remaining columns indicate how many boundaries were within the specified number of words of the actual boundary location. The 0 column measures the number of exact matches. We show the best entry in each column in bold in this and subsequent performance tables. As expected, performance improved when guesses were restricted to lie between paragraphs. Morphological normalization boosted performance as well.

We reimplemented *TextTiling* prior to discovering that a version was freely available. In addition, we implemented other segmentation algorithms based on the vector space which are similar to Hearst's *TextTiling* technique. Implementing these algorithms allowed us to explore the validity of the decisions Hearst made regarding the design of *TextTiling* for these data. *TextTiling* computed the similarity between fixed-length blocks of pseudo-sentences rather than paragraphs to eliminate length variations which can cause problems for the vector space model. It also smoothed and placed boundaries at gaps with large depth scores [Hearst, 1994a].

The vector space model text segmentation techniques we implemented can be parameterized as described below. The parenthesized abbreviations are used to more concisely indicate parameter settings.

- Was smoothing performed? (Yes) or (No)
- Were sentences (Sent), paragraphs (Para) or pseudo-sentences used as the unit of analysis? If pseudo-sentences were used, were boundaries restricted to lie between sentences (PS) or paragraphs (PP)?
- Were boundaries identified using the relative depths of valleys as Hearst recommended (Rel) or by identifying the point between the least similar adjacent blocks (Min)?

When testing each of these variants, if the text was too short to allow for a block size of six pseudo-sentences, as Hearst recommended, we decreased the block size until a boundary could be identified. Table 5.5 shows the performance of these variations on the 100 document concatenations. Results with optional normalization are shown in Table 5.6. The first three columns of these two tables indicate the parameterization tested.

There are 16 possible variants which correspond to all combinations of the 2, 4 and 2 possible settings for each of the parameters. However, when the setting Min was used, we did not smooth, because the location of the minimum score is unlikely to be affected by smoothing. This hypothesis was borne out in preliminary experiments. This eliminated 4 of the 16 possible parameterizations, leaving the 12 found in the table. Our reimplementation of the parameterization Hearst used in *TextTiling* can be found in each table in the row with parameter settings Y, PP and Rel. Our reimplementation performed better than the publicly available version of *TextTiling* due to the adjustment of block size which allowed boundaries to be identified in short concatenations.

We can draw several conclusions from the performance of the publicly available *TextTiling* algorithm and the segmentation techniques based on the vector space model that we implemented. Our enhancement that decreased block size until a boundary could be identified was crucial for our test data, since the concatenations of articles were sometimes shorter than the articles Hearst used to evaluate *TextTiling*. In fact, the publicly available

Smoothing (Y|N)	Unit of Analysis (PS|PP|Sent|Para)	Algorithm (Rel|Min)	# within x words					
			0	20	40	60	80	100
N	PS	Min	22	34	46	53	57	61
N	PP	Min	**29**	**38**	**47**	52	55	61
N	Sent	Min	26	34	44	51	58	54
N	Para	Min	21	25	46	**54**	**61**	**68**
N	PS	Rel	18	27	32	41	51	55
N	PP	Rel	24	30	34	40	49	55
N	Sent	Rel	17	23	31	37	43	48
N	Para	Rel	18	23	40	48	56	62
Y	PS	Rel	10	19	38	47	53	57
Y	PP	Rel	20	23	38	44	50	58
Y	Sent	Rel	7	13	28	36	44	54
Y	Para	Rel	16	18	40	46	52	58

Table 5.5: Results of our implementation of algorithms based on the vector space model when applied to documents without optional normalization. Each algorithm was tested on 100 concatenations of pairs of *Wall Street Journal* articles. The row in gray presents results with the same settings as *TextTiling*.

Smoothing (Y|N)	Unit of Analysis (PS|PP|Sent|Para)	Algorithm (Rel|Min)	# within x words					
			0	20	40	60	80	100
N	PS	Min	28	**48**	**66**	**77**	**86**	**87**
N	PP	Min	29	45	58	62	67	67
N	Sent	Min	**31**	**48**	49	74	76	79
N	Para	Min	26	47	61	64	69	77
N	PS	Rel	27	45	60	70	75	77
N	PP	Rel	24	34	45	56	60	63
N	Sent	Rel	26	45	50	55	59	61
N	Para	Rel	27	46	57	60	65	71
Y	PS	Rel	13	31	58	72	78	81
Y	PP	Rel	20	34	48	59	61	62
Y	Sent	Rel	8	22	46	53	65	69
Y	Para	Rel	14	23	41	48	58	65

Table 5.6: Results of our implementation of algorithms based on the vector space model when applied to normalized data. Each algorithm was tested on 100 concatenations of pairs of *Wall Street Journal* articles. The row in gray presents results with the same settings as *TextTiling*.

Unit of Analysis (Sent|Para)	Normalization (Y|N)	# within x words					
		0	20	40	60	80	100
Sent	N	9	11	15	17	25	29
Para	N	**14**	17	21	23	31	33
Sent	Y	11	14	22	31	41	46
Para	Y	**14**	**19**	**25**	**34**	**44**	**48**

Table 5.7: Results of several variants of Youmans' technique when applied to data consisting of 100 concatenations of pairs of *Wall Street Journal* articles.

version of *TextTiling* failed to guess a boundary location for 11 of the 100 *Wall Street Journal* articles because they were too short. Contrary to what Hearst suggested, we found smoothing to be detrimental to performance. We found that using pseudo-sentences, as Hearst proposed, improved performance on the concatenations. On our test corpus, algorithms which used depth-scores, as *TextTiling* did, fared worse than those which used the simpler technique of locating boundaries where neighboring blocks were least similar. In fact, this simpler technique identified the maximum number of exact matches found by any text segmentation algorithm based on the vector space model.

5.4.2 Vocabulary Management Profiles

We also implemented and tested Youmans' VMP technique. We used a window of 35 words as Youmans suggested. Our automation of the VMP technique placed a boundary immediately preceding the point in the concatenation where the most new words were introduced. We evaluated the VMP technique with boundaries constrained to lie at sentence boundaries and paragraph boundaries. Table 5.7 presents the results of evaluations with and without optional normalization.

Youmans' algorithm performed better when restricted to placing boundaries between paragraphs. Also, it performed slightly better when applied to normalized data. It was less accurate than the best of the algorithms based on the vector space model.

5.5 Compression Algorithm

The first of our algorithms relies only on repetitions of character n-grams to identify topic boundaries. It is based on an elegant approach to compressing data that capitalizes on the inherent self-similarities within text. The crucial assumption underlying this compression algorithm is that similarity in terms of distributions of characters within topics is greater than across topics. We can exploit this assumption to identify topic boundaries by observing compression performance, which is measured by the ratio of the size of the original text to the size of the compressed text. We identify boundaries by locating the point where the compression ratio is minimized, since at that point the text being compressed is least like the preceding text. The location of maximum dissimilarity should be at a topic boundary.

5.5.1 Lempel-Ziv 1977

We performed text compression using a method developed by Ziv and Lempel commonly called LZ77 [Ziv and Lempel, 1977]. LZ77 is widely used and is implemented in GZIP and other popular compression packages. An important contribution of Ziv and Lempel's work was showing that self-similarity could be exploited to compress data. Many earlier compression methods used dictionaries or relied on assumptions about the frequency of individual characters, similar to the way frequency information was used to determine how to encode letters using dots and dashes in Morse code.

The LZ77 algorithm incrementally compresses text. The location of the portion of text about to be compressed is marked with a pointer and the text following the pointer is stored in the *lookahead buffer*. The portion of text immediately before the pointer is retained in a *fixed-size window*, whose size is usually an integer power of 2. The window of already-compressed text is used to perform additional compression by identifying the maximal string match between text immediately following the pointer and text in the window. When the algorithm begins, the window will be empty because no text will have yet been compressed.

Compression proceeds as follows:

1. Set the pointers s and c to precede the first character of the text.
2. Set ω to be the character following s. Move the pointer c one character right.
3. If ω does not match a string in the fixed-sized window go to step 6.
4. If c follows the last character of the document go to step 7.
5. Concatenate the character following c in the lookahead buffer to ω. Move c one character to the right. Go to step 3.
6. Remove the final character from ω if ω is longer than 1 character. Move c one character to the left.
7. If the length of ω is 1, encode ω as a literal. Otherwise encode it as a pair containing the distance from the pointer to the location in the window at which a string identical to ω begins and the length of ω.
8. Quit if c follows the last character of the document.
9. Move s right by the length of ω and go to step 2.

An example will elucidate this process. Assume the text to be compressed is the string `ababa` and the window size is 2 characters. Compression will proceed as shown in Table 5.8. Either the literal column or both the pointer and length columns will be empty for each row in the table, since on each iteration the algorithm encodes text using either a literal or a pair consisting of a pointer and a length.

The sample text `ababa` is encoded as the sequence a, b, $< 2, 2 >$, $< 2, 1 >$. In this short example, the output of the compression algorithm is longer than the original text. Generally, however, LZ77 reduces the length of texts by about 50 percent. Performance usually improves with text length.

The LZ77 algorithm has been modified in many ways to suit various kinds of data. It has also been optimized to reduce compression time and modified to perform additional

Text in window	Lookahead buffer	Literal	Pointer	Length
	ababa	a		
a	baba	b		
ab	aba		2	2
ab	a		2	1
ba				

Table 5.8: Sample LZ77 compression.

compression on portions of the resulting encoding.[4] To perform topic segmentation, we implemented a variant of LZ77 similar to the one found in GZIP. This variant differs from the original LZ77 algorithm in that literals and lengths are compressed using one codebook and pointers are compressed with a second codebook. We used a window size of 4096 characters.

5.5.2 Complicating Factors

One difficulty with using character sequences as an indicator of topic segmentation is that spurious substring repetitions are more likely to occur than accidental word repetitions. For instance, strings associated with inflection, such as the suffix *ing*, are likely to repeat both within and across topic segments. This makes morphology normalization crucial. For example, without normalization the word *making* could be compressed if the previous context contained the word *kayaking*. After reducing both words to their root forms, *make* and *kayak*, the overlap between the strings, and therefore the degree of compression, is reduced.

Coincidental substring repetitions are still likely to occur because some letter sequences are quite frequent. For instance, the string *th* occurs more than $100,000$ times in a 4.5 million character portion of the Penn Treebank *Wall Street Journal* corpus. We assume that non-topic-based repetitions will be similarly distributed within and between topic segments and as a result will not significantly hamper identifying segment boundaries.

[4] See [Bell et al., 1990] for descriptions of many of the variations.

Unit of Analysis (Sent|Para)	Normalization (Y|N)	# within x words					
		0	20	40	60	80	100
Sent	N	8	13	15	19	25	30
Para	N	13	18	32	35	41	45
Sent	Y	6	12	20	30	39	48
Para	Y	**14**	**32**	**42**	**49**	**51**	**57**

Table 5.9: Results of the compression algorithm on 100 concatenations of pairs of *Wall Street Journal* articles.

5.5.3 Evaluation

Table 5.9 shows the performance of the compression algorithm. Although simple and elegant, this algorithm performs poorly. In fact, it does only slightly better than the baseline algorithm which guesses the boundary for each concatenation between the middle paragraphs. As a result, we will not test this algorithm on any of the corpora described in the next chapter. We concluded from the poor performance of this algorithm that character sequences are not as useful an indicator of text segmentation as words. If segmentation was to be performed on data without word boundaries, then character sequences might be useful. Otherwise, more informative features should be used.

5.6 Optimization Algorithm

Our second text structuring method is based on lexical cohesion [Halliday and Hasan, 1976] and segments text using an optimization algorithm applied to patterns of word repetition. It was initially motivated by a technique called *dotplotting* [Helfman, 1994].

5.6.1 Dotplotting

A dotplot is a visual aid for viewing data from a matrix. Dotplots can display large quantities of similarity information and permit this information to be analyzed visually. To display data from a binary-valued matrix, for example, a point would be placed at coordinate (x, y) on a dotplot whenever the value in cell (x, y) of the matrix is 1.

	to	**be**	**or**	**not**	**to**	**be**
to	1	0	0	0	1	0
be	0	1	0	0	0	1
or	0	0	1	0	0	0
not	0	0	0	1	0	0
to	1	0	0	0	1	0
be	0	1	0	0	0	1

Table 5.10: Sample word repetition matrix.

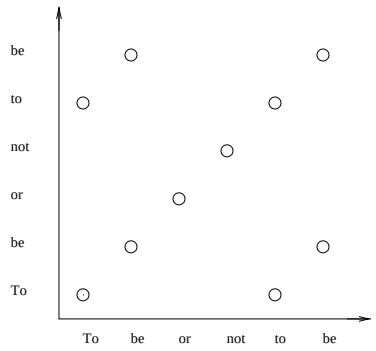

Figure 5.1: Dotplot of the matrix from Table 5.10 which shows word repetitions in the phrase *to be or not to be*.

Helfman used dotplotting for text analysis and as an aid to software engineering. By analyzing repetitions of character n-grams, he detected similar sections within documents and document collections. He also identified modules that contained similar code fragments in a database of computer source code. Church later used dotplotting to align parallel translations in pairs of languages with many cognates [Church, 1993].

If we number the words of a document sequentially beginning with 1, then we can build a matrix in which each cell (x, y) contains a 1 if the words numbered x and y are identical and a 0 otherwise. We can then build a dotplot from this matrix. For example, the matrix shown in Table 5.10 represents word repetitions in the phrase *to be or not to be*. We created the dotplot shown in Figure 5.1 from this matrix. Note that for clarity we labeled the axes of the graph with words rather than word numbers.

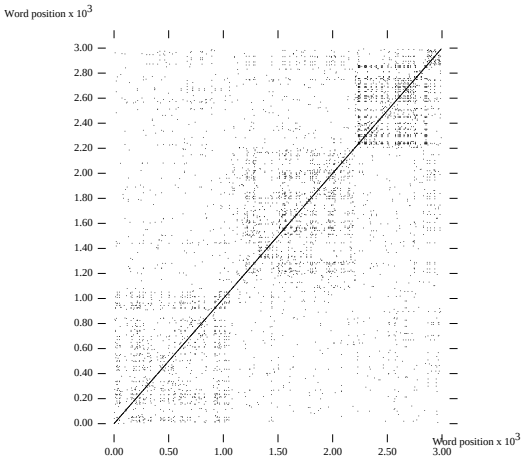

Figure 5.2: The dotplot of four concatenated *Wall Street Journal* articles.

5.6.2 Dotplots for Text Segmentation

To segment text, we generate a matrix in which cells (x, y) are set to 1 when word number x and word number y are the same or have the same root. Thus, if the same word appears at word positions x and y in a text, and $x \neq y$, we set the values in four cells to 1, namely (x, x), (x, y), (y, x) and (y, y). In this type of matrix, cells (x, y) where $x = y$ will have the value 1 because words are identical to themselves. In some of the evaluations we describe below, this condition does not hold because we preprocessed documents so that function words were ignored.

Figure 5.2 shows the dotplot of a matrix built in this way. We constructed the word repetition matrix from four concatenated *Wall Street Journal* articles. The boundaries between documents are located immediately following the words numbered 1085, 2206 and 2863. Due to the number of repeated words within documents, each individual document appears as a dark square region on the dotplot. The boundaries between at least the first 3 documents should be visually apparent on the figure.

5.6.3 Algorithmic Boundary Identification

The fact that boundaries can be identified visually suggests that they could be identified algorithmically by processing the dotplot or the word repetition matrix. The extent of

each topic segment is apparent on the dotplot because segments correspond to regions along the line $x = y$ that are darker than other regions. The darkness arises in regions of high density, where density is a measure of the number of points present per unit area and is computed simply by dividing the number of points in a region by the area of that region. For example, if a region 4 words by 4 words on a dotplot contained 2 points, then its density would be $\frac{2}{4 \cdot 4} = 0.125$.

We propose two related algorithms for identifying topic segments by measuring density. The first technique identifies boundaries which maximize the density within segments that lie along the diagonal of the dotplot. The second method locates boundaries which minimize the density of regions off of the diagonal where $x = y$. Intuitively, the first algorithm finds topic segments by maximizing self-similarity and the second identifies them by minimizing the similarity between different segments.

The algorithm that identifies boundaries by minimizing density proceeds as follows:

1. Posit a boundary at a particular location
2. Compute the overall density of the regions off of the main diagonal with this boundary and any previously identified boundaries in place
3. Record the density and the location of the hypothesized boundary
4. Repeat steps 1 through 3 for all putative boundaries
5. Select the boundary which results in the lowest overall density

These steps can be repeated to find more boundaries if necessary. The maximization algorithm follows the same steps but computes the density of regions on the main diagonal and concludes by selecting the boundary which corresponds to the maximum density score.

A graphical example will make the steps of the minimization algorithm clearer. In Figure 5.3, the algorithm has posited a boundary, as in step 1 above. This boundary divides the dotplot, which is shown without any points plotted for simplicity, into 4 regions. Regions 1 and 3 are potential topic segments because they lie on the main diagonal. Regions 2 and 4 do not lie on the main diagonal and are not, therefore, potential topic segments. In step 2, the algorithm would count the number of points in regions 2 and 4, which are

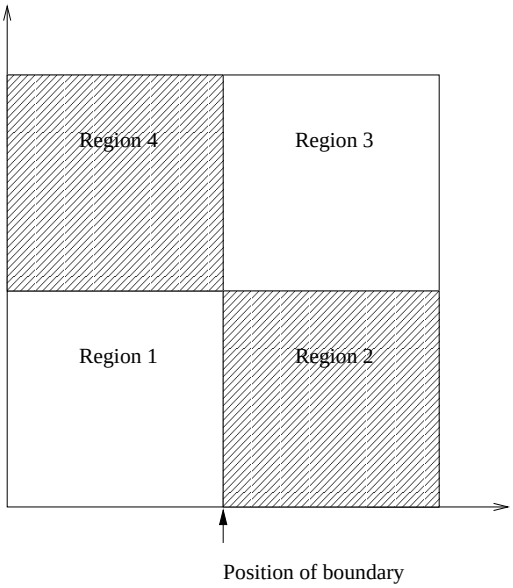

Figure 5.3: Graphical illustration of the working of the optimization algorithm.

shaded on the figure, and divide that number by the combined area of those two regions to yield a density score. This score and the location of the boundary would be recorded in step 3. The process would then repeat for the remaining potential boundaries. Finally, the location that gave rise to the minimum density score would be selected as the best site for a topic boundary.

Figure 5.4 shows the situation after one boundary has been identified and the search for a second boundary is under way. In step 2, the algorithm would sum the number of points plotted in regions 2, 3, 4, 6, 7 and 8, which are shaded on the figure, and compute the density of these areas by dividing the sum by the combined area of these regions. In step 3, the density and the location of the putative boundary would be recorded.

5.6.4 Minimization versus Maximization

The minimization and maximization algorithms are similar, but yield different results, as can be seen from a simple example. Consider this 5 word text: $x\ x\ y\ x\ y$. The algorithm which identifies the best boundary by maximizing self-similarity posits a boundary after the

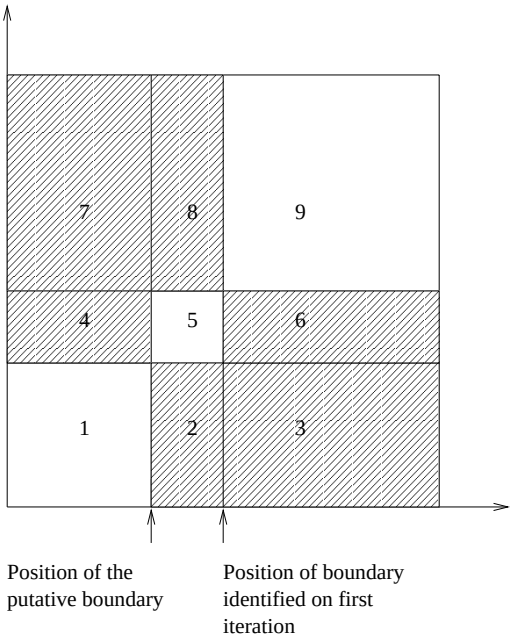

Figure 5.4: Graphical illustration of the working of the optimization algorithm after one boundary has been identified.

second x, while the algorithm which places boundaries by minimizing similarity between regions predicts a boundary after the final x. Table 5.11 shows the density scores for placing boundaries in each possible position using both algorithms. The symbol | represents the location of the hypothesized boundary. The density score associated with the best boundary according to each algorithm is shown in bold.

Putative segmentation	**Minimization score**	**Maximization score**
x | x y x y	$\frac{4}{8} = 0.50$	$\frac{9}{17} = 0.53$
x x | y x y	$\frac{4}{12} = 0.33$	$\frac{9}{13} = \mathbf{0.69}$
x x y | x y	$\frac{6}{12} = 0.50$	$\frac{7}{13} = 0.54$
x x y x | y	$\frac{2}{8} = \mathbf{0.25}$	$\frac{11}{17} = 0.65$

Table 5.11: The application of two optimization algorithms for topic segmentation to the sample text *x x y x y*.

