Minimum Cut Model for Spoken Lecture Segmentation

by

Igor Malioutov

B.S., Northeastern University (2004)

Submitted to the Department of Electrical Engineering and Computer Science
in partial fulfillment of the requirements for the degree of

Master of Science

at the

MASSACHUSETTS INSTITUTE OF TECHNOLOGY

November 2006

Author . . . .
Department of Electrical Engineering and Computer Science
November 29, 2006

Certified by . . . .
Regina Barzilay
Assistant Professor
Thesis Supervisor

Accepted by . . . .
Arthur C. Smith
Chairman, Department Committee on Graduate Students

Minimum Cut Model for Spoken Lecture Segmentation

by

Igor Malioutov

Submitted to the Department of Electrical Engineering and Computer Science
on November 29, 2006, in partial fulfillment of the
requirements for the degree of
Master of Science

Abstract

We introduce a novel unsupervised algorithm for text segmentation. We re-conceptualize text segmentation as a graph-partitioning task aiming to optimize the normalized-cut criterion. Central to this framework is a contrastive analysis of lexical distribution that simultaneously optimizes the total similarity within each segment and dissimilarity across segments.

Our experimental results show that the normalized-cut algorithm obtains performance improvements over the state-of-the-art techniques on the task of spoken lecture segmentation. Another attractive property of the algorithm is robustness to noise. The accuracy of our algorithm does not deteriorate significantly when applied to automatically recognized speech. The impact of the novel segmentation framework extends beyond the text segmentation domain. We demonstrate the power of the model by applying it to the segmentation of raw acoustic signal without intermediate speech recognition.

Thesis Supervisor: Regina Barzilay
Title: Assistant Professor

Acknowledgments

I would like to thank my advisor Prof. Regina Barzilay for her invaluable guidance through the course of my research endeavors and for granting me ample freedom to pursue my wide-ranging interests.

This work bore much of its fruit from an ongoing collaboration with T.J. Hazen, Scott Cyphers, Jim Glass, and most recently Alex Park at the Speech and Language Systems Group. I am grateful to David Karger for an illuminating discussion on Minimum Cuts. I also would like to thank Alex Gruenstein for sharing his valuable insights on the segmentation problem and providing the NOMOS annotation toolkit for the segmentation study.

Many thanks also to Michael Collins, Jacob Eisenstein, Terry Koo, John Lee, Zvika Marx, and Eli Barzilay for providing critical feedback on the various drafts of the ACL paper and to Mirella Lapata for hosting my visit at the University of Edinburgh.

I am especially grateful to have had an opportunity to work alongside and share ideas with some of the Ph.D students in the lab. Paul Hsu, Gabriel Zaccak, and Luke Zettlemoyer have formed the kernel of an informal scientific and technical board for the last two and a half years, influencing much of my understanding of statistical NLP and inspiring me with many of their ideas. My graduate life would not be complete without my fellow students and researchers in the group - Kevin Simler, Philip Bramsen, Ben Snyder, Pawan Deshpande, and Yoong Keok Lee, who enriched my imagination with their fascinating daily exploits and endless philosophical debates. I am also thankful for the continuing friendship of fellow CSAILers - Harr (the Chairman) Chen, Ali Mohammad, Natasha Singh, Rajvinder Singh, Federico Mora, Yuan Shen, and Tara Sainath.

Finally, I would like to thank Kate Nyhan for keeping me revitalized with her critical sense of humor and her effusive charm through the roller coaster of graduate student life. I am also deeply indebted to my family for providing me with a sense of purpose and unfaltering support.

The work described in this thesis was supported in part by the National Science Foundation, under CAREER grant IIS-0448168, grant IIS- 0415865, and the NSF Graduate Fellowship.

Bibliographic Notes

Portions of this thesis are based on the paper "Minimum Cut Model for Spoken Lecture Segmentation" with Regina Barzilay (Malioutov and Barzilay, 2006), which appeared in the Proceedings of the 44th meeting of the Association for Computational Linguistics.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The limits of my language are the limits of my mind.

-Ludwig Wittgenstein

Natural language understanding is arguably one of the most compelling scientific frontiers, only now beginning to be probed through advances in statistical natural language processing, machine learning, linguistics, and cognitive science. In this thesis, we address one of the structural pieces in the required scaffolding, the problem of text segmentation.

The task is to partition a text into a linear sequence of topically coherent segments and thereby induce a content structure of the document. Apart from laying the groundwork for the development of more realistic semantic models for natural language understanding, the immediate applications of the derived structural information are broad, encompassing information retrieval, question-answering, and text summarization.

1.1 Problem Motivation

Text segmentation is an active area of research in natural language processing. However, until recently, much of the work has been hampered by strong oversimplifying assumptions about the distributional properties of the data, the availability of certain structural information such as paragraph and sentence boundaries, and artificial restrictions on the language domain. These assumptions have undercut the effectiveness of the models in more challenging contexts.

A critical dimension that has received relatively little attention is the distinction between

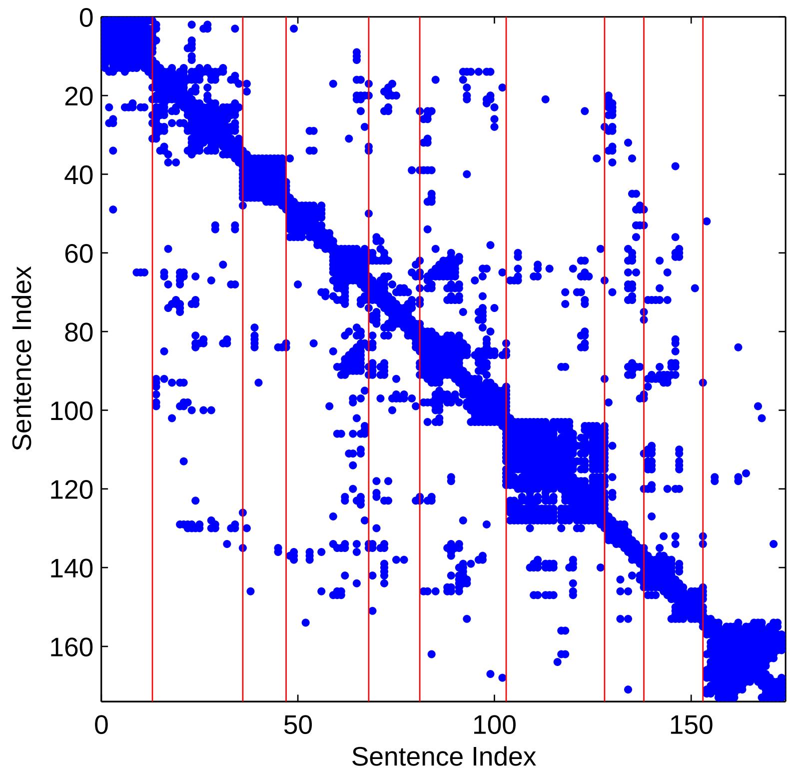

Figure 1-1: Synthetic Text Similarity Plot

topic and sub-topic segmentation. A substantial portion of the work on segmentation addresses the problem of recovering documents or fragments of different documents from a stream of concatenated texts. In this case, the definition of a topic boundary is clear-cut, because it corresponds to a document boundary. There are real-world problems where this scenario is relevant. For example, research work has been conducted on broadcast news segmentation, where the goal is to partition the broadcast news transcripts into a set of distinct news segments (Beeferman et al., 1999; Allan et al., 1998). In more challenging domains, such as spoken language segmentation, however, segmentation has to be executed at the level of a sub-topic. This new objective makes it much more difficult to develop effective models and also be able to evaluate these models, since the concept of a sub-topic is much more fluid.

Following the first unsupervised segmentation approach by Hearst (1994), most approaches assume that variations in lexical distribution indicate topic changes. When docu-

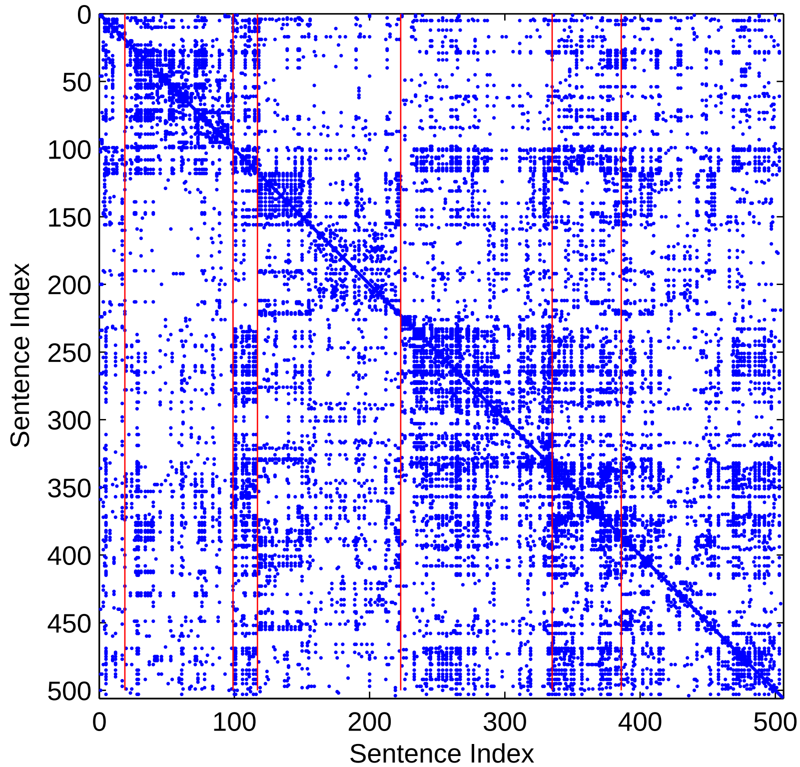

Figure 1-2: Spoken Lecture Transcript Similarity Plot

ments exhibit sharp variations in lexical distribution, these algorithms are likely to detect segment boundaries accurately. For example, most algorithms achieve high performance on synthetic collections, generated by concatenation of random text blocks (Choi, 2000). The difficulty arises, however, when transitions between topics are smooth and distributional variations are subtle. Consider, for example, the pairwise sentence similarity plots in Figures 1-1 and 1-2, computed for a synthetic text and a spoken lecture transcript, where vertical lines indicate true segment boundaries. For clarity, in both of these plots only the cosine similarity scores above the 90-th quantile were plotted. Clearly, the synthetic text exhibits much more sharp transitions, while there is considerable lexical overlap between segments in spoken language. This discrepancy is evident in the performance of existing unsupervised algorithms on less structured datasets, such as spoken meeting transcripts (Galley et al., 2003). Therefore, a more refined analysis of lexical distribution is needed.

Past models have typically been evaluated on written language or clean transcribed data. It is not clear whether these models will be able to tolerate transcription errors and spoken language irregularities. Segmentation in the spoken language domain is challenging in several respects. Being less structured than written text, speech transcripts exhibit digressions, disfluencies, and other artifacts of spontaneous communication. In addition, the output of speech recognizers is fraught with high word error rates due to specialized technical vocabulary and lack of in-domain spoken data for training.

In order to be able to segment transcripts of speech, it is also necessary to cast off assumptions about available structural information. The segmentation approach by Hearst (1994), for example, requires paragraph structure. Many of the other unsupervised and supervised models require sentence-level segmentation. In the spoken language domain these extra sources of information are not available.

In this thesis, we address these limitations by effectively expanding the coverage of unsupervised segmentation models to new domains, while advancing the state-of-the-art in text segmentation.

1.2 Our Approach

Most of the past unsupervised segmentation algorithms rest on intuitive notions of similarity density. In this thesis, we formalize the empirical basis for segmentation by casting text segmentation in a graph-theoretic framework. We abstract a text into a weighted undirected graph, where the nodes of the graph correspond to sentences and edge weights represent the pairwise sentence similarity. In this framework, text segmentation corresponds to a graph partitioning that optimizes the normalized-cut criterion (Shi and Malik, 2000). In contrast to previous approaches, the homogeneity of a segment is determined not only by the similarity of its words, but also by their relation to words in other segments of the text. Thus, our approach moves beyond localized comparisons and takes into account long-range variations in lexical distribution. Global analysis enables us to detect subtle topical changes, yielding more accurate segmentation results than local models.

1.3 Contributions

Below, we summarize the main contributions of our thesis.

- We formalize the text segmentation objective in a general, principled framework. With this objective we are able to model the global characteristics of the lexical distribution and simultaneously maximize within-segment similarity and minimize between-cluster similarity, merging the strengths of different unsupervised approaches to segmentation.

- We attain the new state-of-the-art results in spoken lecture segmentation. In contrast to much of the other work on unsupervised segmentation, we evaluate our algorithm on a corpus of spoken lectures, with more subtle lexical variations. Our experiments demonstrate that the minimum-cut segmentation approach yields superior performance when compared to other state-of-the-art segmentation algorithms in the spoken lecture domain. We outperform the method of Utiyama and Isahara (2001) by 9% P_k measure and the method of Choi (2000) by 24.4% P_k measure.

- Another attractive property of the algorithm is robustness to noise. The accuracy of our algorithm does not deteriorate significantly when applied to automatically recognized speech.

- The impact of our novel segmentation framework extends beyond the text segmentation domain. We demonstrate the power of the model, by applying it to the segmentation of raw acoustic signal. We represent the acoustic signal by an inter-word-fragment acoustic similarity matrix, and partition the resulting similarity matrix with the Minimum Cut segmentation algorithm.

1.4 Thesis Overview

This thesis is organized as follows. In the next chapter we provide an overview of linguistic theory with connections to the segmentation problem. We review existing work on supervised and unsupervised approaches to text segmentation as well as related approaches in vision segmentation.

We introduce the minimum cut algorithm in chapter 3. We first formulate the minimum cut problem, and then describe how it can be applied naturally to the text segmentation task. Finally, we flesh out the implementation details for the text segmentation system based on the Minimum Cut model.

In chapter 4, we analyze the performance of the minimum cut algorithm on spoken lecture data and compare our system with other state-of-the-art text segmentation systems. First, we explain the evaluation metrics used in our analysis and the human agreement results on the data. Then we examine the effect of long-range lexical dependencies employed by the model. In order to gauge its effectiveness, we compare our system with other leading segmentation systems on synthetic and spoken lecture data-sets. We also examine the effect of speech recognition error on segmentation accuracy. Finally, we experiment with the problem of identifying lecture topic boundaries directly from acoustic features of the speech signal.

In chapter 5, we conclude the thesis by highlighting the main points, outlining some of the experimental extensions to the model that did not contribute to further performance gains, and discussing future directions for the work.

Chapter 2

Related Work

Many of the assumptions underlying existing automatic segmentation methods were first formulated in the context of linguistic theory. In this chapter we will outline these theories and distill their connections to the segmentation problem. We then provide an overview of the different computational approaches to text segmentation. We begin by surveying developments in supervised segmentation. Then, we discuss previous work in unsupervised text segmentation that relates most closely to our approach, and conclude by describing a computational model for image segmentation which influenced our work.

2.1 Linguistic Foundations

2.1.1 Lexical Cohesion Theory

One common assumption that threads its way into the design of many segmentation algorithms is the notion that lexical repetition indicates topic continuity, while changes in lexical distribution signal topic changes.

This principle was first formalized in the linguistic work of Halliday and Hasan (1976) on Cohesion Theory. The theory postulates that discourse is constrained by certain grammatical and lexical cohesion requirements. At the semantic and syntactic level these constraints include devices of reference, substitution, ellipsis, and conjunction. At the lexical level, the narratives are tied together by way lexical cohesion or word repetition.

We illustrate these concepts with an analysis of a text fragment reproduced in Figure 2-1 from a transcribed Artificial Intelligence lecture, used in the evaluation of our segmentation system. In the first paragraph, the speaker is giving an overview of agents, and then she moves on to a route planning example. Content words repeated in the span of the text fragment are shown in bold.

> Last time we talked about different ways of constructing **agents** and why it is that you might want to do some sort of **on-line** thinking. We have this idea that if you knew enough about the domain, that off-line you could do all this compilation and figure out what the program that should go in the **agent** and put it in the **agent**. And that's right. But, sometimes when the **agent** has a very rich and complicated environment, it seems easier to leave some of that not worked out, to let the **agent** work some of it out **on-line**. . . .
> The example problem that we'll use in looking at these methods is, for instance, route planning in a **map**. If I give you a **map**, you know the **world** dynamics, because you know that you are in this place and you travel down that road, then you're going to end up at this other place. The **world** state is finite, again as an **abstraction**. If I give you a **map** that has dots on it, which are the towns that they thought were big enough to merit a dot, somebody decided that was a good level of **abstraction** to think about driving around this place. The **world** is deterministic. Again, in the view of a **map**, there aren't probabilities that tell you how likely it is that if you're trying to go here, you'll end up over there

Figure 2-1: Lecture extract from the Artificial Intelligence corpus illustrating lexical cohesion.

Lexical cohesion in these two distinct segments can be observed at the surface level of sentence realization through repetition of key topical words. For example, the word "agent" is repeated in almost all of the sentences of the first paragraph. This is hardly surprising since it is the subject under discussion in that segment. Note also that the word does not reappear in the subsequent segment which moves on to a new topic. Likewise, "map" is repeated several times in the second segment because it relates to the topic of route planning, but it is absent from the first paragraph. In general, if the topics are sufficiently different, it should be expected that the associated key topical words will be different as well.

This property can be exploited for the differentiation of topics within text by preserving continuity of text spans where the lexical distribution is homogeneous and choosing boundaries at locations of prominent change in lexical distribution. The analysis extends to the recurrence of common word stems, synonyms, hyponyms, and word collocations. If words tend to appear in similar contexts, then they are likely to be semantically related, as demonstrated by the cooccurrence of closely related pair of words "program" and "compilation" in the first segment. Despite being patently obvious, the idea of lexical cohesion is very powerful, since the degree of lexical cohesion can be quantified through simple word matching.

Besides lexical cohesion, Halliday and Hasan establish that the presence of certain semantic devices in the text can crystallize the latent thematic structure. Conjunctions such as “for example” in the above text, point to associations between adjoining clauses or sentences. Referential links between anaphors and their antecedents also preserve continuity of the spanned text fragments, because of the persistence of the underlying object. So, in the first paragraph, “that” is referring to the previously mentioned idea. Finally, substitution and ellipsis are also quite common devices that elicit cohesion. These correspond to cases where certain word phrases are implicitly acknowledged to have been either replaced by simpler referring expressions or removed altogether.

In the context of text segmentation, all of these devices can be used to eliminate or identify potential segment boundaries. For example, lexical items and cue words that usually tend to signal references, substitutions, and conjunctions can be readily identified. These trigger words are often employed as lexical features in feature-based segmentation systems. Reynar (1998) observes that anaphoric links tend to occur much more frequently within segments than across different segments and registers the presence of anaphoric links as a feature in his segmentation system. This analysis is consistent with the linguistic function of reference in eliciting cohesion.

2.1.2 Empirical Basis for Lexical Cohesion

Lexical cohesion theory can be grounded empirically with simple graphical representations of lexical distributions in text. Church (1993) achieves this by plotting the cosine similarity scores between every pair of sentences in the text. The intensity of a point (i, j) on the plot indicates the degree to which the i-th sentence is similar to the j-th sentence.

Figure 2-2 is a DotPlot for a lecture transcript from an undergraduate Physics class. The true segment boundaries are denoted by vertical lines. This similarity plot reveals a block structure where true boundaries delimit blocks of text with high inter-sentential similarity. Sentences found in different blocks, on the other hand, tend to exhibit low similarity.

Under multiple domains in both written and spoken genres of language, this representation consistently bears out the claim that repetition of content words is a strong indicator of thematic cohesion, while changes in the lexical distributions usually signal topic transitions. In fact, the representation serves as a basis for many unsupervised algorithms, including

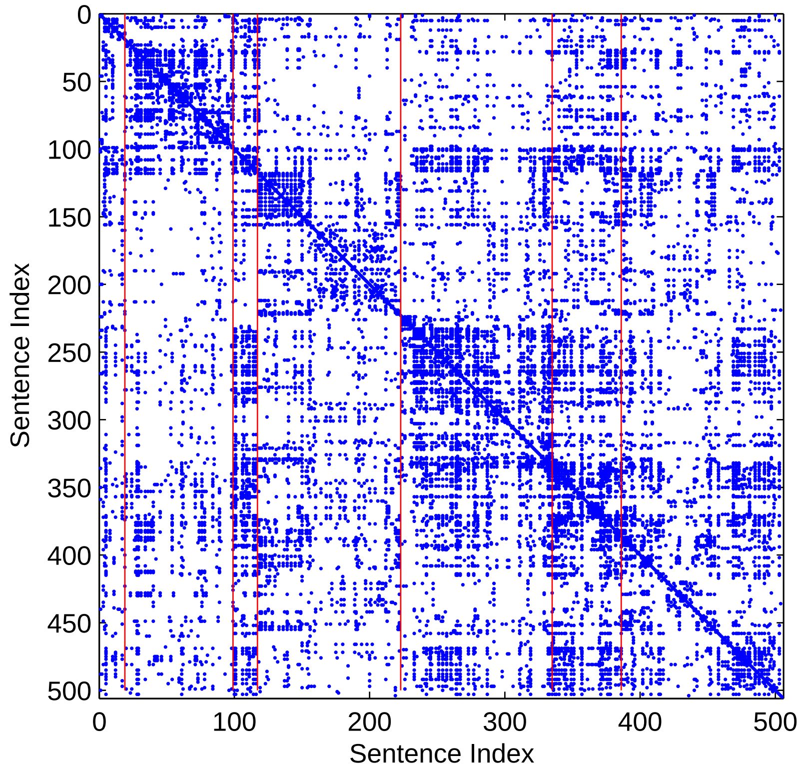

Figure 2-2: DotPlot for a Physics lecture, with vertical lines indicating true segment boundaries.

the approach proposed in this thesis.

2.1.3 Models of Discourse Structure and Coherence

More refined linguistic representations of narratives also shed light on the conceptualization of topic structure. Theories of discourse are concerned in the main with how natural language fits together to produce coherent, easily interpretable narratives that convey meaning and how that meaning is recovered. Approaches to the segmentation problem should be able benefit from an insight into how the thematic structure of text is generated at a higher semantic level of abstraction captured by the notion of coherence.

Textual coherence is a property that is imparted by the global semantic structure embedded in text. For example, Rhetorical Structure Theory (Mann and Thompson, 1987) posits that this sense of logical flow is pieced together by an implicit rhetorical tree of relations among phrasal constituents, relations such as cause and elaboration. Grosz and Sidner (1986), on the other hand, argue that beyond inter-segmental and thematic relations, coherence is conveyed in how the thematic structure relates to the message that the speaker intended to convey and how the target audience actually processes that information.

Even though there are many different discourse theories, the underlying idea of discourse coherence has important implications for segmentation modeling. In general, the goal of segmentation should be to provide the *coherent* constituent structural blocks, whereas most current segmentation systems only aim to provide the set of *cohesive* segments in a text. After all, we are interested in exposing the underlying semantic layers and not just the surface grammatical or lexico-distributional regularities.

In theory, modeling coherence is much more powerful than merely being able to model lexical cohesion. Many of the current segmentation systems fail to take into account the global distributional properties of text that tie into coherence. The approach proposed in this thesis provides part of the framework for modeling coherence by considering the long-range lexical relationships. Since many theories suggest that segmentation should be modeled hierarchically in order to capture the relational structure underlying coherence, our approach could be used as the first step in full semantic relational parsing.

2.2 Supervised Methods

Although our focus in this thesis will be on unsupervised, similarity-based models for segmentation, we will briefly highlight some of the supervised approaches. These methods usually require large amounts of in-domain training, and are sensitive to noise, speech recognition errors, and data sparsity. The supervised methods for segmentation typically fall into one of the two classes, namely binary classification or sequential models.

2.2.1 Classification and Sequential Models

Under the classification framework, each candidate boundary location in the text is evaluated independently by the model, and then the top scoring candidate boundaries are selected. Some of the approaches applied to text segmentation in this class of learning algorithms in the past include Decision Trees (Passonneau and Litman, 1997; Gruenstein et al., 2005), Maximum Entropy (Beeferman et al., 1999), Support Vector Machines (Kauchak and Chen, 2005), and Boosting (Sporleder and Lapata, 2006). The strength of these models lies in their ability to encode arbitrary local contextual features. However, the fact that hypotheses are evaluated independently detracts from their effectiveness, since segment boundaries are inter-dependent. For example, these types of models will not be able to capture the fact that very short segments should be unlikely.

