Topical Structure in Long Informal Documents

Anna Kazantseva

Thesis submitted to the
Faculty of Graduate and Postdoctoral Studies
in partial fulfillment of the requirements
for the degree of Doctor of Philosophy in Computer Science

School of Electrical Engineering and Computer Science
Faculty of Engineering
University of Ottawa

Abstract

This dissertation describes a research project concerned with establishing the topical structure of long informal documents. In this research, we place special emphasis on literary data, but also work with speech transcripts and several other types of data.

It has long been acknowledged that discourse is more than a sequence of sentences but, for the purposes of many Natural Language Processing tasks, it is often modelled exactly in that way. In this dissertation, we propose a practical approach to modelling discourse structure, with an emphasis on it being computationally feasible and easily applicable. Instead of following one of the many linguistic theories of discourse structure, we attempt to model the structure of a document as a tree of topical segments. Each segment encapsulates a span that concentrates on a particular topic at a certain level of granularity. Each span can be further sub-segmented based on finer fluctuations of topic. The lowest (most refined) level of segmentation is individual paragraphs.

In our model, each topical segment is described by a segment centre – a sentence or a paragraph that best captures the contents of the segment. In this manner, the segmenter effectively builds an extractive hierarchical outline of the document. In order to achieve these goals, we use the framework of factor graphs and modify a recent clustering algorithm, Affinity Propagation, to perform hierarchical segmentation instead of clustering.

While it is far from being a solved problem, topical text segmentation is not uncharted territory. The methods developed so far, however, perform least well where they are most needed: on documents that lack rigid formal structure, such as speech transcripts, personal correspondence or literature. The model described in this dissertation is geared towards dealing with just such types of documents.

In order to study how people create similar models of literary data, we built two corpora of topical segmentations, one flat and one hierarchical. Each document in these corpora is annotated for topical structure by 3-6 people.

The corpora, the model of hierarchical segmentation and software for segmentation are the main contributions of this work.

Acknowledgements

Think where man's glory most begins and ends,
And say my glory was I had such friends.

William Butler Yeats

It took many years to complete this thesis, and it was nearly abandoned several times. This dissertation would have never been finished if not for the generous help and unfailing support of my friends and family. Only a few are listed here and many more have helped with advice, a kind word, a joke or stories of their own. Thank you all.

I am particularly grateful to two people who helped me along this journey and who had almost angelic patience with me: my husband Alex and my supervisor Dr. Stan Szpakowicz.

Stan has completely shaped me as a researcher in NLP. He had shown by his own example that asking sharp research questions and being honest and thorough when looking for answers is far more important then any external recognition. Having him as a supervisor allowed me never to worry about submitting a thesis that was less then solid: he would never have let me. He waited for me during the years when my daughter was born and my research was only going backwards. Without his patience, wisdom and high standards this work would never have existed.

My husband Alex allowed me the luxury of pondering abstract questions and wondering about the language while he readily took upon himself the baby, the house and other pressing everyday issues. He supported me through doubts, deadlines, acceptances and failures. No less important, he always provided a fresh, intelligent, critical look at my work. Thank you for being my love, my partner and my friend. There are many more things I am grateful for, but those I will tell him in person.

Many thanks go to my parents, Evgeniya and Vladimir Kazantsev for their support, but especially to my sister Rimma. By believing in me as much as she did, she helped me in the darkest hours.

I am incredibly lucky to have the friends that I had during these long years. Many thanks to Christina Hatziandreou, Mahshid Farhoudi, Anna Pisareva, Rachel Wallace, Aida Alves, Lianne Rossman-Bhatia and Ramanjot Bhatia, Adam Lafrance, Monica Tanase, Monique

Brugger, Diman Ghazi, Dianne Wennerwald and others. Your personalities, support and enthusiasm provided on-going inspiration and made the whole process fun. Thank you all.

Special thanks to Chris Fournier for many discussions of text segmentation and its evaluation and for helping me use his SegEval software.

I am grateful to Lucien Carroll for making the EvalHDS software available and to Christian Smith for allowing me to use a beta version of CohSum.

Many thanks go to the annotators who created the corpora described in Chapter 3 of this dissertation.

Last but not least, many thanks to the members of my examining committees – Ana Arregui, Ash Asudeh, Diana Inkpen, Maite Taboada and Peter Turney – for their insightful, wise comments.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

1.1 Overview

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.

"Beware the Jabberwock, my son
The jaws that bite, the claws that catch!
Beware the Jubjub bird, and shun
The frumious Bandersnatch!"

He took his vorpal sword in hand;
Long time the manxome foe he sought–
So rested he by the Tumtum tree,
And stood awhile in thought.

And, as in uffish thought he stood,
The Jabberwock, with eyes of flame,
Came whiffling through the tulgey wood,
And burbled as it came!

One, two! One, two! And through and through
The vorpal blade went snicker-snack!
He left it dead, and with its head
He went galumphing back.

"And hast thou slain the Jabberwock?
Come to my arms, my beamish boy!
O frabjous day! Callooh! Callay!"
He chortled in his joy.

'Twas brillig, and the slithy toves
Did gyre and gimble in the wabe;
All mimsy were the borogoves,
And the mome raths outgrabe.

Figure 1.1: *Jabberwocky* by Lewis Carroll

Lewis Carroll's famous poem is an example of nonsense verse – a light genre of poetry usually read to children, which does not make sense in the context of what we know about the world. It cannot be fully processed semantically because the reader does not know who Jabberwock is, or why one should be afraid of Bandersnatch. This, however, presents little problem for children, to whom a dentist is no less mysterious (or frightening) a creature. The poem contains enough syntactic, morphological and temporal information for a reader to flesh out an interpretation, even with missing variables. Without creating a full interpretation, the reader can easily understand the initial idyllic setting with slithy toves gimbling in the wabe, then the warning a son receives, his battle with the Jabberwock, the parent's joy and eventually the return to the initial setting.

This particular example contains quite a bit of regular lexemes to aid the understanding. However, a lot of burden also falls on structural devices: conjunctions that mark the syntactic structure of the sentences, commas, exclamation marks, markers of direct speech, shifting in tense and verbal aspect and, of course, the traditional shape of the 19^{th} century English poem. While lacking in lexical semantics, the poem can be read and enjoyed because its structure is familiar and interpretable.

People leverage the knowledge about the structure of discourse quite heavily, both in terms of its comprehension and its creation. It is useful in conversations and institutional

rhetoric [van Dijk, 2007, Jacobs and van Hout, 2009, Rocci, 2009]. It has been shown to affect how people summarize and recall information [Endres-Niggemeyer, 1998]. People structure their writing according to expectations for a particular genre as well as building an argument so as to convey the desired message [Jacobs and van Hout, 2009].

This dissertation describes a model of discourse structure of a document that is based on topic. A document is modelled as a tree of topical segments, with the top-most segment corresponding to the entire document and bottom-most segments to individual paragraphs. Each segment is characterized by relatively constant topic at a certain level of granularity. Nodes close to the root of the tree capture major topical components of a document. The focus of discourse shifts continually and is never exactly the same even in two adjacent sentences. These finer fluctuations are captured by lower levels of the segmentation tree, with leaves denoting individual paragraphs.

Figures 1.2 and 1.3 illustrate the ultimate objectives of this project, using Chapter 2 of the novel *The Moonstone* by Wilkie Collins as an example.[1] The novel and this particular chapter will be used as an example throughout this dissertation. The full text of Chapter 2 is available in Appendix A. The paragraphs of the text in Appendix A are numbered. These numbers are used throughout Figures 1.2 and 1.3.

Figure 1.2 contains a manually created outline of that chapter. The annotator was requested to segment the chapter in places where there is a perceptible shift of topic, first identifying the most noticeable topic fluctuations and then segmenting each identified span further, according to more subtle fluctuations of topic, all the way down to the level of individual paragraphs. The annotator was also requested to label each segment in a way that briefly describes what it is about. The outline in Figure 1.2 is the result of her work. Figure 1.3 contains the corresponding segmentation tree. Here, the chapter is modelled as a tree of topical segments. The leaves of the tree correspond to individual paragraphs in the chapter. Each level of the tree denotes main topical components in the document when viewed at a particular level of granularity. The root node corresponds to the entire chapter, the level just below it – to the coarsest level of segmentation.

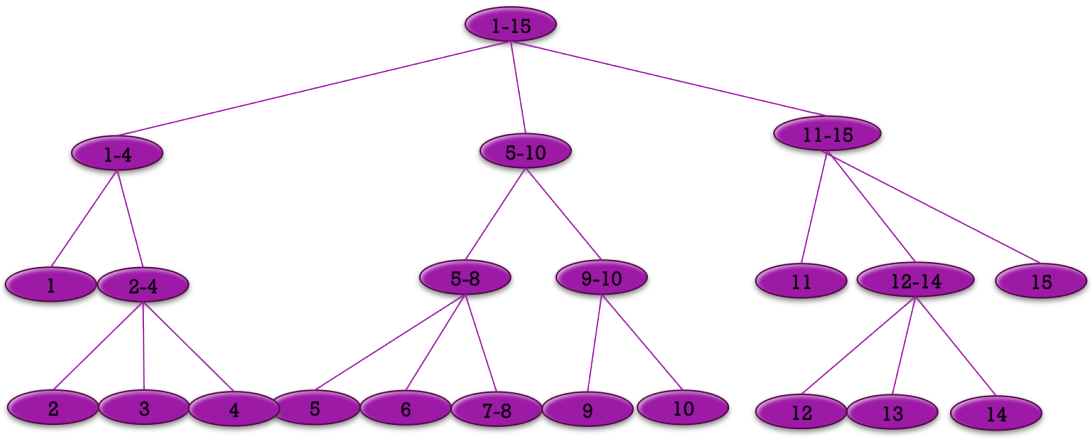
The main objective of this research project was to build such hierarchical structures automatically. While our algorithm outputs a central sentence or a paragraph for each segment in the tree, we focused mainly on building topical trees and not on producing

[1] In these Figures and further in this dissertation we preserve the annotators' original spelling.

Figure 1.2: Example segmentation of Chapter 2 of *The Moonstone* by Wilkie Collins. Annotator 2.

1. Paragraphs 1-4: the single life of the narrator (a note: paragraph one sets the time the story begins)
    (a) paragraph 1: the diamond's location
    (b) paragraphs 2-4: how he came to be in his career
        i. paragraph 2: how he met the lady
        ii. paragraph 3: how he became hired by the lady
        iii. how he came to be promoted to bailiff
2. paragraphs 5-10: life with a love interest before and after marriage
    (a) paragraphs 5-8: motivations for finding a wife
        i. paragraph 5: loneliness
        ii. paragraph 6: economics and love
        iii. paragraphs 7-8: the lady's approval
            A. paragraph 7: asking for approval
            B. paragraph 8: receiving approval
    (b) paragraphs 9-10: experiences of marrying and being married to a woman
        i. paragraph 9: apprehensions about marriage
        ii. paragraph 10: mediocrity in married life
3. paragraphs 11-15: life as a widower and into old age
    (a) paragraph 11: death of the narrator's wife and first mention of daughter Penelope
    (b) paragraphs 12-14: convincing the narrator to alter his occupation
        i. paragraph 12: Lady lulls him into security with kindness
        ii. paragraph 13: narrator realises her plan, and his pride fights against it
        iii. paragraph 14: narrator is convinced after reading Crusoe he will feel better about it tomorrow
    (c) paragraph 15: the difficulty of writing a story without accidentally writing about oneself (a note: only technically in his old age, not really about his old age)

Figure 1.3: Example of a topical tree for Chapter 2 of *The Moonstone* by Wilkie Collins.



coherent labels for each segment. That is why the result is more like the tree in Figure 1.3, not the complete outline in Figure 1.2. Performing the segmentation on its own is difficult enough. In order to find the appropriate labels, we would need to summarize each segment reasonably well and to either extract or generate an appropriate label. Quite often, but not always, the segment centres provide an adequate summary of the segment. At this stage, we made no effort to generate good labels for the nodes in the topical trees.

The main reasons for modelling discourse structure in this manner are computational feasibility and applicability. Topic is defined as a set of entities under discussion [Webber et al., 2012, p. 440]. This definition is rather loose and it is not always possible to provide a more precise view. The definition becomes task-dependent in that it depends on what the final goal of segmentation is [Purver, 2011, p. 292]. In fiction, as the topic changes, the shift is often accompanied by other changes in structure: the narrator may shift, a dialogue may end, the tempo of narration may change, *etc*. Therefore, a topical tree in Figure 1.3 collapses several paradigms into one representation. It subsumes information about temporal structure of the narrative, about dialogue-act structure, about presentational and other types of discourse relations.

Yet, there are several reasons why we feel this is an adequate representation. Mainly, it is a viable trade-off between usefulness and feasibility. Modelling a novel as a topical

tree creates a structure that says something about the main themes in the novel. For many generic NLP applications, the theme, or "aboutness", is of high relevance. One such application is generic automatic summarization, which deals with finding the crucial pieces of information capturing what the document is about. It has been shown that the knowledge of discourse structure can be helpful in summarizing documents, *e.g.,* [Uzêda et al., 2010]. However, automatically determining detailed information about discourse structure is a challenging problem (see Section 2.3 for a review). Topical trees are a useful, if less informative, alternative. To date, models based on topical segments have been used in information extraction, essay analysis and scoring, automatic assessment of coherence of text, automatic dialogue systems, *etc*. [Webber et al., 2012]. On the other side of the issue is computational feasibility. There is a number of detailed and well-researched theories of discourse structure and of narrative structure (see Chapter 2 for a detailed review), yet most of them are described in terms too abstract to be useful for computational modelling, or they do not scale well to such large documents as novels. That is why we decided to choose a simple but useful representation that can be constructed relatively easily.

The model proposed here is suitable for most expository and narrative types of discourse. While we use certain devices to alleviate the problem of low lexical repetition in fiction (see Chapter 6), overall the methods we use are applicable to any data type. The software would need to be fine-tuned and it may work better or worse depending on the genre and on the ultimate objectives. For example, one would expect poorer performance on dialogues, or in situations where the desired segmentations may not correspond to shifts of topic.

The particular implementations described in this proposal are geared towards literary data, especially novels. However, we test our tools on several benchmark datasets that contain data of different genres, namely speech transcripts, chapters of medical textbooks and Wikipedia articles.

Contributions. We hope that this work contributes something to the field of automatic discourse analysis as well as to research on creating tools for processing literature. In our view, there are several key contributions.

The first and possibly most important contributions are the two algorithms for topical segmentation of documents – one for building flat structures and another for producing topical trees of any desired depth. Both have been implemented in Java and are publicly

available. This is particularly important for the hierarchical segmenter, because we are not aware of other hierarchical segmenters available for public use. The segmenters are described in Chapters 4 and 5 of this dissertation.

The second contribution is two high-quality corpora of manual segmentations of literary data. The first corpus consist of single-level segmentations, along with manually created labels summarizing every segment. The second, smaller corpus consists of hierarchical segmentations with labels for every segment in the tree. The corpora contain respectively 20 and 9 chapters of the novel *The Moonstone* by Wilkie Collins. We have 4-6 annotations for each chapter in the first corpus and 3-6 annotations for chapters in the second corpus. Both corpora are publicly available. To the best of our knowledge, there are no comparable datasets for topical segmentation of literature that are available for public use. The datasets are described in Chapter 3.

The third, smaller contribution is a similarity metric that can be used to augment the estimates of topical similarity between sentences and thus improve the quality of topical segmentation. The metric relies on syntactic information about referential expressions and thus can be used for texts with low rates of lexical cohesion. Unfortunately, the improvement brought by this metric is not significant, but it never worsens the results. The metric is described in Chapter 6.

1.2 Motivation

This section explains why studying computational models of discourse structure is a worthy topic and what benefits such models may provide.

Put simply, high-level structure of discourse is a major component of its overall meaning. Semantics of individual sentences are a crucial piece of the puzzle. However, without a meaningful "big" picture individual sentences would remain just that: a string of disjoint ideas. A hierarchy of coherence relations joins individual sentences into a coherent interpretation, hence affecting ultimate interpretation, ease of comprehension, referential connections and also relative importance of various components.

People process discourse sequentially, from left to right one sentence at a time. However, the writing process involves imposing a structure and the reading process involves (re-)construction of this structure in the mental representation. If the structure cannot be

re-created, then we call such discourse incoherent: individual items make sense, but there is no overall meaning to a document.

Ultimately, in the field of Natural Language Processing we are concerned with extracting and utilizing meaning in various ways. In text summarization, we want to find meaning and express it as briefly as possible; in machine translation, we are concerned with rewording meaning in a different language; in question answering, we are concerned with finding a specific piece of meaning, etc. That is why we think that studying computational models of discourse structure is a worthy endeavour – because it is (most likely) necessary to adequately extract and represent the meaning of documents.

None of the aforementioned statements are novel. Young and Becker [1966] have shown that coherent discourse cannot be constructed one sentence at a time. It is also well known that discourse is more than a sequence of unconstrained sentences [Renkema, 2004, pp. 35-50]. Kehler [2002] has shown that discourse relations influence the interpretation and construction of referential expressions, verb ellipsis, gapping and tense; also see [Renkema, 2004, pp. 87-120]. However, Computational Linguistics has been somewhat slow to put this knowledge to use, if not to recognize its importance in theory. In the past 20 years, part-of-speech tagging, syntactic parsing, anaphora resolution and some other intermediate applications have become reliable and indispensable tools. Of course, quite a number of people have worked on robust discourse parsing, but to the best of our knowledge, there are no publicly available robust discourse parsers (see Section 2.3 for a review of relevant related work). What is more important, there is no strong theory of how to make use of them for other NLP tasks, such as machine translation, automatic summarization, question answering *etc*.