5.6.5 Formal Description of the Algorithm

In order to formally specify the algorithm which minimizes the outside density, we will first define some variables and then describe how to compute the values of these variables. The same variables would also be used to describe the algorithm which maximizes self-similarity. However, we will not give a specification for that algorithm because it it is only trivially different from the minimization algorithm.

We define the first element of a vector to have index 1.

Let D be the document to be segmented. Assume D is n words long.

Let T be a vector containing the word tokens of D. Let $T[1]$ be the first word of D, $T[2]$ be the second word of D and so on concluding with $T[n]$ being the final word of D.

Let B be a vector of indices of T corresponding to the location of topic boundaries. B is sorted in ascending order. Initially, B contains only the implicit boundary present at the start of each document, so $B[1] = 0$.

Let A be a vector containing potential boundaries. Each element of A is an index of T. A contains only the locations of sentence or paragraph boundaries.

Let $V_{x,y}$ be a vector containing the number of word tokens of each word type in $T[x]$, $T[x+1]$, ... $T[y]$. Different V vectors are created as needed to compute the values of M, which is defined below.

Let P be a two-dimensional array and let $P[i]$ be the i^{th} row of this array. $P[i][j]$, therefore, is the j^{th} element of the vector $P[i]$. Let $P[i]$ be the vector B with $A[i]$ inserted and then sorted in ascending order. P has dimensionality $|A|$ by $|B|+1$. One of the rows of P will become the vector B in the next iteration of the algorithm.

Let M be a vector. The number of elements in M is the same as the number of elements of A, which in turn is the same as the number of rows in P. Then, for $1 \leq i \leq |A|$, let

$$M[i] = \sum_{j=2}^{|P[i]|} \frac{V_{P[i][j-1],P[i][j]} \cdot V_{P[i][j],n}}{(P[i][j] - P[i][j-1])(n - P[i][j])} \tag{5.6}$$

Let k be $\texttt{argmin}(M)$. k is the minimum density achieved by any of the putative boundaries.

Let l be the index of M such that $M[l] = k$. l is the index in M of the minimum density. l is also the position in A of the boundary that gives rise to the minimum density.

On each iteration, the algorithm finds the best boundary by determining the value of l. After identifying the value of l, the algorithm updates the vector of boundaries B to contain the elements of $P[l]$. Element $A[l]$ is then removed from the vector A. The algorithm can be rerun in this manner until the desired number of boundaries have been located.

The first two variables described above, D and T, remain the same throughout the running of the algorithm. The vector B grows in size as new boundaries are added, while the vectors A and M decrease in size by one element on each iteration of the algorithm. The dimensionality of P changes in accordance with the changes to the size of B and A.

The numerator of the equation used to compute the values of M is the dot product between word vectors associated with two regions of the document D. The denominator is the product of the number of words in each of these regions in D, and is the upper bound for the numerator. The maximum value of the numerator only occurs when each section contains only tokens of a single type. The values of M are densities because the numerator in the formula for M is the number of points within a particular region and the denominator is the area of that region on the dotplot.

Figure 5.5 depicts the density of the regions off of the main diagonal when a boundary is placed at each location on the x-axis. These data are derived from the dotplot shown in Figure 5.2. Only one boundary would be identified using the data on this graph—the boundary at position 1085, which gives rise to the lowest density. On the next iteration, the graph would be updated to reflect the presence of this boundary.

5.6.6 Similarity to Vector Space Model

The dot product in the formula for M, Formula 5.6, reveals the similarity between our optimization algorithm and *TextTiling* [Hearst, 1994b]. The crucial difference between the two lies in the global nature of our approach. Hearst's algorithm identifies boundaries by comparing neighboring regions only, while our minimization technique compares each region to all other regions. Our maximization algorithm is more similar to *TextTiling* since it is essentially local.

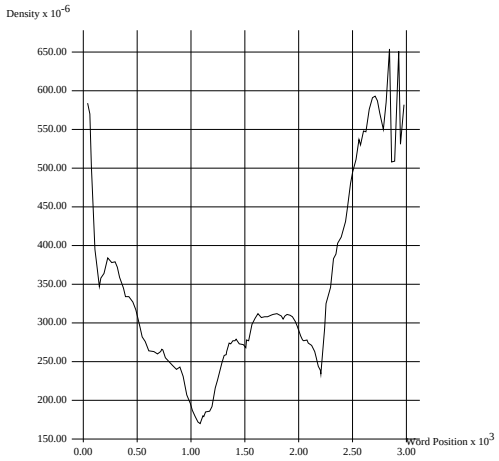

Figure 5.5: The first outside density plot of four concatenated *Wall Street Journal* articles.

5.6.7 Evaluation

We tested several variants of our optimization technique on the corpus of concatenated *Wall Street Journal* articles as well. They can be parameterized as follows:

- Was the density within regions along the main diagonal maximized (Max) or was the density of points outside of regions along the diagonal minimized (Min)?
- Were boundaries constrained to lie between sentences (Sent) or paragraphs (Para)?
- Was the density computation performed using sentences (S) or words (W) to measure area?

The first two parameters are self-explanatory, but the third requires description. Sentence length varies greatly in the *Wall Street Journal*. Hearst addressed this problem in *TextTiling* by dividing documents into equal length pseudo-sentences. We handle length variation by changing the units of the denominator of the density formula from words to sentences. This forces all sentences to be treated equally, regardless of the number of words they contain.

An example will clarify the difference between these two settings. Assume one region of text contains 2 sentences and a total of 20 words and that a second region contains 3

Technique (Min|Max)	Unit of Analysis (Sent|Para)	Area (W|S)	# within x words					
			0	20	40	60	80	100
Max	Sent	W	26	32	**43**	49	55	58
Max	Para	W	**34**	**37**	42	49	55	61
Max	Sent	S	17	24	32	39	49	53
Max	Para	S	28	34	**43**	**51**	**60**	**66**
Min	Sent	W	12	13	20	24	27	32
Min	Para	W	18	20	28	31	37	41
Min	Sent	S	0	0	0	1	5	14
Min	Para	S	1	2	2	3	9	15

Table 5.12: Results of many variants of our optimization algorithm when tested on 100 concatenations of pairs of *Wall Street Journal* articles. We did not reduce words to their roots or ignore frequent words.

sentences and 25 words. Using the W method, we would compute the density of points in this region by calculating the dot product of the word vectors for each region and dividing that value by 500—the product of 20 words and 25 words. With the S method, we would divide the dot product by 6: the product of 3 sentences and 2 sentences.

Table 5.12 presents the results of all of the versions of our optimization algorithm on the task of identifying a single boundary in the corpus of 100 concatenations of pairs of *Wall Street Journal* articles.

As was the case with Hearst's and Youmans' techniques, it was beneficial to restrict boundaries to lie between paragraphs. Accounting for variations in sentence length hurt performance, as the third, fourth, seventh and eighth lines of Table 5.12 show. Also, the maximization algorithm performed better than the minimization algorithm. The maximization technique outperformed Youmans' VMP and Hearst's technique as measured by the number of exactly correct boundaries identified and the number within 100 words of the correct location.

Optional Normalization

There are two ways our optimization algorithms can deal with words that are meant to be ignored because they are on the stop-list. The first, and most intuitive, is to simply disallow

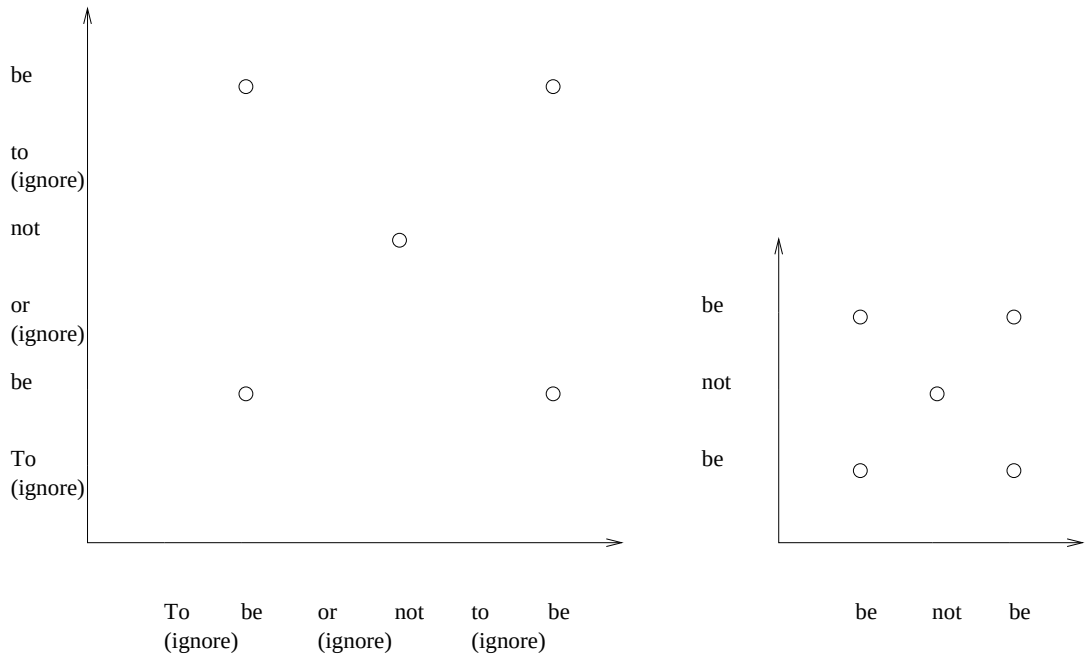

Figure 5.6: Two ways to handle ignored words. On the left dotplot, ignored words may not participate in matches. On the dotplot on the right, ignored words have been eliminated.

matches between ignored words. In that case, the number of words in the document is not affected by ignoring closed-class words. The second possibility is to remove ignored words, thus reducing the number of words in a document. Sample dotplots using each method are shown in Figure 5.6. In preliminary experiments, removing ignored words entirely always performed best. As a result, we only present results using that method here.

Comparing the scores in Table 5.12 and Table 5.13 reveals that the performance of all of the parameterizations of the dotplot technique improved when words were lemmatized and words found on a stop-list were ignored. We found that constraining boundaries to lie on paragraph boundaries improved performance with the optional normalizations performed. Contrary to the results without optional normalization, our minimization algorithm consistently outperformed the maximization algorithm with identical parameter settings. The minimization algorithm identified 86 exact matches and 100 within 100 words of the correct location—more by far than *TextTiling*, the VMP algorithm or our compression technique.

Technique (Min|Max)	Unit of Analysis (Sent|Para)	Area (W|S)	# within x words					
			0	20	40	60	80	100
Max	Sent	W	35	41	52	60	64	71
Max	Para	W	47	49	55	63	68	74
Max	Sent	S	31	41	51	60	65	71
Max	Para	S	39	51	63	70	78	82
Min	Sent	W	75	88	93	95	96	97
Min	Para	W	**86**	**95**	**98**	**100**	**100**	**100**
Min	Sent	S	66	77	82	85	87	87
Min	Para	S	69	76	81	83	84	86

Table 5.13: Results of variants of our optimization technique when tested on 100 concatenations of pairs of *Wall Street Journal* articles. The data was preprocessed to reduce words to their roots and ignore frequent words.

These results suggest several things. First, if possible, boundaries should be placed between paragraphs. Second, normalization is crucial. Third, computing density using words as the unit of area is more accurate. Finally, if we are able to normalize text, then it is best to use the minimization algorithm. Otherwise, the maximization algorithm is more appropriate.

5.7 Word Frequency Algorithm

The goal of language modeling algorithms is to estimate the probability of a sequence of words. *Burstiness* is one phenomenon that a good language model should take into account. Words are considered to be bursty when one appearance in a document is a good indicator that additional occurrences are likely. Put another way, a word that is bursty will more often occur additional times in a document than is implied simply by its overall frequency in a collection of documents [Church and Gale, 1995b]. For example, a contentful word, such as *boycott*, is likely to appear again in a document that contains it one time, presumably because it is crucial to the document's topic. A similarly frequent, but less topic-based word, such as *somewhat* exhibits less burstiness, because documents are unlikely to be about *somewhat* in the way that they can be about a particular *boycott*

or *boycotts* in general. In terms of language modeling, seeing one instance of a bursty word should boost the probability of seeing another instance of that word.

We can determine whether a topic boundary appears between neighboring blocks of text in a document using a language model. We refer to the block of text prior to the putative boundary as block 1 and the text following the boundary as block 2. We can use a language model which accounts for burstiness to determine whether the two blocks are about the same topic or whether there is likely to be a boundary separating them. Using the language model we compute the probability of seeing the words in block 2 as a continuation of block 1—that is, we compute the probability of block 2 conditioned on block 1. We also compute the probability of seeing the words in block 2 in a new segment that is independent of block 1. We can perform both these computations using one language model. The only difference in how the language model is used is whether or not the preceding context plays a role.

If the probability of generating the words in block 2 is sufficiently greater when conditioning on the words in block 1 than without conditioning on those words, then the two blocks of text are probably about the same topic and the putative boundary is unlikely. Otherwise, the two blocks are most likely about different topics and the proposed boundary is a good one. The probability conditioned on the first block should be greater when the two blocks are about the same topic because it is generally more likely that additional instances of bursty words will occur than that the bursty words will occur by chance in block 2.

A brief, qualitative example should clarify this idea. Suppose a document, which is known to be about two topics, has its most contentful, and therefore bursty, words distributed as shown in Table 5.14. Scanning down the list of paragraphs and words present in each paragraph, one might conclude that the most likely location for a boundary between the topics was between paragraphs 3 and 4 because the vocabulary after paragraph 3 is considerably different than the vocabulary of the first 3 paragraphs. We hope that an algorithm using a language model would also determine the best location for a boundary to lie between paragraphs 3 and 4.

Paragraph	Words
1	a,b
2	a,b,c
3	a,c
4	a,d,e
5	d,e

Table 5.14: Example distribution of bursty words in a document.

5.7.1 The G Model

Church and Gale used a document collection to compare the number of documents containing particular words to the expected number of documents those words would appear in if their frequency were approximated by a Poisson [Church and Gale, 1995a]. They concluded that it is useful to independently measure document frequency because the Poisson poorly predicts it. The Poisson is a single parameter model and is therefore unable to capture dependent relationships between word frequency and hidden variables, such as topic. This may account for the discrepancy between the predicted and observed number of documents containing particular words. Church and Gale also showed that both the negative binomial and Katz's K-Mixture—both two parameter models—better predict document frequency than the Poisson. One advantage of the K-mixture model over the negative binomial is that its parameters are easier to estimate.

The probability of seeing k instances of a word w in a document under the K-Mixture model is: $Pr_K(k,w) = (1-\alpha)\delta_{k,0} + \frac{\alpha}{\beta+1}(\frac{\beta}{\beta+1})^k$. α and β should be subscripted with w to indicate their pertinence to a particular word, but for the sake of readability we omit the subscripts. $\delta_{k,0}$ has the value 1 when k equals 0 and is 0 otherwise.

Katz proposed the G model as an improvement upon the K-Mixture model that Church and Gale described [Katz, 1996]. He designed the G model to predict the number of occurrences of content words and phrases in documents and demonstrated that it predicted the number of occurrences of 2 and 3 word phrases in a document collection more accurately than either the negative binomial or the K-Mixture model.

The G model does not compute the probability of seeing words in a particular order, but instead predicts the probability of seeing a particular bag of words. As a result, context does not impact the probability of generating a particular word as it does in most language models. Trigram language models assume language to be a second order Markov process. This simplification permits the probability of a word to be conditioned solely on the two preceding words: $P(w_i|w_{i-1}, w_{i-2})$. The G model makes a much different, even stronger assumption, namely that the probability of generating particular words is completely independent of the surrounding words.

This model also assumes that word probability does not depend on document length. Katz defends this by stating that: "The number of instances of any specific content word in a particular document does not explicitly depend on the document length, it is rather a function of how much this document is about the concept expressed or named by that word. A short document may contain several instances of some content word that names a concept essential for this document, while a much longer document, having an occasional reference to that concept, will have only a single instance of the same word. Only *on average*, longer documents containing a particular content word do usually have more instances of that word than shorter documents." [Katz, 1996, p. 18]

Katz divided words into two categories based on the number of times they appeared in a particular document. He labeled a word that occurred once *nontopical* and assumed that if the word were central to the topic being discussed it would occur again. Words which occured more than once he called *topical* words. It is possible that truly nontopical words may occur more than once in a document and that unequivocally topical words may occur only one time, but the G model does not account for these phenomena.

This view of word repetition led Katz to propose a model with three parameters per word. Each parameter has an intuitive explanation. α represents the probability that a word appears in a document at least one time. γ is the probability that a word appears more than once, if it appears at all. γ measures the degree of topicality of a word. B is the number of times on average a word appears in a document, if it appears more than once. These parameters can be estimated easily from a corpus. Katz estimated them from a collection of technical documents which varied widely in length.

$P_w(k)$ is the probability under the G model of seeing k instances of word w in a document. The computation of $P_w(k)$ can be divided into three cases. The first corresponds to seeing 0 instances of word w. The second represents the probability of seeing exactly 1 instance and the third is the probability of seeing k occurrences, for values of k greater than 1.

$$P_w(0) = 1 - \alpha \tag{5.7}$$

$$P_w(1) = \alpha(1 - \gamma) \tag{5.8}$$

$$P_w(k), k \geq 2 = \frac{\alpha\gamma}{B-1}(1 - \frac{1}{B-1})^{k-2} \tag{5.9}$$

5.7.2 Word Frequency Algorithm Specification

We will write the probability of seeing k instances of occurrences of word w independent of the preceding context as simply $P_w(k)$ and the probability using the context will be written $P_w(k|C)$. The probability of generating all of the words in block 2 is the product of the probabilities of generating the appropriate number of occurrences of each word type. We will call the probability of generating the words in the context of the previous text $P_{\texttt{one}}$ because we treat the words in both blocks as if they are in one segment with regard to the language model. The probability of generating the words in block 2 independent of block 1 is called $P_{\texttt{two}}$, since the blocks are in two different topic segments. Formulae for $P_{\texttt{one}}$ and $P_{\texttt{two}}$ are shown below.

$$P_{\texttt{one}} = \prod_w P_w(k|C) \tag{5.10}$$

$$P_{\texttt{two}} = \prod_w P_w(k) \tag{5.11}$$

In the experiments we present below, we compute $P_{\texttt{one}}$ and $P_{\texttt{two}}$ using blocks of text 230 words long. When examining potential boundaries near the beginning or end of a document, there may not be sufficient text for a block to contain 230 words. In that event,

we reduce the size of both blocks. We chose the block size to be the length of the average topic segment annotators identified in our primary evaluation corpus, the HUB-4 corpus.

We compare $P_{\texttt{one}}$ and $P_{\texttt{two}}$ to determine whether a topic boundary is likely. Since word probabilities are small, we perform word probability calculations in the log domain and compare $P_{\texttt{one}}$ and $P_{\texttt{two}}$ using differences between log probabilities rather than ratios of probabilities. We normalize the log probabilities by dividing by the number of words in each block. This is identical to taking the $n^{\texttt{th}}$ root of the product of probabilities outside the log domain and yields an average probability ratio per word. This facilitates the use of thresholds to determine whether a boundary is present which can be used for any block size.

We compute both $P_w(k|C)$ and $P_w(k)$ using the formulae for Katz's G model described above. Computing $P_w(k)$ requires no knowledge of the number of words of each type in block 1. To compute it, we count the number of word tokens of type w in block 2, which we call k_2, and then compute $P_w(k_2)$ using the formulae for the G model. We then compute $P_{\texttt{two}}$ by taking the product of the probabilities of each word type.

Computing $P_w(k|C)$ is more difficult and requires that we define some conditional probabilities. We divide these conditional probabilities into six cases which correspond to combinations of the number of appearances of a particular word w in block 1 and the number of appearances of w in block 2. Table 5.15 shows the conditional probabilities for all six cases. The symbol + following a number in the table means *that many occurrences or more*. In practice, we do not need to use the first conditional probability listed in the table, which is for the case of 0 occurrences in block 1 and 0 occurrences in block 2. We can ignore this conditional probability because we need only compute probabilities of words which appear in one or both blocks.

The first three rows of the table are simply the probabilities of seeing the number of occurrences in block 2 under the G model. Having seen 0 occurrences in block 1 does not provide any information about how many are likely to be observed in block 2, so the conditional probability is simply the probability under the original model.