Sequential models, as the name implies, model sequences of decisions. Van Mulbregt et al. (1999), Shriberg et al. (2000), and Ponte and Croft (1997) model text streams with Hidden Markov Models over word sequences, with HMM states corresponding to boundary and non-boundary states delimiting segments. Dielmann et al. (2005) employed Dynamic Bayesian Networks for structured multi-party meeting segmentation. These approaches typically require a lot of training data, and they are applied to highly structured domains.

2.2.2 Features

The effectiveness of supervised segmentation models often hinges on choosing a suitable feature representation. In the written language domain, lexical cohesion and linguistically motivated features are used. Cohesion features capture the underlying word distributions, indicating whether segments are lexically cohesive. Beeferman et al. (1999) encode the log likelihood of a context-sensitive and context-independent language model as a feature in their model. Galley et al. (2003) incorporate cosine similarity scores between blocks of text. The linguistic features may register the presence of referential noun phrases which indicate topic continuity or cue words, which usually signal topic changes.

In spoken language segmentation, additional prosodic, acoustic, and discourse features such as speaker activity, speaker overlap, and pause duration have been used to improve segmentation quality (Shriberg et al., 2000; Gruenstein et al., 2005).

2.3 Unsupervised Methods

In this thesis, we focus on the development of unsupervised approaches to segmentation, which tend to differ markedly from their supervised counterparts. Unsupervised segmentation methods can be characterized by the form of the optimization objective, the type of contextual representation and smoothing, and finally by the decoding techniques used for obtaining the segmentation.

2.3.1 Optimization Objective

The optimization objective for segmentation is usually defined either in probabilistic terms or in terms of lexical similarity.

Probabilistic approaches Among approaches with probabilistically motivated objectives, for example, the method developed by Utiyama and Isahara (2001) finds the maximum probability segmentation for the noisy channel model of segmentation. Given a word sequence $W = w_1 w_2 \dots w_n$ and a segmentation $S = s_1 s_2 \dots s_m$ of W the approach aims to maximize $P(S|W) = \frac{P(W|S)P(S)}{P(W)}$. This is equivalent to finding the most likely sequence of segments $\hat{S} = \arg\max_S P(W|S)P(S)$. In order to evaluate this objective, the authors make the simplifying assumption that segments are statistically independent of each other, and words within segments are conditionally independent given the segment. This allows them to decompose the $P(W|S)$ into a product of word emission probabilities, conditioned on the topic:

$$P(W|S) = \prod_{i=1}^{m} \prod_{j=1}^{n_i} P(w_j^i|S_i),$$

where w_j^i is the j-th word in segment i or S_i. Furthermore, $P(W|S)$ is a defined as a smoothed language modeling probability:

$$Pr(w_j^i|S_i) = \frac{f_i(w_j) + 1}{n_i + k},$$

where $f_i(w_j)$ is the frequency of j-th word in the i-th segment and n_i is the number of words in segment i. $Pr(S)$ is defined as a description length prior $2^{-l(S)}$, where $l(S) = m\ logn$ is the description length, m is the number of words in the text, and n is the number of segments. Putting all of these terms together, and taking the log of the posterior, we yield the following objective:

$$\log P(S|W) = \sum_{i=1}^{m} \sum_{j=1}^{n_i} \log \frac{f_i(w_j) + 1}{n_i + k} - m \log n$$

The assumptions of statistical independence for the segments and the conditional independence of words are not borne out in real data. With very short segments, this model will produce noisy estimates for the word emission probabilities. Also, it does not capture the relative importance of words in the process of segmentation.

Other probabilistic models include the work of Purver et al. (2006), who propose a more refined generative model of topic structure, which models the word distributions in segments with a linear combination of distributions over topics.

Similarity-based approaches In many cases pattern recognition problems do not lend themselves readily to a probabilistically-motivated objective, whereas the concept of object or entity similarity may be quite natural. The notion of lexical similarity has been extensively explored and applied in many other natural language tasks.

In the context of segmentation, text is usually decomposed into a series of sentences or blocks, represented by vectors of word counts. Text similarity is measured in terms of cosine similarity of adjacent blocks, $s_i = \langle w_1 w_2 \dots w_n \rangle$, where cosine similarity, $S(s_i, s_j)$, is defined as:

$$S(s_i, s_j) = \frac{s_i \cdot s_j}{||s_i|| \times ||s_j||},$$

In the equation above, $s_i \cdot s_j$ is the dot product of two vectors and $||s_i||$ is the L_2 norm of vector s_i.

Most unsupervised text segmentation algorithms assume that fragments of text with homogeneous lexical distributions correspond to topically coherent segments. So, the homogeneity is typically computed by analyzing the similarity in the distribution of *words within a segment*. The approaches that maximize self-similarity within a segment include (Choi, 2000), (Reynar, 1998), (Kehagias et al., 2003), and (Ji and Zha, 2003). Other approaches determine segment boundaries by locating sharp changes in similarity of *adjacent blocks of text* (Reynar, 1998; Hearst, 1994). Ideally, both of these objectives should be used to evaluate segmentation quality.

2.3.2 Contextual Dependencies

The earliest approaches to text segmentation only took into account local contextual information (Kozima, 1993; Hearst, 1994). For instance, Hearst developed the TexTiling segmentation algorithm for the problem of partitioning expository texts. This approach assumes that drops in the similarity profile of adjacent text blocks correspond to topic changes and that topic changes occur in between paragraph breaks of the text. The Text-

Tiling algorithm determines boundaries by locating local minima in the sequence of cosine similarity scores of adjacent blocks of text. It determines the target number of segments by specifying a similarity cutoff threshold.

The weakness of this approach is that it only considers similarity between adjacent blocks of text, and does not model longer-distance lexical ties. Also, a fixed cutoff for determining boundaries is problematic, since texts may exhibit both sharp and attenuated topic transitions in different parts of the narrative.

Other unsupervised segmentation approaches work with the DotPlotting text representation suggested by Church (1993) first used by Reynar (1994) for segmentation and later adopted by Choi (2000), Kehagias et al. (2003), and Ji and Zha (2003).

These algorithms compute pairwise cosine similarity between every pair of sentences sentences, so the resulting representation is much finer. Then they try to elicit the latent block structure in the similarity matrix. This representation enables the approaches to model long range cohesion dependencies, not just the local context. Our work draws part of its strength from this latest line of research in unsupervised segmentation.

2.3.3 Smoothing and Lexical Weighing

Previous research on similarity-based segmentation methods has analyzed lexical weighting, similarity computation, and smoothing (Hearst, 1994; Utiyama and Isahara, 2001; Choi, 2000; Reynar, 1998; Kehagias et al., 2003; Ji and Zha, 2003). In practice, smoothing has delivered significant performance gains.

Choi (2000) uses similarity ranks in the local context instead of using the actual inter-sentence similarity and further refines the similarity metric by incorporating lexical similarity weights from Latent Semantic Analysis (Choi et al., 2001). Ji and Zha (2003) apply anisotropic diffusion smoothing to the sentence similarity matrix, achieving gains over (Utiyama and Isahara, 2001; Choi, 2000) on a synthetic corpus of concatenated text blocks. We will describe the latter smoothing approach in the next chapter in section 3.4.

The effectiveness of the smoothing approaches is often dependent on the segmentation domain and the underlying characteristics of the segmentation algorithm. For instance, lexical similarity scores obtained from Latent Semantic Analysis will be beneficial in the synthetic domain, because the topics represented in the text are very different. However, when much more subtle distinctions are required for the purpose of sub-topic segmentation,

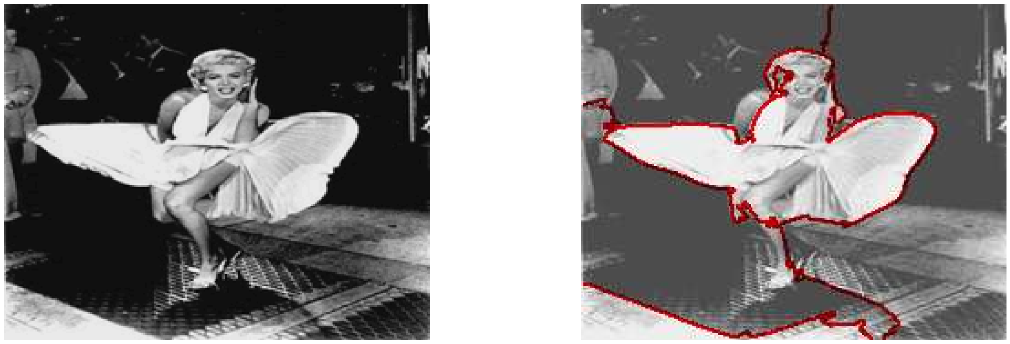

Figure 2-3: (a) Original Image (b) Image segmented with the Normalized Cut Algorithm

this technique may actually degrade performance.

2.3.4 Decoding

The final distinction that can be made among unsupervised segmentation algorithms is based on the type of decoding technique used. The decoding either involves a greedy approximation or performs exact inference. The former class includes the text segmentation algorithm proposed by Reynar (1998), while most of the current state-of-the-art segmentation methods use dynamic programming to obtain the optimal segmentation (Choi, 2000; Utiyama and Isahara, 2001; Kehagias et al., 2003; Ji and Zha, 2003).

2.4 Graph-Theoretic Approaches in Vision Segmentation

In addition to past text segmentation approaches, our model was influenced by the minimum-cut-based segmentation algorithm developed for the problem of image segmentation (Shi and Malik, 2000). The objective of image segmentation is to partition an image into multiple regions corresponding to the different objects and the background. For illustration purposes, consider the original image in Figure 2-3(a) and its counterpart segmented into five regions shown in Figure 2-3(b). The segmentation algorithm delineates the outlines of Marylin Monroe and separates the background into four different regions.

Shi and Malik (2000) approach image segmentation through graph partitioning. Each image pixel is represented as a node in the graph. The feature vectors for the pixels capture intensity, color, and texture information. Edge weights, w_{ij}, between node pairs are defined

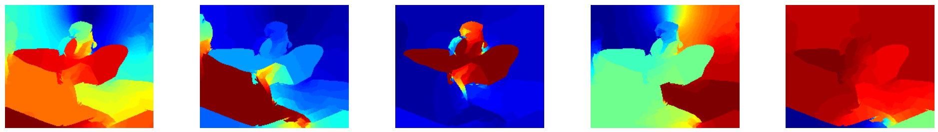

Figure 2-4: Normalized Cut Eigenvectors

as the product of a feature similarity term and a term corresponding to the spatial distance between the pixels i and j: $w_{ij} = e^{\frac{-\|F(i)-F(j)\|^2}{\sigma_I}} \times e^{\frac{-\|X(i)-X(j)\|^2}{\sigma_X}}$, where $\|\cdot\|$ is the L_2 norm, $F(i)$ is the feature vector for pixel i, $X(i)$ is the spatial location of node i, and σ_I and σ_X are parameters. The quality of the partitioning is measured by a new criterion, the normalized-cut, described in the next chapter. Minimizing the normalized cut is NP-complete. However, Shi and Malik reformulate the minimum cut problem in terms of a generalized eigenvalue system subject to discrete constraints on the decision variables. If the decision variables are allowed to take on continuous values, the system can be efficiently solved by finding the second smallest eigenvector of the generalized eigensystem through eigenvalue decomposition.

The cluster assignment is resolved by selecting a threshold such as the median of the eigenvector components and assigning pixels below the threshold to one cluster and those above the threshold to the other cluster. The assignments taken by discretizing the solutions to the relaxed eigenvalue system are only approximate. In general, Shi and Malik show that the eigenvector with the n-th smallest eigenvalue is the real-valued solution that optimally subpartitions the first $n-1$ parts of the overall image. Figure 2-4 shows the five eigenvectors with the smallest eigenvalue.

We note, that one of the principal conceptual differences between text segmentation and image segmentation is that in image segmentation segment boundaries can be drawn up arbitrarily, whereas in text segmentation the boundaries form a linear partitioning of the nodes, so that nodes between two closest boundaries have to belong to the same segment.

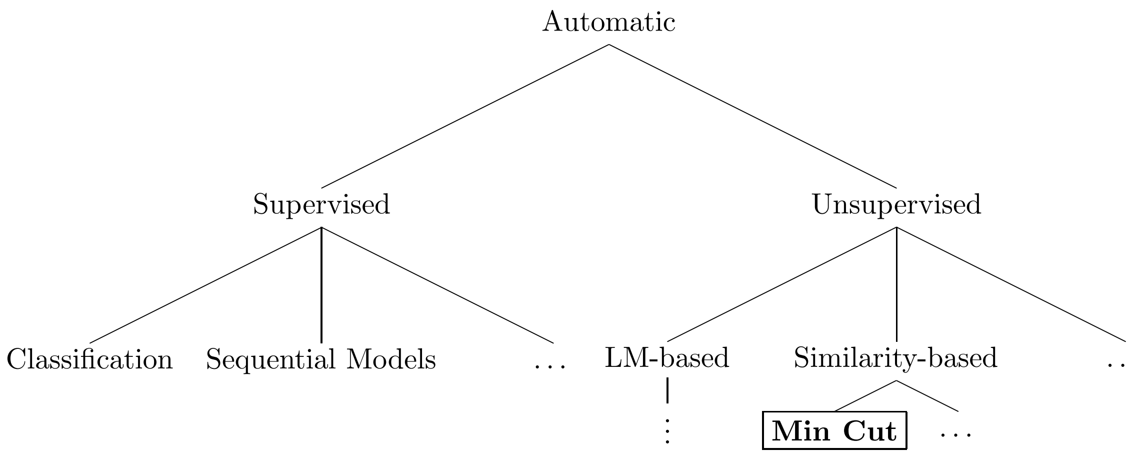

Figure 2-5: Taxonomy of Text Segmentation Models

2.5 Our Approach

Figure 2-5 illustrates the overarching taxonomy of approaches to the segmentation problem. Our algorithm fits into the unsupervised, similarity-based class of approaches to text segmentation. One of the contributions of our work is on the fundamental aspect of text segmentation analysis — the impact of long-range cohesion dependencies on segmentation performance. In contrast to previous approaches, the minimum cut algorithm *simultaneously optimizes the similarity within each segment and the dissimilarity across segments.* Thus, the homogeneity of a segment is determined not only by the similarity of its words, but also by their relation to words in other segments of the text. We show that optimizing our global objective refines the analysis of the lexical distribution and enables us to detect subtle topical changes. Another advantage of this formulation is its computational efficiency. Similarly to other segmentation approaches (Utiyama and Isahara, 2001; Choi, 2000; Reynar, 1998; Kehagias et al., 2003; Ji and Zha, 2003), we are able employ dynamic programming to find the globally optimal solution, because of the linearity constraint on text segmentation.

Chapter 3

Minimum Cut Segmentation

Whereas many of the past unsupervised approaches to segmentation rested on intuitive notions of similarity density, we formalize the objective of text segmentation through cuts on graphs. In this chapter, we first formulate the minimum cut problem, and then describe how it can be applied naturally to the text segmentation task. Finally, we flesh out the implementation details for the text segmentation system based on the Minimum Cut model.

3.1 Background

3.1.1 Minimum Cuts

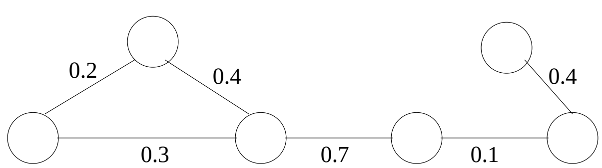

Figure 3-1: Input: Weighted Undirected Graph

Let $G = \{V, E\}$ be an undirected graph, where V is the set of vertices and E is the set of weighted edges (See Figure B-1). We denote the edge weights between every connected pair of vertices u and v by $w(u, v)$. A graph cut is the partitioning of the graph into two disjoint sets of nodes A and B.

The capacity of the cut is defined as the sum of crossing edge weights between A and B. Figure 3-2 includes two possible cuts of the graph in Figure B-1. The edges severed by this cut are shown in dotted lines. The capacity of the left cut in the figure is 0.1, and the capacity of the right cut is 0.5. Note that for notational convenience, we will henceforth refer to the cut capacity and the cut value interchangeably in the thesis.

We are interested in the problem of finding the minimum capacity cut or min cut, for short. The minimum cut is a partitioning of the graph into two disjoint sets of nodes that minimizes the cut capacity. In Figure, 3-2 the left cut is the minimum cut, because it is the configuration that minimizes the sum of the crossing edges.

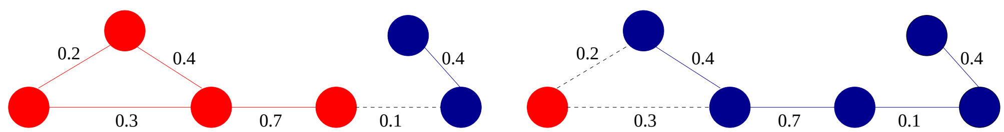

Figure 3-2: Examples of Binary Cuts on a Graph

The minimum cut problem is important in clustering tasks among other applied problems. Wu and Leahy (1993), for example, formulate a method for clustering data with the minimum cut criterion and demonstrate how it can be applied to image segmentation. If the edge weights represent the degree of node similarity, then the capacity of a cut corresponds to the extent of association between the two partitions. Minimizing the cut corresponds to minimizing the degree of association between these partitions, thereby splitting the graph into its two most dissimilar components.

3.2 Variations on the Minimum Cut Objective

There is a problem with the minimum cut objective in its unaltered form. When minimum cuts are employed for clustering, they will often give rise to unbalanced partitions, which can be problematic. Shi and Malik (2000) and Wu and Leahy (1993) observe that small clusters of outlying nodes will tend to be separated from the rest of the graph in many clustering scenarios. This is not a desirable feature for a clustering objective function. In order to address the shortcomings, several alternative forms of the objective have been formulated. We will use the normalized cut objective introduced by Shi and Malik (2000), because it is superior to its alternatives in several important respects.

3.2.1 Normalized Cut

First, we will define the volume of a subset of the graph to be the sum of its edges to the entire graph:

$$vol(A) = \sum_{u \in A, v \in V} w(u, v)$$

Similarly, we can define the association, $assoc(A)$ of a particular cluster of nodes as follows:

$$assoc(A) = \sum_{u \in A, v \in A} w(u, v)$$

Note that volume is simply the sum of the cut value (the sum of cross-partition edge weights) and the association value (the sum of the interpartition edge weights). The new normalized cut criterion ($Ncut$) is a result of normalizing the cut by the volume:

$$Ncut(A, B) = \frac{cut(A, B)}{vol(A)} + \frac{cut(A, B)}{vol(B)}$$

For example, in Figure 3-2, the left segmentation has a cut value of 0.1 and the volume of sets A and B is 1.7 and 0.5, respectively. This results in a normalized cut value of $\frac{0.1}{1.7} + \frac{0.1}{0.5} = 0.2588$. The right segmentation has a cut value of 0.5 and the volumes of the two sets are 0.5 and 2.1, giving a normalized cut value of $\frac{0.5}{0.5} + \frac{0.5}{2.1} = 1.2381$. So, the left partitioning has a smaller normalized cut value.

In general, this alternative form of the objective is sensible, because now the capacity of a cut is measured as a fraction of the overall outgoing weight edges from each subset of nodes. So, for clusters with a small number of points the cut capacity to volume ratio will be large. Therefore, by minimizing this criterion we ensure that the partitions are balanced.

We can identify an even stronger property. Namely, by optimizing this objective we simultaneously minimize the similarity across partitions and maximize the similarity within partitions.

One natural alternative to minimizing the degree of similarity between clusters is to maximize the degree of association within clusters. The normalized association criterion,

$Nassoc$, is defined as follows:

$$Nassoc(A,B) = \frac{assoc(A)}{vol(A)} + \frac{assoc(B)}{vol(B)}$$

We will now show that the normalized cut and the normalized association add up to a constant.

$$Nassoc(A,B) + Ncut(A,B) = \left[\frac{cut(A,B)}{vol(A)} + \frac{cut(A,B)}{vol(B)}\right] + \left[\frac{assoc(A)}{vol(A)} + \frac{assoc(B)}{vol(B)}\right]$$

$$= \left[\frac{cut(A,B) + assoc(A)}{vol(A)}\right] + \left[\frac{cut(A,B) + assoc(B)}{vol(B)}\right] = \frac{vol(A)}{vol(A)} + \frac{vol(B)}{vol(B)} = 2$$

This proves that minimizing the normalized cut criterion is equivalent to maximizing the normalized association objective, as $Ncut(A,B) = 2 - Nassoc(A,B)$.

3.2.2 Average Cut

Another alternative to the plain cut is to normalize the cut by the cardinality of a particular cluster:

$$Ncut(A,B) = \frac{cut(A,B)}{|A|} + \frac{cut(A,B)}{|B|}$$

This will ensure that the clusters are balanced. However, this criterion does not guarantee that the the clusters will have tight inter-cluster similarity.

3.2.3 Average Association

In order to have tight inter-cluster similarity, we can normalize the inter-cluster similarity by the cardinality of a cluster:

$$Nassoc(A,B) = \frac{assoc(A)}{|A|} + \frac{assoc(B)}{|B|}$$

However, this objective will be prone to separating out small clusters with large inter-cluster similarity.

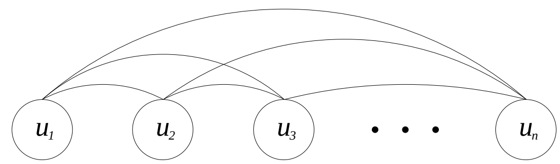

Figure 3-3: Graph-based Representation of Text

3.3 Normalized Mincut Segmentation

We will now show why optimizing the normalized cut objective is a natural fit for the text segmentation problem. Initially, we will consider the binary segmentation problem. Therefore, we will assume that there are only two sections in the text to be segmented. The nodes of the graph will denote adjacent sentences, and the edge weights, $w(u, v)$, will define a measure of similarity between pairs of sentences, where higher scores indicate higher lexical similarity (See Figure 3-3).

Intuitively, we aim to jointly maximize the intra-segmental similarity and minimize the similarity between different segments. In other words, we want to find the segmentation with the most homogeneous set of segments that are also maximally different from each other.

In Chapter 2, we showed an empirical basis for the computational objective of the segmentation problem with the DotPlot representation. That is we observed that identifying the block structure relates directly to the problem of maximizing within-block similarity while minimizing the block similarity between clusters.

This segmentation goal corresponds naturally the normalized minimum cut criterion. By obtaining a minimum cut we split the set of phrases into two maximally dissimilar classes. As shown in the previous section, we simultaneously minimize the similarity across partitions.

In text segmentation, the texts typically consist of more than two segments. Hence, by extension we are interested not just in binary cuts but in multiway cuts on graphs. (See figure 3-4). The normalized cut criterion is naturally extended to a k-way normalized cut:

$$Ncut_k(V) = \frac{cut(A_1, V - A_1)}{vol(A_1)} + \ldots + \frac{cut(A_k, V - A_k)}{vol(A_k)} \tag{3.1}$$

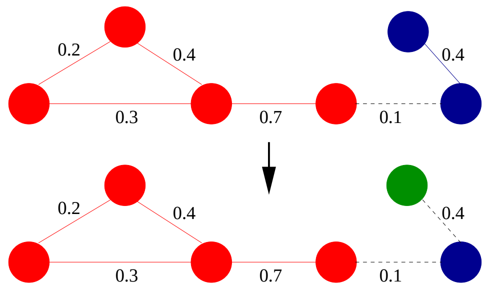

Figure 3-4: Multiway Cuts on Graphs

where $A_1 \ldots A_k$ form a partition of the graph, and $V - A_k$ is the set difference between the entire graph and partition k.

3.3.1 Decoding Algorithm

Papadimitriou proved that the problem of minimizing normalized cuts on graphs is NP-complete (Shi and Malik, 2000). However, in our case, the multi-way cut is constrained to preserve the linearity of the segmentation. By segmentation linearity, we mean that all of the nodes between the leftmost and the rightmost nodes of a particular partition must belong to that same partition.

With this constraint, the space of possible solutions to the minimum cut problem is reduced considerably. In fact, it enables us to formulate a dynamic programming algorithm to find the exact solution to the minimum normalized multiway cut problem in polynomial time.

3.3.2 Dynamic Programming Fundamentals

We will first outline the structure of deterministic dynamic programming problems with a finite number of stages (finite horizon). These problems can be decomposed into a set of *overlapping subproblems*. The solutions to these subproblems are typically saved or *memoized*, and are reused in later stages of the algorithm for solving larger subproblems. Dynamic programming problems exhibit *optimal substructure*, meaning that finding the optimal solutions to the subproblems enables us to find the globally optimal solution to the overall problem.