This is particularly the case for text genres that lack rigid structure: blogs, e-mails, speech transcripts, literature. The few existing tools perform quite poorly on these data, where they would be most useful. This is a pity because, while it is possible to know something about the structure of a scientific paper *a priori* and to encode it manually, it is practically impossible to create such templates for blogs, meeting transcripts and most literary data.

Given all the fascinating dependencies between discourse structure and other linguistic phenomena, it seems that making even a small step in this direction would be beneficial. Modelling discourse structure as a tree of topical segments is a far cry from building a tree

of coherence relations (see Section 2.3 for a review). This representation certainly contains less information, yet it is feasible to build it for a large document, such as a novel.

1.3 Overview

Put simply, in this research, a document is modelled as a tree of segments, each focused on a particular topic. The root of the tree is the complete document. Nodes close to the root correspond to large segments of text; topic shifts between high-level segments are noticeable and rather abrupt. Nodes further down in the tree model finer fluctuations of the topic. A manually constructed topic tree for Chapter 2 of *The Moonstone* is shown in Figure 1.3.

In this work, we explore two main points:

1. An efficient algorithm for flat and hierarchical text segmentation. We use factor graphs as a framework for hierarchical text segmentation. The proposed algorithm is an adaptation of a recent clustering algorithm, Affinity Propagation [Frey and Dueck, 2007]. Given a sequence of sentences (or paragraphs), the algorithm outputs segment boundaries and segment centres – points that best describe what the segment is about. We derive and implement two topical segmenters based on Affinity Propagation: a flat segmenter and a hierarchical one.

2. A similarity measure that helps detect topic fluctuations in running text. It was already mentioned that this work is concerned with informal documents and mostly with literature. In such documents the rate of lexical repetition is much lower than in formal texts [Hoey, 1991, Graesser et al., 2004, Louwerse et al., 2004]. That is why we develop a syntactically motivated measure for tracking topic changes between sentences. The basis for this metric is the theory of functional domains developed by Talmy Givón (1981). According to Givón, the accessibility of concepts mentioned in text is in inverse relation with how much information the author needs to specify in order to (re)introduce a concept. Concepts mentioned most recently will require the least amount of coding and can be expressed by pronominal or zero anaphora. Concepts that are more difficult to access must be invoked explicitly (e.g., using a

proper noun). If they are even less accessible, invoking them will require pre- or post-modification (to remind the reader what they are).

1.4 Justification of the proposed method

1.4.1 Reasons for using topical segmentation

Modelling novels as trees of topical segments is certainly not the only way to go about modelling discourse structure. A large body of research in philosophy, narratology and literary studies describes regularities in literature from different angles (see Section 2.1 for a review). Some researchers have tried to model literary narratives computationally (Section 2.2). There is also a lot of research that deals with how form and function are linked in discourse in general (Section 2.3). Why did we choose topical segmentation as the model of choice?

There are several reasons. They are explained in more detail in Chapter 2 which provides the context for comparisons, but we outline them here briefly.

The main issue with non-computational models of narratives is that they are expressed in terms which do not lend themselves easily to algorithmization. While it makes sense to view a book as consisting of a message (fabula), an event structure (story) and the actual text, this is hardly helpful computationally, since it is very challenging to build a model of event structure for even a toy story, let along get at the core message it is meant to convey. Section 2.1 provides a review of relevant non-computational models. None of them can be applied computationally with any reasonable reliability.

Section 2.2 describes recent work in computational narratology which builds upon traditional narratology, as well as offers its own insight into how narratives can be modelled computationally. While this area has witnessed rapid growth lately, most approaches would also not scale easily to novels. Additionally, to verify their applicability one would need to create a corpus annotated according to the specifications of a model. Since such models are rather detailed, it would be prohibitively expensive and time-consuming to annotate a work as large as a novel.

We also decided against using existing research on generic approaches to modelling discourse structure (Section 2.3). Issues of scalability and creating a corpus of fine-grained

annotations are seen here again. Additionally, existing models of discourse structure, *e.g.,* RST [Mann and Thompson, 1987], were developed for expository texts and it is far from obvious how to apply them to literary data. Literature is full of metaphors, dialogues, criss-crossing topical threads, *etc*. While it is possible to think of extensions, it is neither straight-forward nor guaranteed to be useful computationally.

We decided to use topical segmentation because, despite not being the most informative or detailed model, it is likely to be useful and it can be computed relatively easily given the state-of-the-art in NLP. It is also simple enough cognitively to allow the creation of a benchmark corpus to test our hypotheses.

1.4.2 Reasons for using a syntactically motivated similarity measure

Most available segmenters rely only on word repetition to model topic in discourse. Given how well-studied measures of text similarity and relatedness are, this is far from being the most informative metric. Some research on text segmentation also uses dictionary-based and corpus-based similarity metrics, but none of the publicly available segmenters offer such options (see Section 6.2 for review of related work).

However, lexical cohesion (and synonymy is one of the devices of lexical cohesion) is generally lower for texts that are cognitively simple to process. In other words, the simpler the topic at hand is, the less need there is for the author to explicitly code what he is talking about. Therefore, any similarity metric that relies only on lexical information will have a rather low upper limit on texts that are prevalent in real life: non-scientific books, personal correspondence, speech transcripts.

That is why in this work we propose a similarity metric that uses syntactic information about referential expressions in text. It does not improve the results as much as we would like it to, but it does offer a small improvement.

1.5 Organization

This dissertation is structured as follows. Chapter 2 provides an overview of research on automatic modelling of discourse structure in general and on modelling literary narratives in particular. Chapter 3 explains how we plan to evaluate our progress and describes in

detail the two datasets that we have created for the purposes of studying topical structure of narratives and for the evaluation of our system. Chapter 4 describes the part of the system developed thus far and how it compares with the state-of-the-art. Chapter 6 outlines a syntactically-motivated measure of topical similarity. Chapter 7 concludes this dissertation by discussing the results of this work, possible improvements and outlining our plans for future work.

1.6 Conclusion

This work is concerned with modelling discourse structure of literature, mainly novels. We chose to model structure using topical trees, which capture fluctuations of topic. This way to model structure is not specific to literature, yet we chose it for several reason. In a situation where there is no theoretical framework that can be applied computationally (see next chapter for a comparative review of alternative approaches), topical segmentation provides a light-weight method. Topical trees can be computed with relative ease and encouraging performance. They capture the notion of topic in a way that can be useful for other applications, as we hope to demonstrate in future work.

The main contributions of this work are the algorithms and the software for single-level and hierarchical topical segmentation, and the corpora of manual segmentation for a part of the novel *The Moonstone*.

Topical trees are not specific to literature. They can be built for almost any genre. In fact, in this dissertation we also build them for Wikipedia articles. Similarly, the syntactically-motivated similarity metric outlined in Chapter 6 is also not specific to literature and can hopefully be useful for any coherent text with low rates of lexical repetition. However, the methods used in this work are chosen first and foremost so as to work on literature. We also would like our methods to scale to large corpora of literature and to be fast, light and not require excessive amounts of human labor. If they are applicable to other genres of text (*e.g.,* speech), so much the better. Yet, our main interest and focus is literature. The statistical nature of our methods reflects a trade-off between what needs to be done and what can feasibly be done – for now, of course.

1.6.1 Previous publications

Parts of chapters 3 and 4 have been published in [Kazantseva and Szpakowicz, 2011, 2012]. Parts of chapters 5 and 6 have been published in [Kazantseva and Szpakowicz, 2014a,b].

Chapter 2

Related work

This work focuses on modelling the structure of long literary narratives, mostly novels. This question has not been studied in depth by the community of Natural Language Processing researchers. However, relevant aspects of it have been well explored in several fields.

When attempting to model the structure of novels computationally, one is naturally inclined to look into literary theories of novels. Narratologists and literary critics have been asking questions about main elements of novels, trends, tendencies and laws of literature for many centuries. It is a vastly diverse body of work most of which, unfortunately, is outlined in terms too abstract to be applicable computationally. However, in Section 2.1 we briefly review a few relevant theories of literary narrative.

Another highly relevant body of research is what we loosely term *computational narratology*. Under this broad umbrella, we place research on the structure and interpretation of stories (or, more generally, narratives) by people who are eventually interested in applying their theories computationally. Research on story understanding was a popular topic in the AI community during the 1970s and 1980s (e.g., [Cullingford, 1978, Dyer, 1983]). More recently, it has been picked up with a somewhat different emphasis in the area of Digital Humanities and a growing community of researchers explicitly working on computational models of narratives. Many of these theories are highly relevant and in fact very promising. However, most of them have been developed for short narratives and cannot yet be applied to novels with acceptable accuracy. Section 2.2 reviews the approaches to modelling narratives computationally.

Yet another point of view comes from the community of computational linguists working on discourse structure. There is a number of well-developed and tested theories of how small units of discourse (for example, sentences) combine into larger semantic units, how the combined meaning can be composed and how the process can be modelled computationally. With a fair number of researchers working in this area, an ACM special interest group (Special Interest Group on Discourse and Dialogue (SIGDial)) and several annual workshops, there has been a lot of progress. However, several aspects of this research made us decide against using existing theories of discourse structure to model novels. First of all, from the theoretical point of view, these theories have been developed for expository text and lack explanatory power for narratives (for example, temporal relations or relations between various characters). Secondly, it is far from obvious how these theories can scale to large texts such as novels (or even short stories), since they have been developed and explained on relatively short expository texts. Section 2.3 provides a brief overview of this work and how it relates to the research proposed here.

Section 2.4 provides a brief overview of related work on text segmentation, directly explaining what has been done in the area of modelling discourse using methods such as we propose here.

In Section 2.6, we explain why we chose to pursue the proposed method in the context of what is already known and/or computable for long literary narratives.

2.1 Elements of literary narrative from the perspective of literary theory and narratology

Narrative fiction is defined as narration of a succession of fictional events [Rimmon-Kenan, 1983, p.3]. It is generally agreed upon that a work of literature (or, more generally, a story) has several levels of abstraction:

1. *The text*: the actual discourse that tells the story. It is the only part immediately available to the reader.

2. *The narration*: the act of production or the telling of the story.

3. *The story*: the series of events, abstracted from their disposition in text.

This classification is due to Rimmon-Kenan [1983, p.3]. Similarly, Bal [1985, p.5-10] distinguishes between the *text* (the actual words that tell the story), the *fabula* (a series of events experienced by the actors) and the *story* (a subset of the events from the fabula chosen to convey the message of the story). There are other similar classifications (e.g. [Propp, 1968, Greimas, 1971]) that distinguish layers of meaning in the story based on what the message is as opposed to how it is expressed.

Many have expressed the idea that at a certain level stories share common elements and the variety of the stories in the cultural heritage is due to the variety of ways these elements can be combined, not to the infinite variety of elements. The idea has been pioneered in 1928 by Vladimir Propp [1968] who manually analyzed one hundred Russian folk tales and proposed that they consist of *functions* – low level cognitive elements of the plot. Examples of these functions are *Absentation* (a hero leaves home), *Interdiction* (a hero receives a warning not to perform certain actions or go a certain way), *Trickery* (a villain attempts to deceive the hero), etc. Propp postulated that the order of these functions is restricted even if not all of them are present in every story. The idea was further advanced and built upon by the structuralists in Prague and in France. Claude Lévi-Strauss [1955] proposed that underlying the stories of a given culture are atomic units called *mythemes*. Mythemes are combined together to form the discourse of the myths. Barthes [1970] is another philosopher who attempted to cast narrative discourse as a sequence of surface units stemming from an underlying narrative grammar.

These theories had an enormous influence on the AI community during the 1970s and the 1980s (see Section 2.2 for details). While there is no universally accepted narrative grammar available today (much less one that could be used computationally), the idea of viewing narrative discourse as a sequence of basic cognitive units has laid the foundations for the computational study of narratives as we know it today. However, several shortcomings make it very difficult to apply computationally. First of all, basic cognitive units such as Propp's function or Lévi-Strauss' mythemes are defined in a way that makes it difficult for the-state-of-the art NLP technology to identify them in the actual textual discourse. Additionally, many have questioned the very essence of such an approach. Ryan [1991] points out that the number of possible actions that may be described in a story is probably infinite – as opposed to the number of terminals in any lexicon. Bod et al. [2012] question how objective Propp's definitions of function are by demonstrating that his annotations are

not trivial to reproduce. And, more broadly, more contemporary literary theory (e.g. Eco [1979]) emphasizes the active role of the reader in the interpretation of literature, suggesting that representing literature as a sequence of pre-determined cognitive units may be too simplistic.

2.1.1 Events in stories

We have defined the narrative as the sequence of events, thus suggesting that events are central elements of any story. Indeed, a story can be defined as a sequence of events governed by temporal and causal links.

In the previous section, we have made a distinction between the layers of meaning in the story, separating what is being told from how it is being told. The importance of events in stories can hardly be underestimated. In terms of structural analysis, one of the layers of meaning, inevitably, is the events of the story abstracted from the language and, depending on a specific theory, from particular details of the story. Propp's functions are event-based, as are the basic units of the story structure proposed by Claude Bremond [1973].

Bal [1985, p.19-22] remarks that even though it is agreed upon that events are constituent units of the story, it is far from obvious which sentences specify events, as well we as which events should be included in the storyline. She suggests three main criteria for defining what constitutes relevant events (also called functional events of the story). These criteria are *change, choice* and *confrontation*. While not cast in stone, these rules suggest that an event is a change of state in the story world, which opens a possibility that may or may not be taken up by a character or the author. The third criterion refers to the idea that interesting events in the story are usually those that revolve around a conflict between several characters or groups of characters.

If a story (or fabula, using Bal's terms) is a sequence of events, how are these to be combined? There are two paramount principles of combination: temporal links and causality.

While we may be inclined to think of temporal progression as the natural order of events, in reality, this scenario is almost never encountered in stories. If a story contains at least two characters, events become concurrent, and the narrated timeline becomes jagged, possibly with flashbacks and digressions [Rimmon-Kenan, 1983, p.17].

The second principle of combination of events is causality. Foster [1963, p.93] postulated that the difference between a story and a plot lies in causality.

1. The king died and then the queen died.

2. The king died and then the queen died of grief.

While the former is a story, the latter is a plot.[1] Temporal and causal connections are often closely related in stories and cannot be readily separated.

Events combine into sequences to form the fabula – or the storyline. Bremond [1973] defines stories in terms of narrative cycles – event-based constituents of the story structure. Each cycle is viewed as either an improvement or a deterioration. By comparing the narrative cycles it is possible to compare the stories structurally.

Barthes [1970] categorizes events into *kernels* and *catalysts* based on their function in the story line. Kernels are events that open possibilities for plot development. Having been offered a university admission, a character may or may not accept it. Thus, this is a kernel event. On the other hand, pacing the room, looking out the window while making up one's mind are examples of catalyst events – events that magnify or delay one of the kernels.

2.1.2 Characters in stories

The omni-presence of human or human-like agents in narrative discourse is quite indicative of our fascination with our selves. The human mind is preoccupied with comprehending itself, musing about our values, imperfections, limitations. It is fascinating that the narrative, as we know it, is always about some aspect of human nature, even if related indirectly. It has been shown that people find stories to be more comprehensible when there are well-formed relationships between character actions and recognizable character goals, e.g., [Graesser et al., 1994]. However, it is not clear whether characters should be looked upon as people, or template-like structures that can be reconstructed from words [Rimmon-Kenan, 1983, p.31].

[1] Foster defines a story as a sequence of events arranged in time. This definition is misaligned with the definition from the previous chapter where a story denotes a specific layer of meaning. In our terminology, Foster's *story* is closer to *narrative*.

Literary theory does not quite agree on whether characters supersede events or the other way round. In the *purist* tradition, a character exists only insofar as it is a part of images and events conveyed by the author [Aristotle, 1961, Propp, 1968, Greimas, 1971]. In the *realist* tradition [Ferrara, 1974] characters are modelled after real people and eventually they acquire some degree of independence from the text.

It is possible to evaluate characters along the axes of complexity, development, penetration into inner life and other traits. The basic principles of character reconstruction from the text are repetition, similarity, contrast and implication.

A body of research in cognitive psychology also points to the importance that agents and their goals have in story comprehension and recall. Heider and Simmel [1944], Dik and Aarts [2007] showed that even when people are shown cartoons about inanimate objects, they comprehend them in terms of animate agents, their struggles and goals.

Not surprisingly, a considerable number of representations for modelling narratives is expressed in terms of characters and their affective states. We will discuss formalized models based on characters in Section 2.2.2.

2.1.3 Conclusions

This section offered a very brief overview of narratives and their main components from the point of view of literary theory and narratology. It is agreed upon that stories are multi-layered entities with different levels of abstraction and meaning, from the actual words to the abstract message of the story. Some of the crucial elements of stories are the event and temporal structure, the character networks and the goals and motivations of those characters.

Literary theory and narratology are broad and diverse fields. Studying stories in terms of their characters and event structure are certainly not the only interesting forms of analyses. One may also look at stories in terms of their temporal structure, focalization point, style, etc. In this review, we have included only the aspects most relevant to computational modelling of narratives.