Occurrences in block 1	Occurrences in block 2	Conditional probability
0	0	$1-\alpha$
0	1	$\alpha(1-\gamma)$
0	2+	$\frac{\alpha\cdot\gamma}{B-1}(1-\frac{1}{B-1})^{k-2}$
1	0	$1-\gamma$
1	1+	$\frac{\gamma}{B-1}(1-\frac{1}{B-1})^{k-2}$
2+	0+	$\frac{1}{M(B-1)}(1-\frac{1}{B-1})^{k-2}$

Table 5.15: Conditional probabilities of seeing a particular number of occurrences of a word in block 2 given that a certain number have been observed in block 1. k is the number of occurrences in blocks 1 and 2 combined. M is a normalization constant discussed in the text.

Conditional probabilities, when summed over all possible outcomes, naturally must total 1. The three cases given 0 occurrences in the first block do sum to 1. Proving this will also demonstrate the soundness of the probability model as a whole:

$$1-\alpha+\alpha(1-\gamma)+\sum_{k=2}^{\infty}\frac{\alpha\gamma}{B-1}(1-\frac{1}{B-1})^{k-2}=$$

$$1-\alpha+\alpha-\alpha\gamma+\frac{\alpha\gamma}{B-1}\sum_{k=2}^{\infty}(1-\frac{1}{B-1})^{k-2}=$$

Changing the variable of the summation:

$$1-\alpha\gamma+\frac{\alpha\gamma}{B-1}\sum_{j=0}^{\infty}(1-\frac{1}{B-1})^{j}=$$

Using the identity $\sum_{j=0}^{\infty}q^j=\frac{1}{1-q}$:

$$1-\alpha\gamma+\frac{\alpha\gamma}{B-1}\frac{1}{1-(1-\frac{1}{B-1})}=$$

$$1-\alpha\gamma+\frac{\alpha\gamma}{B-1}(B-1)=$$

$$1-\alpha\gamma+\alpha\gamma=1$$

If we have observed 1 instance of a word with no knowledge of the total number, there are two possibilities: either there no additional instances or there are additional occurrences. The conditional probability of 0 additional occurrences, or a total of 1 occurrence, is $1-\gamma$. This can be explained in two ways. First, by definition, γ measures the probability of seeing a word again once it has been seen one time. Therefore $1-\gamma$ is the probability of not seeing the word again. The second explanation is this: having seen one occurrence, it is impossible to see zero occurrences total. This means that only the terms regarding exactly one occurrence and at least two occurrences from the formulae for the G model are relevant. These terms sum to α but as conditional probabilities they must sum to 1. Dividing each term by α normalizes the probabilities so they do sum to 1. The term for exactly one occurrence in the G model, $\alpha(1-\gamma)$, thus becomes $1-\gamma$.

We can use the trick of dividing by α to determine the conditional probability of additional occurrences as well. The probability of seeing $k \geq 2$ occurrences is: $\frac{\alpha\gamma}{B-1}\sum_{k=2}^{\infty}(1-\frac{1}{B-1})^{k-2}$. Dividing by α yields $\frac{\gamma}{B-1}\sum_{k=2}^{\infty}(1-\frac{1}{B-1})^{k-2}$. This is the conditional probability of seeing 2 or more instances. The conditional probability of seeing exactly k instances where $k \geq 2$ is represented by a single term from the summation: $\frac{\gamma}{B-1}(1-\frac{1}{B-1})^{k-2}$.

When we have observed 2 or more occurrences of a word, the conditional probability of 0 or more additional occurrences depends on the number already observed somewhat differently than in the previous cases. Only the final term of the formula for the G model is relevant to the computation of conditional probabilities in this case. Assuming that exactly 2 repetitions occurred in block 1, we can normalize the probability of encountering 2 through ∞ total occurrences—which accounts for aggregate probability equal to $\alpha\gamma$—by dividing by $\alpha\gamma$ to yield the formula in Table 5.15. Additional normalization is required if we observed more than 2 occurrences in the first block, since the probability of $k > 2$ occurrences accounts for less than $\alpha\gamma$ of the original probability mass. Therefore, we must account for the probability of numbers of repetitions less than the number in block 1 as well. In this case, we tally this probability, which we call M, and compute conditional probabilities by dividing $\frac{\alpha\gamma}{B-1}(1-\frac{1}{B-1})^{k-2}$ by $\alpha\gamma M$.

To locate topic boundaries using this word frequency statistic we compute the ratio $\frac{P_{\texttt{one}}}{P_{\texttt{two}}}$, or, in the log domain, the difference $\log(P_{\texttt{one}}) - \log(P_{\texttt{two}})$. We then compare this difference to a threshold. A natural threshold in the log domain is 0, the difference between the log probabilities when $P_{\texttt{one}}$ and $P_{\texttt{two}}$ are equally likely. If the difference is greater than 0, we will not posit a boundary, since it is more likely that the word sequence was generated in the context of the preceding segment. Otherwise, we will propose a boundary, since it is more probable that the word sequence was generated independent of the preceding text. We can identify more reliable boundaries by decreasing the threshold, thereby improving precision at the expense of recall. Similarly, raising the threshold will improve recall and reduce precision.

5.7.3 The Impact of Individual Words

It is instructive to consider the contributions individual words make to the ratio $\frac{P_{\texttt{one}}}{P_{\texttt{two}}}$. We write the probability of a single word as either $P_{\texttt{one},w}$ or $P_{\texttt{two},w}$. We only need to consider three of the six cases, since in the cases where 0 occurrences of the word occurred in block 1, $P_{\texttt{one},w}$ is equal to $P_{\texttt{two},w}$.

The remaining three cases are: 1 occurrence of a word in block 1 and 0 in block 2, 1 occurrence of a word in block 1 and 1 or more in block 2 and 2 or more occurrences in block 1 and 0 or more in block 2.

We use the following variables in all three cases:

k_1 is the number of occurrences of word w in block 1.

k_2 is the number of occurrences of word w in block 2.

$k_{\texttt{both}}$ is the number of occurrences of word w in blocks 1 and 2. $k_{\texttt{both}} = k_1 + k_2$.

Case 1: there is a single occurrence of a particular word type in block 1 and 0 occurrences of that word type in block 2. From Table 5.15,

$$P_{\texttt{one},w} = 1 - \gamma$$

The probability of independently generating a region containing zero instances of word w is simply

$$P_{\texttt{two},w} = 1 - \alpha$$

The ratio of probabilities is:

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{1-\gamma}{1-\alpha}$$

Case 2: 1 occurrence of word w in block 1 and 1 or more occurrence in block 2. Again, from Table 5.15,

$$P_{\texttt{one},w} = \frac{\gamma}{B-1}(1-\frac{1}{B-1})^{k_{\texttt{both}}-2}$$

There are two subcases that correspond to the number of occurrences in block 2. The probability of seeing exactly 1 instance of w, case 2a, in an independent segment is:

$$P_{\texttt{two},w} = \alpha(1-\gamma)$$

The probability of seeing 2 or more instances of w, case 2b, is:

$$P_{\texttt{two},w} = \frac{\alpha\gamma}{B-1}(1-\frac{1}{B-1})^{k_2-2}$$

As a result, there are two probability ratios. Case 2a, when $k_2 = 1$:

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{\frac{\gamma}{B-1}(1-\frac{1}{B-1})^{k_{\texttt{both}}-2}}{\alpha(1-\gamma)} =$$

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{\gamma}{\alpha(1-\gamma)(B-1)}(1-\frac{1}{B-1})^{k_{\texttt{both}-2}}$$

Case 2b, when $k_2 \geq 2$:

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{\frac{\gamma}{B-1}(1-\frac{1}{B-1})^{k_{\texttt{both}}-2}}{\frac{\alpha\gamma}{B-1}(1-\frac{1}{B-1})^{k_2-2}}$$

Since $k_{\texttt{both}} = k_1 + k_2$,

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{1}{\alpha}(1-\frac{1}{B-1})^{k_1}$$

Case 3: there are 2 or more instances of word w in block 1. The probability of seeing 0 or more occurrences in block 2 as a continuation of block 1 is:

$$P_{\texttt{one},w} = \frac{1}{M(B-1)}(1 - \frac{1}{B-1})^{k_{\texttt{both}}-2}$$

where

$$M = \sum_{j=2}^{k_1-1}(1 - \frac{1}{B-1})^j$$

There are 3 sub-cases. They pertain to 0 instances of w in block 2, 1 instance in block 2 and 2 or more instances in block 2. The probability of case 3a, 0 instances in block 2 arising independently, is:

$$P_{\texttt{two},w} = 1 - \alpha$$

The probability that 1 instance occurred, case 3b, is:

$$P_{\texttt{two},w} = \alpha(1 - \gamma)$$

Finally, the probability of 2 or more occurrences, case 3c, is:

$$P_{\texttt{two},w} = \frac{\alpha\gamma}{B-1}(1 - \frac{1}{B-1})^{k_2-2}$$

The probability ratios for these sub-cases are below. Case 3a:

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{\frac{1}{M(B-1)}(1 - \frac{1}{B-1})^{k_{\texttt{both}}-2}}{1-\alpha} =$$

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{1}{M(B-1)(1-\alpha)}(1 - \frac{1}{B-1})^{k_{\texttt{both}}-2}$$

Case 3b:

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{\frac{1}{M(B-1)}(1 - \frac{1}{B-1})^{k_{\texttt{both}}-2}}{\alpha(1-\gamma)} =$$

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{1}{M\alpha(1-\gamma)(B-1)}(1 - \frac{1}{B-1})^{k_{\texttt{both}}-2}$$

Case 3c:

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{\frac{1}{M(B-1)}(1 - \frac{1}{B-1})^{k_{\texttt{both}}-2}}{\frac{\alpha\gamma}{B-1}(1 - \frac{1}{B-1})^{k_2-2}} =$$

$$\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}} = \frac{1}{M\alpha\gamma}(1 - \frac{1}{B-1})^{k_1}$$

Case #	# in Block 1	# in Block 2	Parameter		
			α	γ	B
1	1	0	+	-	NA
2a	1	1	-	+	-
2b	1	2+	-	NA	+
3a	2+	0	+	NA	-
3b	2+	1	-	+	-
3c	2+	2	-	-	+

Table 5.16: The effect of increasing parameter values on the ratio of $\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}}$.

5.7.4 The Effect of Perturbing the Parameters

Table 5.16 shows the effect on the ratio $\frac{P_{\texttt{one},w}}{P_{\texttt{two},w}}$ of altering one parameter associated with a particular word, w, while holding the others constant. A + in the table indicates that the likelihood of a topic boundary based solely on the number of occurrences of w increases when the parameter is increased and - means that a topic boundary is less likely. The table shows that as the probability of occurrence, α, increases, the likelihood of a boundary decreases assuming w appears in block 2 as well as block 1. This is the case because the more likely a particular word is, the greater the chance that it will appear independently in block 2. In cases 1 and 3a, which correspond to 0 instances in block 2, increasing α increases the likelihood of a boundary, since if w were likely to occur but did not, that would provide weak evidence that the two blocks were in the same segment.

Increasing γ, the probability that a word will be used topically, has no effect in two cases. In case 2a and 3b increasing it improves the chance that blocks 1 and 2 are in the same topic segment since additional occurrences of w are likely to be continuations of the same topic. In cases 1 and 3c, increasing γ increases the likelihood of a topic boundary because it becomes more likely that multiple occurrences of a particular would occur in a single segment, thus making it relatively more likely that a new topic segment with 0 instances of the word occurred than that the previous segment continued with no additional uses of the word.

Altering the value of B has no impact on case 1. As a word becomes burstier, it becomes more likely that multiple occurrences of it are within the same segment. This explains why increasing B decreases the likelihood of a boundary in cases 2b and 3c. In the remaining cases, few instances of the word in block 2 become less likely as B is increased, hence the probability of a boundary increases.

5.7.5 Parameter Estimation

We estimated the parameters for the word frequency model from a corpus of approximately 78 million words of *Wall Street Journal* text. The average document contained 461.4 words.

As in most statistical natural language processing algorithms, unknown words—words which are not in the training corpus, but which occur in test data—pose a problem for the measure of word frequency we used. To account for unknown words, we applied simplified Good-Turing (GT) smoothing to the number of documents in which each word appeared [Good, 1953, Gale and Sampson, 1995].[5] GT smoothing redistributes probability mass among items based on the number of times they appear in a sample. Words in the training corpus have their probability discounted by GT smoothing so that some probability mass is reserved for unseen words. Following Turing's proposal for accounting for unseen events [Good, 1953], we assumed that the number of unseen words was identical to the number of observed word types and distributed the probability mass equally among them.

GT smoothing partially solves the problem of handling unseen events, but since the G model has three parameters, additional smoothing is necessary. GT smoothing addressed the difficulties caused by 0 values of the α parameter, but 0 and 1 values of the parameter γ are also problematic. When the value of γ estimated from the training corpus is 0, the value of B is not computable. We estimated the values of γ and B by counting the number of times each word type appeared 1, 2 and 3 or more times in a document. We smoothed these counts by averaging in the average counts for 10 randomly selected content words in order to eliminate problems estimating γ and B. We also used these average counts to determine the parameter values for unknown words. Table 5.17 shows the ranges of the parameters estimated from the training corpus after smoothing.

[5] Thanks to Dan Melamed for use of his implementation of Good-Turing smoothing.

Parameter	Minimum	Maximum
α	$8.0\text{x}10^{-9}$	0.999
γ	$1.4\text{x}10^{-4}$	0.999
B	2.0001	26.4

Table 5.17: Range of parameters for the G model estimated from the *Wall Street Journal* training corpus with Good-Turing smoothing applied.

Unit of Analysis (Sent|Para)	# within x words					
	0	20	40	60	80	100
Sent	77	**86**	**90**	**90**	**92**	**93**
Para	**79**	82	85	87	90	90

Table 5.18: Results of the word frequency algorithm when given 1 guess about the location of a boundary. The data consisted of 100 concatenations of pairs of *Wall Street Journal* articles.

5.7.6 Evaluation

We tested our word frequency algorithm on the corpus of 100 randomly selected concatenations of pairs of *Wall Street Journal* articles. Table 5.18 presents the algorithm's performance on text normalized by ignoring words found on a stop-list and reducing words to their roots. We did not evaluate the technique without normalization because the G model was intended to predict the frequency of open-class words only.

While initially experimenting with this model on other data, we observed that it frequently erred by guessing too early or late in each concatenation. To address this problem, we did not allow boundaries to be placed too close to the beginning or end of the concatenations. When hypothesized boundaries were restricted to lie between sentences, the algorithm was not permitted to propose a boundary until after the third sentence, nor could one be placed before the two final sentences. Similarly, when guesses were forced to lie between paragraphs, they could not be placed before the third paragraph or before the two final paragraphs. This restriction prevented guesses from occurring too early or late but occasionally caused legitimate boundaries to be discarded.

Unlike both the compression algorithm and the optimization algorithm we presented, this algorithm requires training data. As a result, this technique should better model topic

boundary detection when tested on data from the same source. In the evaluation presented above, the test data was from the same source as the training data but was not from the same time period. The difference in topics discussed and changes in writing style may be responsible for the surprising fact that this algorithm performed slightly poorer than our optimization algorithm.

5.7.7 No Training Corpus Required

One drawback of the word frequency algorithm is its dependence on training data. However, if the method is relatively insensitive to the vicissitudes of training corpora, then this drawback is only a minor one. To test the sensitivity of the model to the quantity and quality of training text, we could retrain it using different quantities of data or using corpora containing text from several sources. Rather than testing the model's robustness in this way, we chose to answer a more radical question: What happens if we use effectively no training data?

In order to compute the probability of seeing a word a particular number of times, the G model uses the values of three parameters for that word. To estimate sensible values for these parameters required smoothing. The smoothing method we employed—Good-Turing smoothing—allowed us to estimate the α parameters for unknown words. Additional, ad hoc smoothing permitted us to estimate values for γ and B. In this final smoothing step, we averaged the parameters associated with ten randomly selected content words.

To test performance with effectively very little training data, we discarded the probabilities estimated for the words found in our training corpus and instead relied solely on the parameters estimated for unknown words. To be clear, this does not mean that we ignored the identities of individual words when testing, but that the parameters used by the word frequency model for each word type were identical. That is, the parameters used for each word type were those estimated for unknown words primarily through smoothing. Table 5.19 presents the results of our evaluation of this version of the word frequency algorithm on the 100 concatenations of *Wall Street Journal* articles.

The algorithm hobbled by using only parameters for unknown words performed surprisingly well. It scored 14 percent fewer exact matches than the word frequency algorithm.

Unit of Analysis (Sent|Para)	# within x words					
	0	20	40	60	80	100
Sent	**73**	**77**	**84**	**86**	**89**	**90**
Para	68	72	75	78	82	83

Table 5.19: Results of the word frequency algorithm when only the parameters for unknown words were used. The data consisted of 100 concatenations of pairs of *Wall Street Journal* articles.

Although this algorithm, even with training data, performs slightly poorer than the optimization algorithm presented above, it has a number of advantages. First, when there is training data, it can take advantage of it. This could be useful for segmenting documents from other domains where word repetition alone is less of an indicator of structure. Second, it is not iterative like the optimization algorithm, but can identify any number of boundaries in a single pass. Third, it is completely local and therefore less costly to compute than the minimization version of our optimization algorithm. Due to these advantages, we will use the output of this algorithm in the statistical model we present in the next section which incorporates some of the other topic segmentation clues presented in Chapter 3.

5.8 A Maximum Entropy Model

Word frequency is a good indicator of topic shift. In Chapter 3, we described a number of other indicators as well, including some which have not previously been used. We incorporated a number of these features into a statistical model built with Ratnaparkhi's maximum entropy modeling tools [Ratnaparkhi, 1997b]. This model predicts the probability that a topic boundary is present at a particular location in a document using features about the surrounding context. It uses these features:

- Did the Word Frequency Algorithm suggest a topic boundary?
- Were any domain cues in each of the categories from Table 3.2 present?
- How many word bigrams occurred in both the region before and the region after the putative topic boundary?

she	her	hers
herself	he	him
his	himself	they
their	them	theirs
themselves		

Table 5.20: Pronouns used as indicators of topic boundaries.

- How many Named Entities were common to the regions before and after the putative topic boundary?
- How many words in the two regions were synonyms according to WordNet?
- What percentage of words in the region following the putative boundary were used for the first time?
- Were any of the pronouns from the list shown in Table 5.20 present in the first 5 words following the potential topic boundary?
- How many words were in the previous segment?

We trained the model on a 30 file subset of the HUB-4 1996 Broadcast News Corpus, which we will use in the next chapter for evaluation. These files contained a total of 534 topic segments which had been annotated by the Linguistic Data Consortium. The training subset did not contain any files relevant to the queries which constituted the Spoken Document Retrieval task. We chose the files, which were a subset of the 36 files from which we manually identified cue phrases, by randomly selecting 2 or 3 documents from 9 of the 11 broadcast sources. We did not train on *Nightline* or *Washington Journal* stories because, while identifying cue phrases, we observed that these two sources were structured differently than the others in the HUB-4 corpus. *Nightline* documents contained few topic shifts and *Washington Journal* documents shifted topic frequently.

We did not test this model on the concatenations of *Wall Street Journal* articles since some of the features used were unlikely to be useful for identifying topic boundaries in this corpus. We will leave the evaluation of this model to the next chapter.

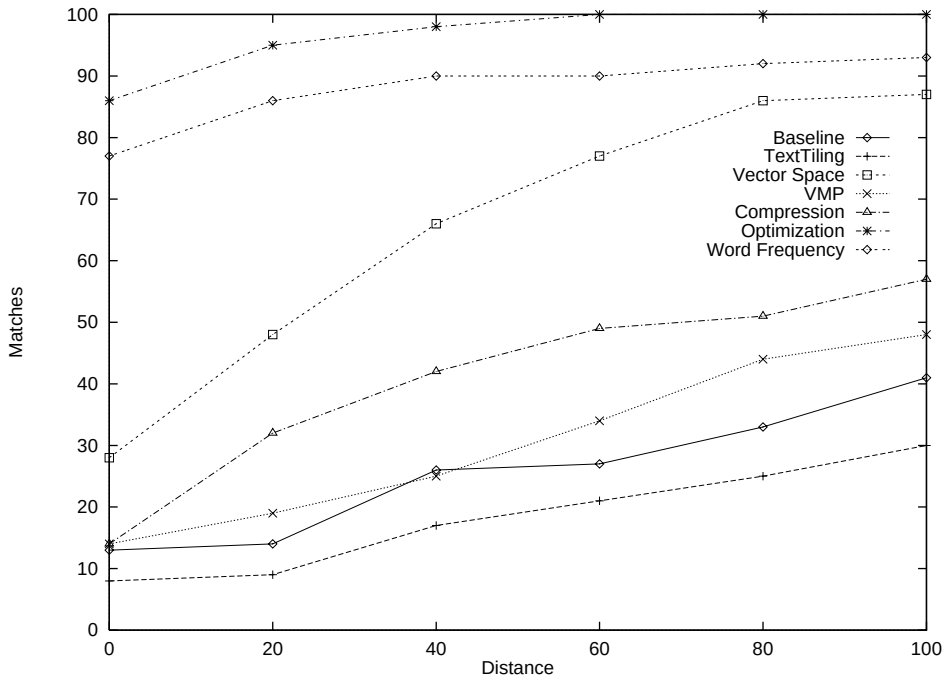

Figure 5.7: The performance of the best performing version of each algorithm. A perfect score would be 100 exact matches—at distance 0 on the graph.