More formally, we are given the following discrete-time system, specifying the progression of states with respect to decisions made at discrete points in time (Bertsekas, 2001):

$$x_{k+1} = f_k(x_k, u_k) \quad k = 0, 1, \ldots, N-1,$$

where x_k is the state of the system at stage or time index k, N (horizon) is the number of stages that the system goes through starting at state x_0, u_k are decision variables selected at time k, and $f_k(x_k, u_k)$ are functions that specify how the state is updated on the basis of the current state x_k and the chosen decision variable u_k.

The states x_k are elements of space S_k, corresponding to each stage in the evolution of the system. In general, the states are not constrained to be discrete-valued and may not be bounded. The controls u_k belong to the space C_k and are dependent on the current state, x_k. A cost function, $c(x_k, \mu(x_k)$, maps the k-th state and its corresponding control to some cost, c. A policy π is a set of functions μ_i over a span of stages or time points, mapping states x_i to their decision variables u_i: $\pi_t = (u_0, u_1, \ldots, u_t)$

Assuming that the system starts out at state x_0, the policy π_t incurs a cumulative cost $J_{\pi_t}(x_0) = J(x, u_0, u_1, \ldots u_t)$. So each transition incurs a cost, and the problem is to find the optimal policy $\pi^* \in \Pi$ that minimizes the overall cost:

$$J_{\pi^*}(x_0) = \min_{\pi \in \Pi} J_\pi(x_0),$$

where Π is the set of all possible policies. In other words, the goal is to choose the optimal sequence of decision controls to minimize the overall cost.

3.3.3 Bellman's Optimality Principle

Assume that the cost function is additive, meaning that the overall cost of a policy is the sum of the costs incurred at each of the stages. More formally, the cost function is additive if the objective function satisfies the following requirement:

$$J_\pi(x_0) = \sum_{k=0}^{T-1} c_k(x_k, u_k) + k_T(x_T), \tag{3.2}$$

where $k_T(x_T)$ is the terminal cost and $c_k(x_k, u_k)$ corresponds to the individual transition cost at time k, state x_k and control u_k.

Let $\pi^* = (u_0^*, u_1^*, \ldots, u_{N-1}^*)$ be an optimal policy; i.e. the policy minimizing the overall cost. Consider the subproblem, where we wish to minimize the cost from time i to time N. Let state x_i be the starting point in this new subproblem, corresponding to time i. Bellman's principle of optimality establishes that the truncated policy $u_i^*, u_1^*, \ldots, u_{N-1}^*$ is optimal for this subproblem (Bertsekas, 2001).

Intuitively, if the optimal sequence of states from the start to the end state hits state x_i at stage i, then the sub-policy from step i to $N-1$ should be optimal. Otherwise, if there is a policy with a lower cost, then we could combine it with the initial subsequence of the optimal policy to get a policy with an even lower cost, which would lead to a contradiction.

3.3.4 Dynamic Program for Constrained Mincut

The constrained multiway normalized minimum cut objective can be shown to exhibit optimal substructure. Note that our problem involves a finite set of states (the last chosen boundaries) and also a finite set of controls (the potential set of terminal segment boundaries). The cost to place a boundary at a given stage in the segmentation is only dependent on the current state, captured by the location of the previous boundary. This is true, because of the linearity constraint on the segmentation. Since segments need to be contiguous, the last boundary marks the start of the new segment. The control to be picked at this stage corresponds to the location of the next boundary, which must be placed further along the text.

Let C_k be the cost incurred at the k-th decision stage: $c_k = \frac{cut(A_k, V-A_k)}{vol(A_k)}$, and u_k be the value of the decision variable at stage u_k. Again, since segments need to be contiguous, $u_{k-1} \leq u_k$. So, choosing the i-th segment corresponds to choosing a single boundary point to finish the segment. The term $cut(A_k, V - A_k)$ can be computed from the current state which is the value of the previous decision boundary and the current decision variable value. Likewise, $vol(A_k)$ can be computed from the current state and the decision variable.

The objective function is clearly additive, as it is the sum of individual costs incurred by each of the segments. Hence, according to Bellman's optimality principle we can formulate the following dynamic program to optimize the minimum cut objective:

$$C\left[i,k\right] = \min_{j<k} \left[C\left[i-1,j\right] + \frac{cut\left[A_{j,k}, V - A_{j,k}\right]}{vol\left[A_{j,k}\right]} \right] \tag{3.3}$$

$$B[i,k] = \operatorname*{argmin}_{j<k} \left[C[i-1,j] + \frac{cut[A_{j,k}, V - A_{j,k}]}{vol[A_{j,k}]} \right] \tag{3.4}$$

$$\text{s.t. } C[0,1] = 0, C[0,k] = \infty,\ 1 < k \leq N \tag{3.5}$$

$$B[0,k] = 1,\ 1 \leq k \leq N \tag{3.6}$$

$C[i,k]$ is the normalized cut value of the optimal segmentation of the first k sentences into i segments. The i-th segment, $A_{j,k}$, begins at node u_j and ends at node u_k. $B[i,k]$ is the back-pointer table from which we recover the optimal sequence of segment boundaries. The initial conditions in Equations 3.5 and 3.6 capture the constraint that the first segment starts with the first node.

The time complexity of the dynamic programming algorithm is $O(KN^2)$, where K is the number of partitions and N is the number of nodes in the graph or sentences in the transcript.

3.4 Implementation Mechanics

The performance of our model depends on the underlying representation, the definition of the pairwise similarity function for texts, and various other model parameters. In this section we provide further details on the process of constructing the target graph that will be partitioned into segments and implementing the overall segmentation system.

3.4.1 Text Preprocessing

Before building the graph, we apply standard text preprocessing techniques to the text. We stem words with the Porter stemmer (Porter, 1980) to alleviate the sparsity of word counts through stem equivalence classes. Since many frequently occurring words in the text such as determiners or personal pronouns are poor indicators of the actual thematic similarities between segments, we remove words matching a list of stop words. We make use of the stop-words list used in several other segmentation systems (Choi, 2000; Utiyama and Isahara, 2001) This stop-words list is reproduced in Appendix C.

3.4.2 Graph Construction

The normalized cut criterion considers long-term similarity relationships between nodes. This effect is achieved by constructing a fully-connected graph. However, considering all pairwise relations in a long text may be detrimental to segmentation accuracy. Therefore, we discard edges between sentences exceeding a certain threshold distance. This reduction in the graph size also provides us with computational savings.

Also, note that in the formulation above we use sentences as our nodes. However, we can represent graph nodes with non-overlapping blocks of words of fixed length. This is desirable, since the lecture transcripts lack sentence boundary markers, and short utterances can skew the cosine similarity scores. The optimal length of the block is tuned on a heldout development set.

3.4.3 Similarity Computation

In computing pairwise sentence similarities, sentences are represented as vectors of word counts and the objective is to identify sentences with similar semantic content. So, we have to make sure that the semantically salient words are given predominant weight in the computation. Previous research has shown that weighting schemes play an important role in segmentation performance (Ji and Zha, 2003; Choi et al., 2001). Apart from being able to distinguish between functional and content-bearing words, particularly important are words that may not be common in general English discourse but that occur throughout the text for a particular lecture or subject.

For example, in a lecture about support vector machines, the occurrence of the term "SVM" is not going to convey a lot of information about the distribution of sub-topics, even though it is a fairly rare term in general English and bears much semantic content. The same words can convey varying degrees of information across different lectures, and term weighting specific to individual lectures becomes important in the similarity computation.

In order to address this issue, we introduce a variation on the *tf-idf* scoring scheme used in the information-retrieval literature (Salton and Buckley, 1988). A transcript is split uniformly into N chunks; each chunk serves as the equivalent of documents in the *tf-idf* computation. In equation 3.7, n_i is the number of chunks in which word i appears, idf_i is the inverse segment frequency of word i in the transcript, and $tf_{i,j}$ is the term frequency of word i in chunk j. The lexical weights are computed separately for each transcript, since topic and word distributions vary across lectures.

$$w(i,j) = tf_{i,j} \times idf_i, \text{ where } idf_i = log(\frac{N}{n_i}) \tag{3.7}$$

After determining the lexical weights, we compute cosine similarity scores between every sentence pair with word frequencies weighted by their *tf-idf* weights:

$$sim(x,y) = \frac{\sum_k [f_{x,j} \times w(k, cid(x)) \times f_{y,j} \times w(k, cid(y))]}{||\vec{w_x} \cdot \vec{x}|| \times ||\vec{w_x} \cdot \vec{y}||} \tag{3.8}$$

In equation 3.8, $f_{x,j}$ is the frequency of word j in sentence x, $\vec{w_x}$ is the vector of weights for sentence x, and $cid(x)$ is the word chunk index containing the sentence.

Finally, in computing the actual edge weight, $e_{i,j}$ between nodes i and j in the graph, the exponent of the cosine similarity score is used to accentuate differences between low and high lexical similarities.

$$e_{i,j} = e^{sim(i,j)} \tag{3.9}$$

3.4.4 Smoothing

The similarity matrix, specifying edge weights between nodes in the graph, will capture the similarity profile at the sentence level. Even though similarity scores of sentences belonging to the same segment will tend to be higher than scores of sentence pairs belonging to different segments, the individual scores are highly variable. This is problematic, because it is not always possible to tell whether a sudden shift in scores in the vicinity of a sentence signifies a transition or it is really just an artifact of the data and the similarity computation.

Consider the case when a sentence is a sequence of stop words and very infrequent lexical items. The similarity score between this and other sentences will be set to the minimum possible score, even though the immediate context may share many content words in common with other parts of the text. Without proper smoothing, these cases will lead the system astray. We considered two smoothing approaches - the Exponentially Weighted Moving Average (EWMA) smoothing and Anisotropic Diffusion.

EWMA The exponentially weighted moving average smoothing developed by S. W. Roberts (Roberts, 1959) is computed by adding counts of words that occur in adjoining sentences to the current sentence feature vector. These counts are weighted in accordance to their distance from the current sentence: $\tilde{s}_i = \sum_{j=i}^{i+k} e^{-\alpha(j-i)} s_j$, where s_i are vectors of word counts, and α is a parameter that controls the degree of smoothing. Hence, when computing the similarity between two sentences, we effectively take into account similarity between their immediate neighborhoods. Empirically, we found that incorporating only previous words in the neighborhood works better than incorporating words on both sides of the target word in the text.

Anisotropic Diffusion Anisotropic diffusion smoothing is a technique developed for image enhancement (Perona and Malik, 1990), and it has been applied previously to lexical smoothing in the context of text segmentation (Ji and Zha, 2003). The method is based on the anisotropic heat diffusion equation (Equation 3.10), which describes temperature as a function of time and space.

$$I(x, y, t) = (c(x, y, t)\nabla^2 I + \nabla c \cdot \nabla I)|_{(x,y)} \tag{3.10}$$

In equation 3.10, I is the brightness or intensity function, $c(x, y, t)$ is the space-dependent diffusion coefficient at time t corresponding to the point (x, y) in the space, ∇ is the gradient and ∇^2 the Laplacian operator, both with respect to the space variables.

On a square lattice, or a gray scale image with nodes corresponding to pixels, the above equation is discretized by approximating the Laplacian with 4-nearest neighbor differences. In Equation 3.11, the term $\bigtriangledown$ indicates the nearest neighbor differences in appropriate directions (North, South, East, or West corresponding to subscripts N, S, E, W), and c^t are the corresponding heat diffusion coefficients. The diffusion flow conduction coefficients are chosen locally to be the inverse of the magnitude of the gradient of the brightness function, because then the flow increases in homogeneous regions which have small gradients.

$$I_{i,j}^{t+1} = I_{i,j}^{t} + \lambda[c_{N_{i,j}}^{t} \cdot \bigtriangledown_N I_{i,j}^{t} + c_{S_{i,j}}^{t} \cdot \bigtriangledown_S I_{i,j}^{t} + c_{E_{i,j}}^{t} \cdot \bigtriangledown_E I_{i,j}^{t} + c_{E_{i,j}}^{t} \cdot \bigtriangledown_W I_{i,j}^{t}] \tag{3.11}$$

$$
\begin{array}{rclcrcl}
\bigtriangledown_N I_{i,j} & = & I_{i-1,j} - I_{i,j} & \quad & c^t_{N_{i,j}} & = & g(| \bigtriangledown_N I^t_{i,j} |) \\
\bigtriangledown_S I_{i,j} & = & I_{i+1,j} - I_{i,j} & & c^t_{S_{i,j}} & = & g(| \bigtriangledown_S I^t_{i,j} |) \\
\bigtriangledown_E I_{i,j} & = & I_{i,j+1} - I_{i,j} & & c^t_{E_{i,j}} & = & g(| \bigtriangledown_E I^t_{i,j} |) \\
\bigtriangledown_W I_{i,j} & = & I_{i,j-1} - I_{i,j} & & c^t_{W_{i,j}} & = & g(| \bigtriangledown_W I^t_{i,j} |)
\end{array}
$$

$$
g(\nabla I) = \frac{1}{1 + (\frac{||\nabla I||}{\kappa})^2} \tag{3.12}
$$

The particular function $g(.)$ in Equation 3.12 was chosen by Perona and Malik to favor diffusion in wide regions over smaller ones.

Anisotropic diffusion has the effect of increasing flow in homogeneous regions and preventing flow across region boundaries in the image. Again, this is consistent with the idea of minimizing between-block similarity and maximizing within-block similarity in the similarity matrix. In practice, our experiments showed that the anisotropic diffusion smoothing technique is much more stable and effective in smoothing the similarity matrices. We use it in the final configuration of the Min Cut system. This method takes as input the κ and λ parameters, as well as the desired number of iterations. The parameters are tuned on the development set.

3.5 Min Cut Segmentation Pseudo-Code

We conclude this chapter by providing the implementation pseudo-code for the Min Cut segmentation system.

Function: `ComputeTfIdfWeights(`*WordFrequencyMap, text, nSegments*`)`

Returns : map of sentences and word types to word counts

```
begin
    TfIdfMap ← makeNewMap() ;
    segmentedText ← generateUniformSegmentation(text, nSegments) ;
    /* Compute chunk count of each word type in the text                     */
    foreach segment in segmentedText do
        foreach wordType in segment do
            documentFrequency(wordType) ← documentFrequency(wordType) + 1 ;
        end
    end
    /* Compute word token counts in each chunk and the tfIdf weights         */
    foreach segment in segmentedText do
        foreach word in segment do
            termFrequency(word,segment) ← termFrequency(word) + 1 ;
        end
        foreach wordType in getWordTypes(segment) do
            idf ← log (nSegments ÷ documentFrequency(wordType));
            TfIdfMap(wordType, segment) ← termFrequency(word,segment) × idf ;
        end
    end
    return TfIdfMap ;
end
```

Function: `MinCutSeg`(*text, nSegments, params*)

Returns : the optimal segmentation of the text into the target number of segments

```
begin
    text ← Stem(text) ;
    WordFrequencyMap ← ComputeWordFrequencies(text) ;
    TfIdfWeights ← ComputeTfIdfWeights(WordFrequencyMap, text) ;
    WeightedFrequencyMap ← ApplyTfIdfWeights(WordFrequencyMap,TfIdfWeights) ;
    SentenceVectorNorms ← ComputeSentenceVectorNorms(WeightedFrequencyMap) ;
    foreach sentence_i in text do
        foreach sentence_j in text do
            s ← 0 ;
            foreach wordType in getWordTypes(sentence_i) ∩ getWordTypes(sentence_j) do
                s ← s + WeightedFrequencyMap(sentence_i,wordType) ×
                WeightedFrequencyMap(sentence_j,wordType) ;
            end
            s ← s ÷ [SentenceVectorNorms(sentence_i) × SentenceVectorNorms(sentence_j)] ;
            SimilarityMatrix(sentence_i, sentence_j) ← e^s ;
        end
    end
    S ← ApplyAnisotropicDiffusion(SimilarityMatrix,params) ;
    return ComputeOptimalSegmentation(S, nSegments) ;
end
```

Function: `ApplyAnisotropicDiffusion(`S, *params*`)`

Returns : apply anisotropic diffusion smoothing to the similarity matrix

```
begin
    numRows ← getNumRows(S) ;
    κ ← params.κ ;
    λ ← params.λ ;
    U ← makeNewMatrix(numRows, numRows) ;
    Temp ← makeNewMatrix(numRows, numRows) ;
    for t ← 0 to params.nIterations do
        ;
        for i ← 0 to numRows do
            for j ← 0 to numRows do
                dN ← dS ← dE ← dE ← cN ← cS ← cE ← cW ← 0 ;
                if i > 0 then dN ← S(i-1,j) - S(i,j) else dN ← S(i-1,j) - S(i,j)
                if i + 1 < numRows then dS ← S(i + 1,j) - S(i,j) else dS ← −S(i, j)
                if j + 1 < numRows then dE ← S(i,j + 1) - S(i,j) else dE ← −S(i, j)
                if j > 0 then dW ← S(i,j − 1) - S(i,j) else dW ← −S(i, j)
                cN ← 1 / (1 + (dN²) /(κ²)) ;
                cS ← 1 / (1 + (dS²) /(κ²)) ;
                cE ← 1 / (1 + (dE²) /(κ²)) ;
                cW ← 1 / (1 + (dW²) /(κ²)) ;
                U(i,j) ← S(i, j) + λ · (cN · dN + cS · dS + cE · dE + cW + dW) ;
                /* Swap the matrix for the previous iteration with the updated
                   similarity matrix                                            */
                Temp ← S ;
                S ← U ;
                U ← Temp ;
            end
        end
    end
    return S;
end
```

Function: `ComputeOptimalSegmentation(`*S, nSegments*`)`

Returns : the optimal segmentation of the text into the target number of segments. The boundary indices specify the index of the sentence before which the boundary is placed. The indices are 0-based, and the last boundary is always placed after the last sentence. The boundary before the first sentence is implicit.

```
begin
    nCutTable ← precomputeNormalizedCuts(S) ;
    backTraceTable ← runDynamicProgramming(nCutTable, nSegments) ;
    nRows ← getNumRows(backTraceTable) ;
    nCols ← getNumCols(backTraceTable) ;
    seg = makeNewVector() ;
    seg.add(nCols) ;
    i ← nRows -1 ;
    j ← nCols -1 ;
    /* The backtrace indices are inclusive:  i j ==> |i ..  j| ;
    So, add 1:  i j ==> |i ...j|j+1                              */
    while i > 0 do
        j ← backTraceTable(i,j);
        seg.add(j +1);
        i ← i -1 ;
    end
    reverseArray(seg) ;
    return seg ;
end
```

Function: `precomputeNormalizedCuts(`S`)`

Returns : the precomputed matrix of partial normalized cut terms $\frac{cut[A_{j,k}, V-A_{j,k}]}{vol[A_{j,k}]}$

```
begin
    nRows ← getNumRows ;
    nCols ← getNumCols ;
    nCutsTable ← makeNewMatrix (nRows,nCols) ;
    columnSum ← makeNewVector (nCols) ;
    for i ← 0 to nCols do
        for j ← 0 to numRows do
            columnSum(i) ← columnSum + S(j,i) ;
        end
    end
    /* Sum of entries S(startIndex:endIndex, startIndex:endIndex)          */
    intraSegmentVolume ← 0 ; lastIntraSegmentVolume ← 0 ;
    /* The Sum of columns from startIndex to endIndex                      */
    volume ← 0 ; lastVolume ← 0 ;
    for startIndex ← 0 to nRows-1 do
        for endIndex ← 0 to numRows-1 do
            if endIndex = startIndex then
                lastVolume ← 0; lastIntraSegmentVolume ← 0;
            end
            intraSegmentVolume ← 0 ;
            for i ← startIndex to endIndex-2 do
                intraSegmentVolume ← intraSegmentVolume + S(endIndex,i) ;
            end
            intraSegmentVolume ← intraSegmentVolume * 2 ;
            intraSegmentVolume ← intraSegmentVolume + lastIntraSegmentVolume +
            S(endIndex,endIndex) ;
            /* volume = assoc(A,V): associativity score of intraClass nodes and
               all other nodes in the graph                                    */
            volume ← lastVolume + columnSum(endIndex) ;
            cutValue ← volume - intraSegmentVolume ;
            nCutsTable(startIndex,endIndex) ← cutValue / volume ;
            lastIntraSegmentVolume ← intraSegmentVolume ;
            lastVolume ← volume ;
        end
    end
    return nCutsTable ;
end
```

```
Function: runDynamicProgramming(nCutsTable, numCuts)
Returns : The backTrace matrix which contains the optimal Normalized Cut segmentation
begin
    nRows ← getNumRows(nCutsTable); nCols ← getNumCols(nCutsTable) ;
    costMatrix ← makeNewMatrix(numCuts+1,nRows);
    backTrace ← makeNewMatrix(numCuts+1,nRows) ;
    for i ← 0 to nRows-1 do
        for j ← 0 to numCuts+1 do
            costMatrix(j,i) ← MAX_VALUE ;
            backTrace(j,i) ← -1 ;
        end
    end
    for i ← 0 to nCuts-1 do
        for j ← 0 to nRows-1 do
            if i = 0 then
                /* Assume first boundary is before the first sentence          */
                startIndex ← 0 ; endIndex ← j ;
                costMatrix(i,j) ← nCutsTable(startIndex,endIndex) ;
                backTrace(i,j) ← 0 ;
                continue;
            end
            if j = 0 and i > 0 then  continue ;
            scoreList ← makeNewVector() ;
            for k ← 0 to j-1 do
                cost ← costMatrix(i - 1,k) ;
                startIndex ← k + 1 ;
                endIndex ← j ;
                updatedCost ← cost + nCutsTable(startIndex,endIndex) ;
                pair ← makeNewPair(k, updatedCost) ;
                scoreList.add(pair) ;
            end
            minPair = findMin(scoreList) ;
            costMatrix(i,j) ← minPair.getValue() ;
            backTrace(i,j) ← minPair.getKey() ;
        end
    end
    return backTrace ;
end
```

Chapter 4

Experimental Results

In this chapter, we will analyze the performance of the minimum cut algorithm on spoken lecture data and compare our system with other state-of-the-art text segmentation systems. First, we explain the evaluation metrics used in our analysis and the human agreement results on the data. Then we examine the effect of long-range lexical dependencies employed by the model. In order to gauge its effectiveness, we compare our system with other leading segmentation systems on synthetic and spoken lecture data-sets. We also examine the effect of speech recognition error on segmentation accuracy. We conclude by experimenting with the problem of identifying lecture topic boundaries directly from acoustic features of the speech signal.

4.1 Evaluation Metrics

The scoring of text segmentation systems can be problematic in several respects. First, the true segment boundaries against which a hypothesized segmentation is to be scored may not be the only sensible way of partitioning a text. Different human subjects may segment a text at different levels of granularity and rely on different subjective criteria in judging whether a given text fragment constitutes a coherent topic. By choosing a single reference segmentation, we may penalize the system for not adhering to one segmentation standard among many admissible alternatives. We can control for this factor by looking at the extent of human agreement on spoken lecture data. This problem will be further explored in section 4.3 on human agreement analysis.

A second challenge is that the evaluation measures must be discriminating enough to pick up small differences between systems. For text segmentation, traditional classification evaluation measures such as precision and recall will be too coarse-grained to capture cases where there is a near mismatch between hypothesized and reference boundaries. It is necessary to employ a more flexible penalty measure, which will not use the zero-one loss to penalize near misses. We follow past segmentation literature in scoring the segmentation systems with the P_k and WindowDiff measures (Beeferman et al., 1999; Pevzner and Hearst, 2002). We also plot the Receiver Operating Characteristic (ROC) Curve to measure system performance at a finer level of discrimination (Swets, 1988).

4.1.1 P_k Measure

We can decompose the segmentation problem into a set of sub-tasks which aim to establish whether pairs of sentences from the text belong to the same segment. With this interpretation, a natural error metric is the probability that there would be a mismatch in the way that the hypothesis and the reference associate or disassociate a randomly chosen pair of sentences. We can compute this probability by marginalizing the joint probability of error and sentence pairs, s_i and s_j, conditioned on the reference (`ref`) and hypothesis (`hyp`) segmentations:

$$P(\text{error}|\text{ref, hyp}) = \sum_{i,j} P(\text{error}, s_i, s_j|\text{ref, hyp}) = \tag{4.1}$$

$$\sum_{i,j} P(s_i, s_j|\text{ref, hyp}) \cdot P(\text{error}|s_i, s_j, \text{ref, hyp}) \tag{4.2}$$

In order to compute this probability, we need to define a distribution over possible sentence pairs. One possible candidate is the uniform distribution. In practice, however, it is not desirable to assign equal weight to mistakes on pairs with different spans. For example, consider that sentences at different ends of a lecture will be classified correctly by most segmentation systems. Hence, a distribution for sentence pairs is chosen so that all probability mass will be distributed equally among word pairs that are exactly k words apart. Another common modification is to define the error metric over pairs of words, since sentences tend to vary markedly in length.