As we will see in Section 2.2, much of the terminology and agreed-upon knowledge about the structure of stories has been borrowed by computational narratologists. Much of the analyses described in this section cannot be easily automated. For example, the

structuralist approach seems very intuitive to computer scientists, but automatically identifying functions or mythemes in literature is not possible with the state of the art in Natural Language Processing. Classifying micro-sequences of events into deteriorations and improvements in the style of Claude Bremond would allow finding similarities across stories, but even precise identification of events is a challenging task, let along classifying event sequences into positive and negative ones (although this is perhaps a more feasible task than finding mythemes automatically).

2.2 Computational approaches to modelling narratives

The theories of narratives outlined above (along with many more in-depth studies) have influenced an interdisciplinary field known as computational narratology. Computational narratologists are concerned with studying the laws and regularities of the narratives using computers and corpus-based methods, as well as how this knowledge can be used for automatic generation of narratives.

The theories of Vladimir Propp and those of the structuralist school have had a serious influence of the AI community during the 1970s and the 1980s. The idea that surface structure of a story (i.e., its text) is generated from an underlying grammar is very appealing to the mind of a computer scientist. Additionally, simultaneous research in cognitive psychology had shown that previous experience and knowledge affects perception of the world. In the now well-known experiment, Brewer and Treyens [1981] have invited students to participate in an experiment where they were asked to describe the contents of a room. The students were told they would need to wait in a colleague's office while waiting for the experiment to start. After a brief period of waiting, they were led to a different room and asked to describe the contents of the previous office. The descriptions provided by the students contained items that one would expect to see in an office: books, notebooks, etc. Curiously, the students "saw" some office-like items that were not there and failed to notice items which are normally not found in offices.

BORIS [Dyer, 1983] is an example of a well-known story-understanding system from that period. The system is based on the assumption that common world knowledge about typical situations can be organized using *scripts* – general and somewhat stereotyped knowledge about a particular situation. A script about eating at a restaurant, for example, would

"know" about menus, waiters, paying the bill, etc. A script about divorces would include information about the two spouses, some conflict that led to the divorce, the knowledge that before the divorce they lived together and that this will not continue to be so after the fact. BORIS processed stories word-by-word, invoking a rich semantic representation for each lexeme, which was manually encoded in an elaborate lexicon. Additionally, it relied on a manually encoded set of scripts which encoded common-sense knowledge about the situations that BORIS was asked to process. BORIS was able to "understand" the stories it "knew" about and answer questions about them.

A story-understanding system with a slightly different flavour is PAM (Plan Application Mechanism) [Wilensky, 1977]. PAM's understanding of stories is based on identifying the intentions of its characters and connecting all the events in the story by causal links in as far as they are related to characters' goals. As PAM "read" a story, it produced a representation using conceptual dependencies, linking them into a knowledge structure Interpretation.

A story-generation system that relied on hand-coded knowledge is UNIVERSE [Lebowitz, 1983]. UNIVERSE was able to create brief stories, focusing on melodramatic interpersonal situations, creating highly believable characters. UNIVERSE was organized around plans – those of the author and those of the characters. It had at its disposal a library of manually encoded plot fragments, which it combined to create interesting story-lines.

BORIS, PAM and UNIVERSE exemplify many of their contemporaries *e.g.,* [Norvig, 1989, Cullingford, 1978, Thorndyke, 1975]. Most of these systems suffered from a knowledge acquisition problem: while awing audiences with performance on familiar scripts, they were helpless in unfamiliar situations. Worse yet, extending such a system is a painstaking and a labor-intensive process. Several attempts have been made to circumvent this problem [Norvig, 1983] or acquire the knowledge automatically [Chambers and Jurafsky, 2009], but it largely remains an unresolved issue.

Later efforts have revolved around finding more abstract representations that would encode higher-level regularities in narratives and would not require so much manual effort. In the following two subsections, we will describe some of the representative approaches to modelling narratives computationally, roughly subdividing them into event-based and character-based. However, it should be noted that this sub-division is somewhat illusory, as all of them model both aspects to some extent. (This is in line with the ongoing debate in narratology as to whether characters are subordinate to events or the other way round.)

We placed them into one category or another depending on what is of primary importance to a given representation.

2.2.1 Event-based approaches to modelling narratives

Aristotle considered the plot to be the soul of tragedy [Aristotle, 1961]. A fair number of more contemporary formal representations of narratives are based on modelling main events in a story and how they are related causally or temporarily.

Trabasso and Sperry [1985], Trabasso and Stein [1997] model the process of narrative comprehension based on causality. The model, Recursive Transition Causal Networks (RTCNs), builds a representation consisting of the fragments of a story connected with causal links. There are six classes of such fragments: Settings, Initiating Event, Reactions, Goals, Actions and Outcome of goals. There is a link between two nodes in the graph if one event necessarily follows another, in the eyes of the reader. The model is primarily a model of how humans process narrative discourse. It has been extensively used to study inferences occurring during the reading process.

One possible way to represent narratives computationally (especially for the purposes of narrative generation) is by using plans. Planning allows building plot lines that are temporarily and causally sound. IPOCL (Intentional Partial Order Causal Link planner) [Riedl and Young, 2010] is a story generation system built with two primary goals in mind: logical causal progression of the plot and believable characters (or, more, precisely, sequences of character actions that are believable). Planning is a convenient mechanism to model the progression of the story world from the initial state to the desired (goal) state. However, the authors observe that the goal state of the story need not be the same as the goal state of characters (indeed, it is rarely the case). IPOCL simultaneously searches in the space of temporal and causal solutions in terms of the author's goal and also in the space of the characters' intentions which may be provided by the author or discovered in the process of generation. Such a process is intended to mimic how the reader may process a story.

Indexter [Cardona-Rivera et al., 2012] is a system that extends the planning-based model of IPOCL so as to model salience of events as they are processed by the reader. The system uses information already available in the planning representation as inputs to an event-indexing situation model. In their model, each event is assigned several indices,

such as time index, space index, protagonist index, causal index or intention index. The salience of each new event is determined by how many indices it shares with the previously seen events. The authors stipulate that salience is crucial to comprehension and that such a way to model narratives will allow generating effective stories. The system is not yet fully implemented, however.

RTCNs and planning-based approaches offer a rather in-depth way of modelling the narratives. These are characterized by relatively fine-grained details and special attention to how humans may process the text. However, it is not quite clear how to apply these models to narrative understanding. RCTNs have been used to model the inference process, but IPOCL and Indexter are systems geared toward story-generation rather then understanding. They offer considerable degree of flexibility compared to the earlier script-based systems, but the process remains still fairly labor-intensive.

Recent work by Chambers and Jurafsky [2008, 2009] attempts to alleviate the knowledge acquisition bottleneck by learning typical event chains and eventually narratives schemas automatically in a fully unsupervised fashion. Chambers and Jurafsky [2008] propose a representation they call *narrative chains* – a typical set of events centred around a common protagonist. For example, one such chain may be $\{$*admits, pleads, convicted, sentenced*$\}$. The authors remark that events that share an actor are likely to belong to the same narrative chain. They use coreference resolution and thus select events with the same protagonist. The system then chooses actor-verb pairs with the highest point-wise mutual information scores and orders them temporally to build a set of typical narrative chains. Chambers and Jurafsky [2009] extend that work to include typical semantic roles of the actors involved in the events, thus building up to a database of narrative schemas. In the same vein, Li et al. [2012] use a corpus of short narratives created through crowd sourcing to find typical events in common social situations.

Mani [2010] also develops an event-centred, and even more so a time-centred, approach to narrative understanding. Mani describes a system that is capable of building elaborate time-lines and tracking spatio-temporal trajectories of individual characters throughout discourse. The usage of the TARSQI [Verhagen et al., 2005] toolkit allows the identification of events, time-stamping them where possible and ordering them temporarily. It then identifies individual characters and their development (and evaluation by the reader) throughout discourse.

Another very ambitious and expressive representation for modelling narratives is proposed by Elson [2012]. Story Intention Graphs (SIGs) are designed with three goals in mind: to provide a robust representation that covers a wide range of stories while having enough formal power to allow finding deep thematic similarities across stories, answering question and performing inferences. The last requirement is that the representation be expressed in simple enough terms to allow persons without special training to annotate previously unseen data. SIGS consist of three layers. The first is *the textual layer* which contains the actual discourse as the reader sees it. The second layer is *the timeline layer* which contains propositions that encode events and states as they occur in the story. The last layer is *the interpretative layer*. This layer contains the nodes indicating beliefs and goals of the characters. It also connects to the prepositions in the timeline layer. The arcs in the timeline layer mostly encode temporal ordering of the events, while the arcs in the interpretative layer mostly indicate causality, conditions and actualization states. This is a very detailed and powerful representation. However, at this stage, unfortunately, there is no obvious way to build such structures automatically.

Having reviewed event-based computational models of narratives, it is important to situate this research in their context. According to the objectives outlined in Section 1.3, we aim to build a representation with an emphasis on computational feasibility and, ultimately, on applicability. To this end, we model novels as trees of topical segments. Unlike the systems described in this section, topical trees are a rather shallow representation – we do not explicitly track characters, or events, nor do we build timelines or account for how a reader may process narrative discourse.

RCTN and the planning-based approaches are also different in their intended usage: our long-distance focus is more on high-level discourse understanding rather than on narrative generation or on modelling of narrative comprehension. The approaches closest to ours (although they are still wildly different) are those of [Chambers and Jurafsky, 2008, 2009] and [Mani, 2010] in that they favour large-scale processing with little or no supervision, using either unsupervised learning or off-the-shelf tools that have already been pre-trained.

The system described here is intended more as an intermediate tool that could help the algorithms that do in-depth processing of narratives perform better by identifying early on portions of the tree that are relevant to a particular task and discarding those that are not. When looking for narrative schemas, for example, it is feasible to assume that the events

of a typical schema would occur within a single topical segment, not across several. When performing anaphora resolution (employed by by both [Chambers and Jurafsky, 2009] and [Mani, 2010] and in general likely to be crucial in such a character-oriented genre), the knowledge of the topical structure is likely to be helpful when choosing the most probable antecedents.

Topical segmentation is not unique to narratives – in fact it does not explicitly account for any of the qualities that make narratives so distinct: the temporal progression and complications, the importance of characters, multiple points of view, etc. However, it is a useful and simple high-level tool. In this work, we focus on creating corpora and tools specifically geared toward working with this genre.

2.2.2 Character-based computational models of narratives.

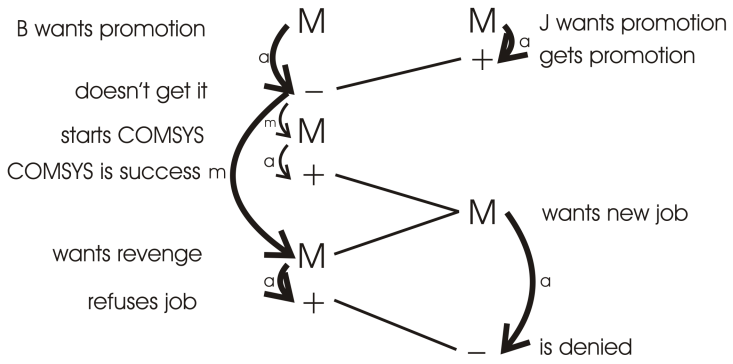
We have already mentioned that human-like protagonists are central to stories. It is, in fact, debatable whether it is the character or the events that are more essential to stories (see Section 2.1.2 for discussion). Be as it may, research in cognitive psychology [Graesser et al., 1994] has shown that characters and their intentions make stories more cohesive and easier to process.

In addition to their intrinsic importance to story-telling, it is relatively easy to track characters computationally. Event identification and temporal parsing and ordering are still not as accurate as we would like them to be. However, named entity recognition and anaphora identification (if not resolution) pose much lesser challenges. In this section we will review a handful of character-based approaches to modelling narratives computationally.

Lehnert [1982] proposed a model called the *plot units*. The author hypothesized that characters and their emotions, expressed through *affectual states*, are central to story understanding and eventually, to retaining the main points. At its core, a plot unit representation is a directed graph where nodes correspond to affectual states of the characters and arcs – to causal links. There are three types of nodes: positive nodes (+), negative nodes (-) and mental states (M). There are 4 types of causal arcs: motivation (m), actualization (a), termination (t) and equivalence (e). Additionally, the nodes may be connected by inter-character nodes, indicating how a particular state of one character may affect another. Figure 2.1 contains a toy story and a corresponding plot unit representation for it [Lehnert, 1982,

Figure 2.1: Example plot-unit representation for the following toy story from [Lehnert, 1982, p. 389].

John and Bill were competing for the same job promotion at IBM. John got the promotion and Bill decided to start his own consulting firm, COMSYS. Within three years COMSYS was flourishing. By that time John had become dissatisfied with IBM so he asked Bill for a job. Bill spitefully turned him down.



p.389].

Plot units are a flexible and powerful representation with an important advantage of modelling the affectual states of characters in an explicit manner. It it also quite detailed. However, it has an unfortunate disadvantage of being difficult to compute. Recently Goyal et al. [2010] attempted to build plot unit representations automatically. The plot-unit recognition system called AESOP was tested on a set of 34 of Aesop's fables. Each fable was assumed to have at most two characters that must be mentioned in the title of the fable. The authors built a *Patient Polarity Verb* lexicon (PPV) which contains verbs and their polarity effects on their arguments (for example, in *The fox ate the chicken* the effect of eating is positive for the fox but rather unfortunate for the chicken). Using a simple coreference resolution system and the PPV, AESOP constructs affectual state nodes. The system's capabilities for producing arcs are quite limited. While the approach is overall promising, the accuracy of the system is rather poor even on the constrained dataset.

Kazantseva and Szpakowicz [2010] propose a summarization system for the 19^{th}-20^{th} century short stories that also leverages information about the characters. Their system attempts to create summaries such that they do not reveal the plot but help the reader decide

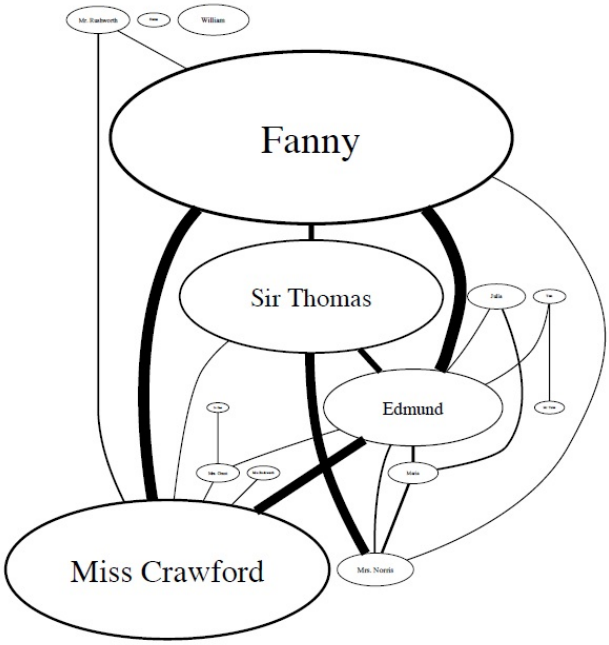
how interested they would be in reading the story. This extractive summarization system selects sentences that 'talk' about one of the main characters and describe states and not events, thus hoping to capture the most relevant aspects of the setting of the story.

Elson et al. [2010], Agarwal et al. [2012] model narratives yet from a different point of view. A narrative contains a network of characters. The character network in many ways defines and constrains the story to be told. The network may be tightly or loosely connected. It may be complete (corresponding to a story world where every character is in a direct relation to any other character) or have a number of relatively disconnected cliques (corresponding to a story with several disjoint communities of characters). In this representation the nodes in the network are characters – identified using named-entity recognition and coreference resolution. The arcs between the nodes correspond to the amount of direct interaction between a pair of characters in the story identified through dialogues in which both characters participate. An example social network extracted from Jane Austen's Mansfield Park is shown in Figure 2.2 (from [Elson, 2012, p.36]) This model has an advantage of being very computationally feasible and, given the state of the art of the NLP tools, quite precise.

Elsner [2012] also takes a character-centred approach to modelling novels, with an emphasis on finding similarity between two works and on distinguishing novels from non-novels. He defines a convolution kernel [Haussler, 1999] which measures similarity between two character-based feature vectors x and y. The system collects information about characters in the novel: for each one, it identifies information about lemmas co-occurring with it in the output of a dependency parser, the information about the frequency of character mentions and also about the emotional trajectory of each character. Two novels are found to be similar if they have characters with similar distributions of these features.

In this subsection, we have reviewed computational models of narratives that are character-centred. It is easy to notice that, compared to the event-based models, these are considerably easier to compute with reasonable accuracy. On the other hand, with the exceptions of plot units, character-based models offer a somewhat static view of the narrative and do not model temporal progression and the development of characters in an explicit manner.

Figure 2.2: An example of a social network extracted from Jane Austen's *Mansfield Park*. From [Elson, 2012, p.36]



2.3 Computational approaches to modelling discourse structure

In our discussion of various ways to model literary discourse, we have so far focused on its narrative nature. However, in addition to being a narrative, any literary work is simply a coherent document (at least most of the time). In fact, in this research, we do not leverage the narrative aspect of fiction in any specific way. Therefore, for the purposes of establishing proper context, it is useful to look at the body of work on modelling high-level semantic structure of any document. The field of *discourse studies* occupies itself with how form relates to function in the area of communication. In this section, we will briefly review several representative models of discourse structure of documents, with an emphasis on those that have been applied computationally.

It is an obvious statement that the meaning of discourse is more than the sum of meanings of its constituents (e.g., sentences). From the level of the clause to the level of complete document, the meaning of each unit combines with that of its neighbours in ways that are not necessarily sequential.