5.9 Performance Reprise

To conclude the chapter, we present a graph showing the best performing version of each algorithm and the baseline performance on the document concatenation task in Figure 5.7. Note that our optimization algorithm and our word frequency model perform much better than a baseline algorithm, our compression technique, the VMP, *TextTiling* and the best of our vector space models.

Chapter 6

Evaluation

The evaluation described in the last chapter was somewhat unrealistic because we knew the number of topic segment boundaries in advance, which meant that we could program our algorithms to select a set number of boundaries. As a result, we measured performance by computing accuracy: the number of guessed boundaries which matched actual boundaries. Evaluating text structuring systems using realistic data is more difficult. The first challenge is selecting appropriate performance measures. When the number of boundaries is not known in advance, algorithms must determine how many to posit. Consequently, accuracy is an uninformative measure because perfect accuracy can be achieved by proposing a boundary at every potential location. Another challenge is choosing appropriate corpora to use for evaluation. Two solutions to this problem present themselves. Either available annotated corpora can be used, in which case the reliability of the annotation should first be determined, or a new corpus can be annotated. We discuss both evaluation metrics and our decisions regarding corpora below. Following these sections, we present evaluations of our algorithms using various corpora and several evaluation metrics.

6.1 Performance Measures

A number of researchers have used precision and recall to evaluate text and discourse segmentation algorithms [Hearst, 1994b, Passoneau and Litman, 1993], but these metrics have several drawbacks. First, they are overly strict: a hypothesized boundary close

to an actual segment boundary is equally detrimental to performance as one far from a boundary. To address this, Passoneau and Litman proposed allowing fuzzy boundaries when annotators agreed there was a boundary in a few utterance range but disagreed about its exact location [Passoneau and Litman, 1996]. We suggested counting boundaries correct that appeared within a fixed-size window of words of an actual boundary, as we did in the previous chapter [Reynar, 1994]. Neither approach solves the problem adequately. Both approaches produce several performance figures and are overly generous in that they weight inexact matches the same as exact ones.

Precision and recall can frequently be exchanged for one another. That is, algorithms can be tuned to increase recall, in exchange for a loss in precision, and vice versa. This means that full knowledge of the performance of an algorithm requires a precision-recall graph or a measure which combines precision and recall. Various combination methods have been proposed in the IR literature, including the precision-recall product and the F-measure.

Beeferman, Berger and Lafferty proposed a new performance measure which solves both of these problems. Their metric weights exact matches more than near misses and yields a single score [Beeferman et al., 1997b]. They suggested measuring performance by determining the probability that two randomly selected sentences, or other units of text, were located similarly in an algorithm's segmentation and the reference segmentation. Similarly located means that if the two sentences are in the same topic segment in the gold-standard segmentation then they are in the same segment in the hypothesized segmentation as well. Alternatively, if they are in different segments in the answer key they must be in different segments in the hypothesized segmentation. Beeferman *et al.* proposed using the formula below to compute this probability for a corpus containing n sentences.

$$P_\mu(\texttt{ref}, \texttt{hyp}) = \sum_{1 \le i \le j \le n} D_\mu(i,j)\, \delta_{\texttt{ref}}(i,j) \overline{\oplus}\, \delta_{\texttt{hyp}}(i,j)$$

where

$$D_\mu = \gamma_\mu e^{-\mu|i-j|}$$

D_μ is an exponential distribution with mean $1/\mu$. Beeferman *et al.* determined μ based on the average topic segment length in their collection. γ_μ is a normalization constant

which normalizes D_μ to be a probability distribution. D_μ effectively weights comparisons between closer sentences more highly in the computation of P_μ. $\delta_{\texttt{hyp}}(i,j)$ and $\delta_{\texttt{ref}}(i,j)$ are binary valued functions which are 1 when sentences i and j are in the same topic segment in the appropriate corpus. The symbol $\overline{\oplus}$ represents the XNOR function, which is 1 when its arguments are equal, and 0 otherwise.

Beeferman *et al.* observed that computing this metric requires some knowledge of the collection since the value of μ must be chosen. The presence of a constant dependent on the collection in the formula for P_μ means that performance figures for different collections may not be comparable. For instance, it is not safe to say that an algorithm which scores 0.90 on one collection is better than one which scores 0.85 on a different collection.

A second disadvantage of the metric exists because it was designed to evaluate segmentation algorithms on single files containing hundreds of concatenated stories each. It is straightforward to compute aggregate precision and recall when segmenting collections of documents in different files, but it is not obvious how to combine the scores from the probabilistic metric. The files could be concatenated prior to computing the score, but this would distort the evaluation because algorithms would be given credit for "guessing" the boundaries between the files when they were actually known in advance. Nonetheless, we believe this method of evaluation is more useful than any proposed to date when used on appropriate corpora.

6.2 Comparison to human annotation

It is both difficult and time-consuming to annotate corpora with topic boundaries. Detailed instructions, like those developed by Nakatani *et al.* [Nakatani et al., 1995], must be written for annotators if they are expected to label consistently. And, even with specific instructions, annotators are unlikely to unanimously agree about the location of topic boundaries in samples of text or speech [Passoneau and Litman, 1993, Hirschberg and Grosz, 1992].

After a collection of documents has been annotated, the problem remains of determining whether the annotations are consistent enough to be useful. Carletta proposed using

ABC Nightline	ABC World News Now	ABC World News Tonight
CNN Early Edition	CNN Early Prime News	CNN Headline News
CNN Primetime News	CNN The World Today	C-SPAN Washington Journal
NPR All Things Considered	NPR/PRI Marketplace	

Table 6.1: News programs found in the HUB-4 Broadcast News Corpus.

the kappa statistic, which is widely used in the field of content analysis, to measure interannotator agreement [Carletta, 1996]. The kappa statistic, K, is defined as $K = \frac{P(A)-P(E)}{1-P(E)}$, where $P(A)$ is the fraction of times that annotators agree and $P(E)$ is the fraction they are expected to agree by chance. In content analysis, a kappa score greater than 0.80 indicates high reliability and a score between 0.67 and 0.80 indicates tentative reliability. Scores below 0.67 mean that the judgments are inconsistent.

Below we will present the performance of algorithms described in the previous chapter on a corpus annotated with topic boundaries. Prior to testing our algorithms on that corpus, we measured the reliability of the annotation using the kappa statistic.

6.2.1 Broadcast News

The 1996 HUB-4 Broadcast News Corpus, which was used for the TREC Spoken Document Retrieval task, is composed of radio and television news broadcasts from eleven sources. These sources encompass a number of different formats and focus on various aspects of news coverage. CNN's *Headline News* contains short summaries of current news items without much in-depth analysis. National Public Radio's show *Marketplace* reports almost exclusively financial news and contains both brief and detailed stories. ABC's *Nightline* covers only a few issues in an hour long program. These sources exemplify the amount of variation in average story length and subject matter present in the corpus. Table 6.1 contains a list of all the programs in the corpus.

The LDC recorded the audio portion of each news broadcast and produced two types of text transcript from each recording. First, people manually transcribed the speech in each broadcast. Second, transcripts were automatically generated using a speech recognizer. Both the speech-recognized and human-transcribed versions lacked punctuation,

Labeler	**Potential boundary sites**	*Story* **segments**	*Filler* **segments**	**Unlabeled sites**
LDC	13619	1950	834	10835
Reynar	13619	1902	864	10853

Table 6.2: Statistics about the HUB-4 corpus annotation.

indications of sentence and paragraph boundaries and normal capitalization. The text was, however, annotated with boundaries between the topic segments that annotators perceived. The annotators identified a number of segment types, but only two kinds of segments are significant for the evaluation we present below. Annotators labeled sections of broadcasts which were self-contained and limited to a single news item as type *story*. They marked smaller segments which contained information about upcoming stories or other less self-contained subjects as type *filler*. The LDC filtered out other segment types, such as commercials and traffic reports. Figure 6.1 shows an example labeling.

Interannotator Agreement

The original annotation was performed by the Linguistic Data Consortium, but because the guidelines used to annotate the corpus were relatively brief and somewhat ambiguous, we relabeled the corpus and compared the two annotations. Note that we removed the LDC's annotation prior to reannotating the corpus. Before annotating the corpus ourselves, we examined only enough annotated data to write a script to remove the segment markup. We identified our domain cues after reannotating the corpus. Table 6.2 presents some statistics about the corpus and the two annotations. Our reannotation was quite similar to the LDC's annotation in terms of the number of segments of each type that were labeled, as well as the total number of segments identified. The largest difference in these figures was 2.5 percent.

We used the kappa statistic to determine the reliability of the agreement between the two annotations. Table 6.3 presents these statistics. We computed the kappa statistic in several ways. We measured the reliability of the agreement between annotations of *story* and *filler* segments and found it to be tentatively reliable using the kappa score ranges

filler

hello from new york i'm mark mullen

and i'm dick schaap thalia assuras has the morning off in the news at this hour the white house apologizes for f. b. i. file requests deadly new violence in the mideast an appeal for restitution from some abandoned by the u. s. also coming up in this half hour the brinkley round table hashes out some of the weeks developments and hashing of a very different kind we begin with the top of the news and mark

---

story

dick thank you two hundred protesters turned out last night to greet president clinton in san francisco the demonstrators oppose the presidents policy of opposition to gay and lesbian marriages the president was at a fifty thousand dollar a plate dinner at the home of california senator dianne feinstein earlier in the day in nevada mister clinton ran into questions about a new washington controversy that story from a. b. c.'s jerry king

at a work center for juvenile offenders in las vegas mister clinton began a three day three state political tour hoping to highlight his law and order credentials but there was no escaping the latest flap in washington over the white house obtaining f. b. i. background files on top republican officials the president left it up to his senior aid to apologize

it is an inexcusable mistake i think apologies are owed to those that were involved here um but let's understand that ah that nothing improper was done with this information and ah steps have been taken to ensure that nothing like that will happen again

but republican leaders were not about to let an embarrassed white house off the hook

i think it ought to come from the president in las vegas however the president declined a personal apology he simply referred to the panetta statement and said me too

it appears to have been a completely honest bureaucratic snafu ah when we were trying to straighten out who had the security who should get security clearances to come to the white house but i am completely support what he said

jerry king a. b. c. news

Figure 6.1: Example from the HUB-4 corpus showing the annotation of topic segments produced by the LDC. Vertical whitespace is used to indicate changes in speaker or background recording condition.

Units	Kappa score
Separate	0.735
Story only	0.790
Filler only	0.534
Combined	0.764

Table 6.3: The interannotator reliability of the annotation of the HUB-4 corpus.

from content analysis presented above. The kappa score for labeling only *story* segments was also tentatively reliable. The kappa score for annotating only *filler* segments was not reliable. Finally, the kappa score for labeling segments as either *filler* or *story*, thus conflating the categories, fell into the tentatively reliable range. However, this score was higher than the score we measured for treating *filler* and *story* segments separately. As a result, we do not differentiate between these segment types in the experiments we describe below.

Structuring the HUB-4 Corpus

The HUB-4 corpus consists of 174 files. We used 36 files to manually identify cue phrases and 30 of those files to train a maximum entropy model for text segmentation. We segmented the remaining 138 files with several algorithms that were restricted to posit topic boundaries only at particular sites. In addition to annotating the corpus with topic boundaries, the LDC's annotators divided the corpus into units based on criteria significant for speech recognition. The annotation of these units was meant to facilitate experiments to determine whether aspects of the broadcast, such as the presence of background music or whether the speech was spontaneous or planned, significantly impacted speech recognition or IR performance. Our text segmentation algorithms placed topic boundaries between these units, since paragraphs and sentences were not labeled.

The units designed for speech recognition experiments varied greatly in length. Sometimes they contained only a few words, while they were occasionally hundreds of words long. Also, the number of units in a typical topic segment depended heavily on the source of the data, and even then varied widely. As a result, we will evaluate text segmentation

Algorithm	Morphology Normalization	Precision	Recall
Random Guess	N	0.16	0.16
Guess All	N	0.16	1.00
TextTiling[a]	N	0.20	0.38
TextTiling[b]	Y	0.21	0.40
Optimization	N	0.19	0.19
Optimization	Y	0.36	0.20

[a] The publicly available version of Hearst's *TextTiling* algorithm produced no output for one long document. When that document was eliminated from the test set recall improved slightly to 0.40, while precision remained at 0.20.

[b] Again, one file could not be processed using the publicly available version of *TextTiling*. Performance excluding this file was 0.21 precision and 0.41 recall.

Table 6.4: Performance of various algorithms on 138 files from the 1996 HUB-4 Broadcast News Corpus.

algorithms on these data using precision and recall rather than the metric Beeferman *et al.* proposed. Also, using precision and recall permits us to easily aggregate the performance scores measured on individual documents.

Table 6.4 presents the performance of several text structuring algorithms on the test data from the HUB-4 corpus. We measured precision and recall against the LDC's annotation of story and filler boundaries after we merged them into a single category indicating a topic shift. We evaluated both *TextTiling* and our optimization algorithm with and without ignoring words on a stop-list and converting words to their lemmas. We also measured the performance of two baseline algorithms. The first algorithm randomly selected boundaries and the second posited all possible boundaries, and scored perfectly in terms of recall.

Hearst's *TextTiling* algorithm does slightly better in terms of precision and considerably better in terms of recall than guessing randomly. It correctly identifies nearly twice as many boundaries using normalized rather than unnormalized text. Our optimization algorithm outperformed guessing randomly but was not as good as *TextTiling*. We did not test our compression algorithm or Youmans' VMP technique, because they performed poorly on the document concatenation task.

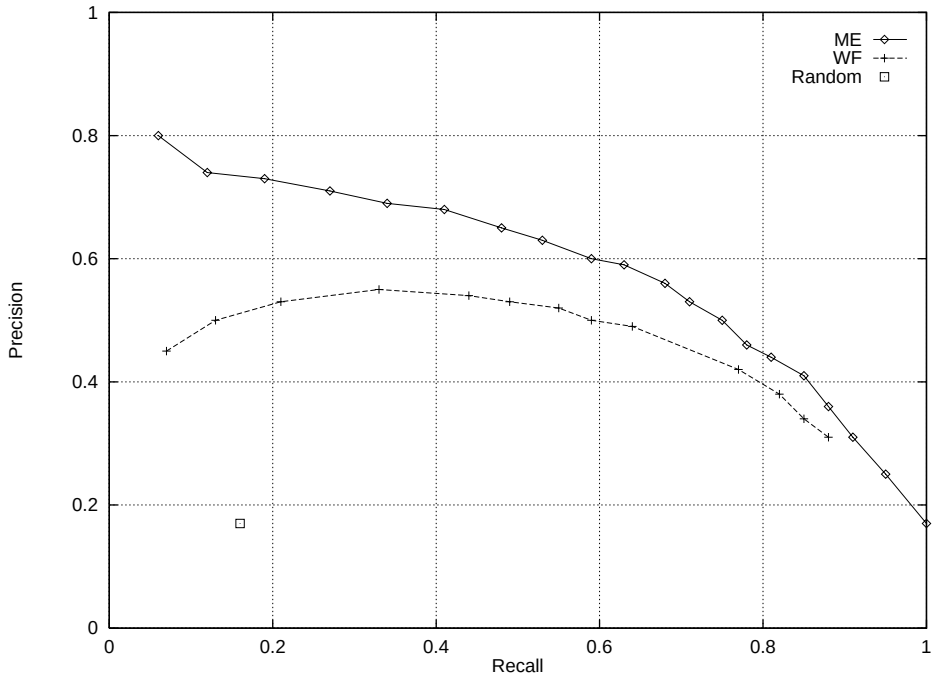

Figure 6.2: Precision-Recall curve for algorithms ME and WF when tested on the 1996 HUB-4 news broadcast Data. The lone point at 0.16, 0.16 represents baseline performance.

Figure 6.2 presents the performance of our word frequency algorithm (called WF, hereafter) and the maximum entropy model (ME) which uses the output of WF and additional cues. This precision and recall graph shows the performance of both algorithms when normalization was performed. The performance of WF is considerably better than any of the algorithms listed in Table 6.4. The point where precision and recall were nearly equal was 0.52. ME performed even better: precision was equal to recall at approximately 0.60.

Performance without Training Data

Algorithm WF outperformed our optimization algorithm on the transcribed HUB-4 data. However, one disadvantage of WF that we noted earlier is that it requires a training corpus to estimate the parameters associated with each word. This means that we cannot apply WF or ME to text in other languages or genres with different word frequencies, as we could

the optimization algorithm. To address this, we evaluated the model when all words were treated as unknown words, as was discussed in the previous chapter. The performance of this version of WF on the HUB-4 corpus is shown by the precision-recall graph in Figure 6.3.

It is quite surprising that this method of identifying topic boundaries performs as well as it does. Not only does it outperform the baseline algorithm, it outperforms *TextTiling* and our optimization algorithm. In fact, it performs nearly as well as WF with word frequency statistics. Several factors account for the good performance of this version of WF. The dotplot algorithm performs moderately well on this corpus and has no prior information about word frequency, so we can conclude that word frequency is not crucial to correctly identifying some boundaries. Another factor is that we garnered the word frequency statistics from *Wall Street Journal* articles which exhibit similar, but not identical, word distributions to broadcast news sources. Using word frequency statistics from a broadcast news corpus would probably widen the gap between the hobbled version of WF and the version that used all the available statistics. Finally, we believe that burstiness, which WF tracks, accounts for the good performance of WF when evaluated without accurate statistics. If the performance were simply due to word repetition, then both *TextTiling* and our optimization algorithm would perform nearly as well.

Speech-Recognized Broadcast News

We did not have access to the speech-recognized transcripts of the HUB-4 corpus that were used for the SDR task. However, the data for the 1997 SDR task was available to us. The 1997 SDR corpus consisted of a subset of the files from the 1996 HUB-4 corpus. There is one crucial difference between the annotation of the 1997 and 1996 versions. Rather than segment the corpus at points where background conditions important for speech recognition changed, the annotators divided the corpus into segments based on speaker changes. This is a more useful division, since speech recognition systems can recognize speaker changes reliably. Recognizing such changes is related to a well-studied problem—determining the identity of a speaker from a sample of their speech. [Ramalho and Mammone, 1994] Note that restricting topic boundaries to lie at speaker turn changes is likely to eliminate the

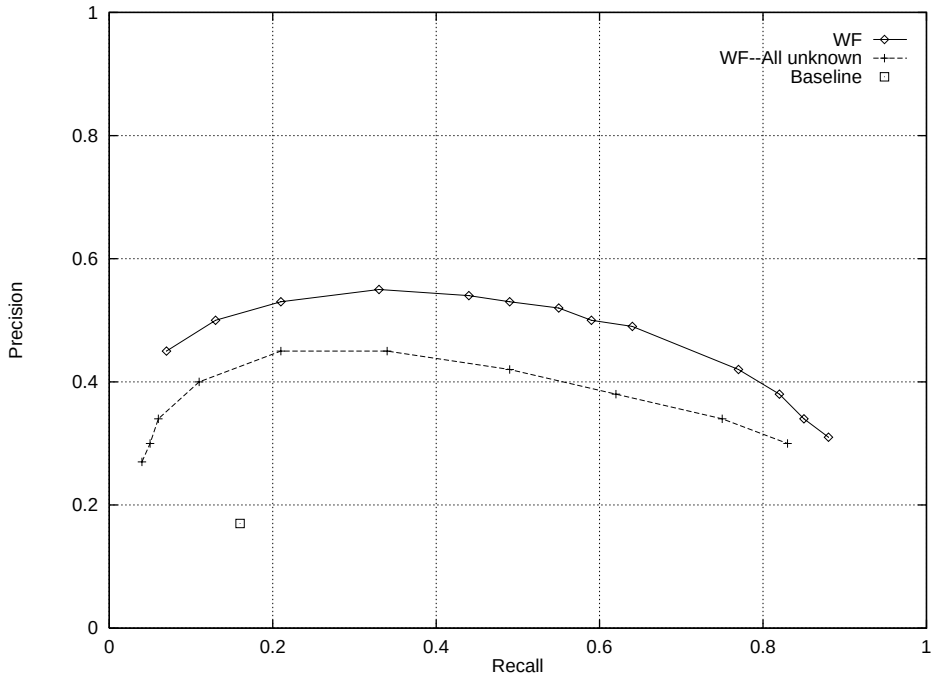

Figure 6.3: Precision-Recall curve for algorithm WF on data from the 1996 HUB-4 Corpus when all words in the corpus were treated as unknown. Performance of the original WF algorithm is shown for comparison.

possibility of perfect performance, since a single speaker may discuss several different topics.