The P_k measure then is the probability that a randomly chosen pair of k words apart in the text is incorrectly classified. That is if in the hypothesis, the two words belong to the same segment, while in the reference they belong to different segments or vice versa. Since $P(\text{error}|s_i, s_j, \text{ref}, \text{hyp})$ is either 1 in case of mismatch or 0 in case of a match, and $P(s_i, s_j|\text{ref}, \text{hyp})$ is uniform over words placed k words apart, equation 4.2 reduces to the following formula:

$$P_k(ref, hyp) = \frac{1}{N-k} \sum_{i=1}^{N-k} (\delta_{ref}(i, i+k) \overline{\oplus} \delta_{hyp}(i, i+k)) \tag{4.3}$$

where $\overline{\oplus}$ is the xnor operator (it evaluates to 1 only if the two arguments are not equal), N is the number of words in the text. $\delta_{ref}(i, j)$ and $\delta_{hyp}(i, j)$ are indicator functions which evaluate to 1 if the two word indices fall within the same segment in the reference and hypothesis segmentations and 0 otherwise. k is a parameter typically set to half the average segment length. We follow Choi (2000) and compute the mean segment length used in determining the parameter k on each reference text separately

Intuitively, formula 4.3 can be interpreted as follows. We shift a window of k words across the text and determine if the terminal words at the ends of the window belong to the same segment for the reference and hypothesis segmentations. The overall penalty is the fraction of cases where the two indicator functions disagree.

In practice, the P_k measure exhibits high variability on real data. In fact, the notion of statistically significant difference in the P_k measure mean is ill-defined, because, strictly speaking, the P_k measure score is not comparable across two different transcripts with different mean segment lengths. Nevertheless, in order to be able to compare with past segmentation results we take the average of P_k measure scores across all the individual transcripts.

4.1.2 WindowDiff

Pevzner and Hearst (2002) presented a critique of the P_k measure. One of the problems they identify is that with greater variation in segment length, the measure becomes more lenient. The primary reason for this is that a penalty is registered only if the reference and hypothesis differ in their assignment of the word pair to the same segment or to two different segments. This approach will not identify errors where both the reference and the hypothesis assign words to different segments, yet in one segmentation there are more intervening segments than in the other. In other words, false positives or false negatives near actual boundaries may not be penalized.

To remedy this problem, Pevzner and Hearst introduced a variant on the P_k measure, the WindowDiff metric, which exacts a penalty only if the number of boundaries between positions i and j placed in the reference segmentation conflicts with the number of boundaries in the same span of the hypothesized segmentation. In other words, the new criterion becomes:

$$WindowDiff(ref, hyp) = \frac{1}{N-k} \sum_{i=1}^{N-k} (|b_{ref}(i, i+k) - b_{hyp}(i, i+k)| > 0),$$

where $b(i, i+k)$ represents the number of boundaries placed between the positions i and $i+k$ in the text.

4.1.3 Receiver Operating Characteristic Curve

Receiver Operating Characteristic (ROC) Curves are one of the standard ways of evaluating binary classifiers in machine learning literature (Swets, 1988). We apply this criterion for the evaluation of segmentation quality to yield a more refined analysis than the one possible with WindowDiff and P_k metrics.

Most classifiers assign test instances a score and decide the actual class of the instance by comparing this score against a threshold. As the threshold is adjusted to allow for more true positives, the false positives rate also goes up. The ROC plot is the plot of the true positive rate against the false positive rate for various settings of a decision threshold. Ideally, the true positive rate will increase at the cost of a minimal increase in false positives. So sharper ROC curves with larger areas under the curve indicate better discrimination performance.

In our case, the concept of a true positive and a false positive is not as straightforward as in many other settings, since the output of the system is not a single binary classification decision, but an entire set of boundaries. To be able to make use of this metric we take the threshold to be the distance from the original hypothesized boundaries within which all of the word positions will be considered hypothesized boundaries. In our case, the true positive rate is the fraction of boundaries correctly classified, and the false positive rate is the fraction of non-boundary positions incorrectly classified as boundaries. At zero distance

Corpus	Lectures	Segments per Lecture	Total Word Tokens	ASR WER Accuracy
Physics	33	5.9	232K	19.4%
AI	22	12.3	182K	$\times$

Table 4.1: Lecture Corpus Statistics

the original boundaries are taken as the set of hypotheses, and the raw true positive and false positive rates are computed. As the threshold distance is increased more and more of the reference boundaries will fall within the range of the hypothesized spans, but the number of false positives will increase as well. The advantage of the ROC curve is that it allows us to aggregate the error statistics from all of the test hypotheses and to visualize the correspondence between increasing accuracy and false positives.

4.2 Data

We evaluate our segmentation algorithm on three sets of data. Two of the datasets we use are new segmentation collections that we have compiled for this study, and the remaining set includes a standard collection previously used for evaluation of segmentation algorithms. In Appendix A, we provide examples of segmented transcripts from each of these sets. Various corpus statistics for the new datasets are presented in Table 4.1. Below we briefly describe each corpus.

4.2.1 Physics Lecture Corpus

Our first corpus consists of spoken lecture transcripts from an undergraduate Physics class. In contrast to other segmentation datasets, our corpus contains much longer texts. A typical lecture of 90 minutes has 500 to 700 sentences with 8500 words, which corresponds to about 15 pages of raw text. We have access both to manual transcriptions of these lectures and also output from an automatic speech recognition system. A speaker-dependent model of the lecturer was trained on 38 hours of lectures from other courses using the MIT Summit Speech Recognition System (Glass, 2003). The word error rate for the latter system on Physics lecture data is 19.4%, which is representative of state-of-the-art performance on lecture material (Leeuwis et al., 2003; Furui, 2003; Cettolo et al., 2004; Fugen et al., 2006).

In section 4.6, we will analyze the effect of speech recognition error on segmentation accuracy with speaker independent models.

The Physics lecture transcript segmentations were produced by the teaching staff of the Physics course at the Massachusetts Institute of Technology. Their objective was to facilitate access to lecture recordings available on the class website. This segmentation conveys the high-level topical structure of the lectures. On average, a lecture was annotated with six segments, and a typical segment corresponds to two pages of a transcript.

4.2.2 AI Lecture Corpus

Our second lecture corpus differs in subject matter, lecturing style, and segmentation granularity. The graduate Artificial Intelligence class has, on average, twelve segments per lecture, and a typical segment is about half of a page. One segment roughly corresponds to the content of a slide. This time the segmentation was obtained from the lecturer herself. The lecturer went through the transcripts of lecture recordings and segmented the lectures with the objective of making the segments correspond to presentation slides for the lectures that she intended to use the next time that she was going to teach the class. Due to the low recording quality, we were unable to obtain the ASR transcripts for this class. Therefore, we only use manual transcriptions of these lectures.

4.2.3 Synthetic Corpus

Also as part of our analysis, we used the synthetic corpus created by (Choi, 2000) which is commonly used in the evaluation of segmentation algorithms. This corpus consists of a set of concatenated segments randomly sampled from the Brown corpus. The length of the segments in this corpus ranges from three to eleven sentences. Again, it is important to underscore that the lexical transitions in these concatenated texts are very sharp, since the segments come from texts written in widely varying language styles on completely different topics.

4.3 Human Agreement Analysis

In order to be able to reliably score systems on the non-synthetic data, there needs to be a well-defined and consistent notion of a reference segment boundary.

	O	A	B	C
Mean Segment Count	6.6	8.9	18.4	13.8
Mean Segment Length	69.4	51.5	24.9	33.2
Segment Length Std. Dev.	39.6	37.4	34.5	39.4

Table 4.2: Annotator Segmentation Statistics for the first ten Physics lectures.

Ref/Hyp	O	A	B	C
O	0	**0.243**	0.418	0.312
A	0.219	0	0.400	0.355
B	0.314	0.337	0	0.332
C	0.260	0.296	0.370	0

Table 4.3: P_k annotation agreement between different pairs of annotators. Note that the measure is not symmetric.

Spoken lectures are very different in style from other corpora used in human segmentation studies (Hearst, 1994; Galley et al., 2003). We are interested in analyzing human performance on a corpus of lecture transcripts with much longer texts and a less clear-cut concept of a sub-topic.

As part of our human segmentation analysis, we asked three annotators to segment the Physics lecture corpus. These annotators had taken the class in the past and were familiar with the subject matter under consideration. We wrote a detailed instruction manual for the task,[1] with annotation guidelines for the most part following the model used by Gruenstein et al. (2005). The annotators were instructed to segment at a level of granularity that would identify most of the prominent topical transitions necessary for a summary of the lecture. The annotators used the NOMOS annotation software toolkit, developed for meeting segmentation (Gruenstein et al., 2005).

The annotators were provided with recorded audio of the lectures and the corresponding text transcriptions. We intentionally did not provide the subjects with the target number of boundaries, since we wanted to see if the annotators would converge on a common segmentation granularity.

Table 4.2 presents the annotator segmentation statistics. We see two classes of segmentation granularities. The original reference (O) and annotator A segmented at a coarse

[1]The instructions are included in appendix A

level with an average of 6.6 and 8.9 segments per lecture, respectively. Annotators B and C operated at much finer levels of discrimination with 18.4 and 13.8 segments per lecture on average. We conclude that multiple levels of granularity are acceptable in spoken lecture segmentation. This is expected given the length of the lectures and varying human judgments in selecting relevant topical content.

Following previous studies, we quantify the level of annotator agreement with the P_k measure (Gruenstein et al., 2005).[2] Table 4.3 shows the annotator agreement scores between different pairs of annotators. The majority of the three annotators agree on the exact placement of a third of all of the boundaries, not counting the boundaries at the very beginning and end of the texts.

P_k measures ranged from 0.24 and 0.42. We observe greater consistency at similar levels of granularity, and less so across the two classes. Note that annotator A operated at a level of granularity consistent with the original reference segmentation. Hence, the 0.24 P_k measure score serves as the benchmark result with which we can compare the results attained by segmentation algorithms on the Physics lecture data. As an additional point of reference we note that the uniform and random baseline segmentations attain 0.469 and 0.493 P_k measure, respectively, on the Physics lecture set. From the agreement results, we can conclude that the lecture segmentation problem is difficult even for humans. However, the task exhibits a high degree of regularity, and most cases of disagreement correspond either to different conceptions of granularity or different approaches of addressing spoken discourse artifacts such as off-topic remarks, audience-speaker interaction, or non-topical, presentational changes. Barring these peculiarities, the concept of a topic is uncontroversial and quite natural.

4.3.1 Setup and Parameter Estimation

A heldout development set of three lectures is used for estimating the optimal window length, the distance thresholds for discarding node edges, the number of uniform chunks for estimating Tf-Idf lexical weights, and the anisotropic diffusion smoothing parameters which

[2] Kappa measure would not be the appropriate measure in this case, because it is not sensitive to near misses, and we cannot make the required independence assumption on the placement of boundaries. Cochran's Q test used previously to assess agreement in text segmentation also is not applicable here. Passoneau and Litman (1997) assume that annotators assign a fixed number of boundaries, which does not hold in our case.

include the lambda and kappa parameters, and the target number of iterations.

One problem is that we do not have access to derivatives of the P_k or WindowDiff metric with respect to the parameters. The functional dependence between these metrics and parameters is a step function with discontinuities at every point of change in the dependent variable. What's more this function is highly non-linear and sensitive to the features of the data. Nevertheless, there are several search and optimization algorithms which could potentially be used, including line search and simulated annealing. One point to keep in mind is that each evaluation of the objective function involves the evaluation of the Minimum Cut algorithm on three development lectures, which may take up to a second. So, the number of evaluations should ideally be kept to a minimum.

We use a greedy search procedure for optimizing the parameters, because it has a small footprint in terms of both time and memory requirements. Each parameter is optimized on a grid of parameters values, with other parameters kept fixed. After all of the parameters have been optimized, the search is repeated on a refined grid, until the objective value converges to a local minimum. Apart from computational efficiency, an added advantage of this method is that it will be unlikely to overfit the parameters on the development data.

Finally, in our experiments, the number of target segments is set to that of the reference segmentation for both the Minimum Cut system and the baselines.

4.4 Long-Range Dependency Analysis

We first determine the impact of long-range pairwise similarity dependencies on segmentation performance. Our key hypothesis is that considering long-distance lexical relations contributes to the effectiveness of the algorithm. To test this hypothesis, we discard edges between nodes that are more than a certain number of sentences apart. We test the system on a range of data sets, including the Physics and AI lectures and the synthetic corpus created by Choi (2000).

The results in Table 4.4 confirm our hypothesis — taking into account non-local lexical dependencies helps across different domains. On manually transcribed Physics lecture data, for example, when the algorithm takes into account edges separated by up to a hundred sentences, it yields 26% lower P_k measure (0.279) than when it considers dependencies up to ten sentences (0.380). Figure 4-1 shows the ROC plot for the segmentation of the Physics

Edge Cutoff						
	10	25	50	100	200	none
Physics (Manual)						
PK	0.3802	0.3527	0.3149	**0.2788**	0.3034	0.3200
WD	0.3927	0.3632	0.3292	**0.2962**	0.3281	0.3505
AI						
PK	0.4375	0.3893	**0.3610**	0.3680	0.4035	0.3936
WD	0.4515	0.4046	**0.3799**	0.3892	0.4296	0.4186
Choi						
PK	**0.1483**	0.1693	0.1830	0.1855	0.1855	0.1855
WD	**0.1840**	0.2104	0.2347	0.2337	0.2337	0.2337

Table 4.4: Edges between nodes separated beyond a certain threshold distance are removed.

lecture data with different cutoff parameters, again demonstrating clear gains attained by employing long-range dependencies. As Table 4.4 shows, the improvement is consistent across all spoken lecture datasets. We note, however, that after some point increasing the threshold may degrade performance, because it introduces too many spurious dependencies (see the last column of Table 4.4). The speaker will occasionally return to a topic described at the beginning of the lecture, and this will bias the algorithm to put the segment boundary closer to the end of the lecture.

Long-range dependencies do not improve the performance on the synthetic dataset. This is expected since the segments in the synthetic dataset are randomly selected from widely-varying documents in the Brown corpus, even spanning different genres of written language. So, effectively, there are no genuine long-range dependencies that can be exploited by the algorithm.

4.5 Comparison with Local Models

We compare our system with the state-of-the-art similarity-based segmentation system developed by Choi(2000). We use the publicly available implementation of the system and optimize the system on a range of mask-sizes and different parameter settings described in (Choi, 2000) on a heldout development set of three lectures. To control for segmentation granularity, we specify the number of segments in the reference segmentation for both our system and the baseline. Table 4.5 shows that the Minimum Cut algorithm consistently outperforms the similarity-based baseline on all the lecture datasets. We attribute this

	Choi	UI	MinCut
Physics (Manual)			
PK	0.372	0.310	**0.281**
WD	0.385	0.323	**0.301**
AI			
PK	0.445	**0.374**	0.378
WD	0.478	0.420	**0.393**
Choi			
PK	0.110	**0.105**	0.133
WD	0.121	**0.116**	0.154

Table 4.5: Performance analysis of different algorithms on the corpora, with three lectures heldout for development.

gain to the presence of more attenuated topic transitions in spoken language. Since spoken language is more spontaneous and less structured than written language, the speaker needs to keep the listener abreast of the changes in topic content by introducing subtle cues and references to prior topics in the course of topical transitions. Non-local dependencies help to elucidate shifts in focus, because the strength of a particular transition is measured with respect to other local and long-distance contextual discourse relationships.

Our system does not outperform Choi's algorithm on the synthetic data. This again can be attributed to the discrepancy in distributional properties of the synthetic corpus which lacks coherence in its thematic shifts and the lecture corpus of spontaneous speech with smooth distributional variations. We also note that we did not try to adjust our model to optimize its performance on the synthetic data. The smoothing method developed for lecture segmentation may not be appropriate for short segments ranging from three to eleven sentences that constitute the synthetic set.

We also compared our method with another state-of-the-art algorithm which does not explicitly rely on pairwise similarity analysis. This algorithm (UI) computes the optimal segmentation by estimating changes in the language model predictions over different partitions (Utiyama and Isahara, 2001). We used the publicly available implementation of the system that does not require parameter tuning on a heldout development set.

Again, our method achieves favorable performance on a range of lecture data sets (See Table 4.5), and both algorithms attain results close to the range of human agreement scores.

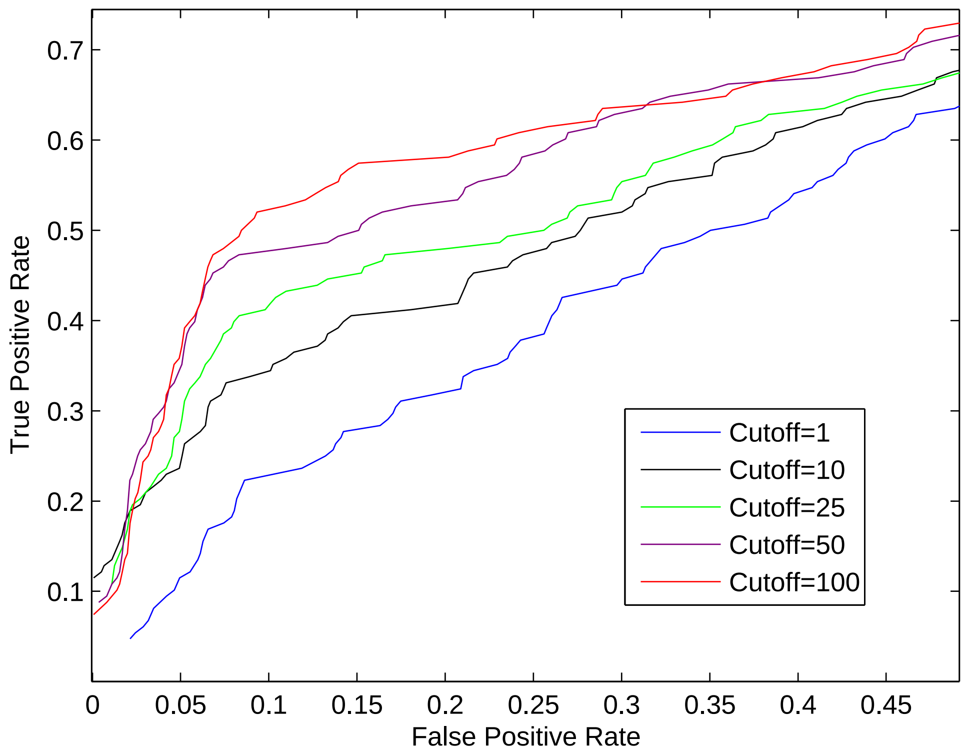

Figure 4-1: ROC plot for the Minimum Cut Segmenter on thirty Physics Lectures, with edge cutoffs ranging from one to hundred sentences.

	SD	SI$^+$	SI$^-$
%WER	18.4	32.7	44.9

Table 4.6: Word Error Rates for different ASR Models

4.6 Effect of Speech Recognition Accuracy

In order to determine how robust our method is in the presence of transcription errors, we analyzed its performance on Automatic Speech Recognition (ASR) transcripts with various levels of word error.

The three speech recognition models used to generate these transcript sets were the speaker-dependent model (SD), the speaker independent model (SI$^+$) with speech samples of the speaker included in the training data, and finally the speaker independent model (SI$^-$) with all instances of the test speaker's utterances removed from training (See Table 4.6 for for their respective word error rates).

System	SD	SI$^+$	SI$^-$
P_k Measure			
MinCut	0.3023	0.3329	0.3302
UI	0.3220	0.3183	0.3527
WindowDiff Measure			
MinCut	0.3183	0.3469	0.3474
UI	0.3369	0.3324	0.3664

Table 4.7: Segmentation Results on transcripts with different levels of word error

The MinCut and the UI segmentation system were tested on each of these ASR transcript sets. The results in Table 4.7 show that the minimum cut system is robust in noisy speech environments. In fact for two of the three test conditions it outperforms the UI baseline, and it comes close to the results derived from the manually transcribed data.

4.7 Speech Segmentation Experiments

In this section, we demonstrate that our algorithm is not only applicable in settings where words and lexical similarity information is available. We include a proof-of-concept experiment with segmentation of acoustic signal without any intermediate speech recognition processing.

4.7.1 Unsupervised Pattern Discovery in Speech

We obtain the representation of speech from automatically derived word clusters, generated by Park's unsupervised word acquisition method (Park, 2006). We note that we only use the intermediate similarity representation derived from this method, as the actual word clusters computed would be too sparse to give us a rich enough representation which could enable us to discern changes in lexical distribution. Many of the words occurring only a few times in the text are pruned away by this method, even though the cumulate sum of these items is enough to have a dramatic impact on the results. Below, we outline the steps for the feature extraction approach.

Signal Processing The speech is converted into a time series of Mel-scale cepstral coefficients (MFCCs), the representation most commonly used in speech recognition. The

Target Words		
direction	half seconds	acceleration
Aligned Words		
direction which	per second	acceleration
direction and	per second squared	acceleration
that action	a second square	acceleration
y direction	seconds	explanation
direction the	per second squared	rotation
direction trays		calculation
direction		acceleration

Table 4.8: Aligned Word Paths

SUMMIT speech recognizer front-end is used for signal processing (Glass, 2003).

This process can be summarized as follows. After capturing the acoustic signal as a digital waveform sampled at a rate of 16 kHz, the waveform mean and magnitude is normalized. The short-time Fourier transform is taken with a frame interval of 10 ms, a 25.6 ms Hamming window, and a 256 point discrete Fourier transform. The spectral energy from the Fourier Transform then is weighted by the Mel-frequency filters, and finally the discrete cosine transform of the log of Mel-frequency spectral coefficients is computed, yielding a series of 14-dimensional MFCC vectors.

Segmental DTW A variation of the Dynamic Time Warping algorithm is used to align most similar fragments of speech in the lecture (Park and Glass, 2006). First, the distance matrix is generated by computing distances between the MFCC vectors for pairs of utterances. The matrix is cut into diagonal bands with a fixed width to limit the amount of distortion in the aligned paths. Optimal paths with the lowest distortion cost through the bands are found by the Dynamic Time Warping Algorithm. Each path is then trimmed to the least average subsequence (See Figure 4-2). The average of the sequence distortion profile is subtracted from the maximum distortion, yielding a similarity profile over time. Table 4.8 shows some examples of aligned word paths in a Physics transcript.

4.7.2 Speech Minimum Cut

Once the highest scoring paths are extracted for each pair of utterances and the similarity score is computed, we employ this information to develop a suitable representation for the

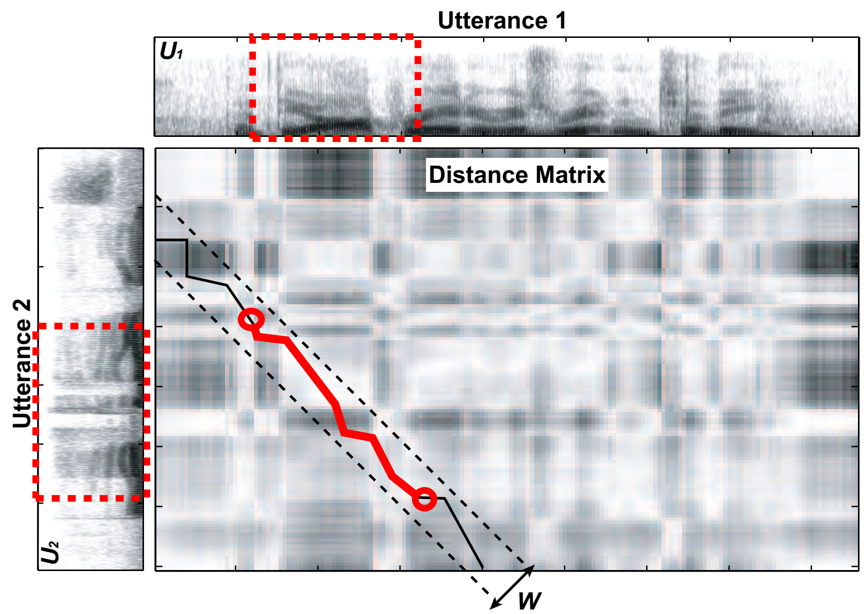

Figure 4-2: Illustration of Dynamic Time Warping from (Park and Glass, 2006).

mincut algorithm. In its original form the similarity profile is too sparse There are gaps between aligned utterance fragments and they also differ in duration.