1. a. Jane fell on the sidewalk. b. She slipped on ice.

2. a. Jane fell on the sidewalk. b. She was taken to the emergency.

In the example above, the relation between 1a and 1b is a causal one: sentence 1b states the cause of 1a. In 2, the relation is that of consequence: 2a is likely to be interpreted as a consequence of 2a.

In a similar manner, larger spans of texts combine to form a meaningful interpretation. A chapter of a novel may be a continuation of the events describes in the previous one, it may present a parallel plot-line or a flashback explaining previous events. The property of a document where each piece has a functional role that contributes to the overall interpretation is called coherence. In more intuitive terms, a coherent document is a document that makes sense. The semantic relations holding between constituent elements are called coherence relations (also sometimes rhetorical, or discourse relations). Consider the following example:

3. a. Jane fell on the sidewalk. b. The president of the country was re-elected.

In 3, both 3a and 3b are well-formed and semantically valid sentences. Yet the fragment cannot be interpreted easily: there are no signals to explain how 3b relates to 3a and the generally assumed world knowledge is not sufficient to suggest an interpretation. This is an example of an incoherent text.

A good number of theories attempt to describe coherence relations, possible taxonomies of the resulting graphs and how they may be used in computational analysis. In this subsection we will review a few representative approaches and then explain why we did not use coherence relations to model novels.

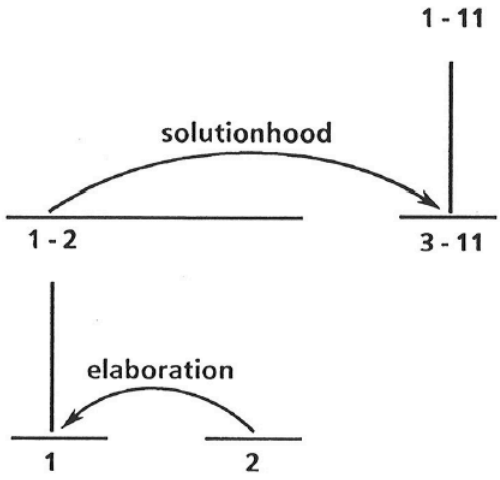
Perhaps the most computationally tried and tested theory of discourse analysis is the Rhetorical Structure Theory (RST) [Mann and Thompson, 1988]. RST was initially developed with text generation in mind. It describes a set of 23 coherence relations that may hold between adjacent spans of text. Examples of these relations are *Elaboration, Condition, Cause and Effect*, etc. In the set of relations described by Mann and Thompson [1988] 20 are mononuclear and 3 are multinuclear. Mononuclear relations are relations where one node (satellite) is less salient than the other (nucleus). For example, in the *Elaboration* relation the matter of the situation is presented in the nucleus and the details are in the satellite. In multinuclear relations both nodes are equally salient. Figure 2.3 shows an example of a small text annotated with RST-style coherence relations. The authors emphasize that the set of relations they propose is not complete and is not meant to cover all text types.

Since then a few researchers have attempted to automate the process of analysing existing texts for rhetorical relations. Marcu [2000] developed one of the first discourse parsers. Using an extended set of 54 relations he created a shift-reduce parser that output RST-style trees for input documents. The parser relied on a detailed analysis of cue phrases that potentially signal relations, as well as on syntactic information and lexical semantics. Additionally, he proposed an algorithm for automatic text summarization that used discourse structure trees to decide the most salient units of text. Marcu's rhetorical parser and summarizer were important in that they opened a new frontier for research in discourse parsing. However, the performance of the parser both in terms of coverage and precision was not very high, which is not surprising given the complexity of the problem (for example, when parsing the Brown Corpus, precision was 35.3% and recall – 26.7% [Marcu, 2000, p.171]). Additionally, the software was never released publicly.

Another discourse parser (and a summarizer based on it) were developed by Thione

Figure 2.3: An example of an RST-tree from [Mann and Thompson, 1987, p.51]

1. One difficulty is with sleeping bags in which down and feather fillers are used as insulation.
2. This insulation has a tendency to slip towards the bottom.
3. You can redistribute the filler
4. . . . 11.



et al. [2004]. The parser used a different set of relations and the summaries were created by pruning the discourse tree. The parser, however, is not publicly available.

Following the seminal work of Marcu [2000], Carlson et al. [2001] developed a large-scale corpus of articles selected from Penn Treebank [Marcus et al., 1993] annotated for discourse structure. The articles were first segmented into Elementary Discourse Units (EDUs) by at least two annotators and the differences were reconciled through discussion. Then, using the agreed upon EDUs, the discourse structure trees were manually constructed. The set of RST relations was also further extended to 78 relations. The resource, known as the RST Treebank has become an important benchmark for research in discourse parsing.

Another important benchmark collection for studying discourse structure and relations is the Penn Discourse Treebank 2.0 (version 1.0 is no longer available) [Prasad et al., 2008]. The resource consists of a corpus of about 1,000,000 words (also a subset of the Penn Treebank) annotated with discourse relations. The annotation is centred around 4 types of discourse connectives: subordinating and coordinating conjunctions and discourse adverbials. A connective acts as a predicate with arguments slots for sentences or clauses. The fourth type includes cases where the connective is implicit, and also cases where the relation is signalled in another manner (lexically or through entity coherence). However, the consequence of this annotation scheme is such that most of the annotated relations hold either within sentences or between adjacent ones.

A number of researchers in the following years attempted building automatic discourse parsers. The problem of text-level parsing is a very challenging one, so some researchers have tried to simplify it and concentrate on sentence-level parsing [Soricut and Marcu, 2003]. This is obviously a more manageable task since syntactic information can tell a lot about the sentence structure. However, such a limited approach is also limited in terms of its usefulness.

Despite the fact that there is no wide-coverage accurate discourse parser available today, there is a number of parsers with promising performance. Subba and Eugenio [2009] developed a discourse parser that relies on Inductive Logic Programming. In addition to using the regular inventory of cue phrases and lexico-syntactic information, the parser also derives knowledge from semantics inferred from WordNet [Fellbaum, 1998] as well as semantic similarity between textual spans.

Baldridge and Lascarides [2005] develop a probabilistic discourse parser within a different framework, Segmented Discourse Representation Theory [Asher and Lascarides, 2003] and successfully apply it to dialogues. Reitter [2003] and Hernault et al. [2010] propose text-level discourse parsers that use SVM to label the relations. HILDA parser [Hernault et al., 2010] uses shallow syntactic and lexical information, information about textual organization and structural information to label the relations. The system achieves an f-score of 48.8% [Hernault et al., 2010, p.19] on a subset of the RST Treebank. Feng and Hirst [2012] modify the set of features used by HILDA and slightly improve the performance of the parser.

One of the problems with determining coherence relations automatically is that few of them are signalled in ways that can be easily detected computationally. Cue phrases (e.g., *because, therefore*) provide good hints, but they are not always present. Taboada [2004, p.149] reports that only 30.86% of relations in a corpus of English conversations were marked by cue phrases. There are other ways, however, in which coherence relations can be signalled. There may be morphological or syntactic signals, tense and mood may signal certain relations as well as other mechanisms and, finally, world knowledge [Taboada, 2009, p.129].

Despite the fact that discourse parsing is a very active research area, there are several reasons why we decided not to pursue this line of research to model literature. First of all, the performance of the existing parsers is not very good. While it is acceptable at the sentence level, it gets worse as the texts get larger. Even a short story is considerably larger than a Wall Street Journal article from the Penn Treebank. To the best of our knowledge, there is no available discourse parser that could be feasibly used on fiction.

But the technical perspective is only part of the issue. First of all, as we have already mentioned, theories of discourse structure that have been used computationally (e.g., RST, SDRT) were developed for different genres of texts, mostly expository ones. It is far from obvious how to model novels using the inventory of those relations, whether it is appropriate and how useful and insightful such a model would be.

Clearly, there remains an issue of creating a benchmark corpus. Annotating texts for discourse relations is an extremely time-consuming task, requiring well-trained (or expert) annotators which renders it prohibitively expensive. There is also another side to the issue. In creating any benchmark resource, its reliability must be verified: among other

parameters, one must show that the human annotators agree between themselves enough to indicate that they understood the instructions well and were indeed annotating the same phenomenon. When using RST-style analysis, the annotators agree quite well at the sentence level, but the agreement decreases as the spans of text grow larger. Above the paragraph level, the agreement falls and the task becomes quite complicated cognitively (the latter observation comes from our initial attempts to re-create RST annotations for the RST Treebank). Additionally, Wolf and Gibson [2006, p.53-67] change the annotation scheme for the RST Treebank so as to allow the annotators to introduce crossing links and multiple parents, effectively allowing a graph, not a tree-based representation. They report significant inter-annotator agreement. In the resulting annotations, 12.5% of the arcs cross other arcs and 41.22% of all nodes have more than one parents [Wolf and Gibson, 2006, p.52, 60]. These findings suggest that trees may not be an appropriate way to model coherence relations after all.[2]

To conclude, we have decided not to use RST-style modelling of coherence relations for literature for the following reasons. Annotating a bench-mark corpus would be very difficult and prohibitively expensive. Also, considering lack of evidence that this type of modelling is well-suited to literature and many unanswered theoretical questions, we decided to use a less constrained model of a tree of topical segments. Obviously, such a model is considerably less insightful, but it can be built automatically and it can provide useful information for other NLP tasks.

2.4 Topical text segmentation

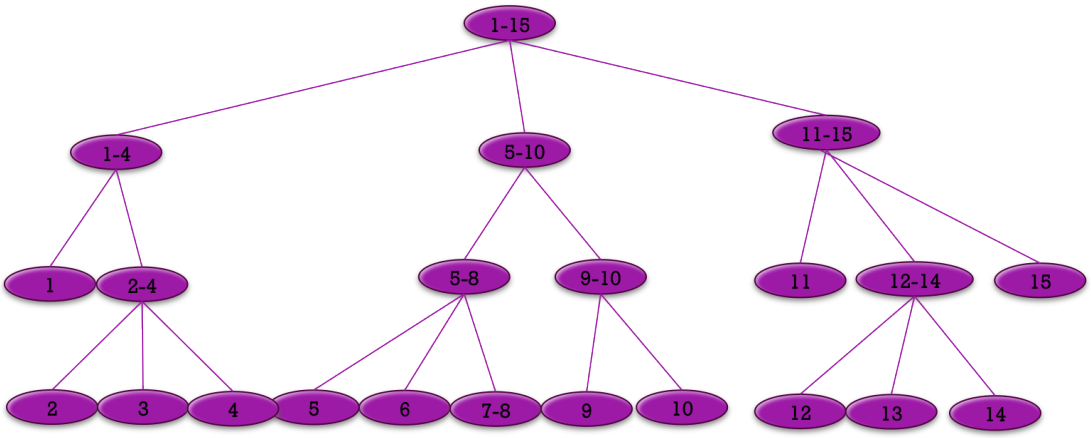
As we have mentioned in Chapter 1, in this research, we model a novel as a tree of topical segments, with the higher levels in the tree corresponding to large spans of text, the lower levels – to subsegments of those spans and the leaves of the tree – to individual paragraphs. Figures 2.4 and 2.5 have already appeared in Chapter 1, but we show them again here for convenience. They illustrate our ultimate objective – building an outline-like structure for chapters in novels (but without the goal of succinct well-formed labels as shown in Figure 2.5).

[2]Marcu [2003] questions the results pointing out that the whole annotation scheme may be not well defined.

Figure 2.4: Example segmentation for Chapter 2 of *The Moonstone* by Wilkie Collins. Annotator 2.

1. Paragraphs 1-4: the single life of the narrator (a note: paragraph one sets the time the story begins)
    (a) paragraph 1: the diamond's location
    (b) paragraphs 2-4: how he came to be in his career
        i. paragraph 2: how he met the lady
        ii. paragraph 3: how he became hired by the lady
        iii. how he came to be promoted to bailiff
2. paragraphs 5-10: life with a love interest before and after marriage
    (a) paragraphs 5-8: motivations for finding a wife
        i. paragraph 5: loneliness
        ii. paragraph 6: economics and love
        iii. paragraphs 7-8: the lady's approval
            A. paragraph 7: asking for approval
            B. paragraph 8: receiving approval
    (b) paragraphs 9-10: experiences of marrying and being married to a woman
        i. paragraph 9: apprehensions about marriage
        ii. paragraph 10: mediocrity in married life
3. paragraphs 11-15: life as a widower and into old age
    (a) paragraph 11: death of the narrator's wife and first mention of daughter Penelope
    (b) paragraphs 12-14: convincing the narrator to alter his occupation
        i. paragraph 12: Lady lulls him into security with kindness
        ii. paragraph 13: narrator realises her plan, and his pride fights against it
        iii. paragraph 14: narrator is convinced after reading Crusoe he will feel better about it tomorrow
    (c) paragraph 15: the difficulty of writing a story without accidentally writing about oneself (a note: only technically in his old age, not really about his old age)

Figure 2.5: Topical tree for the segmentation.



Most research on automatic text segmentation today revolves around a simple idea: when the topic shifts, so does the vocabulary [Youmans, 1991]. We can roughly subdivide existing approaches into two categories: locally informed and globally informed.

Locally informed segmenters attempt to identify topic shifts by considering only a small portion of the complete document. A classical approach is TextTiling [Hearst, 1997]. It consists of sliding two adjacent windows through text and measuring lexical similarity between them. Drops in similarity correspond to shifts of topic. More recently, Riedl and Biemann [2012] propose a variant of TextTiling called TopicTiling which uses Latent Dirichlet Allocation to find segment boundaries. Instead of computing similarity using the bag-of-words approach, TopicTiling leverages ids provided by a latent topic model. Other examples include text segmentation using Hidden Markov Models [Blei and Moreno, 2001] or Conditional Random Fields [Lafferty et al., 2001]. Locally informed methods are often very efficient because of lean memory and CPU time requirements. Due to a limited view of the document, however, they can easily be thrown off by short inconsequential digressions in narration.

Globally informed methods consider "the big picture" when determining the most likely location of segment boundaries. Choi [2000] applies divisive clustering to segmentation, while [Yaari, 1997] uses agglomerative clustering. Malioutov and Barzilay [2006] show that the knowledge about long-range similarities between sentences improves segmentation

quality. They cast segmentation as a graph-cutting problem. The document is represented as a graph: nodes are sentences and edges are weighted using a measure of lexical similarity. The graph is cut in a way which maximizes the net edge weight within each segment and minimizes the net weight of severed edges.

Another notable direction in text segmentation uses probabilistic generative models to find segment boundaries. Eisenstein and Barzilay [2008] treat words in a sentence as draws from a multinomial language model. Segment boundaries are assigned so as to maximize the likelihood of observing the complete sequence. Misra et al. [2011] use a Latent Dirichlet allocation topic model [Blei et al., 2003] to find coherent segment boundaries. Du et al. [2013] use a hierarchical Bayesian model to to come up with the best linear segmentation of a document. [Chen et al., 2009] describe another Bayesian model that is particularly successful at modelling topical structure when many documents from the same domain are available. The system incorporates two crucial assumptions into the Bayesian framework: the documents from the same domain tend to "talk about" similar topics and the ordering of topics within such documents is similar.

Such methods output segment boundaries and suggest a lexical distribution associated with each segment. Generative models tend to perform well, but are less flexible than the similarity-based models when it comes to incorporating new kinds of information.

Globally informed models generally perform better, especially on more challenging datasets such as speech recordings, but they have – unsurprisingly – higher memory and CPU time requirements.

The algorithm proposed in this research (see Chapter 4) is a globally informed algorithm that combines several desirable properties. It is unsupervised and only requires a small development set to fine-tune several parameters. Unlike most other segmenters, it does not require specifying the desired number of segments as an input parameter. It also provides some information as to what the segment is about, because each segment is associated with a segment centre.

In this work, we compare our algorithm (called *APS*) with the *Minimum Cut Segmenter* and the *Bayesian Segmenter* – two publicly available segmenters that have the best reported performance for informal data such as ours. While at this point the gain in performance is not wide, we feel that *APS* is an important alternative approach to topical segmentation.

Similarly to the *Minimum Cut Segmenter*, *APS* takes as input a matrix of similarities and uses all or most of the available information. Decoupling the computation of similarity from the actual algorithm offers flexibility which so far has not been explored by the text segmentation researchers. At the moment, the best automatic segmeters rely almost exclusively on the information about lexical similarity to track topical changes. However, this does not have to be this way. Apart from obvious additions such as synonymy and semantic similarity, we hope that the use of stylometric analyses will soon be explored in more depth (e.g., [Brooke et al., 2012]). In Chapter 6, we describe one possible (syntactically motivated) measure of topical similarity that we expect would be useful for literary data. Lexical similarity is far from the only source of information signalling a topical shift, even if so far this is what the research has been focused on.

While segmentation methods based on probabilistic generative models (e.g., [Eisenstein and Barzilay, 2008, Misra et al., 2011]) perform well and are quite efficient, they are quite inflexible when it comes to the kinds of information that can be used. When using generative models, individual lemmas are usually modelled as draws from several underlying distributions (which hopefully correspond to the main topics in the document). Apart from lexical distributions, many other markers signal topical fluctuation – discourse markers, anaphoric expressions and even the usage of function words. Incorporating this kind of information into a generative model is not straightforward because modelling these markers as draws from several underlying topical distributions does not make as much sense. This is why we chose to concentrate on a similarity-based model for topical segmentation.