Due to this change in format, we could not use the version of algorithm ME trained on the 1996 data, nor could we retrain it because of the limited number of speech-recognized files available. Consequently, Figure 6.4 presents the performance of only algorithm WF with all parameters used. This figure also shows the performance of a baseline algorithm which randomly selects boundaries and the performance of WF on manual transcriptions of the same documents. WF outperforms the baseline algorithm using speech-recognized data. Performance on the speech-recognized transcripts is poorer than performance on the manually produced transcripts, but still considerably better than baseline.

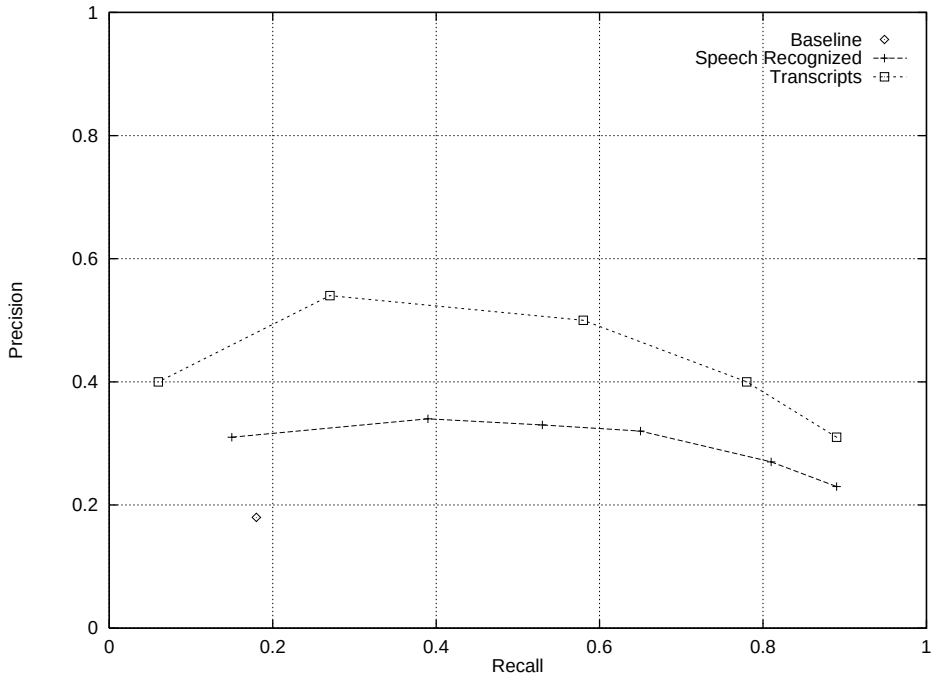

Figure 6.4: Precision-Recall curve for 1997 HUB-4 news broadcast data. Performance is shown for algorithm WF on speech-recognized data and manual transcriptions of the same data. Baseline performance is also shown.

Spanish News Broadcasts

We also evaluated the usefulness of WF on broadcast news data in a language other than English. A portion of the 1997 HUB-4 Corpus is Spanish broadcast news. That data was transcribed, divided into units based on changes of speaker and annotated with topic boundaries in the same way as the English-language data. We conflated the filler and story categories for Spanish data just as we did for the English HUB-4 data, since distinguishing between them in English data was problematic. Figure 6.5 shows the performance of WF on this corpus. We could not evaluate ME because there was too little data to separate it into the requisite training and test portions. As was the case for English data, WF on Spanish language data performs much better than baseline. To segment this data, we again used the version of WF that did not rely on any word frequency statistics, but instead treated all word types as unique unknown words. We anticipate that performance would

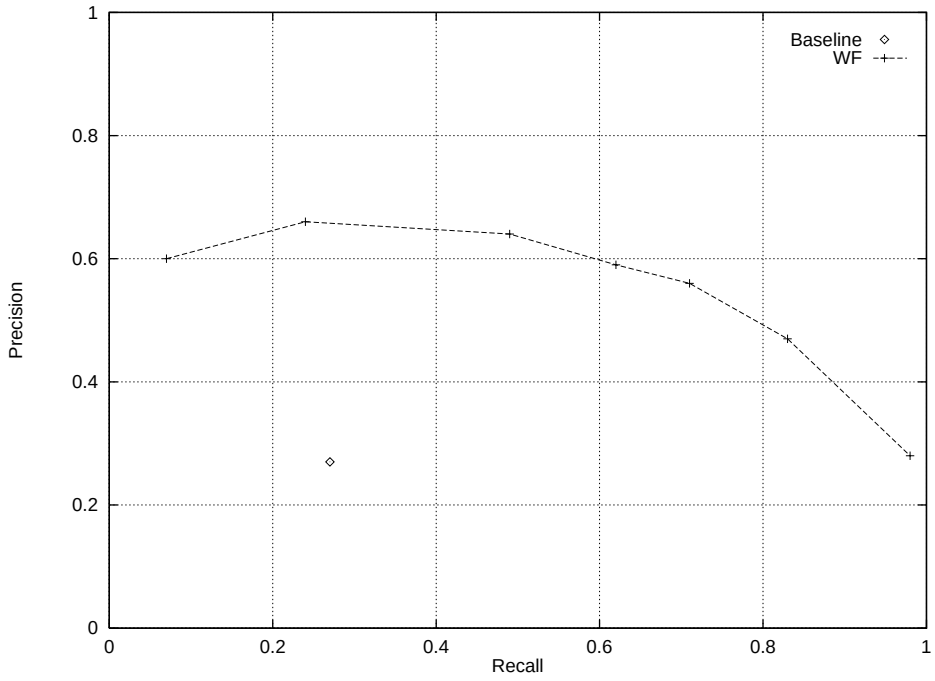

Figure 6.5: Precision-Recall curve for Spanish news broadcast Data from the 1997 HUB-4 Corpus.

improve given a corpus of Spanish news broadcasts from which to learn word probabilities and tools to eliminate closed class words from the data.

6.3 Topic Detection & Tracking Corpus

The Topic Detection and Tracking (TDT) Evaluation contains a task similar to the document boundary identification task presented in Chapter 5. Topic tracking is the process of following the progression of news stories through time, usually using data from a news feed. Although boundaries between articles on the news feed are generally annotated, the TDT Pilot Evaluation included a document segmentation task. The goal was to identify the boundaries between articles without relying on the markup. This task was included because the evaluation will eventually be conducted on speech-recognized news broadcasts which will be divided into segments prior to tracking topics.

Algorithm	Probabilistic Metric	Precision	Recall
WF	0.80	0.15	0.94
ME	0.81	0.27	0.38

Table 6.5: Performance of algorithms WF and ME on the test portion of the TDT corpus.

A significant difference between the TDT corpus and the HUB-4 corpus, which we discussed in the previous section, is its composition. The TDT corpus contains broadcast material transcribed from CNN and text from the Reuters newswire. Unlike documents in the HUB-4 corpus, stories from both sources have accurate punctuation and capitalization.

The TDT corpus contains approximately 7.8 million words of text. We divided the corpus into training and test portions. Our training corpus, which we also used for development, contained approximately 3.8 million words and the test corpus contained roughly 4 million words. The corpus was annotated with paragraph boundaries, but to facilitate comparison with the algorithm Beeferman *et al.* presented, we identified sentence boundaries using a statistical sentence-boundary disambiguation algorithm prior to testing our algorithms [Reynar and Ratnaparkhi, 1997].

6.3.1 Performance on the TDT Corpus

The evaluation metric used for the TDT pilot evaluation was Beeferman, Berger and Lafferty's probabilistic metric. Table 6.5 presents the performance of algorithms WF and ME on our test corpus as measured by the probabilistic metric[1] and precision and recall.[2]

WF and ME performed well according to the metric Beeferman *et al.* proposed, but fared poorly in terms of precision and recall. The performance of our algorithms compared favorably to the algorithm Beeferman *et al.* presented. They tested two versions of their algorithm: one trained only on broadcast news data from outside the TDT corpus and the other trained on 2 million words of TDT data. Their first algorithm scored 0.82 on their metric when tested on 4.3 million words of TDT data, while the second scored 0.88. Note that our algorithm WF scored 0.80 and was trained on no broadcast news data and that

[1] With μ set to 25, the value Beeferman *et al.* used.

[2] Thanks to Doug Beeferman for providing us with their scoring software.

ME was trained on a small quantity of transcribed spoken data which differed significantly from the TDT data.

6.3.2 Adapting Algorithm WF for TDT Data

The poor performance of WF and ME in terms of precision and recall led us to examine their performance on the training portion of the TDT corpus. We found that WF postulated a number of boundaries in the immediate vicinity of actual boundaries. This happened when testing on the TDT corpus but not on the HUB-4 corpus for two reasons. WF posited boundaries between units in the HUB-4 corpus that were more than twice as long as the sentences we algorithmically identified in the TDT corpus. As a result, changes in vocabulary affected the similarity scores of more potential boundaries in the TDT corpus. Also, the HUB-4 corpus, because it lacked punctuation, contained a greater proportion of open-class words than the TDT corpus. Only open-class words figure in the computation of similarity scores, and as a result, there were fewer indicators of segmentation for WF to exploit in the TDT corpus than in the HUB-4 corpus.

To address this, we modified algorithm WF to reduce the number of boundaries guessed in close proximity to one another. We tried selecting the highest scoring boundary from contiguous hypothesized boundaries, selecting all boundaries which were local maxima and imposing a minimum separation between guessed boundaries. Of these, only selecting the highest scoring boundary from neighboring boundaries improved performance on our training corpus. As a result, we evaluated this method on the test corpus as well. Table 6.7 presents the evaluation of this algorithm on the TDT test corpus. Our updated algorithm performed better than the version of Beeferman *et al.*'s algorithm not trained on TDT data, but not quite as well as the version trained on TDT data.

6.3.3 Inducing Domain Cues

The TDT corpus did not exclusively contain broadcast material, but the domain cues we identified by hand for model ME were solely drawn from broadcasts. To address this, we induced domain cues from our training corpus. We automatically determined which word unigrams, bigrams and trigrams occurred more frequently immediately before and

vicki barker ,
susan reed ,
susan king for
skip loescher ,
rowland , c.
rutz , c.
ressa , c.
n. , tokyo
n. , dallas
n. , chicago

Table 6.6: The 10 best cue phrases induced from the TDT training corpus.

Algorithm	**Probabilistic Metric**	**Precision**	**Recall**
WF	0.86	0.46	0.62
ME	0.84	0.70	0.50

Table 6.7: Performance of algorithm WF modified to select the best boundary among neighboring hypothesized boundaries and ME trained on cues induced from the TDT training corpus.

after boundaries. The domain cues we identified occurred at least 5 times in the training corpus and at least 10 times more frequently immediately before or after boundaries than elsewhere in the training corpus.

The domain cues we identified pertained almost exclusively to the CNN portion of the TDT corpus. Despite this drawback, the induced domain cues confirmed that those we selected manually from the HUB-4 corpus were of appropriate types. The best induced cues—those which occurred much more frequently in the context of boundaries than elsewhere—contained reporters names, station identifiers and places reporters broadcast from. Table 6.6 shows the 10 best domain cues. Table 6.7 presents the performance of model ME when trained on the training portion of the TDT corpus using these domain cues, the output of WF and the other indicators of structure we used previously in ME.

The performance of WF modified for TDT data is better than the first model proposed by Beeferman *et al.* and slightly poorer than their second model, which they trained on a large amount of broadcast data. Model ME was, however, much more precise than either

Indicator	Probabilistic Metric	Precision	Recall
Word Frequency	0.71	0.87	0.29
Cue Phrases	0.81	0.46	0.46
Named Entities	0.83	0.70	0.49
Bigrams	0.82	0.76	0.46
Synonyms	0.83	0.74	0.48
First Uses	0.82	0.77	0.46
Pronouns	0.81	0.76	0.45

Table 6.8: Performance of algorithm ME when trained on the training portion of the TDT corpus using all indicators of text structure except the one listed in each row.

of their models. It achieved precision of 0.70 while their best model scored only 0.60.[3]

6.3.4 The Value of Different Indicators

To determine the importance of the features we used in model ME, we trained the model with all of the features except for one and then measured performance on the test portion of the TDT corpus. Table 6.8 shows the results of these tests.

[3]The precision and recall scores Beeferman *et al.* reported [Beeferman et al., 1997b] may be slight overestimates due to a glitch in their scoring software [Lafferty, 1998]. We measured precision and recall independent of their scorer. The computation of the probabilistic metric was not affected by the glitch.

Performance suffers greatly when either the output of the word frequency model or the cue phrases are omitted. Using Named Entities improved recall by a single percentage point, while precision remained constant. Removing each of the other indicators altered the balance between precision and recall in the same way: precision rose, but recall declined. However, performance according to the probabilistic measure declined when each of the indicators was removed from the model. From these results we conclude that each of the indicators we used were useful for these data, but that the word frequency statistics and cue phrases were far and away the most helpful. These findings are particular to the TDT data, but nonetheless demonstrate that the cues we selected are useful for identifying topic boundaries.

6.4 Recovering Authorial Structure

Authors endow some types of documents with structure as they write. They may divide documents into chapters, chapters into sections, sections into subsections and so forth. We can exploit these structures to evaluate topic segmentation techniques by comparing algorithmic determinations of structure to the author's original divisions. This method of evaluation is viable because numerous documents are available in electronic form. Like the document concatenation task, evaluating algorithms this way sidesteps the labor intensive task of annotating a corpus.

We tested algorithm WF on four randomly selected texts from Project Gutenberg. The four texts were Thomas Paine's pamphlet *Common Sense* which was published in 1791, the first volume of *Decline and Fall of the Roman Empire* by Edward Gibbon, G. K. Chesterton's book *Orthodoxy* and Herman Melville's classic *Moby Dick*. We permitted the algorithm to guess boundaries only between paragraphs, which were marked by blank lines in each document.

In assessing performance, we violated our earlier recommendation and set the number of boundaries to be guessed to the number the authors themselves had identified. As a result, this evaluation focuses solely on the algorithm's ability to rank candidate boundaries and not on its adeptness at determining how many boundaries to select. To evaluate

Work	Boundaries	WF		Random	
		Acc.	Prob.	Acc.	Prob.
Common Sense	7	0.00	0.66	0.036	0.72
Decline & Fall of the Roman Empire	53	0.21	0.76	0.024	0.70
Moby Dick	132	0.55	0.80	0.173	0.50
Orthodoxy	8	0.25	0.76	0.033	0.72
Combined	200	0.43	NA	0.059	NA

Table 6.9: Accuracy of the algorithm WF on several works of literature. Columns labeled *Acc.* indicate accuracy, while those labeled *Prob.* show performance using the probabilistic metric of Beeferman *et al.*

performance, we computed the accuracy of the algorithm's guesses compared to the chapter boundaries the authors identified. We also measured performance with the probabilistic metric Beeferman *et al.* proposed. The documents we used for this evaluation may have contained legitimate topic boundaries which did not correspond to chapter boundaries, but we scored guesses at those boundaries incorrect.

Table 6.9 presents results for the four works. The algorithm performed better than a baseline algorithm, which randomly assigned boundaries, on each of the documents except the pamphlet *Common Sense*. *Common Sense* had a small number of chapters, so poor performance on this document is less significant than poor performance would have been for the other, longer works. Performance on the other three works was significantly better than chance and ranged from an improvement of a factor of three in accuracy over the baseline to a factor of nearly 9 for the lengthy *Decline and Fall of the Roman Empire*. These results suggest that algorithm WF could be used to recover authorial structure or to suggest to authors how to divide documents into segments as they write.

6.5 Conclusions

We demonstrated that our word frequency algorithm WF is useful for segmenting a number of different types of documents. It outperformed our optimization algorithm and *TextTiling* on the HUB-4 Broadcast news corpus. This is the most important of the evaluations we

performed, since the most likely uses of text segmentation pertain to broadcast documents. WF also performed well on the TDT corpus, but this evaluation is slightly less interesting because the TDT corpus contains broadcast documents interspersed with Reuters data. We also demonstrated the algorithm's utility independent of training corpora using Spanish language data and English data for which no frequency statistics were available. Finally, we showed that WF was also useful for recovering chapter divisions in works of literature.

Algorithm ME performed better than WF on the HUB-4 data and greatly improved precision on the TDT corpus. With additional training data, it is likely that performance on the HUB-4 corpus would improve further. In the tests we conducted for the HUB-4 corpus, we trained ME using documents from various broadcast sources, each with their own idiosyncratic hints about structure. But, if we knew the source of the documents to be segmented, we could build source-specific models. Such models would be cognizant of the names of reporters frequently on the program, as well as the particular stylistic conventions employed. We demonstrated that we could algorithmically identify domain-specific cues such as these and verified the appropriateness of the domain cues we identified by hand by learning cue words and phrases from the TDT corpus.

Chapter 7

Applications

Automatic techniques for linearly structuring text have many potential uses. We will discuss a number of these below, including 3 areas in which we have confirmed the useful of the segmentations produced by our algorithms: information retrieval, coreference resolution and language modeling.

7.1 Information Retrieval

Information retrieval is the task of identifying the documents in a collection which satisfy a user's request for information about a particular subject. Documents meeting this criterion have come to be known as "relevant" documents, despite the discrepancy between the usual usage of relevant, meaning related, and what is implied: that these documents are specifically about the topic of study [Harter, 1992].

In general, the process of information retrieval proceeds as follows: the user formulates a query, usually in either natural language or in a logical language which uses boolean connectives. The IR system then searches the collection, or more frequently, a precompiled index, to identify relevant documents. The system then presents a list of these documents to the user for her perusal. The returned documents may be ranked in order of relevance if the retrieval technique being employed computes a similarity score. Generally, boolean systems divide the collection into sets containing relevant and irrelevant documents while systems which employ a similarity metric rank documents according to that metric.

A common belief about information retrieval is that performance could be improved if documents, especially long ones, were first divided into sections and IR systems then indexed the sections as if they were separate documents. This belief arises because word usage drives IR and depends on topic. As we mentioned earlier, the number of times a word occurs in a document depends on the topic of the document. Short documents, and even some lengthy ones, address only one topic. However, many documents pertain to multiple topics or, at the very least, sufficiently different facets of a single topic that the vocabulary used will change throughout the document as the topic shifts. As a result, word frequency statistics collected from entire documents, which are the driving force in most IR algorithms, may be unrepresentative of any single topic section. It would be more useful if the statistics were taken from individual topic segments instead.

For example, one document from the HUB-4 broadcast news corpus has a short segment about the travels of the Olympic torch. A query associated with this collection is "Does the torch ever travel by motorcycle?" The query does not exhibit much similarity to the entire document—in fact, the word *motorcycle* only occurs once in 4619 words. However, the query is more similar to the section about the Olympic torch. In a section only 211 words long, *motorcycle* occurs once, and *torch* appears 9 of the 10 times it appears in the entire document. Assuming function words are eliminated and that there are no content words in common between the document and the query besides *motorcycle* and *torch*, the similarity score between the query and the document as a whole would be 0.023 according to the cosine distance measure, while the score for the relevant subsection would be much greater: 0.380. The difference in these scores demonstrates the usefulness of sectioning documents prior to comparing them to queries using the vector space model.

7.1.1 Previous Work

Attempts have been made to show that IR systems perform better when units of text smaller than documents are indexed. The units used have varied from sentences and paragraphs to fixed length blocks of varying sizes to subtopic segments.

Stanfill & Waltz

Stanfill and Waltz demonstrated that IR benefited from indexing 30 word segments. They tested their information retrieval system, which ran on a massively parallel Connection Machine, on a collection of news articles and found a precision-recall product of 0.65 [Stanfill and Waltz, 1992].

Hearst & Plaunt

Hearst and Plaunt used *TextTiling* with a form of term weighting known as term frequency, inverse document frequency weighting ($tf \cdot idf$) to structure texts prior to IR. $tf \cdot idf$ weights words which occur in few documents in a collection more highly than those that occur in many documents. The vector space model is still used to compute similarity, but the values in the word vectors are weights rather than merely the number of times each word appears in a document. With $tf \cdot idf$ weighting, the weight in document D_i of word k, W_{ik}, depends on both the number of times word k appears in document D_i, referred to as $\mathtt{tf}_{ik}$, and the number of documents in which the word appears in a collection consisting of N documents, which is called n_k. The equation below determines the weight associated with each word.

$$W_{ik} = \frac{\mathtt{tf}_{ik} \cdot \log(N/n_k)}{\sum_{k=1}^{t}(\mathtt{tf}_{ik})^2 \cdot (\log(N/n_k))^2} \tag{7.1}$$

Hearst and Plaunt segmented text using *TextTiling* and then indexed each subtopic segment separately for use in an IR system. They measured performance improvements of between 18.9 and 28.2 percent. However, they obtained similar improvements on the same collection of documents when they indexed paragraphs rather than the segments *TextTiling* identified [Hearst and Plaunt, 1993].