In order to use our system, we quantize the data by splitting the lecture into contiguous time blocks to make the nodes in the similarity profile more uniform. We aggregate the similarity scores for paths that fall within these time blocks. More formally if $S(p_i, p_j)$ is the similarity score for the aligned paths p_i and p_j, and $B(p_i)$ is the index of the time block within which the start-time of path p_i falls, then the similarity between the time blocks is computed as follows: $S(b_i, b_j) = \sum_{p_i \in A, p_j \in B} S(p_i, p_j)$, where $A = \{p_i | B(p_i) = b_i\}$ and $B = \{p_i | B(p_j) = b_j\}$.

After quantization, we use the anisotropic diffusion method proposed in (Perona and Malik, 1990) to smooth the similarity matrix (See section 3.4). A sample lecture similarity matrix is shown in figure 4-3. Since the matrix is symmetric, only the upper portion of the matrix is shown. Each element of the matrix corresponds to a rectangular patch in the image. The matrix entries determine the color of each patch. The values are scaled to the range of a colormap ranging from blue to red. The intensity of the red color indicates the degree of acoustic similarity. Vertical lines in the image are reference segment boundaries. Again, here we see that concentrated patches of similarity correspond to topical segments in a lecture.

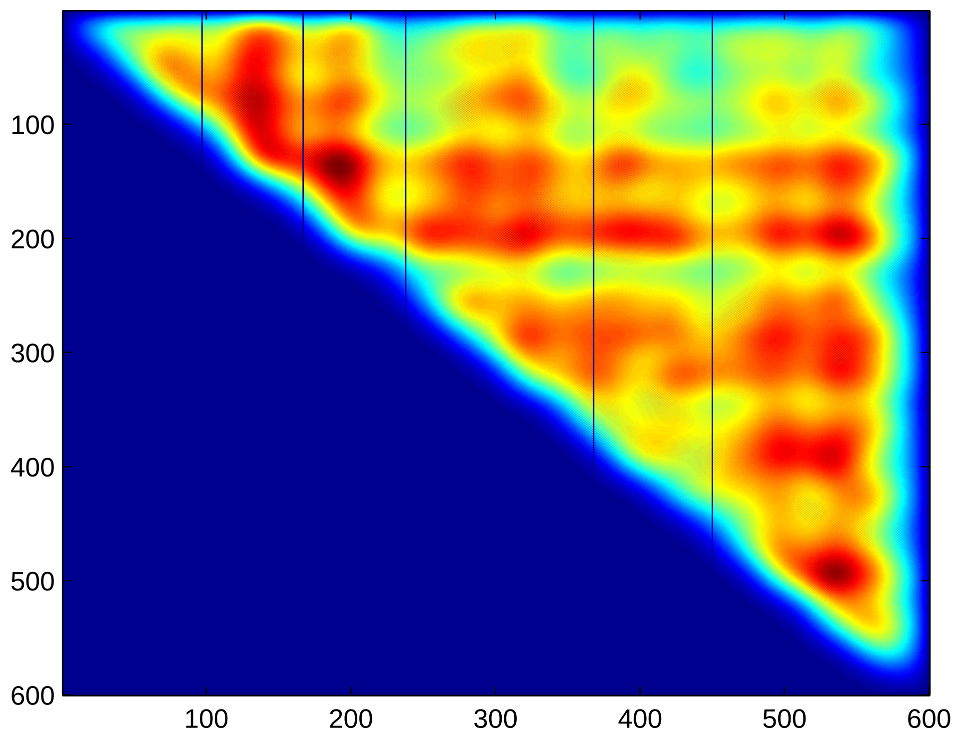

Figure 4-3: Smoothed word-fragment similarity matrix of a Physics lecture. The degree of block similarity is captured by the intensity of the color, ranging from blue (lowest similarity) to red (highest similarity).

We tuned the number of quantized blocks as well as the kappa, lambda parameters, and the number of iterations in the anisotropic algorithm on a heldout set of three development lectures.

With the target number of segments set to the reference number of segments, the minimum cut segmenter on the set of 33 Physics lectures yields 0.38 average P_k measure and 0.3933 average WindowDiff measure. This result is significantly better than the scores attained by uniform and random segmentations, and is close to the performance of the Choi baseline on the Physics lecture set. In some of the individual lectures, the resulting segmentation actually improved on the minimum cut text segmentation result, but the overall result is worse perhaps owing to noise and acoustic irregularities.

We note that it would not be possible to incorporate the acoustic similarity information for the UI baseline, because this algorithm operates over text ngrams.

4.8 Conclusion

In this chapter, we layed out the the experimental basis for the effectiveness of our algorithm. In particular, we showed that the task of spoken lecture segmentation is qualitatively different from previous segmentation tasks with written language on synthetic corpora. The new features of this problem are not well modeled by the previous algorithms, which principally relied on assumptions of locality in lexical similarity changes to discern boundaries. Our results show that being able to exploit the global characteristics of the similarity distribution is critical in our ability to model spoken discourse topics.

Our new framework attains the new state-of-the-art baseline in spoken lecture segmentation. Moreover, we demonstrate that the method is applicable in a variety of other segmentation scenarios where object similarity information is available. We show that our framework allows us to find topics from raw acoustic information, which is a highly promising result, since it obviates the need for any intermediate speech recognition.

Chapter 5

Conclusions and Future Work

In this thesis we presented a novel framework for domain-independent text segmentation. We modeled text segmentation as a graph-partitioning task aiming to simultaneously optimize the total similarity within each segment and dissimilarity across various segments. We showed that our method is able to handle long-range lexical dependencies through global analysis of lexical distribution and is robust in the presence of recognition errors. Combining this type of analysis with advanced methods for smoothing (Ji and Zha, 2003) and weighting could further boost the performance of algorithms on the problem of lecture segmentation.

We analyzed variations in the segmentation performance on a range of testing conditions. Not surprisingly, the performance of the algorithms depends on the distributional properties of the input text. We found that the segmentation accuracy on the synthetic set is not predictive of the performance on real data. These results strongly suggest that segmentation algorithms have to be evaluated on a collection of texts displaying real-world variability.

In the course of our work we experimented with techniques for refining the lexical similarity measure with Latent Semantic Analysis, more powerful lexical weighting techniques, and various clustering methods. We also tested ways of merging unsupervised and supervised models for text segmentation. However, we were not able to improve upon the current segmentation system. These ideas are worth further exploration.

Our current implementation also does not automatically determine the granularity of a resulting segmentation. This issue has been explored in the past (Ji and Zha, 2003; Utiyama and Isahara, 2001), and we will explore the existing strategies in our framework. We believe that the algorithm has to produce segmentations for various levels of granularity, depending on the needs of the application that employs it.

Finally, we would like to test our system on the other spoken language corpora, and attempt to model hierarchical segmentation within the minimum cut framework.

Our ultimate goal is to be able to summarize spoken lectures. We will explore how the interaction between the segmentation and content selection, ordering and generation components can improve the performance of such a system as a whole.

Appendix A

Physics and AI Lecture Examples

A.1 Physics Lecture

1 <section 1>

2 In physics, we explore the very small to the very large.

3 The very small is a small fraction of a proton and the very large is the universe itself.

4 They span 45 orders of magnitude-- a 1 with 45 zeroes.

5 To express measurements quantitatively we have to introduce units.

6 And we introduce for the unit of length, the meter; for the unit of time, the second; and for the unit of mass, the kilogram.

7 Now, you can read in your book how these are defined and how the definition evolved historically.

8 Now, there are many derived units which we use in our daily life for convenience and some are tailored toward specific fields.

9 We have centimeters, we have millimeters kilometers.

10 We have inches, feet, miles.

11 Astronomers even use the astronomical unit which is the mean distance between the Earth and the sun and they use light-years which is the distance that light travels in one year.

12 We have milliseconds, we have microseconds we have days, weeks, hours, centuries, months-- all derived units.

13 For the mass, we have milligrams, we have pounds we have metric tons.

14 So lots of derived units exist.

15 Not all of them are very easy to work with.

16 I find it extremely difficult to work with inches and feet.

17 It's an extremely uncivilized system.

18 I don't mean to insult you, but think about it-- 12 inches in a foot, three feet in a yard.

19 Could drive you nuts.

20 I work almost exclusively decimal, and I hope you will do the same during this course but we may make some exceptions.

21 </section 1>

22 <section 2>

23 I will now first show you a movie, which is called The Powers of Ten.

24 It covers 40 orders of magnitude.

25 It was originally conceived by a Dutchman named Kees Boeke in the early '50s.

26 This is the second-generation movie, and you will hear the voice of Professor Morrison, who is a professor at MIT.

27 The Power of Ten-- 40 Orders of Magnitude.

28 Here we go.

29 MORRISON: 1 October.

30 We begin with a scene 1 meter wide which we view from just 1 meter away.

31 Now, every 10 seconds we will look from 10 times farther away and our field of view will be 10 times wider.

32 This square is 10 meters wide and in 10 seconds, the next square will be 10 times as wide.

33 Our picture will center on the picnickers even after they have been lost to sight.

34 100 meters wide-- the distance a man can run in 10 seconds.

35 Cars crowd the highway, powerboats lie at their docks.

36 The colorful bleachers are Soldiers' Field.

37 This square is a kilometer wide-- 1,000 meters-- the distance a racing car can travel in 10 seconds.

38 We see the great city on the lake shore.

39 10 to the fourth meters-- 10 kilometers the distance a supersonic airplane can travel in 10 seconds.

40 We see first the rounded end of Lake Michigan then the whole Great Lake.

41 10 to the fifth meters-- the distance an orbiting satellite covers in 10 seconds.

42 Long parades of clouds, the day's weather in the middle west.

43 10 to the sixth-- a 1 with six zeros, a million meters.

44 Soon the Earth will show as a solid sphere.

45 We are able to see the whole Earth now just over a minute along the journey.

46 Earth diminishes into the distance but those background stars are so much farther away they do not yet appear to move.

47 A line extends at the true speed of light.

48 In one second, it half crosses the tilted orbit of the moon.

49 Now we mark a small part of the path in which the Earth moves about the Sun.

50 Now the orbital paths of the neighbor planets.

51 Venus... and Mars... then Mercury.

52 Entering our field of view is the glowing center of our solar system, the Sun followed by the massive outer planets swinging wide in their big orbits.

53 That outer orbit belongs to Pluto.

54 A fringe of a myriad comets too faint to see completes the solar system.

55 10 to the 14th-- as the solar system shrinks the one bright point in the distance our sun is plainly now only one among the stars.

56 Looking back from here we note four southern constellations still much as they appear from the far side of the Earth.

57 This square is 10 to the 16th meters, one light-year not yet out to the next star.

58 Our last 10-second step took us 10 light-years further.

59 The next will be a hundred.

60 Our perspective changes so much in each step now that even the background stars will appear to converge.

61 At last, we pass the bright star Arcturus and some stars of the Dipper.

62 Normal but quite unfamiliar stars and clouds of gas surround us, as we traverse the Milky Way galaxy.

63 Giant steps carry us into the outskirts of the galaxy and as we pull away, we begin to see the great flat spiral facing us.

64 The time and path we chose to leave Chicago has brought us out of the galaxy along a course nearly perpendicular to its disc.

65 The two little satellite galaxies of our own are the Clouds of Magellan-- 10 to the 22nd power, a million light-years.

66 Groups of galaxies bring a new level of structure to the scene.

67 Glowing points are no longer single stars but whole galaxies of stars seen as one.

68 We pass the big Virgo cluster of galaxies, among many others.

69 100 million light-years out.

70 As we approach the limit of our vision we pause to start back home.

71 This lonely scene, the galaxies like dust is what most of space looks like.

72 This emptiness is normal.

73 The richness of our own neighborhood is the exception.

The trip back to the picnic on the lakefront will be a sped-up version
reducing the distance to the Earth's surface by one power of 10 every two
74 seconds.

In each two seconds, we will appear to cover 90% of the remaining distance
75 back to Earth.

Notice the alternation between great activity and relative inactivity a
rhythm that will continue all the way into our next goal-- a proton in the
nucleus of a carbon atom beneath the skin on the hand of the sleeping man
76 at the picnic.

10 to the ninth meters... 10 to the eighth... seven... six... five...
77 four... three... two... one.

78 We are back at our starting point.

79 We slow up at 1 meter, 10 to the zero power.

Now we reduce the distance to our final destination by 90% every 10
80 seconds, each step much smaller than the one before.

At 10 to the minus two-- 1/100th of a meter, one centimeter-- we approach
81 the surface of the hand.

In a few seconds, we'll be entering the skin crossing layer after layer
82 from the outermost dead cells into a tiny blood vessel within.

Skin layers vanish in turn-- an outer layer of cells, felty collagen a
83 capillary containing red blood cells and a ruffly lymphocyte.

84 We enter the white cell.

85 Among its vital organelles the porous wall of the cell nucleus appears.

The nucleus within holds the heredity of the man in the coiled coils of
86 DNA.

As we close in, we come to the double helix itself a molecule like a long,
twisted ladder whose rungs of paired bases spell out twice in an alphabet
87 of four letters the words of a powerful genetic message.

88 At the atomic scale, the interplay of form and motion becomes more visible.

We focus on one commonplace group of three hydrogen atoms bonded by
89 electrical forces to a carbon atom.

90 Four electrons make up the outer shell of the carbon itself.

91 They appear in quantum motion as a swarm of shimmering points.

At 10 to the minus 10 meters, one angstrom we find ourselves right among
92 those outer electrons.

93 Now we come upon the two inner electrons held in a tighter swarm.

As we draw toward the atom's attracting center we enter upon a vast inner
94 space.

95 At last, the carbon nucleus.

So massive and so small this carbon nucleus is made up of six protons and
96 six neutrons.

97 We are in a domain of universal modules.

There are protons and neutrons in every nucleus electrons in every atom
98 atoms bonded into every molecule, out to the farthest galaxy.

As a single proton fills our scene we reach the edge of present
99 understanding.

100 Are these some quarks at intense interaction?

101 Our journey has taken us through 40 powers of 10.

If now the field is one unit then, when we saw many clusters of galaxies
102 together it was 10 to the 40th, or 1 and 40 zeroes.

103 </section 2>

104 <section 3>

I already introduced, as you see there length, time and mass and we call
105 these the three fundamental quantities in physics.

I will give this the symbol capital L for length capital T for time, and
106 capital M for mass.

All other quantities in physics can be derived from these fundamental
107 quantities.

108 I'll give you an example.

109 I put a bracket around here.

110 I say [speed] and that means the dimensions of speed.

The dimensions of speed is the dimension of length divided by the dimension
111 of time.

112 So I can write for that: [L] divided by [T].

113 Whether it's meters per second or inches per year that's not what matters.

114 It has the dimension length per time.

115 Volume would have the dimension of length to the power three.

Density would have the dimension of mass per unit volume so that means
116 length to the power three.

117 All-important in our course is acceleration.

118 We will deal a lot with acceleration.

119 Acceleration, as you will see, is length per time squared.

120 The unit is meters per second squared.

121 So you get length divided by time squared.

122 So all other quantities can be derived from these three fundamental.

123 </section 3>

124 <section 4>

125 So now that we have agreed on the units-- we have the meter, the second and the kilogram-- we can start making measurements.

126 Now, all-important in making measurements which is always ignored in every college book is the uncertainty in your measurement.

127 Any measurement that you make without any knowledge of the uncertainty is meaningless.

128 I will repeat this.

129 I want you to hear it tonight at 3:00 when you wake up.

130 Any measurement that you make without the knowledge of its uncertainty is completely meaningless.

131 My grandmother used to tell me that... at least she believed it... that someone who is lying in bed is longer than someone who stands up.

132 And in honor of my grandmother I'm going to bring this today to a test.

133 I have here a setup where I can measure a person standing up and a person lying down.

134 It's not the greatest bed, but lying down.

135 I have to convince you about the uncertainty in my measurement because a measurement without knowledge of the uncertainty is meaningless.

136 And therefore, what I will do is the following.

137 I have here an aluminum bar and I make the reasonable, plausible assumption that when this aluminum bar is sleeping-- when it is horizontal-- that it is not longer than when it is standing up.

138 If you accept that, we can compare the length of this aluminum bar with this setup and with this setup.

139 At least we have some kind of calibration to start with.

140 I will measure it.

141 You have to trust me.

142 During these three months, we have to trust each other.

143 So I measure here, 149.9 centimeters.

144 However, I would think that the... so this is the aluminum bar.

145 This is in vertical position.

146 149.9. But I would think that the uncertainty of my measurement is probably 1 millimeter.

147 I can't really guarantee you that I did it accurately any better.

148 So that's the vertical one.

149 Now we're going to measure the bar horizontally for which we have a setup here.

150 Oop!

151 The scale is on your side.

152 So now I measure the length of this bar.

153 150.0 horizontally.

154 150.0, again, plus or minus 0.1 centimeter.

155 So you would agree with me that I am capable of measuring plus or minus 1 millimeter.

156 That's the uncertainty of my measurement.

157 Now, if the difference in lengths between lying down and standing up if that were one foot we would all know it, wouldn't we?

158 You get out of bed in the morning you lie down and you get up and you go, clunk!

159 And you're one foot shorter.

160 And we know that that's not the case.

161 If the difference were only one millimeter we would never know.

162 Therefore, I suspect that if my grandmother was right then it's probably only a few centimeters, maybe an inch.

163 And so I would argue that if I can measure the length of a student to one millimeter accuracy that should settle the issue.

164 So I need a volunteer.

165 You want to volunteer?

166 You look like you're very tall.

167 I hope that... yeah, I hope that we don't run out of, uh... You're not taller than 178 or so?

168 What is your name?

169 STUDENT: Rick Ryder.

170 LEWIN: Rick-- Rick Ryder.

171 You're not nervous, right?

172 RICK: No!

173 LEWIN: Man!

174 (class laughs) Sit down.

175 (class laughs) I can't have tall guys here.

176 Come on.

177 We need someone more modest in size.

178 Don't take it personal, Rick.

179 Okay, what is your name?

180 STUDENT: Zach.

181 LEWIN: Zach.

182 Nice day today, Zach, yeah?

183 You feel all right?

184 Your first lecture at MIT?

185 I don't.

186 Okay, man.

187 Stand there, yeah.

188 Okay, 183.2. Stay there, stay there.

189 Don't move.

190 Zach... This is vertical.

191 What did I say?

192 180?

193 Only one person.

194 183?

195 Come on.

196 .2-- Okay, 183.2. Yeah.

197 And an uncertainty of about one... Oh, this is centimeters-- 0.1 centimeters.

198 And now we're going to measure him horizontally.

199 Zach, I don't want you to break your bones so we have a little step for you here.

200 Put your feet there.

201 Oh, let me remove the aluminum bar.

202 Watch out for the scale.

203 That you don't break that, because then it's all over.

Okay, I'll come on your side.
I have to do that-- yeah, yeah.
Relax.
Think of this as a small sacrifice for the sake of science, right?
Okay, you good?
ZACH: Yeah.
LEWIN: You comfortable?
(students laugh) You're really comfortable, right?
ZACH: Wonderful.
LEWIN: Okay.
You ready?
ZACH: Yes.
LEWIN: Okay.
Okay.
185.7. Stay where you are.
185.7. I'm sure... I want to first make the subtraction, right?
185.7, plus or minus 0.1 centimeter.
Oh, that is five... that is 2.5 plus or minus 0.2 centimeters.
You're about one inch taller when you sleep than when you stand up.
My grandmother was right.
She's always right.
Can you get off here?
I want you to appreciate that the accuracy... Thank you very much, Zach.
That the accuracy of one millimeter was more than sufficient to make the case.
If the accuracy of my measurements would have been much less this measurement would not have been convincing at all.
So whenever you make a measurement you must know the uncertainty.
Otherwise, it is meaningless.
</section 4>
<section 5>
Galileo Galilei asked himself the question: Why are mammals as large as they are and not much larger?

234 He had a very clever reasoning which I've never seen in print.

235 But it comes down to the fact that he argued that if the mammal becomes too massive that the bones will break and he thought that that was a limiting factor.

236 Even though I've never seen his reasoning in print I will try to reconstruct it what could have gone through his head.

237 Here is a mammal.

238 And this is one of the four legs of the mammal.

239 And this mammal has a size S.

240 And what I mean by that is a mouse is yay big and a cat is yay big.

241 That's what I mean by size-- very crudely defined.

242 The mass of the mammal is M and this mammal has a thigh bone which we call the femur, which is here.

243 And the femur of course carries the body, to a large extent.

244 And let's assume that the femur has a length l and has a thickness d.

245 Here is a femur.

246 This is what a femur approximately looks like.

247 So this will be the length of the femur... and this will be the thickness, d and this will be the cross-sectional area A.

248 I'm now going to take you through what we call in physics a scaling argument.

249 I would argue that the length of the femur must be proportional to the size of the animal.

250 That's completely plausible.

251 If an animal is four times larger than another you would need four times longer legs.

252 And that's all this is saying.

253 It's very reasonable.

254 It is also very reasonable that the mass of an animal is proportional to the third power of the size because that's related to its volume.

255 And so if it's related to the third power of the size it must also be proportional to the third power of the length of the femur because of this relationship.

256 Okay, that's one.

257 Now comes the argument.

258 Pressure on the femur is proportional to the weight of the animal divided by the cross-section A of the femur.

259 That's what pressure is.

And that is the mass of the animal that's proportional to the mass of
the animal divided by d squared because we want the area here, it's
260 proportional to d squared.

261 Now follow me closely.

262 If the pressure is higher than a certain level the bones will break.

Therefore, for an animal not to break its bones when the mass goes up by
a certain factor let's say a factor of four in order for the bones not to
263 break d squared must also go up by a factor of four.

264 That's a key argument in the scaling here.

265 You really have to think that through carefully.

266 Therefore, I would argue that the mass must be proportional to d squared.

267 This is the breaking argument.

268 Now compare these two.

The mass is proportional to the length of the femur to the power three and
269 to the thickness of the femur to the power two.

Therefore, the thickness of the femur to the power two must be proportional
to the length l and therefore the thickness of the femur must be
270 proportional to l to the power three-halfs.

271 A very interesting result.

272 What is this result telling you?

It tells you that if I have two animals and one is ten times larger than
the other then S is ten times larger that the lengths of the legs are ten
times larger but that the thickness of the femur is 30 times larger because
273 it is l to the power three halves.

If I were to compare a mouse with an elephant an elephant is about a
hundred times larger in size so the length of the femur of the elephant
would be a hundred times larger than that of a mouse but the thickness of
274 the femur would have to be 1,000 times larger.

And that may have convinced Galileo Galilei that that's the reason why the
275 largest animals are as large as they are.

Because clearly, if you increase the mass there comes a time that the
276 thickness of the bones is the same as the length of the bones.

277 You're all made of bones and that is biologically not feasible.

278 And so there is a limit somewhere set by this scaling law.

279 Well, I wanted to bring this to a test.

After all I brought my grandmother's statement to a test so why not bring
280 Galileo Galilei's statement to a test?

And so I went to Harvard where they have a beautiful collection of femurs
281 and I asked them for the femur of a raccoon and a horse.

A raccoon is this big a horse is about four times bigger so the length of
282 the femur of a horse must be about four times the length of the raccoon.

283 Close.

284 So I was not surprised.

285 Then I measured the thickness, and I said to myself, "Aha! "

If the length is four times higher then the thickness has to be eight times
286 higher if this holds.

And what I'm going to plot for you you will see that shortly is d divided by l, versus l and that, of course, must be proportional to l to the power
287 one-half.

288 I bring one l here.

So, if I compare the horse and I compare the raccoon I would argue that the thickness divided by the length of the femur for the horse must be the
289 square root of four, twice as much as that of the raccoon.

And so I was very anxious to plot that, and I did that and I'll show you
290 the result.

291 Here is my first result.

292 So we see there, d over l.

293 I explained to you why I prefer that.

294 And here you see the length.

295 You see here the raccoon and you see the horse.

And if you look carefully, then the d over l for the horse is only about
296 one and a half times larger than the raccoon.

297 Well, I wasn't too disappointed.

298 One and a half is not two, but it is in the right direction.

299 The horse clearly has a larger value for d over l than the raccoon.

300 I realized I needed more data, so I went back to Harvard.

I said, "Look, I need a smaller animal, an opossum maybe maybe a rat, maybe
301 a mouse," and they said, "okay. "

302 They gave me three more bones.

They gave me an antelope which is actually a little larger than a raccoon
303 and they gave me an opossum and they gave me a mouse.

304 Here is the bone of the antelope.

305 Here is the one of the raccoon.

306 Here is the one of the opossum.

307 And now you won't believe this.

308 This is so wonderful, so romantic.

309 There is the mouse.

310 (students laugh) Isn't that beautiful?