Hierarchical segmentation is a more complicated task than single-level or flat segmentation. This is the case both in terms of writing a program to segment a text hierarchically and in terms of asking people to do so. Most research on text segmentation has been focused on building and evaluating single-level segmenters (e.g., [Hearst, 1997], [Choi, 2000], [Malioutov and Barzilay, 2006], [Eisenstein and Barzilay, 2008], [Misra et al., 2011]). This, however, is in a certain conflict with almost any linguistically-grounded theory of discourse structure ([Mann and Thompson, 1987], [Marcu, 2000], [Kamp and Reyle, 1993], [Wolf and Gibson, 2006]) – all of which suggest either multiple levels or multiple types of structures within texts.

Trivially speaking, any single-level text segmentation algorithm can be turned into a hierarchical one by simply applying it iteratively: first, the top-level segments are identi-

fied; then, each such segment is examined again and its-subsegments are identified. The process can be repeated as many times as necessary (or according to the assumptions about a pre-determined number of levels in a tree). However, such an approach has a serious shortcoming: at each level the automatic segmenter is effectively forced to make decisions without any knowledge about discourse structure higher or lower in the tree. As a result, while it is possible to guarantee optimality within levels, the global optimality is probably sacrificed (or at least cannot be guaranteed).

Research on non-iterative algorithms for hierarchical text segmentation is somewhat scarce. Yaari [1997] applied agglomerative clustering in order to produce hierarchical segmentations. Eisenstein [2009] describes a global hierarchical segmentation algorithm and evaluates it using a medical textbook. The data consists of 12 high-level book parts segmented into chapters and sections (thus corresponding to two-level trees). When evaluating the segmenter, Eisenstein reports *windowDiff* and *Pk* at each of the two levels separately (see Section 3.1.4 for definitions and formulae for these measures). Song et al. [2011] develop an algorithm for hierarchical segmentation that iteratively splits a document in two at a place where cohesion links are weakest. As a second pass, the system transforms a deep binary tree into a shallow and broad structures.

2.5 Applications of models of structure

So far, we have looked at modelling the structure of novels from three different angles. However, it is important to also discuss why would one be interested in building such a model and how it can be useful computationally.

Considering the multitude of possibilities for modelling structure, its applications have been somewhat under-explored, perhaps due to the fact that many of the systems are not very precise.

This is especially true of the narratological models. One of the popular applications of the models of narratives is story generation. Planning-based approaches such as IPOCL and Indexter [Riedl and Young, 2010, Cardona-Rivera et al., 2012] (Section 2.2.1) have been used to generate believable stories. This is very important in game development, as well as in teaching applications.

A more recent application of the models of narrative structure is finding similar stories.

What elements of stories make them similar, what is a useful representation for finding commonalities in stories? Story Intention Graphs [Elson, 2012] were proposed with finding analogies between stories in mind (among several other objectives). Michael [2012] proposes to consider both the author's intentions and the readers's search for commonalities, and outlines a computational model for that. Krakauer and Winston [2012] use patterns consisting of multiple events to find similarities across stories.

Some models of discourse structure have also been used for synchronic and diachronic studies of literary trends using corpora. Elson [2012] uses networks of characters to disprove two generally accepted literary theories. Other studies have tried to use similar corpus-based methods to analysis large corpora of historical novels (e.g., [Moretti, 2011, Heuser and Le-Khac, 2012]).

Computational models of discourse structure have been applied even more extensively. Text summarization has been one of the more popular applications (e.g. [Teufel and Moens, 2002, Das and Srihari, 2009, Darling and Song, 2011]). This is especially true about probabilistic topic modelling (e.g., [Haghighi and Vanderwende, 2009]). Hazen [2011], Wang and Cardie [2012]) use topic modelling to create decision summaries from a corpus of meeting transcripts.

Many have remarked upon the inter-relatedness between discourse structure of a document and possible anaphoric relations in it [Kehler, 2002, Jasinskaja et al., 2007]. Cristea et al. [1998] propose the *Veins Theory*, an extension of RST that describes a set of constraints between discourse structure of a document and domains of accessibility of referential expressions in it. According to this theory, relations of coreference in document depend not so much on the distance between the reference and its antecedent, as on the structural relation between discourse units where they are found. The "veins" in discourse are domains of accessibility for referring expressions. For examples, a leaf in an RST tree may refer to entities mentioned higher up in the tree if they are found in one of the nuclei nodes on the path between the leaf in question and the root of the tree.

Rotaru and Litman [2006] use discourse structure to predict how well students learn using automatic tutoring system. They use a corpus of 95 dialogues annotated in line with the theory of intentions [Grosz and Sidner, 1986]. They study how well labels of the discourse structure nodes correlate with student learning; they also look at the types of trajectories in the hierarchical model of the dialogue that are associated with better learning.

The results suggest that discourse structure is not useful in and of itself. However, when the knowledge about discourse structure is used as a context for other information, the results are highly predictive of the learning rate, as well as of the performance of the system.

Compared to other models of discourse structure, text segmentation has achieved encouraging performance and has been used in a number of applications. Speech processing is one of such applications, since browsing through a series of topical segments is easier for the user then scrolling through a long unstructured transcript [Purver, 2011]. For example, the MIT Lecture Browser aims to make lectures available to students for browsing and searching [Glass et al., 2007]. Text segmentation has also been extensively applied in the processing of meeting transcripts (e.g., [Galley et al., 2003, Janin et al., 2003, Gruenstein et al., 2005, Banerjee and Rudnicky, 2007]). In genres other than the speech, segmentation of news broadcasts has been thoroughly explored, especially throughout the Topic Detection and Tracking initiative (TDT) [Allan et al., 1998].

2.6 Conclusions

In this chapter, we have reviewed several ways to model structure of literary narratives and positioned our model of choice in their context.

The models of narratives proposed by philosophers, narratologists and literary critics are powerful, expressive and abstract. However, it is not yet feasible to build such models automatically.

The models proposed by computational narratologists are can be built computationally, yet we decided not to rely on the existing ones because of the scalability issues and because of how expensive it would be to create a benchmark corpus.

We decided not to use generic computational models of discourse structure because most of them have been developed for short expository texts and not for long literary narratives. Additionally, we would still have the issue of creating a corpus of very detailed annotations.

We have decided to model a novel as a tree of topical segments. This is far from being the most detailed and expressive model, yet it can be built automatically and we hope to show that it can still be useful for other NLP tasks.

Chapter 3

Corpora for topical segmentation of literature

The objective of this work is to automatically model the structure of literary texts as topical trees. In order to guide our efforts and to gauge our progress it would be desirable to have examples of how people may go about creating such structures.

There are several existing benchmark datasets for evaluating automatic topical segmenters (on the task of flat segmentation). The examples include a synthetic corpus compiled by Choi [2000], the newswire dataset used by Misra et al. [2011], a corpus of manually segmented lectures [Malioutov and Barzilay, 2006], meeting transcripts [Gruenstein et al., 2005, Janin et al., 2003], medical textbooks [Eisenstein and Barzilay, 2008] and a number of others. However, none of these datasets is suitable to validate the research project described here: literature is quite a unique genre and to the best of our knowledge there are no datasets of literary data annotated for topical shifts.[1]

Literature is a genre with characteristics quite different from newspapers, textbooks and even speech transcripts. There are many differences, of course, but for the purposes of text segmentation the most important one is the relatively low rate of repeating vocabulary. When writing fiction, it is generally bad practice to repeat the same word over and over.[2]

[1] A notable exception is the dataset developed by Brooke et al. [2012]. It consists of T.S. Elliot's *The Wasteland* manually segmented by a large group of undergraduate students. However, in this work we concentrate on prose, not poetry. Besides, the dataset is not yet publicly available.

[2] This practice is more acceptable in technical texts because explicit repetition of vocabulary facilitates the reading process of otherwise difficult text. We discuss this distinction in more detail in Chapter 6.

However, vocabulary repetition is precisely what almost all topical segmenters rely on to identify topic shifts. This means that even if our segmenter performed well on one of the aforementioned datasets, we have no reasons to expect that the performance would generalize. Therefore, we needed to create a new benchmark dataset for topical segmentation of literature.

Since the task has not been explored for this data type, we needed to verify that it makes sense to ask people to perform it (i.e, that an untrained annotator can perform topical segmentation of literature with reasonable reliability). We also wanted to create a dataset that would be extensive enough to study this process in humans – that is to examine the levels of inter-annotator agreement, to examine the patterns of disagreements and to see what difficulties the annotators had.

This need is even more dire for hierarchical segmentations. While there is a fair number of single-level corpora of topical segmentations in the public domain, only very few are available for hierarchical segmentation. Gruenstein et al. [2005] describe a corpus of 2-level annotations of meeting transcripts (which, unfortunately, is not free). It is more frequent, however, to use metadata to evaluate hierarchical segmenters. Eisenstein [2009] used section and subsection annotations in medical textbooks to evaluate a hierarchical segmenter. Carroll [2010] collected a set of 66 Wikipedia articles and used the structure of the articles to create a corpus of hierarchical segmentations. However, not only are these datasets for different genres of data, most of them capture only one possible view of the topical structure of the document - that of the author. It is well known that the annotators tend to disagree significantly about the granularity of segmentations and about the exact boundary positions. That is why it is important to have more than one annotation per document.

For all these reasons we decided to ask people to perform flat and hierarchical segmentation of a part of the novel *The Moonstone* by Wilkie Collins.

The annotators were asked to perform two distinct tasks. In the first task, they had to read chapters of the novel and to mark the places where the topic shifts perceptibly, briefly summarizing each identified segment. The result of this first experiment is a corpus of single-level segmentations and corresponding outlines of each chapter. In the second task, we made the process hierarchical. The subjects had to find the most perceptible topical shifts, then for each identified segment they had to repeat the procedure until they reached

the level of individual paragraphs. Once again, each segment and sub-segment had to be briefly summarized, effectively creating a hierarchical outline of each chapter. Each chapter was annotated by 3-6 people, resulting in two high quality corpora of flat and hierarchical segmentations.

These corpora are suitable for evaluating flat and hierarchical segmenters and, perhaps even more importantly, for studying how people perform topical segmentation. Because multiple annotations and outlines are available for each chapter, it is possible to study agreement and disagreement patterns as well as use them for developing better metrics for evaluating topical segmenters.

This chapter is structured as follows. Section 3.1 described the dataset for single-level topical segmentation created in the course of this work (further *the flat Moonstone dataset*). In Subsection 3.1.3 we describe the analysis of the corpus, the patterns of agreement and disagreement between the annotators. Section 3.2 describes the second, hierarchical dataset of topical segmentations of literature. We analyze this dataset in Subsection 3.2.3.

3.1 The Moonstone dataset 1: flat segmentation

3.1.1 Overview

To study how people identify topical shifts in written literary texts, we asked 27 annotators to segment 20 chapters of the novel *The Moonstone* by Wilkie Collins. Each chapter was annotated by 4-6 people. The results of our analysis indicate that untrained annotators can agree reasonably well on how to segment literature. However, we noticed that the agreement is not uniform, even within individual chapters. People agree well on a subset of topical boundaries (about 50% of topical boundaries are marked by at least half of all annotators working on that chapter). However, the agreement on the remaining topical boundaries is quite low – they are marked by a minority of annotators or only by a single person. These findings suggest that when evaluating the performance of automatic segmenters, it is important to consider not only the overall similarity between human and machine segmentations, but also to examine the regions of disagreement. When a program misses or misplaces a prominent topic shift – a segment boundary marked by all annotators – it should be penalized more than if it was mistaken about a boundary marked by one per-

son. Similarly, a false positive in the region where none of the annotators found a change in topic is worse than a boundary inserted in a place where at least one person perceived a topic change.

Sections 3.1.2-3.1.6 describe the dataset and patterns of disagreement in more detail.

3.1.2 Corpus description

In order to study how people find topic fluctuations in literature, we chose a XIX century novel, *The Moonstone* by Wilkie Collins, as the text to be annotated. We used two chapters for a pilot study and then another 20 for the large-scale experiment.[3] The annotators worked with individual chapters and were required to align segment boundaries with paragraph breaks. Compiling a corpus consisting of chapters of different works would provide a more stereoscopic picture of how people segment literature in general. However, one of the application for which we would like to use our segmenter is automatic summarization. That is why we chose consecutive chapters of a single work.

Objectives. The main question behind this study was this: "How do people identify topical shifts in literature?" This vague question can be mapped to several more specific objectives. First, we sought to verify that topical segmentation of literature was a sensible task from the viewpoint of an untrained annotator. Next, it was important to examine inter-annotator agreement to make sure that the annotators in fact worked on the same phenomena and that the resulting corpus is a reasonable approximation of how people segment literature in general. Third, in addition to analyzing the overall agreement we also took a close look at the type of common disagreements, in search of patterns and insights to evaluate automatic segmenters.

These were the formulated questions, but of course the most interesting observations occur for questions that have not been asked. Therefore, we were also on the lookout for surprises.

Subjects. The participants were undergraduate university students of English at the University of Ottawa. They were recruited by email and received 50 dollars each for their participation. Everyone had to annotate four chapters from *The Moonstone*, not necessar-

[3]The pilot study tested the overall procedure and the annotation guidelines, and estimated expected agreement and hence the desired number of annotations per chapter. The pilot was successful; the study described in this chapter closely resembles the original experiment design, apart from several small details.

ily consecutive ones. The chapters were divided so as to ensure an approximately equal workload.

We had planned 6 independent annotations for each chapter of the novel.[4] The annotators were divided into five groups and each group was asked to read and annotate four distinct chapters. In the end we had three groups with six people, one group with five and one group with four.

Procedure. The experiment was conducted remotely. The students received by email packages with detailed instructions and an example of a segmented chapter from a different novel. They had two weeks to annotate the first two chapters and then two more weeks to annotate another two chapters.

The annotators were instructed to read each chapter and split it into episodes – topically continuous spans of text demarcated by the most perceptible shifts of topic in the chapter. We asked the annotators to provide a brief one-sentence description of each episode, effectively creating a chapter outline. The students were also asked to record places they found challenging and to note the time it takes to complete the task. The instructions that the annotators received can be found in Appendix B. The actual dataset is publicly available under GNU license at `http://www.site.uottawa.ca/~ankazant/Annas_page/Downloads.html`.

Because even short chapters of most traditional novels are rather lengthy, we chose to use paragraphs as the basic units of annotation (sentences are more common in text segmentation literature).

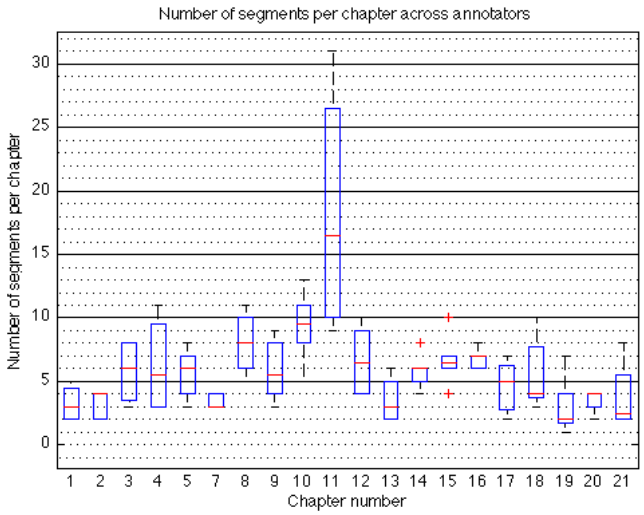
3.1.3 Corpus analysis

Time. On average, an annotator required 137.9 minutes to complete both tasks. The standard deviation was σ = 98.32 minutes appropriately reflecting the fact that some students are very fast readers and besides had already read the novel in one of their classes, while others are quite slow.

The average chapter length is 53.85 paragraphs (σ = 29.31), the average segment length across all annotators is 9.25 paragraphs (σ = 9.77). On average the annotators identified 5.80 episodes (σ = 2.45) per chapter. Figure 3.1 shows the distribution of the number of

[4] We hired 30 students. Three did not complete the task.

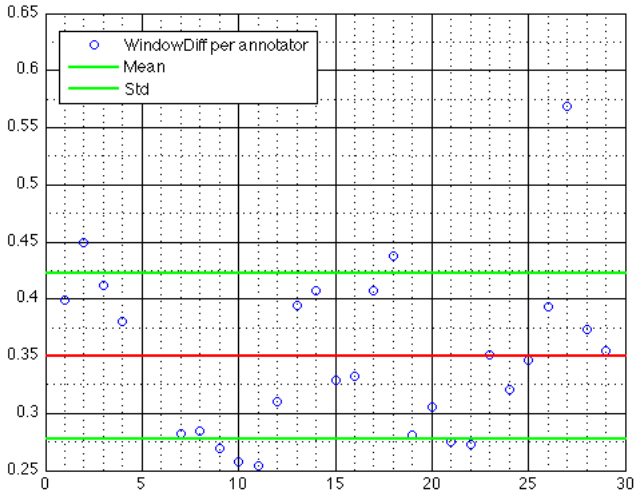
Figure 3.1: Distribution of segment counts across chapters.



segments identified in each chapter. An individual box plot is compiled using all available annotations for the that chapter – six for most, four or five for several. Since the data are plotted for individual chapters, the only source of variance is the disagreement between annotators as to what is the appropriate level of detail for the task. Figure 3.1 confirms the findings by other researchers that people find topical shifts at different levels of granularity (e.g., Malioutov and Barzilay [2006], Gruenstein et al. [2005]).