Salton, Allen and Buckley

Salton, Allen and Buckley improved the performance of an IR system by incorporating information about passages [Salton et al., 1993]. Their method involved two steps. First, they used the vector space model to identify documents that were globally similar to each query. They then used the vector space model again to compute the similarity between

the query and individual sentences and paragraphs of the text. If a globally similar text did not contain relevant passages, then they assumed it was erroneously identified in the first step. They found that precision improved by up to 22.5 percent using this method. Recall did not improve with this technique because performance is measured relative to the set of documents identified on the first pass. For recall to improve, the second pass would need to identify additional relevant documents.

Callan

Callan studied the effect of indexing passages with the INQUERY information retrieval system [Callan et al., 1992], a probabilistic IR system which identifies relevant documents using Bayesian networks [Callan, 1994]. He suggested two reasons to use units smaller than documents. First, "If each portion of text, or passage, is ranked an interface can quickly direct a user to the relevant information in a document." Second, "Long documents, documents with complex structure, and even short documents summarizing many subjects, are a challenge for algorithms that do not distinguish where in a document the text matches a query. If the algorithm cannot distinguish a few matches scattered across a document from a dense region of matches, it may have difficulty retrieving long documents and newswire 'news summaries.'"

Callan suggested that three types of units could be exploited in an IR system:

Discourse units Units of a document, such as paragraphs or sections, which are identified explicitly by the document's author.

Semantic units Units of a document not explicitly marked by the document's author which divide a document into non-overlapping regions.

Fixed-size units Units of a document neither based on content nor authorial annotation, but solely on a predetermined word-window size. Windows may or may not be permitted to overlap.

Callan only tested IR performance using units of the first and third types. He concluded that passages based on overlapping word windows were more useful for IR than those comprised of paragraphs.

Kaskziel & Zobel

Kaskziel and Zobel compared a number of passage retrieval techniques using the Federal Register collection [Kaszkiel and Zobel, 1997]. They evaluated retrieval performance using the cosine distance measure with $tf \cdot idf$ on documents, paragraphs, pages of documents, sections of documents, tiles produced by *TextTiling*, fixed length and variable length passages, and text windows of several sizes. They also evaluated the vector space model with pivoted document length normalization [Singhal et al., 1996] on the same set of units. We will discuss pivoted document length normalization in Section 7.1.2.

Several conclusions can be drawn from the wide range of experiments Kaskziel and Zobel conducted. First, pivoted document length normalization generally improves performance over simply using $tf \cdot idf$. Second, retrieving documents using fixed length, non-overlapping windows of words improves performance considerably–from 12.7 to 18.9 percent in terms of average precision. The best performance, however, consistently comes from using fixed-length passages starting at every word in each document. Using passages of sizes 150 and 350 words improved performance by 18.0 to 37.7 percent.

Although the improvements that come from indexing all fixed-length passages are considerably better than those that come from using any other unit, there are drawbacks. If the goal is to identify the most relevant document, as was the case in Kaskziel and Zobel's experiments, the sole drawback is in terms of indexing. Dividing documents into fixed length passages beginning at every word in a document greatly increases the size of the index used for IR. Needless to say, this can put an unbearable burden on an IR system used to index large collections, especially if many of the documents in the collection are long, because longer documents are more costly to index than short ones.

For example, compare the storage costs of indexing two collections, each consisting of 3500 words, using fixed length passage retrieval with the passage length set to 350 words. Suppose the first collection has one document of 3500 words and the second collection has 10 documents, each exactly 350 words in length. Indexing the first collection would require indexing $3500 - 350 = 3150$ documents of length 350 for a total of 1102500 word index entries, while the second would require only 3500 entries.

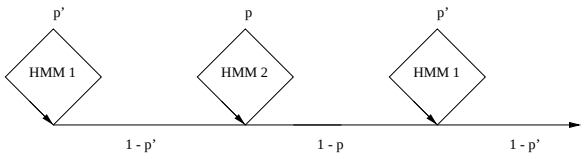

Figure 7.1: Diagram of the HMMs Mittendorf and Schäuble used for passage retrieval. The transition probabilities p and p' were both set to 0.9999.

There is, however, a potentially more significant disadvantage if the items being retrieved are sections of documents rather than entire documents. Using fixed length passage retrieval the user of an IR system will likely be shown passages which begin in the middle of a topic, in the middle of a paragraph and even in the middle of a sentence. Use of this technique should be limited to situations where the units of retrieval are entire documents, and even then it is crucial to consider the increased storage cost.

Mittendorf & Schäuble

Mittendorf and Schäuble proposed a method for retrieving arbitrary length passages using two different hidden markov models (HMM). They used the first type of HMM to compute the probability of seeing a particular stretch of text independent of the query. The second HMM determined the probability that a passage was relevant to the query. Figure 7.1 shows the linkage between the two types of HMM. Mittendorf and Schäuble determined the probability of retrieving a particular passage by computing the probability of seeing the text prior to and following that passage (which could be null if the passage occurred at the beginning or end of the document) with the first type of HMM and then combining that probability with the probability from the second HMM that a particular passage matched the query. They used the Viterbi algorithm to determine the highest probability path through the concatenated HMMs and from that path identified the best ranked passage for a particular query [Mittendorf and Schäuble, 1994].

Mittendorf and Schäuble's technique has the advantage that the document collection need not be segmented prior to IR. But, it has the disadvantage that the passages it

retrieves begin and end with words from the query. The passages are therefore unlikely to begin and end at sentence boundaries, let alone paragraph or topic boundaries.

Mittendorf and Schäuble tested their technique on the MEDLARS collection. They found performance using two variations of their passage retrieval model to be better than the performance of the vector space model. They presented performance figures graphically in their paper, but their passage retrieval model appeared to improve performance measured where precision and recall were approximately equal from 0.40 for the vector space model to 0.50.

Summary

Salton, Allen and Buckley improved the performance of their IR system by comparing queries to individual sentences and paragraphs. Hearst and Plaunt showed that the subtopic segments identified by *TextTiling* improved retrieval performance, but the improvement was no better than that achieved by indexing paragraphs. Stanfill and Waltz, Callan, and Kaskziel and Zobel each demonstrated that indexing unmotivated blocks of text was beneficial for IR performance. Mittendorf and Schäuble built an HMM which identified passages beginning and ending with query words and improved precision and recall.

Only Hearst and Plaunt demonstrated an improvement in performance using motivated segments—segments which were intended to be topically coherent. The advantage of such segments is that they can be presented to users in response to queries. Arbitrary passages which begin in the middle of a sentence or paragraph, like those used by Stanfill and Waltz, Callan, Mittendorf and Schäuble and Kaskziel and Zobel, are less useful for this purpose. We will use topic segments to show a performance improvement on an IR corpus transcribed from spoken data. Previous attempts to improve IR performance using segments have been conducted using pristine textual data, with reliable punctuation, capitalization and, sometimes, indications of sentence and paragraph boundaries. Our data are transcribed from speech and lack the advantages conveyed by these features.

7.1.2 TREC Spoken Document Retrieval Task

We measured the utility of the best performing text segmentation algorithms we presented in Chapter 5 for IR using the data from the TREC Spoken Document Retrieval (SDR) task. This task differs from most IR tasks in several significant ways. First, it is conducted using text derived from speech. IR is not performed using the speech signal. Rather, there are two scenarios. In the first, IR is performed using transcripts produced by human annotators and in the second, a transcript created automatically by a speech recognition system is used. We were only able to test IR performance using manually produced transcripts, because we did not have access to speech-recognized data for the entire corpus.

The second difference pertains to the method of evaluation. Unlike most IR test sets, the SDR test set contains only one document relevant to each query. In fact, the queries were written with knowledge of the contents of the documents. As a result, measuring recall would be pointless: it would either be 0 or 1 for each query. Measuring precision would be equally uninformative, since the maximum number of relevant documents is 1. Also, because the collection contained a relatively small number of documents, 174, IR systems were able to rank the entire collection.

Instead of measuring performance with precision and recall, systems are scored by determining the average rank of the relevant document. A perfect system would rank the sole relevant document first for each query and would therefore assign the relevant document an average rank of 1. The worst possible performance with this collection would be an average rank of 174, which could only be achieved by ranking the relevant document last for every query [HUB-4 Program Committee, 1996].

In our evaluations, we performed IR on sections of documents then eliminated all but the highest ranked section of each document. This yielded an ordering of the entire collection based on the rank of the most relevant section of each document. We then used this ordering to measure the rank of the relevant document.

Our IR Experiments

We conducted our IR experiments using the publicly available SMART system which implements the vector space model [Buckley, 1985]. Within SMART we used both $tf \cdot$

idf weighting and our implementation of pivoted document length normalization (PDLN) [Singhal et al., 1996], which accounts for variations in document length so that short documents are not retrieved overly often. This normalization was inspired by the observation that there is a positive correlation between document length and relevance in the TREC collection [Harmon, 1992]. Singhal, Buckley and Mitra showed that this normalization improves average precision on disks one and two of the TREC corpus by between 9 and 12 percent [Singhal et al., 1996].

PDLN is used with the cosine distance measure. The crucial difference between PDLN and the traditional cosine distance is that the length of the document vector, which is used in the denominator of the formula for the cosine distance, is altered to bias retrieval toward longer documents. The length of the document vector, $|W|$, is computed and then scaled using the formula shown below. In the formula, S is the normalization required to compensate for the tendency to over-retrieve short documents and is frequently set to 0.75—a value which has yielded good results on several collections. $W_{\mathtt{av}}$ is the average document vector length in the collection.

$$W' = (1.0 - S) + S \cdot \frac{|W|}{W_{\mathtt{av}}} \tag{7.2}$$

Table 7.1 presents the results of the experiments we conducted both with and without PDLN on the SDR data. The rows in this table represent different ways we indexed the SDR collection.

The row labeled *Documents* refers to indexing entire documents. Following on the success of Callan and Kaskziel and Zobel on indexing fixed-length, overlapping segments of documents, we also divided documents into 230 word overlapping passages and indexed those. We did not use all overlapping passages, but instead used those beginning every 10 words. We also tested the usefulness of the segments labeled by human annotators. Results of that test are found in the row labeled *Annotator Segments.* The rows labeled *ME* and *WF Segments* in the table present results when we indexed the segments identified by two of our text segmentation algorithms. We also indexed the collection by treating the small segments annotated to enable tests of speech recognition systems as documents. Results of that test can be found in the row labeled *Background Segments* in the table.

Method	Average Rank	Average Rank with PDLN
Documents	12.70	9.52
Annotator Segments	9.42	8.42
Background Segments	12.32	12.32
230 word overlapping passages	9.01	7.40
WF Segments	10.54	9.48
ME Segments	7.92	7.54

Table 7.1: IR performance on the spoken document retrieval corpus with stemming using SMART's built in stemmer.

The results from Table 7.1 demonstrate the usefulness of text segmentation for information retrieval. Using the segments the annotators identified outperformed indexing documents themselves. Indexing overlapping passages performed the best and would probably improve if we indexed all possible 230 word segments. That was not feasible, since even indexing segments beginning every 10 words resulted in a 148 megabyte index for 4.5 megabytes of text. Treating the background segments as documents yielded the worst performance with pivoted document length normalization. When we indexed the segments identified by algorithm WF, performance was marginally better than that achieved by indexing documents, but not as good as when we indexed the annotators' segments. We observed the second best performance when we indexed the segments identified by ME. The average relevant document was ranked 7.54, only 0.14 behind the best performing method, indexing 230 word overlapping passages. Indexing changes in background conditions yielded poor IR performance.

Unlike previous attempts to show the usefulness of segmentation for IR, we have shown that the segments produced by our algorithms improve IR performance more than using any units found in the documents. Our data had no labeled sentence or paragraph boundaries, so we could not compare, as Hearst and Plaunt did, to IR performance when indexing these units.

Example Retrieved Segment

Figure 7.2 shows the segment most relevant to the query discussed above about the travels of the Olympic torch by motorcycle. If an IR system indexed the segments identified by algorithm ME and presented the most relevant segments to users, the text in the figure would be presented for that query. This segment contains only 424 words, while the document containing it is 4619 words long. As a result, the user of an IR system would save a substantial amount of reading time if given only this section to peruse. A perfect segmentation technique which replicated the annotation produced by the LDC would have divided this segment into two nearly equal sized segments, thus saving a user additional time.

7.2 Language Modeling

Language modeling has a number of applications including playing a crucial role in speech recognition. Speech recognition is the notoriously difficult problem of determining what words are encoded in an acoustic signal. It is challenging for a number of reasons. Different speakers pronounce words in subtly different ways. Languages contain homonyms, which are pairs of words that are pronounced the same but spelled differently, like *too* and *two*. Speakers also slur words and speak at different rates. Also, there are no explicit boundaries between words like those marked by whitespace in text.

Continual improvement in speech recognition technology has been made over the past few years and speech recognition is now sufficiently accurate and computationally inexpensive that affordable commercial speech dictation systems are available for personal computers. However, these systems are not perfect, and improving accuracy is still an active area of research.

Most speech recognition is performed using some form of n-gram language model. A simple trigram model is often used. According to this model the probability of a word depends only on the identity of the previous two words. This gross over-simplification is necessary to model language, since limiting the context used by language models prevents sparse data problems from becoming overwhelming. Even word trigrams are relatively

officials in trenton new jersey are waiting the arrival of the olympic flame the torch relay is expected to reach new jersey's capitol around eight o'clock tonight right now the torch should be in bedminster township new jersey it is on schedule on its way to philadelphia the torch will cross into pennsylvania about eight forty five and then by about nine o'clock it should actually hit philadelphia it will travel by motorcycle from buck county along route one roosevelt boulevard going on to ridge avenue and kelly drive and then at kelly drive the local torch bearers in the philadelphia area will carry the torch from the west river drive to its final destination at the art museum where a big celebration party will take place between ten thirty and eleven o'clock the celebration actually starts at around eight but the torch won't get into center city until about ten thirty or eleven o'clock the torch leaves philadelphia starting at five tomorrow morning and will travel through chester and delaware counties before going on to delaware the final stop for the torch tomorrow will be baltimore the torch started its journey april twenty seventh and will end fifteen thousand miles later at the start of the olympics on july nineteenth

wilmington delaware residents will be electing a new mayor this year and republican candidate brad zuber yesterday released his plan to fight wilmington's growing crime problem zuber tells w. h. y. y.'s twelve tonight the city's neighborhoods need help in fighting crime it's not unusual for our police officers to face a barrage of bricks and bottles in certain neighborhood in fact there are certain neighborhoods where the people who deliver pizza and mail are afraid to go and where parents live in fear of their children being caught in the cross fire i have to say that life in certain parts of wilmington is starting to be as dangerous as life in washington d. c. and that is unacceptable incumbent first term democratic mayor jim sills is running for reelection this year and he's facing a stiff challenge in the democratic primary this coming september checking the franklin institute forecast tonight some clouds fog and a few scattered showers or thunderstorms they should not be around the area for when the torch comes into philadelphia around ten thirty eleven o'clock and tomorrow ah humid with scattered showers and thundershowers highs of around eighty right now it's eight four degrees in philadelphia it's five o. six you're listening to ninety one f. m.

Figure 7.2: Example of a segment identified by ME which would be returned to a user of an IR system. The whitespace indicates where an additional boundary was placed by the annotators.

sparse. If a system is expected to handle a vocabulary of 10,000 words, far fewer than typical adults know, then there are $10000^3 = 10^{12}$ possible trigrams—more words than even the largest training corpus contains.

The trigram modeling simplification described above allows speech recognition to be performed but at the same time is a severe handicap. There are much longer distance dependencies in language. A noun phrase consisting of a determiner, three adjectives and a head noun is 5 words long. In a trigram model, the determiner and the first adjective will have no impact on identifying the head noun.

A number of attempts have made to address this handicap in language modeling. Cache-based language models boost the probability of recently seen words (see, for example, [Lau et al., 1993]). Trigger-based language models increase the probability of particular words based on observing sets of other words with which they frequently co-occur (e.g. [Rosenfeld and Huang, 1992]).

7.2.1 Text Segmentation and Language Modeling

More relevant to work on topic segmentation, however, is work done at NYU in conjunction with BBN which focused on improving speech recognition accuracy using sublanguage corpora [Sekine et al., 1995]. While performing speech recognition, the NYU/BBN system employs previously recognized sentences to identify documents with contents related to those sentences. The system then used words from those documents to adapt the word probability model used for speech recognition.

Topic segments could be used in this way rather than entire documents. The advantages of using topic segments to improve speech recognition in this context are similar to the advantages of using them for IR. Long documents are often about numerous topics and documents with relevant sections may contain completely unrelated sections as well. The words in unrelated sections would erroneously have their probability boosted in the same way as the words from the relevant section. For example, if documents from the HUB-4 collection were used in this way, then while recognizing a document about the tobacco industry, the May 23 edition of NPR's *Marketplace* might be used to enhance the probabilities of topic-related words, since it has a long section about lawsuits against the

tobacco companies in the U.S. However, that document also has sections about German auto makers and British tabloid newspapers. Words from these irrelevant sections, such as *Mercedes* and *newspaper*, would have their probability boosted as much as pertinent words like *smoke*. Also, performing the document matching step on a collection of topic segments would identify shorter passages of text, permitting faster execution.

Another application of topic segments within the context of language modeling is to build static topic-dependent language models. Currently, most language models are intended to be relatively broad in coverage. They are often built from *Wall Street Journal* text. We should be able to model language more accurately by taking topic into account. To test this hypothesis, we could first divide documents into topic segments, then group the segments by topic using clustering techniques. We could then build language models from these topic clusters.

Determining which language model is most appropriate would often be difficult. In some cases information from outside the speech stream may be useful for selecting the best language model. For example, if speech recognition is regularly performed on the same news broadcast, correlations between story type and reporters could be identified. This information could then be used to identify the most topically-related language model.

7.2.2 Topic-Dependent Language Modeling

We conducted an experiment using the TDT corpus to test the hypothesis that topic segmentation would be useful for language modeling. We segmented the corpus using algorithm ME and then identified all segments that contained the word *Clinton*. The identification of these segments simulated clustering the articles based on topic. We divided the segments into two groups. We reserved a set of articles containing approximately 4000 words for test data and built a topic-dependent language model from the approximately 20000 remaining words. This language model used trigrams and backed off to bigrams and then unigrams. We also built a separate language model from the first 20000 words of the TDT corpus which did not contain articles in the test corpus.

Language Model	Perplexity
Standard	153.7
Topic-dependent	134.0

Table 7.2: Results of a language modeling experiment conducted on the TDT corpus.

We measured perplexity to determine which model better predicted the test data [Bahl et al., 1977]. Perplexity measures the difficulty of predicting the identity of a word in the text given the words which precede it. Low perplexity scores indicate that a language model is representative of the text. Computing perplexity involves first determining the probability of generating the sequence of words W in the test corpus with a language model. The trigram language model we used assumes that the probability of a word is dependent only on the identity of the two words preceding it. Thus, we can compute the probability of the corpus, $P(W)$, using the formula below. n is the number of words in W.

$$P(W) = \prod_{i=1}^{n} p(w_i|w_{i-1}, w_{i-2}) \tag{7.3}$$

We computed the perplexity, $\texttt{Perp}(W)$ of the text using the formula below.

$$\texttt{Perp}(W) = 2^{-\frac{\log_2 P(W)}{n}} \tag{7.4}$$

The perplexity scores, which are presented in Table 7.2, indicate that segmenting documents and then performing simple topic clustering does improve language modeling. The difference in perplexity is modest, but we trained the language models from only 20000 words of text, and used simple techniques to identify related segments.

7.3 Improving NLP Algorithms

Many natural language processing tools rely on word co-occurrence statistics collected from corpora or examine windows of words in the preceding context of a particular word or phrase. Algorithms of the first type include those for word-sense disambiguation (e.g.

[Yarowsky, 1992]), language modeling (e.g [Beeferman et al., 1997a]), coreference resolution [Kehler, 1997] and identifying the most likely genre for a document [Losee, 1996, Kessler et al., 1997]. Pronoun resolution techniques often search the preceding context for candidate antecedents (e.g. [Baldwin, 1997]).

Using data found in arbitrary windows of words has yielded state-of-the-art systems. However, performance on various tasks should improve if more motivated segments were used. Words related to one topic would not erroneously be counted as co-occurring with words from a neighboring topic segment. Rather than relying, for example, on computing statistics in an arbitrarily sized window of, say, 100 words, independent of whether or not the topic has changed, it would be better to use the location of topic boundaries to limit the size of the window to the confines of a single topic. For example, if we computed frequency statistics for words that occur near the word *industrial* in the HUB-4 corpus, we would erroneously identify the word *churches* as appearing twice within 50 words simply because the discussion of the Dow Jones industrial average was in a topic segment immediately followed by a story about the burning of churches in the southern U.S. in one document. We would not identify this spurious connection between *industrial* and *churches* if we first identified the topic boundary between the segments containing these words and then counted only co-occurrences within segments.