311 Teeny, weeny little mouse?

312 That's only a teeny, weeny little femur.

313 And there it is.

314 And I made the plot.

315 I was very curious what that plot would look like.

316 And... here it is.

317 Whew!

318 was shocked.

319 I was really shocked.

320 Because look-- the horse is 50 times larger in size than the mouse.

321 The difference in d over l is only a factor of two.

322 And I expected something more like a factor of seven.

323 And so, in d over l, where I expect a factor of seven I only see a factor of two.

324 So I said to myself, "Oh, my goodness.

325 Why didn't I ask them for an elephant? "

326 The real clincher would be the elephant because if that goes way off scale maybe we can still rescue the statement by Galileo Galilei and so I went back and they said "Okay, we'll give you the femur of an elephant. "

327 They also gave me one of a moose, believe it or not.

328 I think they wanted to get rid of me by that time to be frank with you.

329 And here is the femur of an elephant.

330 And I measured it.

331 The length and the thickness.

332 And it is very heavy.

333 It weighs a ton.

334 I plotted it, I was full of expectation.

335 I couldn't sleep all night.

336 And there's the elephant.

337 There is no evidence whatsoever that d over l is really larger for the elephant than for the mouse.

These vertical bars indicate my uncertainty in measurements of thickness
and the horizontal scale, which is a logarithmic scale... the uncertainty
of the length measurements is in the thickness of the red pen so there’s no
338 need for me to indicate that any further.

339 And here you have your measurements in case you want to check them.

340 And look again at the mouse and look at the elephant.

The mouse has indeed only one centimeter length of the femur and the
341 elephant is, indeed, hundred times longer.

So the first scaling argument that S is proportional to l that is certainly
what you would expect because an elephant is about a hundred times larger
342 in size.

343 But when you go to d over l, you see it’s all over.

The d over l for the mouse is really not all that different from the
elephant and you would have expected that number to be with the square
root of 100 so you expect it to be ten times larger instead of about the
344 same.

345 </section 5>

346 <section 6>

I now want to discuss with you what we call in physics dimensional
347 analysis.

I want to ask myself the question: If I drop an apple from a certain height
and I change that height what will happen with the time for the apple to
348 fall?

Well, I drop the apple from a height h and I want to know what happened
349 with the time when it falls.

350 And I change h.

So I said to myself, "Well, the time that it takes must be proportional to
351 the height to some power alpha. "

352 Completely reasonable.

If I make the height larger we all know that it takes longer for the apple
353 to fall.

354 That’s a safe thing.

I said to myself, "Well, if the apple has a mass m "it probably is also
355 proportional to the mass of that apple to the power beta. "

I said to myself, "Gee, yeah, if something is more massive it will probably
356 take less time. "

357 So maybe m to some power beta.

358 I don’t know alpha, I don’t know beta.

And then I said, "Gee, there’s also something like gravity that is the
359 Earth’s gravitational pull-- the gravitational acceleration of the Earth. "

So let's introduce that, too and let's assume that that time is also
proportional to the gravitational acceleration-- this is an acceleration;
360 we will learn a lot more about that-- to the power gamma.

Having said this, we can now do what's called in physics a dimensional
361 analysis.

On the left we have a time and if we have a left... on the left side a time
362 on the right side we must also have time.

363 You cannot have coconuts on one side and oranges on the other.

364 You cannot have seconds on one side and meters per second on the other.

365 So the dimensions left and right have to be the same.

366 What is the dimension here?

367 That is [T] to the power one.

That T... that must be the same as length to the power alpha times mass
to the power beta, times acceleration-- remember, it is still there on the
blackboard-- that's dimension [L] divided by time squared and the whole
368 thing to the power gamma so I have a gamma here and I have a gamma there.

369 This side must have the same dimension as that side.

370 That is nonnegotiable in physics.

371 Okay, there we go.

372 There is no M here, there is only one M here so beta must be zero.

There is here [L] to the power alpha, [L] to the power gamma there is no
373 [L] here.

374 So [L] must disappear.

375 So alpha plus gamma must be zero.

There is [T] to the power one here and there is here [T] to the power -2
376 gamma.

377 It's minus because it's downstairs.

378 So one must be equal to -2 gamma.

379 That means gamma must be minus one half.

380 That if gamma is minus one half, then alpha equals plus one half.

381 End of my dimensional analysis.

I therefore conclude that the time that it takes for an object to fall
equals some constant, which I do not know but that constant has no
dimension-- I don't know what it is-- times the square root of h divided
382 by g.

Beta is zero, there is no mass h to the power one half-- you see that
383 here-- and g to the power minus one half.

This is proportional to the square root of h because g is a given and c is
384 a given even though I don't know c.

I make no pretense that I can predict how long it will take for the apple
385 to fall.

386 All I'm saying is, I can compare two different heights.

I can drop an apple from eight meters and another one from two meters and
the one from eight meters will take two times longer than the one from two
387 meters.
The square root of h to two, four over two will take two times longer,
388 right?

If I drop one from eight meters and I drop another one from two meters then
389 the difference in time will be the square root of the ratio.

390 It will be twice as long.

391 And that I want to bring to a test today.

392 We have a setup here.

We have an apple there at a height of three meters and we know the length
393 to an accuracy... the height of about three millimeters, no better.

And here we have a setup whereby the apple is about one and a half meters
394 above the ground.

And we know that to about also an accuracy of no better than about three
395 millimeters.

396 So, let's set it up.

I have here... something that's going to be a prediction-- a prediction of
the time that it takes for one apple to fall divided by the time that it
397 takes for the other apple to fall.

398 </section 6>

399 <section 7>
H one is three meters but I claim there is an uncertainty of about three
400 millimeters.

401 Can't do any better.

And h 2 equals 1.5 meters again with an uncertainty of about three
402 millimeters.
So the ratio h one over h two... is 2.000 and now I have to come up with an
uncertainty which physicists sometimes call an error in their measurements
403 but it's really an uncertainty.

And the way you find your uncertainty is that you add the three here and
404 you subtract the three here and you get the largest value possible.

405 You can never get a larger value.

And you'll find that you get 2.006. And so I would say the uncertainty is
then.006. This is a dimensionless number because it's length divided by
406 length.

And so the time t1 divided by t2 would be the square root of h1 divided by
407 h2.

408 That is the dimensional analysis argument that we have there.

And we find if we take the square root of this number we find 1.414, plus
409 or minus 0.0 and I think that is a two.

410 That is correct.

411 So here is a firm prediction.

412 This is a prediction.

413 And now we're going to make an observation.

So we're going to measure t1 and there's going to be a number and then
414 we're going to measure t2 and there's going to be a number.

I have done this experiment ten times and the numbers always reproduce
415 within about one millisecond.

416 So I could just adopt an uncertainty of one millisecond.

417 I want to be a little bit on the safe side.

418 Occasionally it differs by two milliseconds.

So let us be conservative and let's assume that I can measure this to an
419 accuracy of about two milliseconds.

420 That is pretty safe.

So now we can measure these times and then we can take the ratio and then
we can see whether we actually confirm that the time that it takes is
421 proportional to the height to the square root of the height.

422 So I will make it a little more comfortable for you in the lecture hall.

423 That's all right.

424 We have the setup here.

425 We first do the experiment with the... three meters.

426 There you see the three meters.

And the time... the moment that I pull this string the apple will fall, the
427 contact will open, the clock will start.

428 The moment that it hits the floor, the time will stop.

429 I have to stand on that side.

430 Otherwise the apple will fall on my hand.

431 That's not the idea.

432 I'll stand here.

433 You ready?

434 Okay, then I'm ready.

435 Everything set?

436 Make sure that I've zeroed that properly.

437 Yes, I have.

438 Okay.

439 Three, two, one, zero.

440 781 milliseconds.

So this number... you should write it down because you will need it for
441 your second assignment.

442 781 milliseconds, with an uncertainty of two milliseconds.

443 You ready for the second one?

444 You ready?

445 You ready?

446 Okay, nothing wrong.

447 Ready.

448 Zero, zero, right?

449 Thank you.

450 Okay.

451 Three, two, one, zero.

452 551 milliseconds.

453 Boy, I'm nervous because I hope that physics works.

454 So I take my calculator and I'm now going to take the ratio t1 over t2.

The uncertainty you can find by adding the two here and subtracting the two
there and that will then give you an uncertainty of, I think,.0... mmm,.08.
455 Yeah,.08. You should do that for yourself--.008. Dimensionless number.

456 This would be the uncertainty.

457 This is the observation.

458 781 divided by 551.

459 One point... Let me do that once more.

460 Seven eight one, divided by five five one... One four one seven.

461 Perfect agreement.

Look, the prediction says 1.414 but it could be 1 point... it could be two
462 higher.

463 That's the uncertainty in my height.

464 I don't know any better.

And here I could even be off by an eight because that's the uncertainty in
465 my timing.

466 So these two measurements confirm.

467 They are in agreement with each other.

468 You see, uncertainties in measurements are essential.

469 Now look at our results.

470 We have here a result which is striking.

We have demonstrated that the time that it takes for an object to fall is
471 independent of its mass.

472 That is an amazing accomplishment.

Our great-grandfathers must have worried about this and argued about this
473 for more than 300 years.

474 Were they so dumb to overlook this simple dimensional analysis?

475 Inconceivable.

476 Is this dimensional analysis perhaps not quite kosher?

477 Maybe.

Is this dimensional analysis perhaps one that could have been done
478 differently?

479 Yeah, oh, yeah.

480 You could have done it very differently.

481 You could have said the following.

You could have said, "The time for an apple to fall "is proportional to the
482 height that it falls from to a power alpha. "

483 Very reasonable.

We all know, the higher it is, the more it will take-- the more time it
484 will take.

And we could have said, "Yeah, it's probably proportional "to the mass
485 somehow.

486 If the mass is more, it will take a little bit less time. "

487 Turns out to be not so, but you could think that.

But you could have said "Well, let's not take the acceleration of the Earth
488 but let's take the mass of the Earth itself. "

489 Very reasonable, right?

I would think if I increased the mass of the Earth that the apple will fall
490 faster.

491 So now I will put in the math of the Earth here.

492 And I start my dimensional analysis and I end up dead in the waters.

493 Because, you see, there is no mass here.

There is a mass to the power beta here and one to the power gamma so what
you would have found is beta plus gamma equals zero and that would be end
494 of story.

Now you can ask yourself the question well, is there something wrong with
495 the analysis that we did?

496 Is ours perhaps better than this one?

497 Well, it's a different one.

We came to the conclusion that the time that it takes for the apple to fall
498 is independent of the mass.

499 Do we believe that?

500 Yes, we do.

On the other hand, there are very prestigious physicists who even nowadays
do very fancy experiments and they try to demonstrate that the time for an
apple to fall does depend on its mass even though it probably is only very
501 small, if it's true but they try to prove that.

And if any of them succeeds or any one of you succeeds that's certainly
502 worth a Nobel Prize.

503 So we do believe that it's independent of the mass.

However, this, what I did with you, was not a proof because if you do it
504 this way, you get stuck.

On the other hand, I'm quite pleased with the fact that we found that the
505 time is proportional with the square root of h.

506 I think that's very useful.

507 We confirmed that with experiment and indeed it came out that way.

508 So it was not a complete waste of time.

509 But when you do a dimensional analysis, you better be careful.

I'd like you to think this over, the comparison between the two at dinner
and maybe at breakfast and maybe even while you are taking a shower whether
510 it's needed or not.

It is important that you digest and appreciate the difference between these
511 two approaches.

It will give you an insight in the power and also into the limitations of
512 dimensional analysis.

This goes to the very heart of our understanding and appreciation of
513 physics.

514 It's important that you get a feel for this.

515 You're now at MIT.

516 This is the time.

517 Thank you.

518 See you Friday.

519 </section 7>

A.2 AI Lecture

1 <section 1>

2 If you're going to teach an AI course, it's useful to ask: "What's AI?".

3 It's a lot of different things to a lot of different people.

Let's go through a few things that AI could be and that it usefully is
and situate the ways we will look at AI and situate it within the broader
picture of ways of thinking about AI One thing it could be is "Making
4 computational models of human behavior".

Since you figure that humans are intelligent and therefore models of
5 intelligent behavior must be AI.

There's a great paper by Turing who really set up this idea of AI as making
6 models of human behavior (link).

7 In this way of thinking of AI, how would you proceed as an AI scientist?

One way, which would be a kind of cognitive science is to do experiments on
humans, see how they behave in certain situations and see if you could make
8 computers behave in that same way.

Imagine that you wanted to make a program that played poker, instead of
making the best possible poker-playing program, you would make one that
9 played poker like people do.

10 Another way is to make computational models of human thought processes.

11 This is a stronger and more constrained view of what the enterprise is.

It is not enough to make a program that seems to behave the way humans do;
12 you want to make a program that does it the way humans do it.

A lot of people have worked on this in cognitive science and in an area
13 called cognitive neuro-science.

The enterprise is to affiliate with someone who does experiments that
reveal something about what goes on inside people's heads and then build
14 computational models that mirror those kind of processes.

So here, it is an interesting and a hard question to decide at what level
15 to mirror what goes on inside people's heads.

Someone might try to model it a very high-level, for example, saying that
there's a memory and a vision module, and this kind of module or that kind
of module and so they try to get the modularity to be accurate but they
16 don't worry too much about the details.

17 Other people might pick, e.g.

a neuron, as a kind of computational unit that feels like it's justified in
terms of neurophysiology and then they take that abstract neuron and they
18 make computational mechanisms out of that neuron.

19 They feel "That's cool since brains as made up of neurons."

But, then if you talk to people that study neurons you find that they argue
a lot about what neurons can and can't do computationally and whether they
are a good abstraction or whether you might want to make your models at a
20 lower level.

So, there's a tricky business here about how you might want to try to
match up what we know about brains and how it is that you might make
21 computational models.

22 This is not what we will be doing here.

Another thing that we could do is computational systems that behave
23 intelligently.

24 What do we mean here?

When we talked about human behavior, we said that was intelligent because
humans are intelligent (sort of by definition) so what humans do has to be
25 intelligent.

In this view, we say that there might be other ways of being intelligent
26 besides the way humans do it.

And so what we might want to do is make computational systems drawn from
27 this larger class.

But then you get into terrible trouble because you have to say what it
28 means to behave intelligently.

We might feel that although we can't define what is intelligent, we can
29 recognize it when we see it.

We'll punt on trying to decide what intelligence is and spend our time
30 thinking about rationality.

31 What might it mean to behave rationally?

32 We'll get into that in more detail later.

So, the perspective of this course is that we are going to build systems
that behave rationally - that do a good job of doing what they're supposed
33 to do in the world.

But, we're not going to feel particularly bound to respect what is known
34 about how humans behave or function.

35 Although we're certainly quite happy to take inspiration from what we know.

There's another part of AI that's closer to what we will talk about in this
36 class that's fundamentally about applications.

Some of these applications you might not want to call "intelligent" or
"rational" but it is work that has traditionally been done in the field of
37 AI.

And usually what they are are problems in computer science that don't feel
well specified enough for the rest of the computer science community to
38 want to work on.

For instance, compilers used to be considered AI, because you were writing
down statements in a high-level language and how could a computer possibly
39 understand that stuff.

Well, you had to do work to make a computer understand that stuff and that
40 was taken to be AI.

Now that we understand compilers and there's a theory of how to build
41 compilers and lots of compilers out there, well it's not AI any more.

So, AI people have a chip on their shoulders that when they finally get
42 something working it gets co-opted by some other part of the field.

43 So, by definition, no AI ever works; if it works, it's not AI.

But, there are all kinds of applications of AI, many of these are
applications of learning, which is my field of research and for which I
44 have a soft spot in my heart.

For example, NASDAQ now monitors trades to see if insider trading is going
on, Visa now runs some kind of neural network program to detect fraudulent
transactions, people do cell-phone fraud detection through AI programs,
scheduling is something that used to be AI and is now evolving out of AI
(and so it doesn't really count) but things like scheduling operations in
big manufacturing plants; NASA uses all kind of AI methods (similar to the
ones we're going to explore in the first homework) to schedule payload
bay operations, so getting the space shuttle ready to go is a big and
complicated process and they have to figure out what order to do all the
45 steps.

46 There's all kinds of applications in medicine.

For example, managing a ventilator, a machine that is breathing for a
patient, there is all kinds of issues of how to adjust various levels of
gases, monitor pressure, etc. Obviously, you could get that very badly
47 wrong and so you want a system that's good and reliable.

48 Obviously, if they field these systems they must be ok.

49 There's no end of examples; AI applications are viable.

We're going to spend most of our times thinking, or at least feeling
50 motivated, by computational systems that behave rationally.

But a lot of the techniques that we will be talking about will end up
51 serving a wide variety of application goals as well.

52 That’s my story about what we’re up to.

53 </section 1>

54 <section 2>

55 We’re going to be talking about agents.

56 This word used to mean something that acts.

Way back when I started working on AI, agent meant something that took
57 actions in the world.

Now, people talk about Web agents that do things for you, there’s publicity
58 agent, etc. When I talk about agents, I mean something that acts.

So, it could be anything from a robot, to a piece of software that runs
in the world and gathers information and takes action based on that
information, to a factory, to all the airplanes belonging to United
59 Airlines.

60 So, I will use that term very generically.

When I talk about computational agents that behave autonomously, I’ll use
61 agent as a shorthand for that.

62 So, how do we think about agents?

63 How can we begin to formalize the problem of building an agent?

Well, the first thing that we’re going to do, which some people object
to fairly violently, is to make a dichotomy between an agent and its
64 environment.

There are people in AI that want to argue that that is exactly the wrong
thing to do, that I shouldn’t try to give an account of how I work by
separating me from the world I work in, because the interface is so big
65 and so complicated.

66 And that may be right.

That I can’t get exactly right a description of how I need to operate in
67 the world by separating me from the world.

But, it gives me a kind of leverage in designing the system that I need
68 right now because I’m not smart enough to consider the system as a whole.

69 </section 2>

70 <section 3>

71 Here’s a robot and the world it lives in.

The robot is going to take actions that affect the state of the environment
and it’s going to receive percepts somehow that tell it about what’s going
72 on in the environment.

73 So it ??

loop where the agent does something that changes the state of the
environment then it somehow perceives some new information about the state
74 of the environment.

There’s a whole question of how to draw the line between the agent and the
75 environment.

In this class, we’ll entirely spend our time thinking about the agent as a
76 computational entity.

77 SO, I should really draw this cartoon differently.

Since we’re going to be thinking about what is going on in the agents head
and so the actions instead of going like this are going to be going from
the agent’s head to its wheels and the percepts are coming from the camera
78 into its brain.

79 And, so, here’s another view of the world.

We’re going to be thinking about the agent as the software that runs some
80 big hardware system.

That is not to make light of or say that it’s easy to design the hardware
part and depending on how the hardware part has been designed your problem
81 could be made arbitrarily easier or harder.

82 An example of this is making a walking robot.

83 How hard that job is depends on the design of the hardware.

There are these great walking robots that are called "compass walkers"
that are just two legs hinged together and when you set them on an inclined
plane they will walk down the hill (if you get it balanced right); so you
84 don’t need any computation at all to do that walking.

So, the computation, the intelligence or whatever is in the design of the
85 hardware.

On the other hand, you could imagine building a great big contraption (like
one at CMU) with six or eight legs and is taller than this room and it
runs a whole complicated planning algorithm to decide where to place each
foot, so that’s the opposite extreme of putting all the intelligence in the
86 brain, instead of in the hardware.

We’re going to try to be agnostic about the design of the hardware and
work with people who do a good job of that and take as given computational
87 problems.

88 How can we formalize a computational problem of building an agent?

89 Here’s a formal model.

90 </section 3>

91 <section 4>

What do we need to write down when we talk about the problem of making an
92 agent.

93 How can we specify it really carefully?

94 Well, we’re going to need an action interface.

These all the things that my agent can do, it might be continuous, it might
be very high dimensional but there's some space of possible actions that
95 the agent can take in the world.

And there's a percept space, same sort of thing, what are all the things
96 that the agent can perceive in the world.

These spaces can be continuous; you can imagine that the agent can perceive
how high its arm is raised or the temperature in some reaction vessel or
97 something.

98 But, we're going to assume discrete time, or at least discrete events.

I drew this picture of the interaction between the agent and its
environment and I said that the agent takes an action and the environment
99 updates its state and then the agent observes.

You could imagine modeling this as a set of coupled differential equations
and there are people who do that for fairly simple and low-level systems;
we're going to think of things rather more discretely and combinatorially
and so we're going to model the interaction between the agent and the
100 environment as a turn-taking thing that happens on some cycle.

In the discrete time view you say that every one second, or two seconds or
101 ten seconds or ten minutes there is this kind of turn taking.

In the discrete event view, time marches on contibuously but there are
events of I do this action sort of in an impulse and the world changes
102 state some time later and then I do another action some time after that.

103 You can imagine continuous time with discrete events embedded in it.

104 (20:58) ??

105 discrete-time case.

Time won't enter too much in the stuff we'll talk about but it will a
bit and it's something that's really important to keep in the back of our
106 minds.

So we have a set of actions and a set of percepts and the we need the
107 environment.

We need, in order to say what the problem is for our agent, to describe the
108 world that the agent lives in.

At the most detailed level, we can think of the environment as being a
109 mapping of strings of actions into percepts.

110 You could say, what does the environment do?

Well, there's some history of actions that the agent has done to it and
111 every time the agent does a new action, it generates a percept.

112 That's not a very helpful way of thinking about it.

Usually we'll think of the environment as having some internal state which
113 may not be visible to the agent.

You could think of the envoronment something that instead includes a
mapping from state to percepts, something that says when the world is in
this state what the agent gets to see and another mapping from situations
114 and actions into situations.

115 These things describe how the world works.

We'll call these the world dynamics and sometimes this get called the
116 perception function.

Later on we'll talk about the fact that these things may not be
117 deterministic and they may not really be known.

Suppose you wanted to make a robot that could vacuum the hallways or
118 something in this building.

You'd like not to have to completely specify how this building is laid out
119 and where the chairs are and who has a backpack on the floor today.

So, in fact, rather tahn giving a complete, perfectly nailed down
description of how the environment works, in general when we specify the
problem of designing an agent we'll give some constraints, some parts of an
120 specification of how the environment works.

121 We'll leave a lot to be determined in a lot of cases.

122 One more thing.

123 This so far has no value judgements.

124 We're describing a set of worlds that the agent has to work in.

125 </section 4>

126 <section 5>

And we also have to say what we want the agent to do, what constitutes good
127 or bad behavior of the agent in the environment.

128 We need an utility function.

That;s typically thought of a mapping from states in the world to real
129 values, or maybe sequences of states into real values.

This is just to say, "Agent, these are the states of the world and this how
130 valuable they are from your perspective."

131 SO that kind of tells the agent what you want it to do.

Now, our problem as people who want to design AI systems is to build the
132 agent (the software) in such a way as to get a lot of utility.

133 SO, now is just an optimization problem - that doesn't seem so hard.

134 We'll it's going to turn it to be really quite hard.

135 But, at this level of abstraction, it's straightforward what we want to do.

We want to put the program in the head of the agent that does as well as it
can subject to this specification of how the world works and what we want
136 in the world.

137 </section 5>

138 <section 6>
Let's talk about rationality, since I said that what we wanted to do was to
139 make rational agents.

140 So, what do I mean by that?

The standard definition of rationality is: A rational agent takes actions
141 it believes to achieve its goals.

This is all in high-level pseudo-psychological talk that makes some people
142 nervous.

We can cache it out into something more concrete in a minute but the idea
is that you're rational if you do things that are consistent with what you
143 are trying to do in the grand scheme of things.

Let's say that I don't like to be wet and so when I come out of my office
144 in the morning, I bring an umbrella.

145 Is that rational?

Depends on the weather forecast and whether I've heard the weather
146 forecast.

If I heard the weather forecast and I'm disposed to believe them and I
147 think it's going to rain then it's rational to bring my umbrella.

Whether it's going to rain or not, whether you think it's dumb for me to
want to stay dry or various things like that, given what I'm trying to do
and given what I know we'll say an action is rational if it would lead to
148 doing a good job of what I'm trying to do.

149 Rationality is not omniscient.

For example, some time ago I rode my bike in to work, not knowing that it
was going to snow like crazy and I was going to run into a car on the way
150 home.

You can still argue that it was rational for me to ride my bike, maybe
at some grander level it was irrational not to have watched the weather
151 forecast the night before.

But, given what I knew it was ok to ride my bike, even though it turned out
152 be dumb at some level, because I didn't know what was happening.