In order to see if we had 'rogue' annotators, we compared how much each person agreed with the others in her group. To this end, we computed pair-wise *windowDiff* for each possible pair of annotators and averaged them (see Section 3.1.4 for the definition and discussion of *windowDiff*). Figure 3.2 shows the results. Evidently, annotators 2 and 27 have the highest *windowDiff* values corresponding to the lowest agreement with their group. However, looking at the actual annotations made us decide against discarding them. These two people were simply very thorough and since we also have their outlines, we could verify the quality of their segmentations. Annotator 27 spent the longest amount of time on the task of the whole cohort (8.5 hours). An example segmentation for annotator 2 is available in Figure 3.3 in Section 3.1.5. She too is the most detailed of the group. In the end we preserved all the annotations and included them all in our analysis.

Figure 3.2: Annotator quality.



3.1.4 Inter-annotator agreement

In order to make sure that our guidelines are sufficiently clear and the annotators in fact annotate the same phenomenon, it is important to measure inter-annotator agreement [Artstein and Poesio, 2008]. This is particularly important given the fact that the resulting corpus is intended as a benchmark dataset for evaluation of automatic segmenters.

When looking at inter-annotator agreement independently of the domain, the most commonly used metrics are coefficients of agreement – α [Krippendorff, 2004], κ [Cohen, 1960, Shrout and Fleiss, 1979], π [Scott, 1955] and several others. In this work, we use a multi-annotator version of π, also known in the CL community as Fleiss's κ [Shrout and Fleiss, 1979, Siegel and Castellan, 1988] .

Fleiss's κ is computed as follows:

$$\kappa = \frac{Agreement_{observed} - Agreement_{expected}}{1 - Agreement_{expected}} \tag{3.1}$$

$$Agreement_{observed} = \frac{1}{ic(c-1)} \sum_{i \in I} \sum_{k \in K} n_{ik}(n_{ik} - 1) \tag{3.2}$$

$$Agreement_{expected} = \frac{1}{(ic)^2} \sum_{k \in K} n_k^2 \tag{3.3}$$

where i is the number of items to be classified in set I, k is the number of available cate-

gories in set *K*, *c* is the number of annotators, n_{ik} is the number of annotators who assign item *i* to category *k*, n_k is the total number of items assigned to category *k* by all annotators [Artstein and Poesio, 2008, pp. 562-563]. Effectively κ measures how much the annotators agree above what can be expected by chance. The value of κ is 0 where there is no agreement above chance and 1 where the annotators agree completely.

While we report κ values for our dataset, it is important to note that κ is ill-suited to measuring agreement in segmentation. The main problem is its insensitivity to near-hits. When asked to segment a document, the annotators often disagree about the exact placement of the boundary, but agree that there is a boundary somewhere in the region (e.g., consider paragraphs 9-11 in segmentations in Figure 3.3). It is desirable to give partial credit to such near-hits instead of dismissing them as utter disagreement. This cannot be achieved with κ. The second problem is the independence assumption: the label for each item must be independent from the labels of all other items. In our case, this would amount to claiming, highly unrealistically, that the probability of a topical shift between two sentences is independent of the topical landscape of the rest of the document.

Two other commonly used agreement metrics are *Pk* [Beeferman et al., 1999] and *windowDiff* [Pevzner and Hearst, 2002], both designed to compare a hypothetical segmentation to a reference, not to measure agreement *per se*. A common feature of both metrics is that they award partial credit to near-hits by sliding a fixed-length window through the sequence and comparing the reference segmentation and hypothetical segmentation at each window position. The window size is generally set at half the average segment length.

Pk (Equation 3.4) measures the probability that two units randomly drawn from a document are correctly classified as belonging to the same topical segment. *Pk* has been criticized for penalizing false negatives less than false positives and for being altogether insensitive to certain types of error; see [Pevzner and Hearst, 2002, pp. 22-26] for details. Despite its shortcomings, *Pk* is widely used. We report it for comparison with other corpora.

$$Pk(ref, hyp) = \sum_{1 \leq i \leq j \leq n} D(i,j)(\delta_{ref}(i,j) \ XNOR \ \delta_{hyp}(i,j)) \tag{3.4}$$

Functions δ_{hyp} and δ_{ref} indicate whether the two segment endpoints *i* and *j* belong to the same segment in the hypothetical segmentation and reference segmentation respectively.

windowDiff was designed to remedy some of *Pk*'s shortcomings. It counts erroneous windows in the hypothetical sequence normalized by the total number of windows. A window is judged erroneous if the boundary counts in the reference segmentation and hypothetical segmentation differ; that is (|ref – hyp| $\neq$ 0) in Equation 3.5).

$$winDiff = \frac{1}{N-k} \sum_{i=1}^{N-k} (|ref - hyp| \neq 0) \tag{3.5}$$

Both *Pk* and *windowDiff* produce penalty scores between 0 and 1, with 1 corresponding to all windows being in error, and 0 – to a perfect segmentation.

Recently, Fournier and Inkpen [2012] proposed a new measure for measuring similarity between two segmentations called the *S* metric. Unlike *windowDiff* and *Pk*, *S* is an edit-distance metric and thus it does not require sliding a window through candidate segmentations. Effectively, given two segmentations bs_a and bs_b, *S* measures how many edit operations would be needed to turn a hypothetical segmentation bs_a into a reference one (bs_b), taking into account the total length of the document D in question. The allowed types of editing operations are substitutions (i.e., adding a missing boundary or removing an extra one) and transpositions (a separate operation to align near-hits, that is boundaries misaligned by only a short distance which is set as a parameter). The formula for the *S* metric is as follows:

$$S(bs_a, bs_b, n) = \frac{1 - |boundary\_distance(bs_a, bs_b, n)|}{pb(D)} \tag{3.6}$$

Here $boundary\_distance(bs_a, bs_b, n)$ is the total number of edit operations needed to turn a segmentation bs_a into bs_b, n is the threshold defining the maximum distance of transpositions. $pb(D)$ is the maximum possible number of edits.

S has preserves the desirable qualities of *windowDiff* and *Pk* in that it allows giving partial credit to near-hits. Because it does not require a sliding window, S is more robust: the size of the sliding window in *windowDiff* and *Pk* is of critical importance, even small variations in size lead to relatively large differences in the final values. Additionally, it has been shown that *windowDiff* has a number of undesirable characteristics, including favouring false negatives over near-hits in certain circumstances and not increasing monotonically as a function of undesirable errors [Scaiano and Inkpen, 2012]. Additionally, *S* provides a

Table 3.1: Overview of inter-annotator agreement.

	Mean	Std. dev.
κ	0.29	0.15
Pk	0.33	0.17
windowDiff	0.38	0.09
π_S	0.89	0.07

new flexibility with respect to allowing more than one boundary type – the fact that comes extremely handy when analysing the hierarchical dataset in Section 3.2.1.

Additionally, Fournier and Inkpen propose several versions of the agreement coefficients that use *S* as their basis. The coefficient called π_S corresponds to Fleiss's κ in terms of the assumptions made about distributions. π_S combines the strengths of κ and *windowDiff* while remedying their shortcomings. As an agreement coefficient, it allows measuring true agreement – the agreement that is above what can be expected by chance. On the other hand, it allows taking into account near-hits and near misses when computing the agreement:

In this dissertation we use Fournier's version of the π coefficient and refer to it as π_S.

Table 3.1 reports *Pk*, *windowDiff*, κ and π_s values for our corpus. *Pk* and *windowDiff* are computed pairwise for all annotators within one group and then averaged. We set the window size to half the average segment length as measured across all annotators who worked on a given chapter. The values are computed for each group separately; Table 3.1 shows the averages across five groups.

Even by most relaxed standards, e.g., [Landis and Koch, 1977], the κ value of 0.29 corresponds to low agreement. This is not surprising, since it only includes the cases when the annotators agree exactly where the boundary should be. For the purpose of our task, such a definition is too strict.

The values of *windowDiff* and *Pk* are more reasonable; *windowDiff* = 0.38 means that on average a pair of annotators disagrees on 38% of windows. *windowDiff* was originally designed to compare only two segmentations. Our strategy of computing its values pairwise is perhaps not optimal, but in the absence of another metric allowing to account for near-hits we are practically forced to use it as a primary means of inter-annotator agreement.

The π_S value of 0.89 is considered to be reliable on most scales for computing agree-

ment coefficients. It is computed taking into account near-hits and misses and is more appropriate for evaluating segmentations. However, since the scales such as described by [Landis and Koch, 1977] or [Krippendorff, 2004] were suggested for regular versions of κ and π and since there are no other segmentation corpora for which these values have been reported, it is difficult to put this seemingly good value in appropriate context.

3.1.5 Patterns of disagreement

Figure 3.3: Example segmentation for Chapter 1.

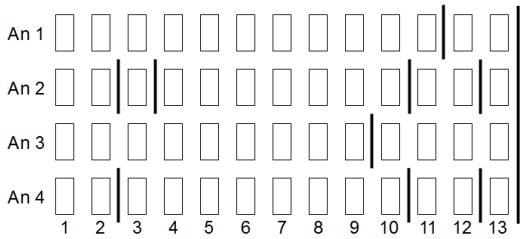
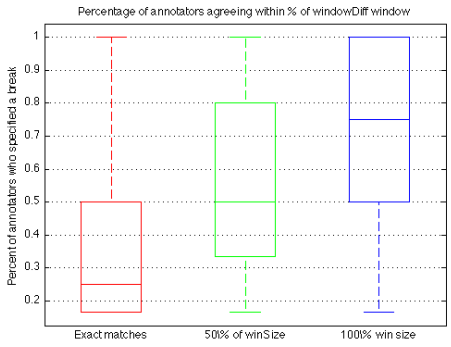


Figure 3.3 shows the segmentation of the shortest chapter in the dataset. The overall agreement is quite low (*windowDiff*=0.38, $\kappa = 0.28$). This is not surprising, since annotators 1 and 3 found two segments, annotator 2 – five segments, and annotator 4 – four. Yet all annotators agree on certain things: everyone found that there was a significant change of topic between paragraphs 9 and 11 (though they disagree on its exact placement). It is therefore likely that the topical shift between paragraphs 9 and 11 is quite prominent. Annotators 2 and 4 chose to place a segment boundary after paragraph 2, while annotators 1 and 3 did not place one there. It is likely that the topical shift occurring there is less prominent, although perceptible. According to these annotations, the least perceptible topic shifts in the chapter occur after paragraph 4 (marked only by annotator 2) and possibly after paragraph 11 (marked only by annotator 1). Overall, glancing at these segmentations suggests that there is a prominent topical shift between paragraphs 9-11, three significant ones (after 2, 10 and 12) and several minor fluctuations (after 3 and possibly after 10 and 11).

Looking at the segmentations in Figure 3.3 it seems likely that the disagreements between annotators 2 and 4 are due to granularity, while the annotators 1 and 3 disagree more fundamentally on where the topic changes. When measuring agreement, we would like to be able to distinguish between disagreements due to granularity and disagreements due to

Figure 3.4: Quality of segment boundaries.



true lack of agreement (annotator 1 and 3). We would also like to leverage this information for the evaluation of automatic segmenters.

Distinguishing between true disagreement and different granularity while taking into account near-hits is not trivial, especially since we are working with multiple annotations simultaneously and there is no *one* correct segmentation.

In order to estimate the quality of individual boundaries and look inside the segmented sequence, we approximate the quality of each suggested segment boundary by the percentage of annotators who marked it. Since the annotators may disagree on the exact placement of the boundaries, our measurement must be relaxed to allow for near-hits.

Figure 3.4 shows the distribution of segment boundaries using three different standards of quality. We consider all segment boundaries introduced by at least one annotator. Then, for each suggested boundary, we compute how much support there is from peer annotators: what percentage of annotators included this boundary in their segmentation. The leftmost box plot in Figure 3.4 corresponds to the most strict standard. When computing support, we only consider perfect matches: segment boundaries specified in exactly the same location (window size = 0). The middle box plot is more relaxed: we consider boundaries found within half of a *windowDiff* window size of the boundary under inspection. The rightmost box plot corresponds to the inclusion of boundaries found within a full *windowDiff* window size of the boundary under inspection.

Looking at exact matches (the leftmost box plot), we observe that at least a half of segment boundaries were specified by less than 25% of annotators (which corresponds to

one person). This explains why κ values in Table 3.1 are so low: this is the only sort of agreement κ captures. Also one can notice that at most 25% of the boundaries have the support of more than 50% of the annotators.

The picture changes if we consider all boundaries within a tight window around the candidate boundary (the middle box plot). This standard is twice as strict as the regular *windowDiff* evaluation. Here 50% of all boundaries are marked by at least 35% at and most 80% of annotators. Only 25% of boundaries are marked by less than 30% of the annotators.

The rightmost plot looks even better. If we consider the support found within a window size of any candidate boundary, then 50% of all boundaries are supported by over 70% of annotators. However, we find this way of measuring support too optimistic. The reason is, again, the difference in the granularity of segmentations. The window size used for these measurements is based on the average segment length across all annotations. For example, the average segment length for segmentation shown in Figure 3.3 is 4, making the window size 2. This size is too relaxed for annotators 2 and 3, who were very detailed. Due to the excessively large window there will almost always be a boundary where fine-grained annotations are concerned, but those boundaries will not correspond to the same phenomena. That is why we think that a stricter standard is generally more appropriate. This is especially the case since we are working with paragraphs, not sentences. A distance of 2-3 sentences is quite tolerable, but a distance of 2-3 paragraphs is considerable, and it is far more likely that a stricter notion of near-hits needs to be considered.

3.1.6 Flat moonstone dataset: conclusions

Overall, the flat Moonstone dataset appears to have enough integrity to be used as a benchmark for segmentation of literature. The inter-annotator agreement is quite reasonable given the subjective nature of the task. However, in the light of findings described in this section, we argue that it is very important to include all available annotations when evaluating topical segments (or to consider the support for each potential boundary position as reflected in the annotations). If we consider Figure 3.3, if a segmenter misses a boundary after paragraph 2, for example, it should be penalized more than if it misses a boundary after sentence 9 (since the former was included by 2 annotators and the latter by one).

Conversely, if an automatic segmenter includes a boundary after paragraph 6, it should be penalized more than if it includes a boundary after paragraph 9 (the former boundary was not included by any of the 4 annotators, while the later was included by one person, making it a more feasible hypothesis). In-depth research on evaluating topical segmenters is beyond the scope of this research project. However, in [Kazantseva and Szpakowicz, 2012] we propose a simple modification to *windowDiff* that improves the reliability of evaluation by considering all available annotations. Alternatively, the S metric of Fournier and Inkpen [2012] can be configured to give weights to the boundary positions according to the support from the human annotators.

3.2 The Moonstone dataset 2: a dataset for hierarchical segmentation

3.2.1 Overview of the hierarchical dataset

One of the research questions addressed in this dissertation is hierarchical segmentation of literature – the task of building a topic-based tree for each chapter/work. Consequently, it is necessary to evaluate the performance of our hierarchical segmenter by comparing its performance with segmentations produced by people. This subsection describes a dataset that we have created for this purpose.

We have already mentioned in the previous section that literature is a genre with unique characteristics and for that reason it is unrealistic to expect that the performance of automatic segmenters tested on other datasets would generalize to literary data. This is why we decided to create our own dataset. Our objectives when creating this dataset were as follows. Firstly, we wanted to study how people performed when asked to identify hierarchical topical structure in literary data. We wanted to study whether it was a reasonable task, whether it was difficult or easy for them and whether they agreed between themselves. Secondly, given the general scarcity of datasets for evaluating hierarchical segmenters, we wanted to build a benchmark dataset and make it available to other researchers.

In order to study how people perform hierarchical segmentation of literature, we chose use a subset of the data used in the flat segmentation study – 9 chapters from the novel *The*

Moonstone.[5] This way, we eventually obtain a dataset that can be used for studying flat or hierarchical segmentations, parallels and differences between them or for studying how segmentations can be applied to higher-level tasks, such as automatic summarization.

In a nutshell, the results of this experiment are slightly less conclusive than those of the previous one. Firstly, the annotators have found the task considerably more difficult than the previous one – this is reflected both in the average time taken to complete the experiment and also in the annotators' notes. The inter-annotator agreement is lower than in the flat dataset but still quite reliable. Interestingly, it seems that the most common type of disagreement between the annotators is not so much whether there is a topic shift or not at a given position, but at what level it occurs. In a certain way, this echoes the findings from the previous experiment about the fact that people segment at different levels of granularity. This may suggest that ultimately hierarchical segmentation is a more appropriate way of modelling topic structure. However, it also becomes clear that comparing hierarchical segmentations and evaluating hierarchical segmenters is considerably more challenging.

The next subsesction (3.2.2) describes in detail how the corpus was constructed. Subsection 3.2.3 provides the analysis of the data and discusses the inter-annotator agreement. Section 3.2.4 concludes with high-level interpretation and discussion.

3.2.2 Hierarchical moonstone dataset: corpus overview

We chose nine previously annotated chapters of *The Moonstone* to be used in the hierarchical segmentation experiment (Chapters 1-2 and 4-10). The choice of the chapters was more restricted than in the previous study. Hierarchical segmentation is considerably more time-consuming than single-level segmentation, therefore there was a very real upper limit on the ultimate length of the chapter.

Objectives. The main objective of this study was studying how people segment literature hierarchically. We wanted to examine the difficulty of the task, inter-annotator agreement and disagreement patterns between the annotators.

Subjects. The subjects participating in the experiment were undergraduate students of the English Department at the University of Ottawa. They were recruited by e-mail and each received 50 dollars compensation for their participation. We aimed to hire 30 students

[5] All nine chapters had already been annotated in the previous study.