Although a number of NLP algorithms would benefit from topic boundary detection, we only measured the potential improvement for pronoun resolution. Pronoun resolution is important for such tasks as summarization, IR and IE and is often performed by ranking candidate resolutions using various heuristics and then choosing the best of them. The size of the set of candidate antecedents can significantly impact the running time of coreference algorithms as well as their performance. To demonstrate the usefulness of topic segmentation for coreference, we counted the number of candidate antecedents for singular pronouns which referred to people mentioned in 4 randomly selected documents from the HUB-4 corpus. We limited the set of possible antecedents to noun phrases which referred to people because most coreference systems incorporate property information.

Table 7.3 presents statistics regarding this investigation. We present statistics for both the gold-standard topic segmentation produced by annotators and the output of model ME.

Source	Avg. candidates in previous context	Avg. candidates in current topic segment	Resolutions outside segment
Human Annotation	42.9	12.0	4
Max. Ent. Model	42.9	14.4	45

Table 7.3: Number of candidate antecedents for singular pronouns that referred to people in 4 randomly selected broadcast news transcripts. 189 pronouns were examined.

These numbers indicate that using the topic-segmented text results in an approximately three-fold reduction in the number of candidate antecedents if the search for pronoun resolutions is limited to the current topic segment. Further reduction would occur if the algorithm produced a segmentation identical to the human annotators.

There is, however, a cost associated with assuming that resolutions are within the current topic segment. 4 resolutions could not be made using only the noun phrases within the current topic segment given the human annotation. This number increased to 45 if we used the automatically generated annotation, but in some of these cases, no candidate antecedents were present in the current topic segment. As a result, a coreference resolution algorithm could easily determine that searching antecedents outside the current context was necessary, thus mitigating the effect of these 45 pronouns on pronoun resolution performance.

7.4 Potential Applications

7.4.1 Summarization

In her presidential address to the ACL in 1994, Sparck Jones stressed the importance of summarization as the focus for future NLP research [Sparck Jones, 1994]. She also suggested that understanding discourse structure was crucial to producing good summaries since text structure is deeply related to text meaning, which, of course, is what summaries attempt to capture.

Documents could be summarized using information about text structure and an algorithm to extract important sentences. A text could first be divided into segments by a structuring algorithm, then representative or important sentences from each of the segments could be selected and concatenated to produce a summary. The difficult part is determining which sentences satisfy these criteria. One possibility is to choose sentences about the most frequently mentioned entity in a segment, as Sparck Jones suggested. Another possibility is to use word frequency information to identify important terms within the segment, as even the earliest summarization by extraction systems did [Luhn, 1958].

Paice observed about summarization systems that "...it is hard to believe that systems relying on selection and limited adjustment of textual material could ever succeed" [Paice, 1990]. However, his observation was made in reference to producing abstracts of articles which could themselves be indexed by IR systems and then quickly perused to determine whether to read a printed document. While a summary produced by a summarization system that takes advantage of text structure to extract sentences may not be adequate for this task, it could be used instead to decide whether to read an electronic version of a document. Lower fidelity summarization techniques are more useful today because the cost of accessing an electronic document is lower than the cost, measured more in terms of time and effort than money, of retrieving an article from a distant library. In addition, since both summaries and summarized documents would be online, hyperlinks could be established from the extracted sentences into the original document to facilitate skimming sections identified as important.

7.4.2 Hypertext

Salton, Buckley and Allen describe a system for automatically linking related portions of the same document or portions of two separate documents [Salton and Buckley, 1992, Salton and Allan, 1993]. They suggest that these links will improve access to large collections of documents by providing an easy means of directed random access, and by highlighting relationships between sections.

There are several reasons it is desirable to automate the linking process. First, in a document or collection with n segments, there are $O(n^2)$ potential links. This is too many

to consider manually even when the number of segments is small. Second, identifying links by hand may require the services of a domain expert. Consequently, automation will make linking feasible for large collections and reduce the cost for technical collections.

An obvious use of text structuring algorithms is to divide lengthy documents into segments, then determine the similarity between segments using the vector space model or another similarity metric. Similar segments could then be linked to allow easier navigation of WWW documents or navigation among segments or documents identified by an IR engine.

7.4.3 Information Extraction

Information extraction (IE) is the task of automatically filling out templates with particular facts from a document. IE systems use different templates to report different types of information. For example, if IE were used to convey the details of new product introductions, one of the *slots*—relevant pieces of information contained in a template—might be the date when the product will become available.

One frequently addressed IE task is identifying management changeovers in newswires [MUC-6 Program Committee, 1995, Baldwin et al., 1997]. For example, given the sentences in Example 7.5, an IE system might generate filled-in templates like those shown in Figure 7.3. Some of the slots are empty because most management-change events specify only a subset of the types of facts that have a corresponding template slot.

(7.5) Bill Smith today resigned from XYZ Computer Corporation citing conflicts with the board of directors. He was immediately hired as Chief Financial Officer of the newly formed WXY Company.

We built tools for information extraction using the EAGLE NLP system as part of a project conducted with Lexis-Nexis [Baldwin et al., 1997]. Although we did not evaluate our information extraction system algorithmically, as is the case for the Message Understanding Conference evaluations, we did perform both a quantitative and a qualitative evaluation. The quantitative performance was encouraging, but only the qualitative evaluation is relevant here. One of the difficulties we encountered was that templates were

Company: XYZ Computer Corporation
Event-type: resigned
Person: Bill Smith
Position:
Reason: conflicts with the board of directors

Company: WXY Company
Event-type: hired
Person: he
Position: Chief Financial Officer
Reason:

Figure 7.3: Sample completed templates from an information extraction task.

often only partially completed. This occurred for a number of reasons, including the failure of various processing components. One of the most frequent causes, however, was that relevant information was not localized but was distributed throughout a document. The text which triggered a pattern to fire is sometimes far from the fragment of text needed to fill one slot of the template.

Our approach to information extraction was highly syntactic and relied on a pattern matching language called MOP [Doran et al., 1996], which had access to various types of analysis including part-of-speech tags and parse trees. One way to address the problem of needing distant information to complete a template in this system would involve structuring texts prior to performing information extraction. Then, when filling templates, our system could identify important fields not filled with local information and look to global information from the current topic segment. We could track the presence of particular types of entities within each segment and use this information to fill empty slots.

Information Extraction and Text Segmentation

We are unable to present examples from the data used for the work with Lexis-Nexis, so a simple example from the *Wall Street Journal* will have to suffice. There are a number of management changeovers in the text in Figure 7.4 which would be identified by an IE

For the sixth time in as many years, Continental Airlines has a new senior executive. Gone is D. Joseph Corr, the airline's chairman, chief executive and president, appointed only last December.

Mr. Corr resigned to pursue other business interests, the airline said. He could not be reached for comment.

Succeeding him as chairman and chief executive will be Frank Lorenzo, chairman and chief executive of Continental's parent, Texas Air Corp. Mr. Lorenzo, 49 years old, is reclaiming the job that was his before Mr. Corr signed on.

The airline also named Mickey Foret as president. Mr. Foret, 44, is a 15-year veteran of Texas Air and Texas International Airlines, its predecessor. Most recently he had been executive vice president for planning and finance at Texas Air.

Top executives at Continental haven't lasted long, especially those recruited from outside. But Mr. Corr's tenure was shorter than most. **The 48-year-old Mr. Corr was hired largely because he was credited with returning Trans World Airlines Inc. to profitability while he was its president from 1986 to 1988.** Before that, he was an executive with a manufacturing concern.

Figure 7.4: Example text indicating how topic segments could be useful for information extraction. A management changeover template should be filled from the sentence in bold.

system. However, the sentence in bold is of particular interest. That sentence refers to the reason D. Joseph Corr was initially hired by Continental. However, there is no mention of Continental in the sentence. When processing this sentence, our IE system might leave the Company field blank or might mistakenly fill it with *Trans World Airlines Inc.*, another company mentioned in the sentence. However, Continental Airlines is the best guess for this field, since it is the company most frequently referred to in the text.

Given only the text below there is no need to perform topic segmentation in order to hypothesize that Continental hired Mr. Corr. But, if this information were in a longer article or were to be gleaned from the transcript of a news broadcast which lacked story boundaries then it would be advantageous to know the portion of the text to search for the most likely company. We could write a pattern in MOP to identify the most frequently mentioned company in the current topic segment and then use the name of that entity to fill empty slots of type company.

7.4.4 Topic Detection and Tracking

There is increased interest today in tracking the evolution of news stories over time by analyzing the contents of various news broadcasts and newswires. This task has two main components: computing the topical similarity of passages of text and identifying the passages which will be compared. The first component is addressed primarily using IR algorithms. The second requires the use of techniques for topic segmentation such as those proposed in this dissertation.

It might seem that topic tracking could be done without first performing topic segmentation, but the data sources may contain only minimal markup. News feeds may or may not indicate the boundaries between stories and even if they do, it may still be beneficial to segment lengthy stories. Speech-recognized broadcast news programs will have virtually no annotation regarding topic structure. In fact, the first step for news broadcasts may be separating the contents of the program from the commercials which are interspersed with it. After commercials have been removed, a topic segmentation algorithm could divide the remaining text into segments according to topic. These segments could then be fed into a topic-tracking system.

7.4.5 Automated Essay Grading

Standardized tests are routinely administered to assess people's qualifications before admitting them to educational institutions. Most standardized tests consist primarily of multiple-choice questions which are easily graded by machine. Essay questions test mastery of subjects in ways multiple-choice tests cannot, but are time-consuming and expensive to grade. Burstein *et al.* describe a system for automatically assigning scores to essays [Burstein et al., 1997]. They designed their system to replace one of two judges who traditionally score the essays in order to reduce grading time and expense.

Prior to scoring, their system divides essays into units intended to contain a single argument or point [Burstein, 1998]. They currently do this using only cue words, but a natural extension of this work is to also use word frequency information and the other cues exploited by our algorithms.[1]

[1] Thanks to Jill Burstein and Karen Kukich of Educational Testing Service for this suggestion.

Chapter 8

Conclusions

We have proposed several new indicators of topic shift and described four algorithms for dividing documents into topic segments using these indicators and other previously used clues. We proposed that topic shifts are accompanied by changes in the distribution of character n-grams and tested this hypothesis by implementing an algorithm that tracked text compression performance. We found that, despite their intuitive appeal, character n-grams are relatively uninformative about text structure.

We also devised a text segmentation algorithm that used patterns of word repetition to detect topic shifts. Although this optimization algorithm performed better than Hearst's *TextTiling* and Youmans' VMP on a topic segmentation simulation using concatenated documents, it did not do as well on the HUB-4 broadcast news corpus as our third and fourth algorithms. This algorithm is not without advantages, however. First, it can easily be applied to documents from a variety of domains and in various languages because it requires no training data. Second, it can be used in conjunction with the dotplotting visualization technique which graphically displays similarity information.

We developed a language model-based algorithm which performed well on our simulation and even better on actual data from the HUB-4 corpus. Although this method requires a training corpus for optimal performance, we showed that it performs nearly as well without training data because it exploits the burstiness of content words to locate topic segments. This model also accurately segmented text produced by a speech recognition system and, more surprisingly, was useful for segmenting Spanish text as well.

Our final technique employed a statistical model built using maximum entropy modeling software and incorporated a number of features: the output of our word frequency algorithm, the first uses of words, synonymy and several novel features including repetition of named entities and bigrams, and domain cue phrases that incorporated named entities. This model performed extremely well on the HUB-4 corpus when trained from a small number of articles and exploited domain-specific cue phrases induced from one portion of the TDT corpus to segment the TDT corpus with very high precision.

We also showed that topic segmentation techniques are useful for language modeling by demonstrating a decrease in the perplexity of a topic-based language model trained from topic-segmented data. We demonstrated the utility of topic segments for improving pronoun resolution algorithms by decreasing the average number of candidate antecedents by restricting resolutions to topic segments.

We demonstrated that indexing the topic segments identified by our text segmentation algorithms improves IR performance on the SDR collection. We also showed a sample segment that would be returned by an IR system in response to one of the queries from the SDR task. This segment contained the most relevant information in the collection and reading it would take much less time than reading the entire document it came from. This provides anecdotal evidence that text segmentation is useful for improving user interactions with IR systems as well as retrieval performance.

8.1 Future Directions

There are many possible directions for this work, some of which we pointed out in previous chapters. Performance of the best of our algorithms was state-of-the-art, but there is obviously room for improvement. We intend to refine the algorithms presented and incorporate additional segmentation cues, as well. We mentioned three such cues in Section 3.2.10: parallelism, enumerated points and text complexity. The named entity detection system we built for speech-recognized data performed well in that domain, but a system trained from data with reliable capitalization and punctuation, like the data from the TDT corpus, would be helpful for segmenting documents from similar sources.

We explored the usefulness of text structuring for language modeling, IR and coreference resolution, but there is ample room for additional work in these areas. It would be interesting to utilize structured text in a statistical coreference resolution system like Kehler's [Kehler, 1997]. We would also like to further explore language modeling applications, possibly in the context of OCR.

We discussed a number of potential applications for text segmentation in the previous chapter. We would like to incorporate our work into systems that address some of these. Summarization is an area of active research that we believe would benefit greatly from using structured text. Hypertext generation is an obvious, but nonetheless interesting, application, especially in light of the growth of the World Wide Web. We would also like to revisit our earlier work on information extraction in order to test the usefulness of structured text for constraining possible fills for some template slots.

We feel that the most interesting and exciting work lies in incorporating the textual features used in this work with those present in the audio and video portions of multimedia documents. We would like to combine work on the relationship between intonation, pauses and discourse structure with the textual features we used to produce a better segmentation system. We further intend to use video cues, such as scene transitions and fades to black, which commonly occur in news broadcasts, to additionally improve segmentation. Ultimately, we would like to build an integrated model encompassing video, audio and textual cues and test it in the context of a real-world video-on-demand application.

We would also like to apply our segmentation techniques to transcripts of dialogues, such as those from the Switchboard corpus. So far we have focused exclusively on types of documents containing primarily planned speech. Segmenting dialogues will provide an additional set of challenges such as speech disfluencies, interruptions of topics followed by returns and more frequent, less well-defined topic shifts than news broadcasts possess. Nonetheless, we would like to apply these techniques to spontaneous speech and study information retrieval on documents of this type.

Bibliography

[Aone et al., 1997] Aone, C., Okurowski, M. E., Gorlinsky, J., and Larsen, B. (1997). A scalable summarization system using robust NLP. In Mani, I. and Maybury, M., editors, *Proceedings of the Workshop on Intelligent Scalable Text Summarization*, pages 66–73, Madrid.

[Bahl et al., 1977] Bahl, L. R., Baker, J. K., Jelinek, F., and Mercer, R. L. (1977). Perplexity—a measure of the difficulty of speech recognition tasks. In 94^{th} *Meeting of the Acoustical Society of America,* in *Journal of the Acoustical Society of America*, volume 62, page S63.

[Bahl et al., 1983] Bahl, L. R., Jelinek, F., and Mercer, R. L. (1983). A maximum likelihood approach to continuous speech recognition. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 5(2):179–190.

[Baldwin, 1997] Baldwin, B. (1997). CogNIAC: High precision coreference with limited knowledge and linguistic resources. In *Proceedings of a workshop on Operational Factors in Practical, Robust Anaphora Resolution for Unrestricted Texts*, pages 38–45, Madrid.

[Baldwin et al., 1997] Baldwin, B., Doran, C., Reynar, J. C., Niv, M., Srinivas, B., and Wasson, M. (1997). EAGLE: An extensible architecture for general linguistic engineering. In *Proceedings of RIAO-97*, pages 271–283, Montreal.

[Baldwin et al., 1995] Baldwin, B., Reynar, J., Collins, M., Eisner, J., Ratnaparkhi, A., Rosenzweig, J., Sarkar, A., and Srinivas, B. (1995). University of Pennsylvania: Description of the University of Pennsylvania system used for MUC-6. In *Proceedings of*

the Sixth Message Understanding Conference, pages 177–192, San Francisco. Defense Advanced Research Projects Agency, Morgan Kaufmann.

[Beeferman et al., 1997a] Beeferman, D., Berger, A., and Lafferty, J. (1997a). A model of lexical attraction and repulsion. In *Proceedings of the* 35^{th} *Annual Meeting of the Association for Computational Linguistics*, pages 373–380, Madrid.

[Beeferman et al., 1997b] Beeferman, D., Berger, A., and Lafferty, J. (1997b). Text segmentation using exponential models. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 35–46, Providence, Rhode Island.

[Bell et al., 1990] Bell, T. C., Cleary, J. G., and Witten, I. H. (1990). *Text Compression.* Advanced Reference Series. Prentice Hall, Englewood Cliffs, New Jersey.

[Berber Sardinha, 1993a] Berber Sardinha, A. P. (1993a). Lexis in annual reports: Text segmentation and lexical threads. Technical Report 8, Development of International Research in English for Commerce and Technology.

[Berber Sardinha, 1993b] Berber Sardinha, A. P. (1993b). Lexis in annual reports: the cluster triangle technique. Technical Report 2, Development of International Research in English for Commerce and Technology.

[Berger et al., 1996] Berger, A., Della Pietra, S. A., and Della Pietra, V. J. (1996). A Maximum Entropy Approach to Natural Language Processing. *Computational Linguistics*, 22(1):39–71.

[Biber, 1989] Biber, D. (1989). A typology of English texts. *Linguistics*, 27:3–43.

[Biber, 1990] Biber, D. (1990). Methodological issues regarding corpus-based analyses of linguistic variation. *Literary and Linguistic Computing*, 5(4):257–269.

[Bikel et al., 1997] Bikel, D. M., Miller, S., Schwartz, R., and Weischedel, R. (1997). Nymble: a high-performance learning name-finder. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 194–201, Washington, D.C.

[Brown et al., 1990a] Brown, P. F., Cocke, J., Pietra, S. A. D., Pietra, V. J. D., Jelinek, F., Lafferty, J. D., Mercer, R. L., and Roosin, P. S. (1990a). A statistical approach to machine translation. *Computational Linguistics*, 16(2):79–85.

[Brown et al., 1990b] Brown, P. F., Pietra, V. J. D., deSouza, P. V., Lai, J. C., and Mercer, R. L. (1990b). Class-based n-gram models of natural language. In *Proceedings of the IBM Natural Language ITL*, Paris.

[Buckley, 1985] Buckley, C. (1985). Implementation of the SMART information retrieval system. Technical Report Technical Report 85-686, Cornell University.

[Burstein, 1998] Burstein, J. (1998). Personal communication.

[Burstein et al., 1997] Burstein, J., Wolff, S., Lu, C., and Kaplan, R. (1997). An automatic scoring system for advanced placement biology essays. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 174–181, Washington, D.C.

[Callan, 1994] Callan, J. P. (1994). Passage-level evidence in document retrieval. In *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 302–310, Dublin, Ireland. Association for Computing Machinery.

[Callan et al., 1992] Callan, J. P., Croft, W. B., and Harding, S. M. (1992). The INQUERY retrieval system. In *Proceedings of the Third International Conference on Database and Expert Systems Applications*, pages 78–83, Valencia, Spain. Springer Verlag.

[Carletta, 1996] Carletta, J. (1996). Assessing agreement on classification tasks: The kappa statistic. *Computational Linguistics*, 22(2):249–254.

[Chafe, 1974] Chafe, W. L. (1974). Language and consciousness. *Language*, 50:111–133.

[Chinchor, 1997] Chinchor, N. (1997). MUC-7 named entity task definition, dry run version, version 3.5. Documentation for the Seventh Message Understanding Conference.

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

[Church, 1988] Church, K. W. (1988). A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the Second Conference on Applied Natural Language Processing*, pages 136–143.

[Church, 1993] Church, K. W. (1993). Char_align: A program for aligning parallel texts at the character level. In *Proceedings of the* 31^{st} *Annual Meeting of the Association for Computational Linguistics*, pages 1–8.

[Church and Gale, 1995a] Church, K. W. and Gale, W. A. (1995a). Inverse document frequency (IDF): A measure of deviations from Poisson. In Yarowsky, D. and Church, K., editors, *Proceedings of the Third Workshop on Very Large Corpora*, pages 121–130. Association for Computational Linguistics.

[Church and Gale, 1995b] Church, K. W. and Gale, W. A. (1995b). Poisson mixtures. *Journal of Natural Language Engineering*, 1(2):163–190.

[Cover and Thomas, 1991] Cover, T. and Thomas, J. A. (1991). *Elements of Information Theory*. John Wiley & Sons, New York.