153 Also, rationality is not the same as succesful.

Imagine that I take my umbrella, I know that it's nice and sunny out and I
154 take the umbrella anyway, which was irrational of me.

155 But, then I use the umbrella to fend off a rabid dog attack.

You might say, well it was rational of her to take the unbrella because
it saved her from the rabid dog, but that wouldn't be right beacuse it was
156 done for the wrong reason.

Even though it was successful and useful; we would not have said that was
157 rational.

158 So this limits the scope of what we want our agents to do.

They don't have to be succesful and they don't have to know everything,
159 they just have to do a good job given what they know and what they want.

160 </section 6>

161 <section 7>

This is still not a good enough notion to decide what goes in the head of
162 our agent or our robot.

DO you see any potential problem with this as a criterion for behavior in
163 real systems?

164 You might not be able to compute the best thing to do.

165 There's a notion that the philosophers have pursued and so have AI people.

People talk instead of complete or perfect rationality, of limited
166 rationality.

And that means exactly "acting in the best way you can subject to the
167 computational constraint that you have."

SO, here we are with soft squishy brains that can't compute very well or very fast and so, for instance, humans are irrational because they're bad at doing task X or Y or Z; they just can't compute the optimal response in
168 certain circumstances.

That we know; there's no question, but yet you might be able to argue that
169 given their squishy brains that's the best they can do.

Or, maybe you want to argue that for this idea of limited rationality that you need to put a program in the agent's head that's going to last for the
170 agent's whole range of things it has to do and life it has to live.

And it might be that brain could conceivably compute the optimal action in
171 one circumstance, it may not in another.

So, we might be able to make a robot that's the end-all and be-all chess
172 player but it might not be able to cross the street.

173 SO, that's probably not ok.

SO, when we think about rationality we may we want to think about it in a much broader context: given all the things that you have to do, given all the circumstances that you're likely to be faced with in the environment
174 that you;ve been put in, how can you respond the best in the aggregate.

SO, any individual response may not be the best, the optimal response, even given your hardware, it may be that the program you're running is the best possible program when measured in an aggregate over all the things that yo
175 have to do.

176 What we're need to make is an agent program.

An agent program is, given all that stuff, we want to find the best possible mapping from P* to A (sequences of percepts to actions) that subject to our computational constraints does the best job it can as
177 measured by our utility function.

178 </section 7>

179 <section 8>

180 Let's imagine that someone was able to write down a specification of the environment that we want our agent to work in.

181 You could say: "Oh, but you can't do that.

182 This is all pretty silly because how is it that anyone could specify the domain that the agent is going to work in?"

183 AT some level I am sympathetic to that complaint, but at some other level I am entirely unsympathetic to that complaint because if you ask me to solve a problem then you have to tell me what problem you want me to solve.

184 So, you might imagine that this whole process is going to operate in a much larger context that's iterative.

185 You give me a specification of the environment you want the robot to work in; I work away to give you the maximally rational robot given your specification, we start running it and then you tell me "Darn, I forgot to tell you about not vacuuming the cat."

186 Then you would have to go back and recompute the robot.

187 In any real application you have this cycle at a high level.

188 But, I don't think you can get out of saying: "Here's what I want the system to do."

189 Given a specification for all this stuff, it seems like our problem is "just" one of coming up with a program that satisfies some specifications.

190 So, you could go study that in software engineering (maybe).

191 But, why not?

192 Why is this not just software engineering?

193 Any of us would be hard-pressed, given all the pieces of the space shuttle and constraints on how they go together, to sit in a chair and write the program that is optimal given all those constraints.

194 The problem is that, although information theoretically this is an specification for the correct program, it is not an effective specification.

195 It's not a specification that the computer can use.

196 There is a huge gap between the specification for what you want the thing to do and what you can write down in a program and actually have run.

197 How do we bridge this gap?

198 There is a part of AI that still goes on (in some places) but people don't talk about much, called "automatic programming".

199 In fact, quite a while ago there was a project going on here in the AI Lab called "The programmer's assistant" which was supposed to enable you to say "I need a linked list that would do whatever..."

200 or "Put these things in a hash table..."

You would give it instructions at that level and it was supposed to write
201 the code to do that for you.

But, the idea in automatic programming was that you would go from some declarative specification of what you wanted the system to do to actual
202 code to do it.

203 But, it's a really hard problem and most people have given up on it.

204 But it seems that's the problem we are faced with here.

205 But, we're not going to do this automatically.

206 So, what's the enterprise that we're going to be engaged in?

We're going to look at classes of environment specifications and utility functions and try to map from classes of environments to structures of
207 programs.

To try to say that "if you need an agent to try to solve this class of problem in that kind of environment, then here is a good way to structure
208 the computation."

209 </section 8>

210 <section 9>

211 This doesn't feel a lot like AI.

We have this idea that AI is about agents thinking in their heads figuring
212 out what they're supposed to do.

213 This feels like it's off-line.

214 Someone (God?)

215 doing all the figuring and blasting the program into the head of the robot.

216 The question we want to ask ourselves is "Why is it ever useful to think?"

If all these thought processes could happen off-line and you could just be
217 endowed with the optimal set of reflexes then who needs cogitation?

Why can't we (for you or a big complicated factory) compile a whole table
218 of reactions?

219 Let's even imagine that the environment is not changing.

220 The problem is that the table is too big.

If P is any size at all or if you live for very long, the table is way too
221 big.

222 Way, way too big.

There are too many ways the world could be, there are too many sequences of
223 percepts that you could have of the world.

224 There is no way that you could off-line anticipate them.

225 Actually, for some domains you can.

226 It’s interesting to know where this line gets drawn.

This is my version of what the direction that we’re going to take in this
227 class relate to the direction that Embodied AI takes.

228 There are two fundamental differences.

One is that the Embodied AI people actually take as one of their
constraints that the mechanims that they develop are somewhat related to
229 the mechanisms that go on in nature.

Another difference is that they entertain a different class of problems and
the class of problems that they entertain are amenable to something like
230 this approach.

It’s not turned out quite so formally, but the way it works is that a human
thinks about a problem, thinks hard about, figures out what the program
231 ought to be structured like and writes the program.

But that program when it runs is pretty direct, it pretty much gets the
232 percepts and computes an action.

It doesn’t feel like it thinks (whatever that might mean to us); it doesn’t
233 entertain alternative realities.

There is certainly a class of problems for which it feels like you can’t
make a table but you can write a fairly compact program that would do the
234 job of being the table.

But there are other domains in which you quite clearly can’t do that and
235 those are the domains that we are going to focus on.

The domains where you can’t think of a compact way to write this program
236 down, this mapping from strings of perceptions to actions.

237 So, we’ll have to think of other ways to construct this program.

And the other ways of constructing this program are going to take advantage
238 of the fact that the vast majority of the things that could happen - don’t.

Think of all the ways the world could be, there are a lot of percept
sequences that you could conceivably have and no matter how long you live
you are going to have only the most minuscule fraction of all the percepts
239 you could possibly have.

So, the work that Nature does for you is that there’s no reason to have
precomputed and stored reactions for what happens if an elephant flies
240 through the window - we don’t have to worry about that.

So, you probably don’t have precompiled reactions for what happens if
an elephant flew in through the window, on the other hand if one did you
wouldn’t be totally incapcitated (like you would be if you were under the
241 elephant).

You’d say "oh, my gosh" and then your brain would kick in and you’d start
242 figuring out what to do about it.

So, you could be very flexible to a very broad range of stimuli but there's
243 some way that you could have canned your responses to those ??

244 (44:10).

245 </section 9>

246 <section 10>
Let me talk a bit about learning; we're going to talk about learning
247 towards the end of this class.

248 So, what happens when the environment changes?

249 When I talk to people about why it's important to build systems that learn.

I say "maybe you don't know very much about the environment when you start
250 out or maybe the environment changes" and so you have to do learning.

And it seems that I haven't accounted for that in this framework, but I
want to say that I have accounted for it because I've said so very little
251 about what this kind of specification might be.

252 So, let's take a very simple case.

Imagine that we're sending a robot to Mars and we don't know the
coefficient of friction of the dust it's going to land on; they don't know
253 what it feels to drive around in that stuff.
I could still say: "Look, I know something about this place we're going to
254 send the vehicle to.
It's going to have gravity, I know what the gravity is going to be like,
I know what's going to go on there; I know a lot about the vehicle but I
255 don't know the coefficient of friction of the dust.
Instead of giving the complete world dynamics; I'm going to have to leave
a free parameter or some disjunction (the world is either going to be like
256 this or like that and I don't know which).
And then part of my job as the agent is, based on the sequence of percepts
that I have, to kind of estimate or to learn or to gather information about
257 the dynamics of the world.

If this specification doesn't have to be full then I'm allowed to learn
258 something about how the world works.

Similarly, I can build into this specification that there is a coefficient
259 of friction that changes over time but I don't know how it changes.

260 So, learning can fit into this framework too.

This is a framework that in the end isn't really that informative in the
261 sense that it isn't that constraining.

In some sense learning isn't very different from perception, they're both
262 about learning something about the world by virtue of your experience.

And we tend to call "learning" things that happen on larger time-scale;
263 things that seem more permanent.

And we tend to call perception, things that like noticing where I am with
respect to a wall, things that are on a shorter time scale things that
264 don't seem so built-in.

But there is no hard and fast distinction between learning and perceiving
265 where I am relative to the wall.

266 </section 10>

267 <section 11>

Let's think about environments and the different kinds of environments that
268 our agents might need to work in.

Now, there's a whole enterprise in this course that will be thinking
about particular properties of the environment that we know hold and what
consequences they might have on how it is that we would design an agent to
269 perform well in that environment.

So this is a nice list that comes out of Russell & Norvig (textbook) - a
270 nice way of thinking about environments.

One dimension along which it is useful to categorize environments is
271 whether they are "accessible".

What they mean by accessible (vs inaccessible) is "Can you see the state of
272 the world directly?".

Most real environments are inaccessible; I can see some aspects of the
state of the world, but I don't know what's happening right out there or
who's opening the door etc. So, my world is not accessible but some kinds
273 of toy worlds are accessible and maybe some kinds of applications.

Imagine I am thinking of where to route all the airplanes for United
274 Airlines.

I like to think that they know where all the airplanes are all the time, so
275 maybe that's an accessible domain.

276 Another dimension is "deterministic" vs "non-deterministic".

Over here I talked about world dynamics, the mapping between a current
state of the world an the action that an agent takes into another state of
277 the world.

278 In some domains that's usefully thought of as being deterministic.

The only domains that are really deterministic are artificial ones, like
279 games.

Even clicking on a link and going to a Web page, you know that doesn't
280 always work.

Most things are not entirely deterministic, some things are reasonably
281 modeled as being deterministic.

And, we'll spend about the second half of this class thinking about
282 non-deterministic environments.

The first half we'll think about deterministic models really as an
283 abstraction and in the second half we'll think about probabilistic models.

284 Another dimension for describing environments is static vs dynamic.

285 Again, one can argue that everything is dynamic but let's talk about it.

286 It has to do with whether the world can change while you're thinking.

287 If the world can't change while you're thinking, then the whole limited rationality thing does not matter as much, because you can think and think until you come up with the best possible thing to do.

288 But, usually the world is changing.

289 If you compute the optimal trajectory for avoiding the truck but you're a little late, it's no good.

290 You have to really worry about the dynamic property of the environment.

291 And then there's "discrete" vs "continuous".

292 Most of these are not really intrinsic properties of the environment but more properties of how we choose to model the environment.

293 So, you can think of your perceptions of the world in different cases as being discrete or continuous.

294 </section 11>

295 <section 12>

296 Let's talk about some environments.

297 Let's talk about playing backgammon.

298 </section 12>

299 <section 13>

300 For an agent playing backgammon, what's the action space?

301 The action space is the set of backgammon moves, e.g.

302 I put a white piece on that point.

303 But you're want to think of the moves in some fairly logical way.

304 You probably don't want to think of the move as the x-y location of the stone on the board.

305 You could.

306 But, that doesn't feel so useful.

307 If you were building the robot to move the pieces, you would have to think of the x-y location; you would have to think of the motor voltages that you send to the joints in order for the arm to move where it needs to go in order to put the stone where it goes on the point on the board.

308 So, this gets to what I said about not worrying about the very best way to frame a problem, the very best way to divide the pieces up - although when we talk about execution we'll talk about that a bit.

But, it’s an interesting question "how are we going to define the action
spaces" do you want to define it in terms of motor voltages, are you going
to define it in terms of x-y locations or are you going to define it in
309 terms of how many guys I have on the board point on my side of the board.

310 There’s logical descriptions of the actions and similarly the percepts.

Your percepts might be images of the backgammon board, they might be
x-y locations of the stones, they might be the facial expression of your
311 oponent or they might be a logical description of where the stones are.

For any one of those levels of description of the environment and of the
312 problem you’re supposed to solve, you’d write the software differently.

Let’s take for now the very abstracted view of playing backgammon, the view
313 that backgammon books take.

Which is the moves are putting the stones somewhere, the percepts are where
314 (again at a logical level) the stones are.

315 </section 13>

316 <section 14>

Backgammon is one of those few domains that is accessible; you can see
317 everything there is to know about the state of a backgammon board.

318 Is it deterministic?

319 No. There are two issues about backgammon that make it non-deterministic.

320 One is the dice.

321 The other is your oponent.

322 Actually, games are not very well modeled in this mode.

There is a nice chapter in the book on games; but we’re not going to do it
323 - there’s too much of it to cover.

324 Certainly, there is no way to predict what your oponent will do.

The typical game theory thing to do is to assume that your opponent is
325 infinitely smart and predict what he’s going to do on that basis.

326 But, there are problems with that.

327 He might not be.

Then you get into learning models where you say, ok my opponent is not
infinitely smart but I am going to learn what my opponent is like so that
I can predict what he might do and react but still you are not going to be
328 able to make deterministic predictions.

329 Is backgammon static or dynamic?

330 Static unless you have a time limit.

331 Discrete vs continuous?

Depends on how you choose to model the percepts and the actions but it is
332 usually thought of as a pretty continuous type game.

333 Why is not discrete "move by move"?

334 But, what if our percepts are images?

They are discrete (quantized) but so fine grained that it is useful to
335 think of them as continuous and what if our actions are motor voltages?

But, if we are thinking about the stones and the points, then it is
336 completely discrete.

The point is that it depends on how you choose the space of actions and
337 percepts.

There are domains (like images) that are discrete, very big and ordered
where it is useful to treat them as continuous because you can get certain
kinds of compactness in the program by treating the image pixels as being
338 related to each other in space as if there's a cntinuous axis.

Sometimes, computer scientists have this reflexive tendency to given a
continuous problems to make it discrete because it's in a domain that they
339 can cope with.

But sometimes it is useful to take a discrete problem and make it
continuous; it gives you certain kinds of generalizations that you might
340 not otherwise have.

341 This will come up again when we talk about learning.

342 </section 14>

343 <section 15>

344 Driving a taxi.

There's so many things to think about here it's hard to know where to
345 begin.

Suppose you wanted to make a taxi driver, how would you even think about
346 it?

347 What would you want the action space to be?

348 There are many levels that it could be.

349 It could be steering angle, accelerator and brake.

350 What other levels of description of the action space might you want to use?

351 Physical position.

352 Addresses.

As we go in this direction, it becomes harder and harder to map one of
353 these commands into the lowest level of how to turn the steering wheel.

354 That's ok.

We might want to say that we really need to think about the problem as
going from addresses to addresses and then I'll hire somebody else to
figure out how to take the command to go to an address and cache that out
355 into which way am I going to turn the steering wheel.

We'll have multiple dimensions - like speech (some taxi drivers speak and
356 other listen).

In perception, there's going to be an analogous range of ways that we can
357 think about the problem.

There's another way of thinking about driving, in terms of lane changes
and passing etc. Earlier I made light of framing and specifying the domain
but that's at least as hard (or harder) than solving the problem once it is
358 written down.

359 And the key questions are: "How do you think about the action spaces?",

360 "How do you think about the percept spaces?"

I can think about the percepts being images; I can think of them as being
361 "there's a red car to my left".

Another thing you have to control (and this is something that comes up
362 often) is where you're looking.

For example, in a car you can look in your rear-view mirror and that tells
you something of where people are around you that is useful to know, for
363 example for lane changes.

But, of course, you can't look in the rear view mirror all the time becuase
364 then you don't know what is happening in front of you.

So, you also have to think about when you should look in the rear-view
365 mirror versus when you should look straight ahead.

So you also have in your action space things that will give you information
366 about the world.

In an inaccessible environment, you may have to do things to find something
367 out.

You have to stop to ask directions, or buy a map or call someone on the
368 telephone to get directions.

369 So, there's an enormous range of actions that you may want to take.

370 </section 15>

371 <section 16>

372 Let me go through a couple of structures of agents.

373 We talked about a table-based agent.

We'll talk a bit more about what the book calls a simple-reflex agent
374 (sometimes called "reactive").

But, there's this huge amount of literature on things called reactive -
reactive robots, reactive planning, and it's gotten to the point that this
375 word means so many things to so many people that it does not mean anything.

The most coherent interpretation is that the structure of the agent is that
it gets percepts in and it generates actions and it only ever maps a single
376 percept to an action and so it has no memory.

377 So, the basic thing here is that there is no memory.

Remember that we've said that in general an agent maps strings of percepts
378 into actions.

379 It could integrate information about time.

380 THere aren't a lot of problems that you can solve in this way.

Maybe you can solve backgammon in this way; maybe you can solve te problem
381 of driving down the hallway and not running into the wall this way.

You look and you see that wall too close and you move away from it, etc.
382 So, there are a bunch of things you can do reactively.

Clearly if the world is accessible (you can see everything there is to see
in one shot) this means that you don't need any memory, you can just look a
383 see where everything is.

384 Doesn't this depend on how complex the goals are?

The programming here has to be kind of complicated and so calling it a
reflex agent might not be right anymore but certainly it doesn't need to
385 have memory (in the sense of remembering previous percepts).

It needs to have memory in the traditional sense that computer programs
386 need to have memory in the VonNeumann model.

387 If the environment is accessible, then everything is visible at once.

This is not usual except in domains like backgammon and perhaps some kinds
388 of information retrieval problems.

If it matters how the environment got into the state it's in; then that has
389 to be part of the state.

FOr example, let's imagine that you arrive somewhere with more or less
390 gasoline.

Now, there's two way of knowing how much gas you have, one is to remember
391 how much driving you've done and the other is to look at the gas gage.

If you have a gas gage then the state of the tank is accessible to you and
392 you don't need to remember how long you've been driving.

393 Accessible and predictable are not the same.

You can read the gas gage but have no idea an hour from now what the gage
394 will say.

395 In that case, we would still say that the environment is accessible.

You do have a problem if the world dynamics is much faster than your
396 interaction with the world, e.g.

397 if you look at the gage only once an hour.

398 COnsider deciding where I should stop for gas.

It may only depend on the reading on the gas gauge and where the gas
399 stations are.

But, it feels like it requires something other than reflex, that it
requires looking into the future, which is something we'll get to in a
minute, simulating possible trajectories about how the world works but it
400 doesn't require remembering more stuff about the past.

This little distinction about whether an agent is reflexive (or reactive vs
non-reactive) depends on whether you have to remember something about the
401 past.

402 </section 16>

403 <section 17>

404 Here's an agent with memory.

Everybody who has taken theory of computation is familiar with this
405 picture.

You can take any finite state machine that you want to and decompose it
406 like this.

The idea is that you have some part that gas feedback and you grab all the
407 feedback and put it together and that's how you get to remember stuff.

Then you have some part that says "given what I remember, what should I
408 do?"

409 We'll often call this mapping from whatever I know to actions a "policy".

410 And so here we would call this part the policy and this part the memory.

But, another way to think about it is that it is an estimate of the state
411 of the world, it is a distilled picture of what's going on outside.

It's what I've chosen to remember about the history of percepts that I've
412 had in the world.

413 In some fields, such as control theory, this is called a state estimator.

Whatever it is, it takes the sequence of percepts you've had over time and
the sequence of actions that you've had over time and somehow remembers
414 something about it.

Another way of thinking about it is that it takes whatever you knew before,
what you just saw and what you just did and maps that into whatever you
415 know now.

416 SO, it is in charge of keeping your mental state updated.

Then you can say that the problem of behavior is "given my mental state
(whatever I remember of what I've seen in the world) what action should I
417 take".

418 </section 17>

419 <section 18>

420 Let's talk about planning for a minute.

So this exactly about the question: What about deciding when to stop for
421 gas??.

Your choice of actions depend not just on what's going on right now but
422 what's going to happen in the future.

423 Intuitively, "should I do this or should I not?"

424 Well, it depends on what downstream events it's going to cause.

425 I want to argue that this is completely consisten with this view.

There is still some mapping between what I see right now into what I'm
426 supposed to do.

But, it's maybe that the justification you have to give for why this is a
427 good thing to do depends on what is going to happen in the future.

But you don't have access to what's going to happen in the future; there's
428 no input here from the oracle.

SO, you're still taking action based on what's happening right now but the
429 way you justify them is in virtue of what they'll cause to happen.

430 Let's look at what I would call a planning agent.

You can imagine an agent which still has the state estimation part, there
still the part that distills what we've seen into a picture of what's going
on in the world, but now the policy (big box) involves a search (you've
431 probably all seen a search tree).

That is, we take the state from the state estimator and imagine "what if
we take action1, what if we take action2, etc. Then, after we take action1,
what if we take action2, etc. I just had a flat tire, what if I call AAA -
432 then wait 6 hours.

What if I fix it myself, probably get fixed in 1/2 hour but I'd get covered
433 in mud.

So, there's different consequence that you can spin out of different ways
434 of doing things.

435 And, how do you evaluate these consequences?

436 With U, you take your utility function and you apply it.

How good is it to be covered in mud but ready to go in 1/2 hour, but first
437 how good is to be here for five hours but clean as can be.

438 Maybe one, maybe another.

439 But, given an utility function we can help pick which is the best.

So, you can imagine spinning out these consequences, picking which is the
440 best and committing to one of these actions.

You pick the action the immediate action that's on the path that looks
441 best.

This computation is really no different than that computation that I
drew over there, it's just a particular way to organize a computation of
442 choosing an action to take next.

But it's a way of organizing it in terms of what you think is going to
443 happen downstream.

Karl Popper was a philosopher of science and he thought about falsification
444 of theories and so on.

But, he says an advantage of being human (I would say of being a planning
445 agent) is that you can let your hypotheses die in your stead.

Rather than jumping off the cliff you can think about what it would be like
446 and not do it.

447 </section 18>

Appendix B

Segmentation Study Instructions

Part I: Task Definition

I. Introduction

The ultimate goal of this research is to be able to automatically generate summaries for spoken lectures. Text segmentation is the first step towards this goal. It involves partitioning a text into a set of coherent segments which reveal the topics discussed in the text. Knowing the underlying topical structure of text simplifies extraction of information and summarization.

A. Task Overview

Your task is to partition a set of transcribed lectures into a sequence of coherent segments and provide short topical descriptions for these segments. These descriptions should then be able to provide a high-level overview of the lecture content and enable easy access to the relevant sections covered in the lecture. The target number of segments for each transcript **will not** be given to you in advance, so you will need to segment the lecture into as many segments as you see fit and natural to convey the overall structure of the lecture.

For your task, you will need to:

- Indicate the major and minor topic breaks (consult Section 4 for further segmentation guidelines)

- Label each resulting topic with a short title

- Take notes about the lecture, recording any noteworthy lecture characteristics

Each lecture is about an hour long, and it may take up to 2 to 3 hours to segment, so allocate enough time to be able to segment each transcript without interruptions.

B. Lecture Materials

You will be asked to segment a set of lectures from an undergraduate Physics class. Recorded audio and a transcript of each lecture will be provided. The segmentation annotation software described in Part II will enable you to listen to the audio and examine the corresponding transcript concurrently. Note that the transcripts may contain occasional omissions or mistakes. When listening to the audio and reading through the transcripts try to understand the lecture content to the best of your ability, even if the material may be unfaimiliar to you.

II. Segmentation Guidelines

During your segmentation task you will be identifying places in the transcript where the topics change. Implicitly, you are also splitting the lecture into segments or blocks of text. After you locate the transitions, the segments are defined as the spans of text in between each neighboring pair of boundaries. *Note that the words "segment" and "topic" are used interchangeably in this manual, since each segment is supposed to convey a topic.*

It often happens that clear-cut transitions may not delimit substantive and coherent topics. For example, a brief digression may interrupt the flow of the main narrative, but in itself it provides little relevant information that contributes to the overall goal of the segment. Thus, while you need to identify strong transitions, you also need to verify that the segments that are thus defined convey a clear, prominent topic or goal.