Figure 3.5: An example of hierarchical segmentation for Chapter 2. (Produced by Annotator 1).

1. Paragraph 1: My Lady and the diamond
2. Paragraphs 2-5 : Gabriel Betteredges service
   (a) Paragraph 2: The Herncastles
   (b) Paragraph 3-4: The relationship between Gabriel and his Lady Miss Julia
      i. Paragraph 3: The page boy
      ii. Paragraph 4: Moving up in positions
   (c) Paragraph 5. Bailiff
3. Paragraphs 6-11: Selina and Gabriel
   (a) Paragraph 6: Picking a wife
   (b) Paragraphs 7-8: Transition paragraphs
   (c) Paragraphs 9-11: Gabriel and Selina, husband and wife
      i. Paragraph 9: Thoughts of breaking off the engagement
      ii. Paragraph 10: After the wedding
      iii. Paragraph 11. Five years of marriage and two deaths
4. Paragraphs 12-14: Life goes on
   (a) Paragraphs 12-13: The wool waist coat
      i. Paragraph 12: Fifty years of service to his Lady
      ii. Paragraph 13: Getting old
   (b) Paragraph 14: Accepting age and the less demanding job as house steward
5. Paragraph 15: Penelope making sure Gabriel stays on track and tells the story of the diamond

but eventually we received only 23 annotations. The cohort contained several participants from the first experiment, but mostly the students were new.

Procedure. The students were divided into five groups and each group had to annotate two chapters. Group 5 was an exception: since the chapter assigned to them was very long, they annotated a single chapter (in two steps with a check-point in the middle).

The experiment was conducted by e-mail. Initially each group had received detailed instructions and an example annotation as well as the first chapter to work on. They were required to submit their annotation within a week. In the second round of the experiment, the students have received the second chapter to work on. Again, they had a week to complete the task.

The instructions asked that the annotator read the chapter and split it into top-level segments – according to where there is a perceptible shift of topic. Similarly to the previous experiment, the annotator had to provide a one-sentence description of what the segment is about. The annotators had to work with paragraphs as the individual units.

Then the task took on a recursive nature: the annotators had to examine each of the top-level segments and split them into sub-segments, providing a brief description of each. The annotators were asked to repeat this procedure recursively until they reached the level of individual paragraphs. However, we allowed them to stop at a higher level if they found it excessively difficult, asking them to record the difficulties.

Effectively, the annotators were building a detailed hierarchical outline for each chapter. An example annotation is shown in Figure 3.5. This outline was created by an annotator for Chapter 2 of *The Moonstone* (see Appendix A for full text).

Metrics: Analyzing and evaluating the quality of the dataset for hierarchical segmentation is considerably more complicated than for flat segmentation. When comparing two single-level segmentations, one needs to take into account extra/missing boundaries and also near hits and misses. If more than two annotations are available, one needs to account for the whole pool. However, when the annotation is hierarchical, in addition to all of the above, one also needs to consider multiple layers of the tree. In the end, when estimating the reliability of a pool of hierarchical segmentations, we need to take into account similarities and differences (including near misses) within and across layers and also across all available annotations.

Traditionally, hierarchical segmenters have been evaluated either by computing *win-*

Figure 3.6: Example segmentation for Chapter 2. Annotator 2.

1. Paragraphs 1-4: the single life of the narrator (a note: paragraph one sets the time the story begins)
    (a) paragraph 1: the diamonds location
    (b) paragraphs 2-4: how he came to be in his career
        i. paragraph 2: how he met the lady
        ii. paragraph 3: how he became hired by the lady
        iii. how he came to be promoted to bailiff
2. paragraphs 5-10: life with a love interest before and after marriage
    (a) paragraphs 5-8: motivations for finding a wife
        i. paragraph 5: loneliness
        ii. paragraph 6: economics and love
        iii. paragraphs 7-8: the lady's approval
            A. paragraph 7: asking for approval
            B. paragraph 8: receiving approval
    (b) paragraphs 9-10: experiences of marrying and being married to a woman
        i. paragraph 9: apprehensions about marriage
        ii. paragraph 10: mediocrity in married life
3. paragraphs 11-15: life as a widower and into old age
    (a) paragraph 11: death of the narrator's wife and first mention of daughter Penelope
    (b) paragraphs 12-14: convincing the narrator to alter his occupation
        i. paragraph 12: Lady lulls him into security with kindness
        ii. paragraph 13: narrator realises her plan, and his pride fights against it
        iii. paragraph 14: narrator is convinced after reading Crusoe he will feel better about it tomorrow
    (c) paragraph 15: the difficulty of writing a story without accidentally writing about oneself (a note: only technically in his old age, not really about his old age)

dowDiff or *Pk* per level (e.g., [Eisenstein, 2009]) or by flattening the tree (e.g., [Song et al., 2011]). However, both methods have weaknesses. When reporting the results per level, we do not take into account situations when the segmenter failed to specify a boundary at the correct level, but perhaps identified it in an adjacent level. When flattening the tree we effectively lose all information about hierarchical structure and treat the errors at the top level equally with the errors at the fine-grained levels.

Carroll [2010] proposes a measure for evaluating hierarchical segmenters. The measure is a version of *Pk* proposed by Beeferman et al. [1999], adjusted so as to spread the error over all levels in the segmentation tree. It is designed to give higher weight to the top levels of the tree and lower weight to the levels towards the bottom:

$$E_{p_k} = \frac{1}{|R|} \sum_i c_i P_k(R_i, H_i) \tag{3.7}$$

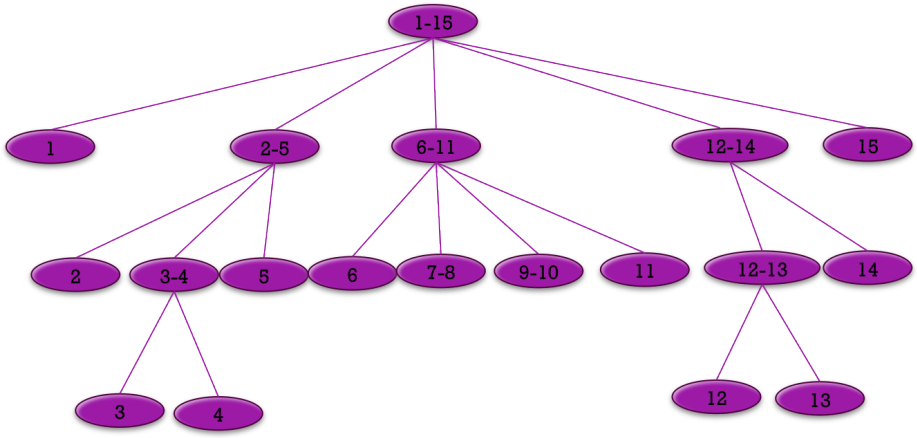
Here *R* is the set of reference boundaries and *H* is the set of hypothetical ones, ordered by their prominence (the level in the tree), c_i is the weighing factor. The following equation must also hold:

$$R_i = \{b_j : b_j \in R \wedge j \leq i\} \tag{3.8}$$

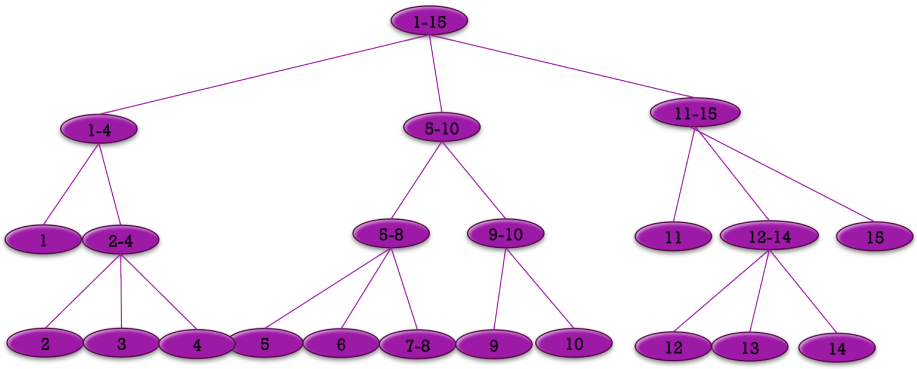
The measure is an improvement over simply applying *windowDiff* or *Pk* at each level, but it has a number of shortcomings. For one thing, it inherits all the problems already found in *Pk* (listed in Section 3.1.4 and also in [Pevzner and Hearst, 2002]). Additionally, it does not take into account similarities across different levels in the segmentation tree, effectively evaluating each level in isolation.

In Section 3.1.4 we briefly listed some of the problems with the window-based metrics (e.g., *windowDiff* and *Pk*). All those issues are just as relevant in hierarchical segmentation as in flat one. For this reason we decided not to use it for analysis. Instead, we decided to use the S metric as our main analytical tool. To be comparable to previous research we also report *windowDiff*. Section 3.2.3 shows the results, but also illustrates why *windowDiff* is not an appropriate metric for hierarchical segmentation analysis.

Recall that the *S* metric can be interpreted as follows: if we were to change a boundary sting bs_a into a boundary string bs_b, what proportion of the boundaries would remain unchanged? In their paper, Fournier and Inkpen [2012] explain in detail how to use the



(a) Chapter 2, annotator 1



(b) Chapter 2, annotator 2

Figure 3.7: Example hierarchical segmentations for Chapter 2.

metic for single-level segmentation. However, they mention that in fact the metric operates not on strings of boundaries, but on strings of sets of boundaries – making it applicable in situation where multiple boundary types are involved, for example in dialogue-act schemas or in hierarchical segmentation.

Figure 3.7 shows two example segmentations for Chapter 2 of *The Moonstone* (full text is available in Appendix A). The textual outlines created by the annotators are shown in Figures 3.5 and 3.6. While the trees are somewhat similar (and especially if we consider the textual outlines), it is difficult to say whether they are in fact similar enough. Worse yet, it is not clear which trees are more similar than the others (since of course we have more than two trees for each chapter).

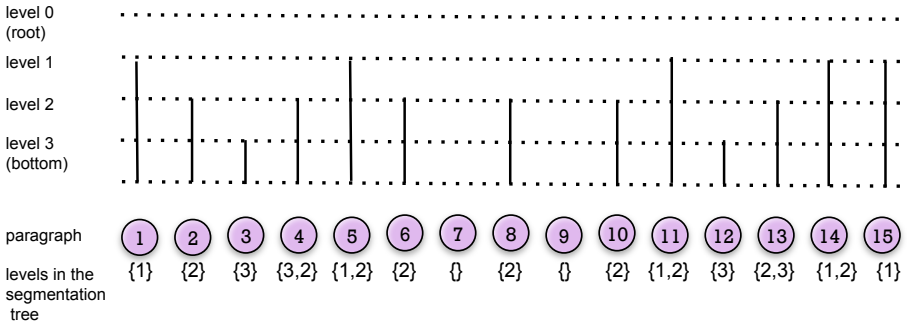
We can notice that often the annotators disagree on the exact placement of the boundary (especially for transition paragraphs) and on where exactly they should be attached. Both annotators indicated a shift of topic after Paragraph 1, but annotator 1 separated it into a top-level segment, while annotator 2 attached it to a second-level segment. (The situation is the same with Paragraph 15. From the annotators' notes and from reading the text in Appendix A, we can see that these are introductory and concluding paragraphs. Additionally, they represent entry and exit points of an embedded narrative, adding to the complexity of the task for the annotators).

Comparing two segmentation trees is not intuitive. However, we need to remember that effectively the branchings in the tree correspond to fluctuations in topic – with branchings near the top of the tree indicating relatively significant fluctuations, while branchings near the leaves correspond to less pronounced fluctuations or minor digressions. Keeping this in mind, we may flatten the tree representation. Figures 3.8a and 3.8b show the corresponding flattened representations.

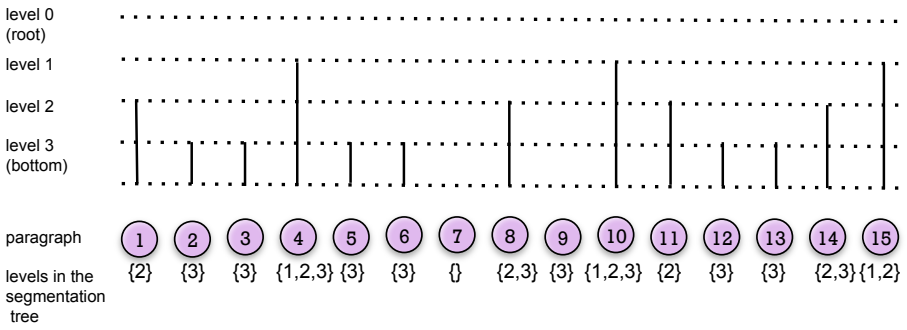
Figure 3.8a shows a flattened representation for the tree shown in Figure 3.7a (and, correspondingly, the representation of the tree from Figure 3.7b is given in Figure 3.8b). In this representation, the vertical lines indicate the presence of a boundary. The length of the vertical lines represents the level at which the boundary was found in the tree – the longer the line, the closer to the root the boundary was (and hence the more pronounced the fluctuation was). The sets of numbers under each line indicate the levels specifically. The *S* metric operates on sets of boundary strings – such as appear in the bottom row. This is how we will compare topical segmentation trees.

Figure 3.8: Flattened representations of topical trees.

(a) Flattened representation of the tree from Figure 3.7a



(b) Flattened representation of the tree from Figure 3.7b



3.2.3 Hierarchical dataset: corpus analysis

Overall, we have collected 23 annotations for the 9 chapters of *The Moonstone*. Table 3.2 shows what groups of annotators worked on what chapters and the number of people per group. Since different groups worked on different chapters, most statistics have been computed per group. Whenever overall averages are given for the whole corpus, they are macro-averages across all groups, unless otherwise indicated.

On average, the annotators took 3.5 hours to complete the task ($std.dev. = 1.6$). Average depth of the tree is 3.00 levels ($std.dev. = 0.65$), suggesting that the annotators prefer shallow but broad structures. Table 3.3 reports average breadth of the tree at different levels. The breadth at Level 4 is lower than that at level 3, indicating that only a small percentage of the entire tree was annotated at a level deeper than 3. We hypothesize that these shallow broad structures are due to the fact that it is difficult for people to create deep recursive structures in their mental representations. However, we do not have any hard data to support this hypothesis.

Many of the annotators specifically commented on the difficulty of the task. Nine out of 23 people included comments ranging from notes about specific places to general comments about their lack of confidence. 4 annotators found several (specific) passages they had trouble with.

One simple way to estimate the quality of the annotations is by computing pair-wise *windowDiff* at each level. Figure 3.9 shows the averages of pairwise *windowDiff* values across all annotations.[6] The values seem to be lowest for the top level (Level 1), indicating the best agreement near the root of the tree. Overall, they rise gradually as we go deeper into the tree, corresponding to worse agreement. However, these values should be taken with caution. In addition to all the shortcomings listed in Section 3.1.4, these values are computed for the whole sequence at each level. This rarely makes much sense except for the top level since the trees are not balanced and not the whole sequence has been segmented at deeper levels (for example in Figure 3.7a only four nodes appear at the bottom level). Due to these concerns, we use the *S* metric as the primary tool for analyzing the hierarchical dataset.

Figure 3.10 reports average pairwise S similarity across all annotators. The values here

[6] *windowDiff* values are reported per annotator here. For each annotator we computed *windowDiff* values against all other annotators in their group. Figure 3.9 reports averages of these values.

Table 3.2: Chapter assignments per groups of annotators.

Group	Chapters	Number of annotators
Group1	ch1, ch4	6
Group 2	ch2, ch6	5
Group 3	ch5, ch7	6
Group 4	ch8, ch9	4
Group 5	ch10	3

are computed pairwise for each possible pair of annotators and then the mean is reported for each annotator. We are not aware of another dataset for hierarchical segmentation with more than one annotator, hence it is somewhat difficult to put these numbers in context. However, we know that *S* ranges from 0 for absolutely dissimilar segmentations to 1 for identical ones. Therefore, the numbers in Figure 3.10 seem to indicate a reasonable level of inter-annotator similarity.

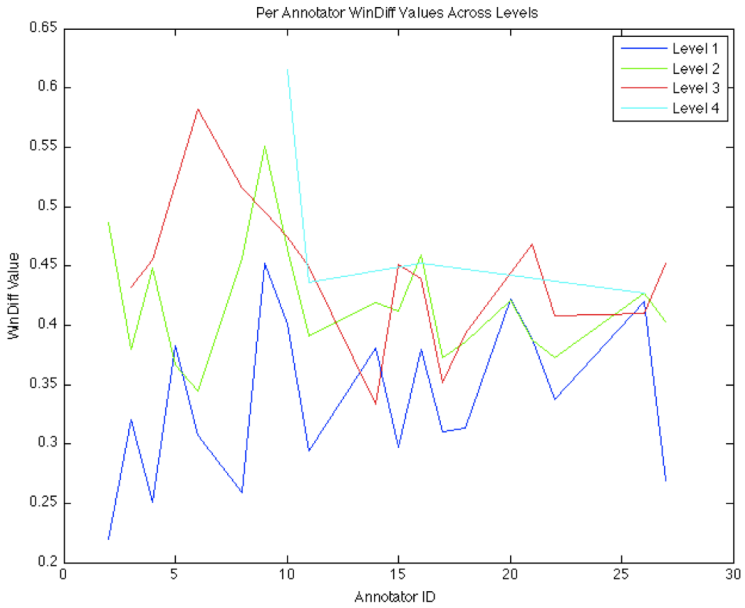
Fournier and Inkpen [2012] also propose versions of the widely used inter-annotator agreement coefficients κ and π that are suitable for text segmentation (i.e., they take into account near hits and near misses). Figure 3.11 shows the values of S-based variants of κ and π (and also, for completeness, the values of pairwise *S*) per chapter. While there is no other hierarchical dataset for which these values have been computed, it is generally thought that agreement above 0.7 is acceptable (see the scale in Landis and Koch [1977]).