[Dahlgren, 1996] Dahlgren, K. (1996). Discourse coherence and segmentation. In Hovy, E. H. and Scott, D. R., editors, *Computational and Conversational Discourse: Burning Issues—An Interdisciplinary Account*, chapter 5, pages 111–138. Springer Verlag, Berlin.

[DiEugenio et al., 1997] DiEugenio, B., Moore, J., and Paolucci, M. (1997). Learning features that predict cue usage. In *Proceedings of the* 35^{th} *Annual Meeting of the Association for Computational Linguistics*, pages 80–87, Madrid.

[Digital Equipment Corporation, 1997] Digital Equipment Corporation (1997). About alta vista. http://www.altavista.digital.com/av/content/about_our_story.htm.

[Doran et al., 1996] Doran, C., Niv, M., Baldwin, B., Reynar, J. C., Srinivas, B., and Wasson, M. (1996). Mother of Perl: A multi-tier pattern description language. In *Proceedings of the International Workshop on Lexically Driven Information Extraction (LDIE97)*, Frescati, Italy.

[Dunning, 1994] Dunning, T. (1994). Statistical identification of language. Technical Report CRL Technical Memo MCCS-94-273, University of New Mexico.

[Food and Agriculture Organization of the United Nations, 1995] Food and Agriculture Organization of the United Nations (1995). WWW database available at http://apps.fao.org.

[Fox et al., 1995] Fox, E. A., Akscyn, R. M., Furuta, R. K., and Leggett, J. L. (1995). Digital libraries. *Communications of the ACM*, 38(4):23–28.

[Gale et al., 1992] Gale, W., Church, K., and Yarowsky, D. (1992). One sense per discourse. In *Proceedings of the Fourth DARPA Speech and Natural Language Workshop*, pages 233–237.

[Gale and Sampson, 1995] Gale, W. and Sampson, G. (1995). Good-Turing smoothing without tears. *Journal of Quantitative Linguistics*, 2.

[Good, 1953] Good, I. J. (1953). The population frequencies of species and the estimation of population parameters. *Biometrika*, 40(3–4):237–264.

[Grimes, 1975] Grimes, J. E. (1975). *The Thread of Discourse*. Mouton, The Hague.

[Grosz and Hirschberg, 1992] Grosz, B. J. and Hirschberg, J. (1992). Some intonational characteristics of discourse structure. In *International Conference on Spoken Language Processing*, pages 429–432.

[Grosz et al., 1995] Grosz, B. J., Joshi, A. K., and Weinstein, S. (1995). Centering: A framework for modeling the local coherence of discourse. *Computational Linguistics*, 21(2):203–226.

[Grosz and Sidner, 1986] Grosz, B. J. and Sidner, C. L. (1986). Attention, intentions and the structure of discourse. *Computational Linguistics*, 12(3):175–204.

[Gunning, 1952] Gunning, R. (1952). *The Technique of Clear Writing*. McGraw-Hill, New York.

[Halliday and Hasan, 1976] Halliday, M. and Hasan, R. (1976). *Cohesion in English.* Longman Group, New York.

[Harmon, 1992] Harmon, D. K. (1992). Overview of the first text retrieval conference. In Harmon, D. K., editor, *Proceedings of the TREC Text Retrieval Conference*, pages 1–20, Washington.

[Harris, 1952] Harris, Z. S. (1952). Discourse analysis. *Language*, 28(1):1–30.

[Harter, 1992] Harter, S. P. (1992). Psychological relevance and information science. *Journal of the American Society for Information Science*, 43(9):602–615.

[Hauptmann and Witbrock, 1997] Hauptmann, A. G. and Witbrock, M. J. (1997). Informedia: News-on-demand multimedia information acquisition and retrieval. In Maybury, M. T., editor, *Intelligent Multimedia Information Retrieval*, chapter 11, pages 215–239. AAAI Press and MIT Press, Menlo Park, California.

[Hearst, 1993] Hearst, M. A. (1993). TextTiling: A quantitative approach to discourse segmentation. Technical Report 93/24, University of California, Berkeley.

[Hearst, 1994a] Hearst, M. A. (1994a). *Context and Structure in Automated Full-Text Information Access.* PhD thesis, University of California, Berkeley.

[Hearst, 1994b] Hearst, M. A. (1994b). Multi-paragraph segmentation of expository text. In *Proceedings of the* 32^{nd} *Annual Meeting of the Association for Computational Linguistics*, pages 9–16, Las Cruces, New Mexico.

[Hearst and Plaunt, 1993] Hearst, M. A. and Plaunt, C. (1993). Subtopic structuring for full-length document access. In *Proceedings of the Special Interest Group on Information Retrieval*, pages 59–68.

[Helfman, 1994] Helfman, J. I. (1994). Similarity patterns in language. In *IEEE Symposium on Visual Languages.*

[Hinds, 1979] Hinds, J. (1979). Organizational patterns in discourse. In *Discourse Analysis*, volume 12 of *Syntax and Semantics*, pages 135–157. Academic Press, New York.

[Hirschberg and Grosz, 1992] Hirschberg, J. and Grosz, B. (1992). Intonational features of local and global discourse. In *Proceedings of the Workshop on Spoken Lanuage Systems*, pages 441–446. DARPA.

[Hirschberg and Litman, 1993] Hirschberg, J. and Litman, D. (1993). Empirical studies on the disambiguation of cue phrases. *Computational Linguistics*, 19(3):501–530.

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

[Hobbs, 1979] Hobbs, J. R. (1979). Coherence and coreference. *Cognitive Science*, 3:67–90.

[Hobbs, 1983] Hobbs, J. R. (1983). Why is a discourse coherent? In Neubauer, F., editor, *Coherence in Natural Language Texts*, Papers in Textlinguistics, pages 29–70. Buske, Hamburg.

[HUB-4 Program Committee, 1996] HUB-4 Program Committee (1996). The 1996 HUB-4 annotation specification for evaluation of speech recognition on broadcast news, version 3.5.

[Hull, 1992] Hull, J. J. (1992). A hidden Markov model for language syntax in text recognition. In *Proceedings of the Eleventh Conference on Pattern Recognition*, volume 2, pages 124–127.

[Hurtig, 1977] Hurtig, R. (1977). Toward a functional theory of discourse. In Feedle, R. O., editor, *Discourse Processes: Advances in Research and Theory*, volume I: Discourse Production and Comprehension, chapter 4, pages 89–106. Ablex Publishing, Norwood, New Jersey.

[Justeson and Katz, 1995] Justeson, J. S. and Katz, S. M. (1995). Technical terminology: Some linguistic properties and an algorithm for identification in text. *Natural Language Engineering*, 1(1):9–27.

[Karp et al., 1992] Karp, D., Schabes, Y., Zaidel, M., and Egedi, D. (1992). A freely available wide coverage morphological analyzer for English. In *Proceedings of the* 15^{th} *International Conference on Computational Linguistics*, Nantes, France.

[Kaszkiel and Zobel, 1997] Kaszkiel, M. and Zobel, J. (1997). Passage retrieval revisited. In *Proceedings of the ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 178–185, Philadelphia. ACM.

[Katz, 1996] Katz, S. M. (1996). Distribution of content words and phrases in text and language modeling. *Natural Language Engineering*, 2(1):15–59.

[Kehler, 1997] Kehler, A. (1997). Probabilistic coreference in information extraction. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, Providence, Rhode Island.

[Kessler et al., 1997] Kessler, B., Nunberg, G., and Schuetze, H. (1997). Automatic detection of text genre. In *Proceedings of the* 35^{th} *Annual Meeting of the Association for Computational Linguistics*, pages 32–38, Madrid.

[Kozima, 1993] Kozima, H. (1993). Text segmentation based on similarity between words. In *Proceedings of the* 31^{st} *Annual Meeting of the Association for Computational Linguistics, Student Session*, pages 286–288.

[Kozima and Furugori, 1993] Kozima, H. and Furugori, T. (1993). Similarity between words computed by spreading activation on an English dictionary. In *Proceedings of the European Association for Computational Linguistics*, pages 232–239.

[Kozima and Furugori, 1994] Kozima, H. and Furugori, T. (1994). Segmenting narrative text into coherent scenes. *Literary and Linguistic Computing*, 9(1):13–19.

[Krupka, 1995] Krupka, G. R. (1995). SRA: Description of the sra system as used for MUC-6. In *Proceedings of the Sixth Message Understanding Conference*, pages 221–336, San Francisco. Morgan Kaufmann.

[Kukich, 1992] Kukich, K. (1992). Techniques for automatically correcting words in texts. *ACM Computing Surveys*, 24(4):377–439.

[Kučera and Francis, 1967] Kučera, H. and Francis, W. N. (1967). *Computational Analysis of Present-Day American English.* Brown University Press, Providence.

[Lafferty, 1998] Lafferty, J. (1998). Personal communication.

[Lancaster, 1978] Lancaster, F. W. (1978). *Towards Paperless Information Systems.* Academic Press, New York.

[Lau et al., 1993] Lau, R., Rosenfeld, R., and Roukos, S. (1993). Adaptive language modeling using the maximum entropy principle. In *Proceedings of the ARPA Human Language Technology Workshop*, pages 110–113.

[Levy, 1984] Levy, E. T. (1984). *Communicating Thematic Structure in Narrative Discourse: The Use of Referring Terms and Gestures.* PhD thesis, University of Chicago, Chicago, Illinois.

[Lexis-Nexis, 1998] Lexis-Nexis (1998). The lexis-Nexis home page. http:/www/lexis-nexis.com.

[Library of Congress, 1997] Library of Congress (1997). The Library of Congress: Acquisitions. http://lcweb.loc.gov/acq/acquire.html.

[Longacre, 1979] Longacre, R. E. (1979). The paragraph as a grammatical unit. In *Discourse Analysis*, volume 12 of *Syntax and Semantics*, pages 115–134. Academic Press, New York.

[Longacre, 1983] Longacre, R. E. (1983). *The Grammar of Discourse.* Topics in Language and Linguistics. Plenum Press, New York.

[Losee, 1996] Losee, R. M. (1996). Text windows and phrases differing by discipline, location in document and syntactic structure. *Information Processing and Management*, 32(6):747–767.

[Luhn, 1958] Luhn, H. (1958). The automatic creation of literature abstracts. *I.B.M. Journal of Research Development*, 2(2):159–165.

[Manabu and Takeo, 1994] Manabu, O. and Takeo, H. (1994). Word sense disambiguation and text segmentation based on lexical cohesion. In *Proceedings of the* 15^{th} *International Conference on Computational Linguistics*, pages 755–761.

[Mann and Thompson, 1988] Mann, W. C. and Thompson, S. A. (1988). Rhetorical structure theory: A theory of text organization. In Polanyi, L., editor, *The Structure of Discourse*. Ablex, Norwood, N.J. Also available as ISI/RR-87-190, June 1987.

[Marcus et al., 1993] Marcus, M., Santorini, B., and Marcinkiewicz, M. A. (1993). Building a large annotated corpus of English: the Penn Treebank. *Computational Linguistics*, 19(2):313–330.

[Miller et al., 1990] Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. (1990). Five papers on WordNet. Technical report, Cognitive Science Laboratory, Princeton University.

[Mittendorf and Schäuble, 1994] Mittendorf, E. and Schäuble, P. (1994). Document and passage retrieval based on hidden Markov models. In *Proceedings of the ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 318–327, Dublin, Ireland. ACM.

[Morris, 1988] Morris, J. (1988). Lexical cohesion, the thesaurus, and the structure of text. Technical Report CSRI-219, Computer Systems Research Institute, University of Toronto.

[Morris and Hirst, 1991] Morris, J. and Hirst, G. (1991). Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics*, 17(1):21–42.

[Mosteller and Wallace, 1964] Mosteller, F. and Wallace, D. L. (1964). *Applied Bayesian and Classical Inference: The Case of* The Federalist *Papers*. Springer Series in Statistics. Springer Verlag, New York.

[MUC-6 Program Committee, 1995] MUC-6 Program Committee (1995). Information extraction task definition version 2.1. In *Proceedings of the Sixth Message Understanding Conference*, pages 345–360, San Francisco. Morgan Kaufmann.

[Nakatani et al., 1995] Nakatani, C. H., Grosz, B. J., Ahn, D. D., and Hirschberg, J. (1995). Instructions for annotating discourses. Technical Report TR-21-95, Center for Research in Computing Technology, Harvard University, Cambridge, MA.

[Nakhimovsky, 1988] Nakhimovsky, A. (1988). Aspect, aspectual class, and the temporal structure of narrative. *Computational Linguistics*, 14(2):29–43.

[Nomoto and Nitta, 1994] Nomoto, T. and Nitta, Y. (1994). Grammatico-statistical approach to discourse partitioning. In *Proceedings of COLING-94*, pages 1145–1150, Kyoto, Japan.

[Paice, 1990] Paice, C. D. (1990). Constructing literature abstracts by computer: Techniques and prospects. *Information Processing & Management*, 26(1):171–186.

[Passoneau and Litman, 1993] Passoneau, R. J. and Litman, D. J. (1993). Intention-based segmentation: Human reliability and correlation with linguistic cues. In *Proceedings of the* 31^{st} *Meeting of the Association for Computational Linguistics*, pages 148–155.

[Passoneau and Litman, 1996] Passoneau, R. J. and Litman, D. J. (1996). Empirical analysis of three dimensions of spoken discourse: Segmentation, coherence and linguistic devices. In Hovy, E. H. and Scott, D. R., editors, *Computational and Conversational Discourse: Burning Issues—An Interdisciplinary Account*, chapter 7, pages 161–194. Springer Verlag, Berlin.

[Phillips, 1985] Phillips, M. (1985). *Aspects of Text Structure: An Investigation of the Lexical Organisation of Text*. North Holland Linguistic Series. North Holland, Amsterdam.

[Polanyi, 1988] Polanyi, L. (1988). A formal model of the structure of discourse. *Journal of Pragmatics*, 12:601–638.

[Ponte and Croft, 1997] Ponte, J. M. and Croft, W. B. (1997). Text segmentation by topic. In *European Conference on Digital Libraries*, pages 113–125, Pisa, Italy.

[Porter, 1980] Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, 14:130–137.

[Ramalho and Mammone, 1994] Ramalho, M. A. and Mammone, R. J. (1994). A new speech enhancement technique with application to speaker identification. In *Proc. ICASSP '94*, pages I–29 – I–32, Adelaide, Austrailia.

[Ratnaparkhi, 1996] Ratnaparkhi, A. (1996). A maximum entropy model for part-of-speech tagging. In *Proceedings of the First Conference on Empirical Methods in Natural Language Processing*, pages 133–142, University of Pennsylvania.

[Ratnaparkhi, 1997a] Ratnaparkhi, A. (1997a). A linear observed time statistical parser based on maximum entropy models. In *Proceedings of the Second Conference on Empirical Methods in Natural Language Processing*, pages 81–89, Providence, Rhode Island.

[Ratnaparkhi, 1997b] Ratnaparkhi, A. (1997b). A simple introduction to maximum entropy models for natural language processing. 97-08, Institute for Research in Cognitive Science, University of Pennsylvania, Philadelphia.

[Reichman, 1981] Reichman, R. (1981). *Plain-speaking: A Theory and Grammar of Spontaneous Discourse*. PhD thesis, Harvard University, Department of Computer Science.

[Reynar, 1994] Reynar, J. C. (1994). An automatic method of finding topic boundaries. In *Proceedings of the* 32^{nd} *Annual Meeting of the Association for Computational Linguistics, Student Session*, pages 331–333, Las Cruces, New Mexico.

[Reynar and Ratnaparkhi, 1997] Reynar, J. C. and Ratnaparkhi, A. (1997). A maximum entropy approach to identifying sentence boundaries. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 16–19, Washington, D.C.

[Richmond et al., 1997] Richmond, K., Smith, A., and Amitay, E. (1997). Detecting subject boundaries within text: A language independent statistical approach. In *Exploratory Methods in Natural Language Processing*, pages 47–54, Providence, Rhode Island.

[Roget, 1911] Roget, P. M. (1911). *Roget's International Thesaurus*. Cromwell, New York, first edition.

[Roget, 1977] Roget, P. M. (1977). *Roget's International Thesaurus*. Harper and Row, New York, fourth edition.

[Rosenfeld and Huang, 1992] Rosenfeld, R. and Huang, X. (1992). Improvements in stochastic language modeling. In *Proceedings of the DARPA Speech and Human Language Technology Workshop*, San Mateo, California. Morgan Kaufmann.

[Rotondo, 1984] Rotondo, J. A. (1984). Clustering analysis of subjective partitions of text. *Discourse Processes*, 7:69–88.

[Salton and Allan, 1993] Salton, G. and Allan, J. (1993). Selective text utilization and text traversal. In *Hypertext-93*, New York. A.C.M.

[Salton et al., 1993] Salton, G., Allan, J., and Buckley, C. (1993). Approaches to passage retrieval in full text information systems. In Korfhagge, R., Rasmussen, E., and Willett, P., editors, *Proceedings of the Sixteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 49–58, Pittsburgh, PA. Association for Computing Machinery.

[Salton and Buckley, 1992] Salton, G. and Buckley, C. (1992). Automatic text structuring experiments. In Jacobs, P. S., editor, *Text-Based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, pages 199–210. Lawrence Erlbaum Associates, Hillsdale, New Jersey.

[Sekine et al., 1995] Sekine, S., Sterling, J., and Grishman, R. (1995). NYU/BBN 1994 CSR evaluation. In *Proceedings of the Workshop on Spoken Lanuage Systems Technology*. DARPA.

[Sibun, 1992] Sibun, P. (1992). Generating text without trees. *Computational Intelligence*, 8(1):102–122.

[Singhal et al., 1996] Singhal, A., Buckley, C., and Mitra, M. (1996). Pivoted document length normalization. In *Proceedings of the ACM-SIGIR Conference on Research and Development in Information Retrieval*, pages 21–29, Zurich, Switzerland. ACM.

[Skorochod'ko, 1972] Skorochod'ko, E. (1972). Adaptive method of automatic abstracting and indexing. *Information Processing*, 71:1179–1182.

[Sparck Jones, 1994] Sparck Jones, K. (1994). Natural language processing: She needs something old and something new (maybe something borrowed and something blue, too). Association for Computational Linguistics Presidential Address.

[Stanfill and Waltz, 1992] Stanfill, C. and Waltz, D. L. (1992). Statistical methods, artificial intelligence, and information retrieval. In Jacobs, P. S., editor, *Text-based Intelligent Systems: Current Research and Practice in Information Extraction and Retrieval*, pages 215–226. Lawrence Erlbaum Associates, Hillsdale, New Jersey.

[Stark, 1988] Stark, H. (1988). What do paragraph markings do? *Discourse Processes*, 11:275–303.

[Suen, 1979] Suen, C. Y. (1979). N-gram statistics for natural language understanding and text processing. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1(2):164–172.

[van der Eijk, 1994] van der Eijk, P. (1994). Comparative discourse analysis of parallel texts. In *Proceedings of the Second Workshop on Very Large Corpora*, Kyoto.

[Walker and Whittaker, 1990] Walker, M. and Whittaker, S. (1990). Mixed initiative in dialogue: An investigation into discourse segmentation. In *Proceedings of the* 28^{th} *Annual Meeting of the Association for Computational Linguistics*, pages 70–78.

[Webber, 1991] Webber, B. L. (1991). Structure and ostension in the interpretation of discourse deixis. *Language and Cognitive Processes*, 6(2):107–135.

[XTAG Group, 1995] XTAG Group (1995). A lexicalized tree adjoining grammar for english. Technical Report IRCS 95-03, University of Pennsylvania.

[Xu and Croft, 1996] Xu, J. and Croft, W. B. (1996). Query expansion using local and global document analysis. In *Proceedings of the Nineteenth Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 4–11, Zurich, Switzerland.

[Yaari, 1997] Yaari, Y. (1997). Segmentation of expository texts by hierarchical agglomerative clustering. In *Proceedings of Recent Advances in Natural Language Processing*, Bulgaria.

[Yarowsky, 1992] Yarowsky, D. (1992). Word-sense disambiguation using statistical models of Roget's categories trained on large corpora. In *Proceedings of COLING-92*, pages 454–460, Nantes, France.

[Yeo and Liu, 1995] Yeo, B.-L. and Liu, B. (1995). Rapid scene analysis on compressed video. *IEEE Transactions on Circuits and Systems for Video Technology*, 5(6):533–544.

[Youmans, 1990] Youmans, G. (1990). Measuring lexical style and competence: The type-token vocabulary curve. *Style*, 24:584–599.

[Youmans, 1991] Youmans, G. (1991). A new tool for discourse analysis: The vocabulary management profile. *Language*, 67(4):763–789.

[Ziv and Lempel, 1977] Ziv, J. and Lempel, A. (1977). A universal algorithm for sequential data compression. *IEEE Transactions on Information Theory*, 23(3):337–343.