Topic segmentation can be done at various levels of granularity. In the course of your segmentation, you may come upon transitions that denote minor subtopics and digressions that are not essential for a high-level summary. These subtopics can be a source of confusion. To make the task more explicit, we introduce an extra segmentation requirement.

In placing the segment boundaries you will need to distinguish between major and minor topic transitions.

A. Major Topics

The major topic transitions signify changes in important subject matter. The objective should be to place major boundaries only when a prominent topic under discussion changes to some other prominent topic. In addition, every segment needs to be cohesive and to a great extent self-contained. A topic change is prominent or significant enough to merit a major boundary when disregarding it impairs high-level understanding of the structure and the content of the lecture.

The sequence of major segments that is delimited by the transitional boundaries is contiguous. In other words, there are no gaps between major topics.

Brief statements that introduce the major segment belong inside the segment. For example, if the speaker says "Now I will talk about ..", then this statement needs to be included in the subsequent discussion of the topic. Likewise, brief concluding statements belong inside the major segment. For example, a speaker may say "So, I showed that ...", and this comment will need to be incroporated into the previous discussion.

B. Minor Topics

The minor topics are used for sub-topics related to the major topic, digressions, remarks, and other prominent transitions that are not crucial to understanding the high-level content of the lecture. These transitions need to be nested inside the major segments, so they can not span two major segments.

It is important that you introduce minor topic only when there is a clear-cut topical transition. You should not mark minor topic breaks after every couple of sentences. The span between boundaries must satisfy segment length constraints (see section 4.4). The minor transitions should not occur much more often than the major transitions.

In general the major/minor topic distinction corresponds to the prominence of a topic.

C. Topic Descriptions

To make sure that the segments are really self-contained and cohesive you are asked to provide descriptions for each of the segments.

In order to come up with a segment description, try to fill in one of the following statements with a specific topical noun phrase:

In this segment, the lecturer talks about ______

OR

In this segment, the lecturer presents _________

Some examples of appropriate descriptions could be *"gravity"*, *"centripetal acceleration"*, *"a proof of Theorem A"*, *"an application of principle B"*.

In a lecture entirely about biology, it would be inappropriate to provide the *"biology"* label for a segment, since it is too general. In general, if two consecutive segments have the same description, then either the descriptions are problematic, or the segments need to be merged.

In selecting descriptions you need to give preference to conceptual descriptions over descriptions that have to do with presentational or administrative issues. For example, if the lecturer draws a diagram that explains a particular phenomenon, then it is more appropriate to provide the name of the phenomenon then a *"diagram"* segment description.

If you are having difficulties coming up with a specific topical description for the segment distinguishing it from previous segments, then the segment may be a continuation of a previous segment. It can also be the case that the lecturer simply rambles on without a particular subject in mind, and you need to come up with a unifying description that ties all of the closely related topics that the speaker mentions.

Wherever appropriate you should also make use of 2 predefined topic labels: **INTRO** and **END**. **END** identifies a section after the point where you deem the lecture has ended. For example, there may be some background noise or an extended pause. This content of this section will be ignored in later analysis. Likewise, **INTRO** signifies the section from the beginning of the audio recording until the point where the actual lecture begins.

D. Segment Length

Segments have to be at least 5 sentences long, but they can range anywhere from 5 to more than 100 sentences. It is unlikely that you will encounter more than 3 consecutive short segments. So, segment with these length constraints in mind.

E. Tips

a. Read several sentences after the boundary in question before making the candidate boundary, because it might sound like a boundary, but the topic really continues

b. Look for trigger words that can signal transitions in topic such as "Now I will talk about ...".

c. Use the following test for determining whether to place segment boundaries: if you can remove one utterance/sentence and make the surrounding text cohesive, do not place a boundary before or after this intervening sentence.

Part II: Annotation Software

The following software user manual is an edited version of meeting segmentation instructions by Alexander Gruenstein and John Niekrasz.

(`http://godel.stanford.edu/twiki/bin/view/Public/NomosMainPage`)

I. Introduction

For segmentation annotation you will be using Annotate! or Nomos, an annotation tool created by Alexander Gruenstein and John Niekrasz at the Stanford Center for the Study of Language and Information (CSLI) for marking discourse features in transcripts of recorded meetings and/or lectures.

Ultimately, the annotations entered by the human annotator will be used to train a computer program to recognize the same discourse features in other recorded lectures. The basic format of Annotate! is straightforward. The tool plays the audio of the recorded lecture and simultaneously shows the transcript. Each speaker's individual transcript is

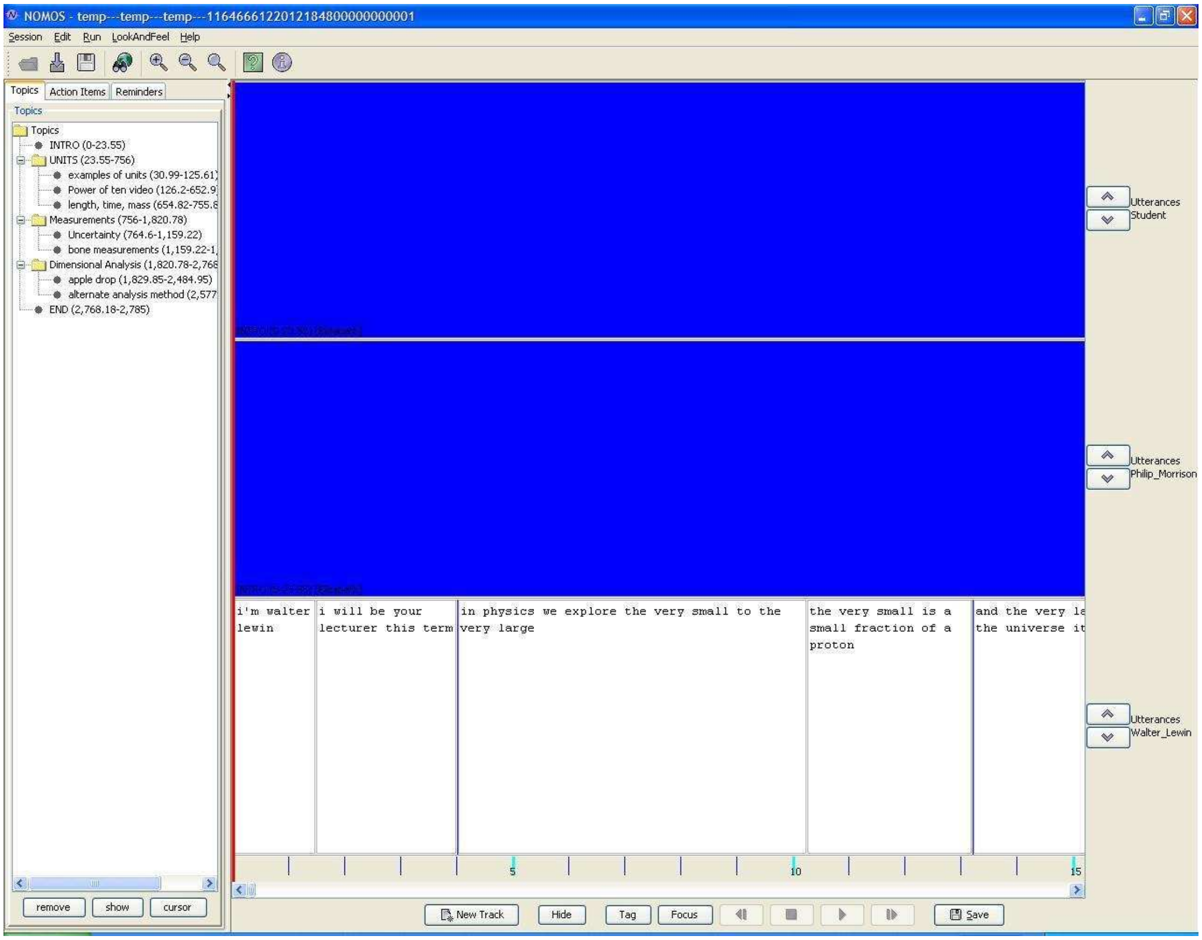

Figure B-1: Screenshot of the NOMOS Annotation System

shown in its own row, and the rows are stacked vertically to show all participants in the conversation. The utterances appear in white segment boxes staggered according to when they occur during the flow of the conversation. During play, a vertical red cursor keeps track of the time location within the lecture. There are also a number of features in the tool that allow the annotator to mark the transcript for discourse features, to take notes, to manipulate the play of the recording, etc.

A. Topic Breaks and Hierarchies

Topic boundaries are marked in the Annotate! tool at the beginning and end of each topic region. If the topic has yet to be named and restricted in its length, it will appear with the label “NAME ME!” and cover the temporal and visual space in the transcript that hasn’t yet been manipulated by the annotator. A boundary between topics is shown as a vertical line that extends through all speaker channels.

In Annotate! there are two kinds of topics: major and minor. A major topic appears as either a dark or light blue background (the color alternates) behind the transcribed speech. A minor topic appears as a set of smaller light or dark grey bands within the boundaries of the major topic. A major topic can include several minor topics, but minor topics cannot extend from one major topic to the next. While major topics are necessarily contiguous, minor topics need not be followed directly by another minor topic. Major topics are also necessarily continuous; no part of the discourse will be without a major topic, even if it is unnamed. The topic hierarchy appears in a window to the upper left of the transcript. Topics appear in chronological order, and those majors containing minors appear as folders with the minors listed below.

You can make use of predefined topic labels, selected from the drop-down menu when marking a shift. The predefined topic labels are always in capital letters to distinguish them from the custom labels given by the annotator. At this point you should only make use of the predefined topics: INTRO and END. These topics are all considered major.

B. Lecture Notes

Under the Edit menu, you will find Lecture Notes, a list of general notes or impressions compiled by the transcriber during their task. There is also a space where you as the annotator should include your own notes about the lecture. Examples of useful notes are the annotator's impression of the difficulty of the annotation task for the lecture, the presence of an agenda, whether the lecture had any troublesome patches (either technically or conceptually), or any items of interest from which others could benefit.

C. Aids to the annotator

The annotator is also expected to take notes about the lecture, recording its level of difficulty, presence of an agenda, degree of structure, or any noteworthy occurrences.

1. The Annotate! tool allows annotators to leave notes to themselves within the transcript. These "reminders" appear as dark pink vertical lines wherever the annotator tags and selects the REMINDER option. The reminder list appears in a window in

the upper right part of the tool, organized chronologically. Since determining topic shifts and action items is often difficult the first time the annotator goes through the lecture, REMINDER is useful for marking a place that the annotator wishes to revise or reconsider later. In particular, these help for marking possible topic shifts so you can go back and formally mark topic shifts and action items as needed.

2. Also under the Edit menu, the Search Annotations option allows you to search the topics in a set of lectures for an annotator-defined regular expression. Once the annotator name is selected and the search string entered, the search returns a list of topics containing the expression, each one listed with the lecture in which it occurs, the start and end times, the annotator's name, and the corpus containing the lecture.

III. USING ANNOTATE!

A. Opening a lecture for annotation:

1. Run Annotate!

2. Choose your name from the list of annotators. If it's not there, create a new one.

3. Click on File- Open to get a list of lectures and select the one you wish to annotate.

B. Transcript tools

There are number of buttons near the bottom of the Annotate! tool used for maneuvering around the transcript.

1. HIDE: This button hides the white segment boxes containing the utterances. This is useful when looking at multiple annotations in the comparison mode since it allows annotators to compare the agreement of the topic shifts and action items without the visual clutter of the utterance segments. Once the segments have been hidden, they can be made to reappear by clicking on the same button again, this time labeled SHOW.

2. TAG: This button allows the annotator to mark a major or minor topic shift or a reminder "on the fly" that is, at the moment the button is clicked. This same action

can be performed by left clicking the mouse in the transcript at the desired moment followed by clicking the TAG button.

3. FOCUS: This button will focus the screen on the red vertical cursor. This is a quick way to bring the screen back to the moment of play from another point. This button is useful for returning to a particular place if the annotator has left that spot to look at (but not listen to) another part of the lecture.

4. PLAY/PAUSE: This button controls the play of the audio files and the scroll of the transcript. Pressing the button when it says PLAY will cause the play to begin, and pressing the button when it says PAUSE will pause the play at that moment.

5. STOP: This button returns the cursor to where it was before the previous click on PLAY.

6. REPEAT: This button pulls the cursor back six seconds.

7. SKIP: This button moves the cursor forward two seconds.

8. SAVE: The SAVE button is self-explanatory. It should be used often. This button appears normal when changes have been made to the transcript and not saved, and dims to grey when all changes have been saved. It is a good idea to make sure that the button dims when clicked. If it does not, it may mean that your work has not been saved.

9. LEFT-CLICK ACTION: These radio buttons switch the mode of the tool so that the left click performs different tasks. ´Audio¡ is the standard mode. In this mode, left-clicking will stop play if the lecture is playing and will move the cursor to that point. ´Action item¡ mode means that left-clicking on utterances will include them in the list of action items. The item under which the utterance will be listed depends on what item is currently selected in the action item list above the transcript. Right-clicking in either mode gives the annotator the option of tagging a topic shift, action item, or reminder at that moment.

Lastly, there are two more features you can use to manipulate the visuals:

1. Scroll bar: click and drag on the scroller to move around in the transcript.

2. Zoom slider: click and drag up/down to zoom in/out. This feature allows you to view large, non-detailed portions of the transcript, and is especially useful for comparing the results of different annotators in Comparison Mode.

C. Marking Topic Breaks

Topics are marked by clicking the TAG button below the transcript or by right-clicking on the moment of shift and making a selection. Whenever a major or minor topic has concluded, the annotator should name it using the dialogue box that appears for editing the topic label. These labels should be composed by the annotator according to his or her best estimation of the subject matter of the topic, and can be edited at any time. The labels are intended to help the annotator conceptualize the topic relationships, and will not be used in future analysis. If a topic changes but the discussion later returns to the same subject matter, the multiple topic regions should be given the exact same labels. Interrupting a previous topic region by marking a new one gives the option of naming the region that just ended (and in the case of major topic boundaries, the region to follow). The annotator can either make a new label or select from the drop-down menu, which includes the predefined topics and the topics the user has already marked (allowing the annotator to informally link section's by using the same name for them). Ending a major topic always starts a new major, but upon ending a minor topic you can either return to the major containing it or start a new minor topic.

The Topic Breaks window has several buttons to help the annotator manipulate the organization of topics.

1. REMOVE: removes the topic. A major with minors within it cannot be removed.
2. SHOW: brings the start of that topic to the center of the transcript window.
3. CURSOR: the same as "Show", but moves the cursor to the center as well.
4. EDIT: used to change the name of the topic.
5. MERGE: unifies two topics under the same name. In order to merge, both topics must be highlighted (use the Ctrl button to highlight more than one topic at a time). The

user must then select a name for the new larger topic. Merging two majors retains their minor topics.

6. PROMOTE: turns a minor topic into a major. If there are minors following it within major topic ”A”, promoting it will keep those later minors in topic ”A”.

7. DEMOTE: turns a major into a minor, incorporating it into either the previous major or the following major.

D. Marking Reminders

A reminder can be marked by clicking the TAG button or by right-clicking at the desired moment, then selecting REMINDER. Once marked, a dialogue box will appear and ask the annotator to label the reminder. The annotator should label it with any phrase or key words that will help him or her. The buttons in the ´Reminders¡ window have the same function as those of the same name in the ´Topic Breaks¡ window.

E. Keyboard Shortcuts

Some actions can be made easier with the help of the keyboard (NOTE: these shortcuts can only be used while the PLAY/PAUSE button is highlighted):

1. Enter/Return: the same as TAG. (Used to mark topic boundaries and reminders.)

2. Right arrow: the same as SKIP.

3. Left arrow: the same as REPEAT.

4. Spacebar: pauses or restarts the play of the transcript.

Appendix C

Stop Words List

yes	becomes	every	ie	not	somehow	un
no	becoming	everyone	if	nothing	someone	under
said	been	everything	in	now	something	until
n't	before	everywhere	inc	nowhere	sometime	up
'm	beforehand	except	indeed	of	sometimes	upon
's	behind	few	interest	off	somewhere	us
're	being	fifteen	into	often	still	versa
'll	below	fify	is	on	such	very
a	beside	fill	it	once	system	via
about	besides	find	its	one	take	vice
above	between	fire	itself	only	ten	was
across	beyond	first	keep	onto	than	we
after	bill	five	last	or	that	well
afterwards	both	for	latter	other	the	were
again	bottom	former	latterly	others	their	what
against	but	formerly	least	otherwise	them	whatever
all	by	forty	less	our	themselves	when
almost	call	found	ltd	ours	then	whence
alone	can	four	made	ourselves	thence	whenever
along	cannot	from	many	out	there	where
already	cant	front	may	over	thereafter	whereafter
also	co	full	me	own	thereby	whereas
although	computer	further	meanwhile	part	therefore	whereby
always	con	get	might	per	therein	wherein
am	could	give	mill	perhaps	thereupon	whereupon
among	couldnt	go	mine	please	these	wherever
amongst	cry	had	more	put	they	whether
amoungst	de	has	moreover	rather	thick	which
amount	describe	hasnt	most	re	thin	while

an	detail	have	mostly	same	third	whither
and	do	he	move	see	this	who
another	done	hence	much	seem	those	whoever
any	down	her	must	seemed	though	whole
anyhow	due	here	my	seeming	three	whom
anyone	during	hereafter	myself	seems	through	whose
anything	each	hereby	name	serious	throughout	why
anyway	eg	herein	namely	several	thru	will
anywhere	eight	hereupon	neither	she	thus	with
are	either	hers	never	should	to	within
around	eleven	herself	nevertheless	show	together	without
as	else	him	next	side	too	would
at	elsewhere	himself	nine	since	top	yet
back	empty	his	no	sincere	toward	you
be	enough	how	nobody	six	towards	your
became	etc	however	none	sixty	twelve	yours
because	even	hundred	noone	so	twenty	yourself
become	ever	i	nor	some	two	yourselves

Table C.1: Continuation of the Stop Words List

Bibliography

J. Allan, J. Carbonell, G. Doddington, J. Yamron, and Y. Yang. 1998. Topic detection and tracking pilot study. In *Procedeengs of DARPA Broadcast News Transcription and Understanding Workshop*, pages 194–218.

Doug Beeferman, Adam Berger, and John D. Lafferty. 1999. Statistical models for text segmentation. *Machine Learning*, 34(1-3):177–210.

Dimitri P. Bertsekas. 2001. *Dynamic Programming and Optimal Control*. Athena Scientific.

M. Cettolo, F. Brugnara, and M. Federico. 2004. Advances in the automatic transcription of lectures. In *Proceedings of ICASSP*, pages 769–772.

Freddy Choi, Peter Wiemer-Hastings, and Johanna Moore. 2001. Latent semantic analysis for text segmentation. In *Proceedings of EMNLP*, pages 109–117.

Freddy Y. Y. Choi. 2000. Advances in domain independent linear text segmentation. In *Proceedings of the NAACL*, pages 26–33.

Kenneth Ward Church. 1993. Char_align: A program for aligning parallel texts at the character level. In *Proceedings of the ACL*, pages 1–8.

A. Dielmann and S. Renals. 2005. Multistream dynamic Bayesian network for meeting segmentation. In *Proceedings Multimodal Interaction and Related Machine Learning Algorithms Workshop (MLMI–04)*, pages 76–86.

Christian Fugen, Matthias Wölfel, John W. McDonough, Shajith Ikbal, Florian Kraft, Kornel Laskowski, Mari Ostendorf, Sebastian Stuker, and Kenichi Kumatani. 2006. Advances in the automatic transcription of lectures: The isl rt-06s evaluation system. In *Proceedings of ICSLP*.

Sadaoki Furui. 2003. Recent advances in spontaneous speech recognition and understanding. In *Proceedings of IEEE Workshop on Spontaneous Speech Processing and Recognition*, pages 1–6.

Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. 2003. Discourse segmentation of multi-party conversation. In *Proceedings of the ACL*, pages 562–569.

James R. Glass. 2003. A probabilistic framework for segment-based speech recognition. *Computer Speech and Language*, 17(2–3):137–152.

Barbara L. Grosz and Candace L. Sidner. 1986. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204.

Alexander Gruenstein, John Niekrasz, and Matthew Purver. 2005. Meeting structure annotation: Data and tools. In *Proceedings of the SIGdial Workshop on Discourse and Dialogue*, pages 117–127.

M. A. K. Halliday and Ruqaiya Hasan. 1976. *Cohesion in English*. Longman, London.

Marti Hearst. 1994. Multi-paragraph segmentation of expository text. In *Proceedings of the ACL*, pages 9–16.

Xiang Ji and Hongyuan Zha. 2003. Domain-independent text segmentation using anisotropic diffusion and dynamic programming. In *Proceedings of SIGIR*, pages 322–329.

David Kauchak and Francine Chen. 2005. Feature-based segmentation of narrative documents. In *Proceedings of the ACL Workshop on Feature Engineering for Machine Learning in Natural Language Processing*, pages 32–39.

Athanasion Kehagias, Pavlina Fragkou, and Vassilios Petridis. 2003. Linear text segmentation using a dynamic programming algorithm. In *Proceedings of the EACL*, pages 171–178.

Hideki Kozima. 1993. Text segmentation based on similarity between words. In *Proceedings of the ACL*, pages 286–288.

Erwin Leeuwis, Marcello Federico, and Mauro Cettolo. 2003. Language modeling and transcription of the ted corpus lectures. In *Proceedings of ICASSP*, pages 232–235.

William C. Mann and Sandra A. Thompson. 1987. Rhetorical structure theory: A theory of text organization. Technical report, USC Information Sciences Institute.

Alex Park and James Glass. 2006. Unsupervised word acquisition from speech using pattern discovery. In *Proceedings of ICASSP*, pages 409–412.

Alex Park. 2006. *Unsupervised Pattern Discovery in Speech: Applications to Word Acquistion and Speaker Segmentation*. Ph.D. thesis, Massachusetts Institute of Technology.

Rebecca J. Passonneau and Diane J. Litman. 1997. Discourse segmentation by human and automated means. *Computational Lingustics*, 23(1):103–139.

P. Perona and J. Malik. 1990. Scale-space and edge detection using anisotropic diffusion. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(7):629–639.

L. Pevzner and M. Hearst. 2002. A critique and improvement of an evaluation metric for text segmentation. *Computational Linguistics*, 28(1):pp. 19–36.

Jay M. Ponte and W. Bruce Croft. 1997. Text segmentation by topic. In *Proceedings of the European Conference on Digital Libraries*, pages 113–125.

Martin F. Porter. 1980. An algorithm for suffix stripping. *Program*, 14(3):130–137.

Matthew Purver, Konrad P. Körding, Thomas L. Griffiths, and Joshua B. Tenenbaum. 2006. Unsupervised topic modelling for multi-party spoken discourse. In *Proceedings of ACL/COLING*, pages 17–24.

Jeffrey C Reynar. 1994. An automatic method of finding topic boundaries.

Jeffrey Reynar. 1998. *Topic segmentation: Algorithms and applications*. Ph.D. thesis, University of Pennsylvania.

S. W. Roberts. 1959. Control chart tests based on geometric moving averages. *Technometrics*, 1(3):239–250.

Gerard Salton and Christopher Buckley. 1988. Term weighting approaches in automatic text retrieval. *Information Processing and Management*, 24(5):513–523.

Jianbo Shi and Jitendra Malik. 2000. Normalized cuts and image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 22(8):888–905.

Elizabeth Shriberg, Andreas Stolcke, Dilek Hakkani-Tur, and Gokhan Tur. 2000. Prosody-based automatic segmentation of speech into sentences and topics. *Speech Communication*, 32(1-2):127–154.

Caroline Sporleder and Mirella Lapata. 2006. Broad coverage paragraph segmentation across languages and domains. *ACM Transactions on Speech and Language Processing*, 3(2):1–35.

John Swets. 1988. Measuring the accuracy of diagnostic systems. *Science*, 240(4857):1285–1293.

Masao Utiyama and Hitoshi Isahara. 2001. A statistical model for domain-independent text segmentation. In *Proceedings of the ACL*, pages 499–506.

P. van Mulbregt, I. Carp, L. Gillick, S. Lowe, and J. Yamron. 1999. Text segmentation and topic tracking on broadcast news via a hidden markov model approach. In *Proceedings of ICSLP*, pages 2519–2522.

Z. Wu and R. Leahy. 1993. An optimal graph theoretic approach to data clustering: Theory and its application to image segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 15(11):1101–1113.