Recall that the value of S can be interpreted as 1 - e, normalized by the length of the string, where e is the number of edit operations necessary to turn one boundary string into another. In the hierarchical version of the software, three edit operations are possible: ad-

Table 3.3: Average breadth of the trees at different levels.

Level	Mean breadth	Std. dev.
Level 1	6.53	3.41
Level 2	17.55	7.03
Level 3	17.63	10.61
Level 4	8.8	9.39

Figure 3.9: *windowDiff* values across levels of hierarchical segmentations.



ditions/deletions, transpositions and substitutions.[7] Additions/deletions and transpositions are intra-level edit operations, while substitutions are inter-level operations. Figure 3.12 breaks down the sum total of necessary edit operations involved in computing S. For clarity, these values are normalized by chapter lengths multiplied by the number of annotations per chapter.

From Figure 3.12 it is easy to see that transposition errors are relatively rare, meaning that in our case near hits and misses do not play a critical role. On the other hand, missing/extra boundaries (shown under the legend *additions*) are common. Equally, if not more, common are substitution errors – situations where the annotators disagreed not on whether or not there is a topic change, but on how prominent it is (i.e., at what level in the segmentation tree it occurs). This is an important finding. It shows that, while overall agreement in hierarchical segmentation tasks is lower, a significant portion of it is about the level, not about the presence of absence of topical shifts.

[7] Additions/deletions correspond to the cases where a boundary needs to be added or removed *within the same level*. Transpositions correspond to near hits/near misses – where the boundaries boundary positions do not align but are near, within a specified transposition window *within the same level*. Substitutions correspond to situations where the levels at which the boundaries are placed differ. For example in Figures 3.7a and 3.7b Annotator 1 placed a boundary after paragraph 1 at level 1, which Annotator 2 placed at level 2.

Figure 3.10: Mean pairwise S values for all annotators.

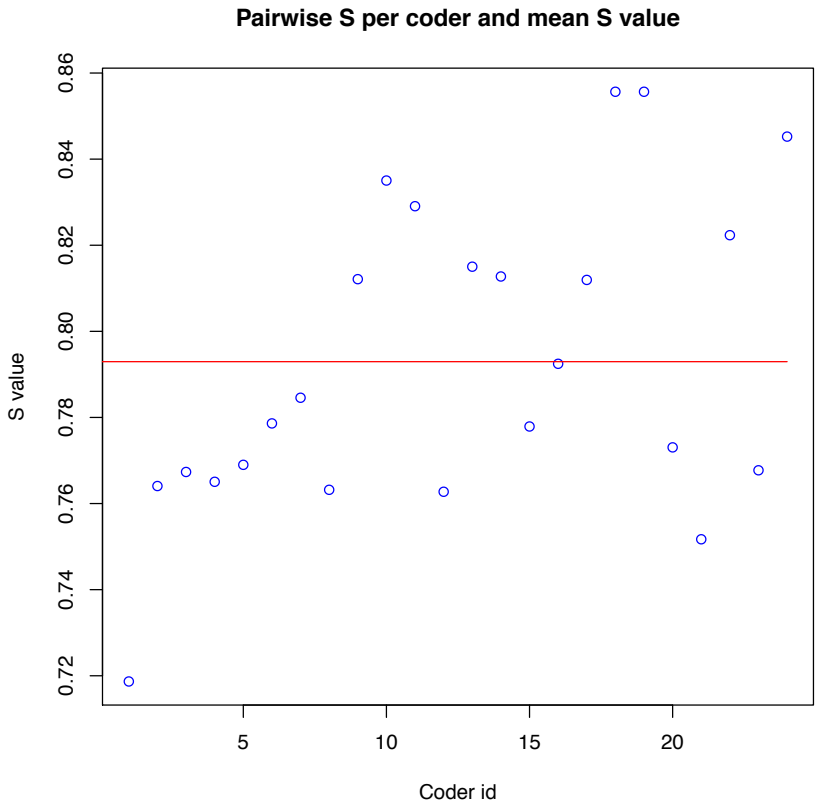
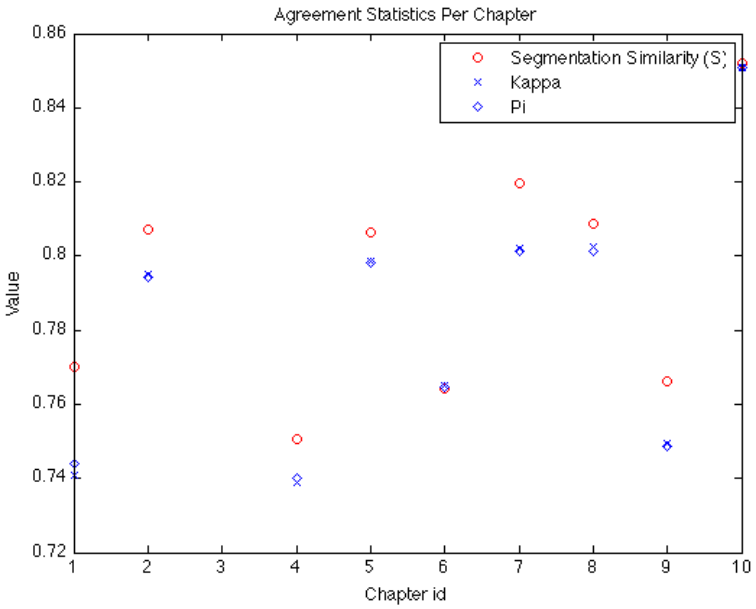


Figure 3.11: S-based inter-annotator agreement per chapter.



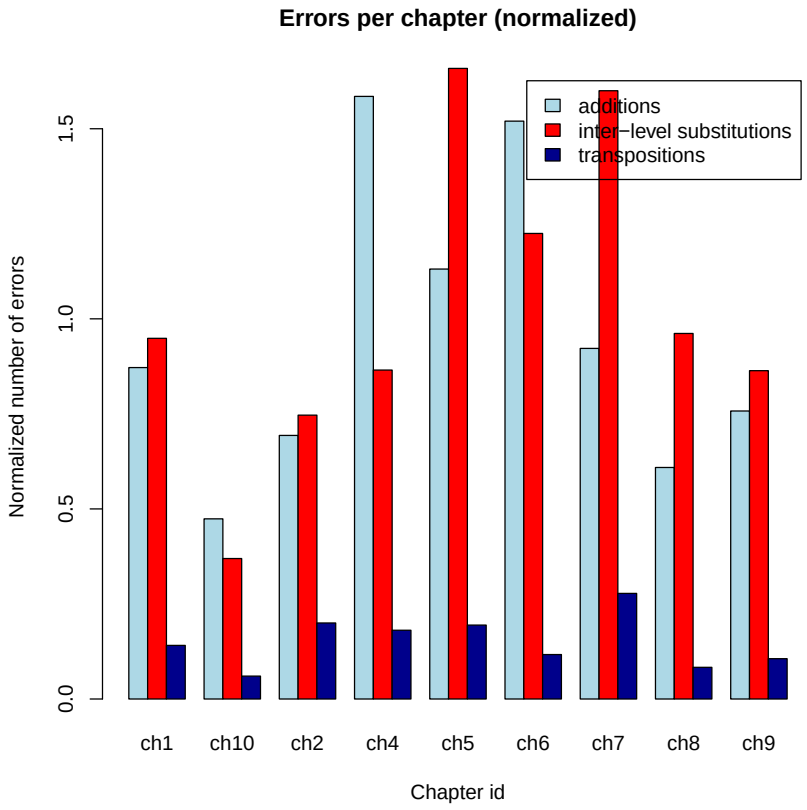
3.2.4 Conclusions about the hierarchical dataset

We have created a dataset for hierarchical segmentation of literary data. Hierarchical segmentation is more difficult than flat segmentation – both cognitively (for people) and computationally (for systems). Yet, our dataset appears to have enough integrity to serve as a benchmark. As expected, there is a fair amount of disagreement between the annotators. However, considering the nature of the task, there is also substantial agreement.

The analysis also revealed that substantial portion of disagreement is disagreement about the magnitude of topic fluctuations, not about whether there is a topic change or not. We do not have the results to support this hypothesis, but it is possible that our requirement to summarize each segment added to this type of disagreement. It is likely that when creating this outline people are more constrained, probably invoking schemas from similar documents that they have previously read, thus influencing the points of attachments in the tree.

However, the dataset appears suitable for our purposes of evaluating automatic hierarchical segmenters.

Figure 3.12: Edit operations when computing S. Number of edits / chapter length * number of annotations



3.3 Conclusions

In this chapter we have described two datasets of topical segmentations of literature that were created in the course of this work. The datasets can be used to evaluate topical segmenters as well as to study how people perform this task.

The results of our analysis suggest that the *Moonstone* corpora have enough integrity to be used as benchmark datasets. The inter-annotator agreement is higher on the flat *Moonstone* dataset. The agreement on the hierarchical dataset is more difficult to measure and to interpret. Creating trees as opposed to sequences of segments is more difficult for both people and automatic segmenters and the drop in agreement is reasonable.

Both datasets are freely available for public use and study. One of the research avenues that we have left unexplored in this work is the study of the manually written outlines created by the annotators. It would be very interesting to compare them qualitatively and to explore if disagreements about the granularity of segmentation and about the levels of particular boundaries can be explained by the annotators' interpretations of the document as captured in the outlines. However, we leave this fascinating project for future work.

Chapter 4

Affinity Propagation for text segmentation

In the previous chapter, we have described two studies of how people segment fiction according to topical fluctuations. Ultimately, it is our objective to create trees similar to those in Figures 3.7a and 3.7b automatically. This chapter describes an algorithm for text segmentation proposed in this work. This document describes an implemented and preliminarily evaluated algorithm for flat text segmentation. (In the next chapter we will extend the segmenter to build tree-like structures). While we are primarily interested in segmenting literary texts, we will also report the results on two other benchmark datasets – one consisting of transcribed speech and one consisting of chapters from medical textbooks. We do this to see how the performance of our segmenter generalizes to other challenging data types and also to allow fair comparison with the other segmenters which were built, fine-tuned and tested by their authors on those datasets. The segmenter described in this chapter is suitable for any type of data and is not literature-specific.

In this work, we use exemplar-based clustering to model the topical structure of the text. Each cluster has a central point – *an exemplar* – that somehow 'summarizes' the unifying quality of the cluster. The points are assigned to clusters based on their similarity to the exemplars. When computing cluster assignments, the program attempts to maximize the quantity known as *net similarity* – the total sum of similarities between all points and their respective exemplars.

If we extrapolate this abstract problem definition into the domain of text segmentation,

then ultimately we have the following. Using exemplar-based clustering, we will attempt to identify spans of text that have certain degree of topical unity to them. The clusters must necessarily consist of adjacent units, so as to qualify as segments. Each such segment/cluster is characterized by a central unit – exemplar– which best captures what it is about. The clusters are computed so as to maximize the total sum of similarities between each unit and its exemplar. The units used will be discussed in detail in Section 4.3 which describes the experimental setting. However, for our main datasets (*the Moonstone* datasets) we will use paragraphs as units.

The problem of exemplar-based clustering is NP-complete and has been extensively studied (*e.g.,* [Mahajan et al., 2009]). Some of the classical solutions are the k-means/k-medoid algorithms [Bishop, 2006] and the Expectation Maximization (*EM*) algorithms [Mclachlan and Krishnan, 1996]. There are two main problems with these approaches. Firstly, they are known to be highly sensitive to well-selected initialization points. Secondly, on each iteration they take into account only a part of the available information, which makes them prone to finding local as opposed to global maxima. Recently, Frey and Dueck [2007], Givoni and Frey [2009] proposed a new clustering algorithm, Affinity Propagation (*AP*) that dramatically improved both the speed of clustering and the quality of solutions. This is the algorithm that we have chosen to address our text segmentation problem.

Affinity Propagation operates by iteratively passing messages between all available data points (in our case, given that we work with text, it would correspond to all paragraphs or all sentences in a document). It takes as inputs a matrix of similarities between all points and on each iteration considers them all as potential exemplars. This global view of the data and of the possible assignments often leads to better-quality assignments than other algorithms. Technically, it is an instance of so-called loopy-belief propagation on a factor graph with cycles. Such algorithms are not guaranteed to find a globally optimal solution. They are not even guaranteed to converge. However, in practice, they have been used to achieve state-of-the-art performance in error-correcting decoding, image processing and data compression, to name a few.

In practice and in our setting, AP will operate in the following manner. On each iteration all nodes (corresponding to paragraphs to be clustered) send two types of messages: an availability message and a responsibility message. A responsibility message reflects how

likely the receiver is to be an exemplar for the sender given all other potential exemplars (all other paragraphs). An availability message reflects how likely the sender is to be an exemplar for the receiver given the evidence from all other paragraphs it exemplifies. The messages are sent iteratively until a solution emerges or until the program runs out of time.

This chapter presents a version for flat segmentation. The system is called *Affinity Propagation for Segmentation – APS*. The results are encouraging: *APS* performs on par or slightly better than two state-of-the art topical segmenters. The hierarchical version (*HAPS*) is described in the following chapter.

The flat version of *APS* operates by iteratively passing messages in a factor graph [Kschischang et al., 2001] until a good set of segments emerges. Each iteration considers all similarities – takes into account all available information. An iteration includes sending at most $O(N^2)$ messages. For the majority of realistic segmentation tasks, however, the upper bound is $O(MN)$ messages, where M is a constant. This is more computationally expensive than the requirements of locally informed segmentation algorithms such as those based on HMM or CRF (see Section 2.4), but for a globally-informed algorithm the requirements are very reasonable. *APS* works on an already pre-compiled similaritiy matrix, so it offers flexibility in the choice of similarity metrics.

In addition to requiring a similarity matrix, the algorithm also takes as input a vector of *preference* values. Preferences reflect *a priori* beliefs of how likely each paragraph is to be an exemplar. For example, if clustering news articles, one may want to give higher preference values to the leading sentences in each paragraph to reflect the fact that the leading sentence often summarizes the paragraph in that genre.

The desired granularity of segmentation can be set by adjusting preferences. This is a desired and often overlooked quality – in a realistic situation the number of segments is almost always unknown. Additionally, since each cluster/segment has an exemplar, we do not only get the information about where the topic changes, but also a rough gist of what each segment is about.

This chapter is structured as follows. Section 4.1 offers some background of message-passing algorithms in factor graphs and on the original Affinity Propagation. Section 4.2 describes the derivation of *APS*. Section 4.3 contains experimental results and Section 4.4 closes the chapter with conclusions and a discussion.

Parts of this chapter are rather technical. We tried to keep such sections self-contained.

The readers not interested in mathematics and in the derivation of the algorithm will not miss much if they skip Sections 4.1 and 4.2.

4.1 Background: factor graphs and affinity propagation for clustering

4.1.1 Factor graphs and the max-sum algorithm

This section provides a brief but hopefully self-sufficient overview of factor-graphs as a framework and also of the original Affinity Propagation (for clustering) algorithm.

The *APS* algorithm is an instance of belief propagation on a cyclic factor graph. In order to explain the derivation of the algorithm, we will first briefly introduce factor graphs as a framework.

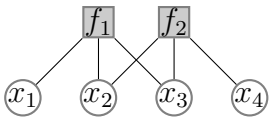
Many computational problems can be reduced to maximizing the value of a multivariate function $F(x_1, \ldots, x_n)$ which can be approximated by a sum of simpler functions. In Equation 4.1, H is a set of discrete indices and f_h is a local function with arguments $X_h \subset \{x_1, \ldots, x_n\}$:

$$F(x_1, \ldots, x_n) = \sum_{h \in H} f_h(X_h) \tag{4.1}$$

Factor graphs offer a concise graphical representation for such problems. A global function F which can be decomposed into a sum of M local functions f_h can be represented as a bi-partite graph with M function nodes and N variable nodes ($M = |H|$). Figure 4.1 shows an example of a factor graph for $F(x_1, x_2, x_3, x_4) = f_1(x_1, x_2, x_3) + f_2(x_2, x_3, x_4)$. The factor (or function) nodes are dark squares, the variable nodes are light circles.

The well-known max-sum algorithm [Bishop, 2006] seeks a configuration of variables which maximizes the objective function. It is guaranteed to find the maximum in acyclic factor graphs, but in graphs with cycles, neither convergence nor optimality are guaranteed [Pearl, 1982]. Yet, in practice good approximations can be achieved. The max-sum algorithm amounts to propagating messages from function nodes to variable nodes and from variable nodes to function nodes. A message sent from a variable node x to a function node f is computed as a sum of the incoming messages from all neighbours of x other than f (the sum is computed for each possible value of x):

Figure 4.1: Factor graph for $F(x_1, x_2, x_3, x_4) = f_1(x_1, x_2, x_3) + f_2(x_2, x_3, x_4)$.



$$\mu_{x \to f} = \sum_{f' \in N(x) \setminus f} \mu_{f' \to x} \tag{4.2}$$

$N(x)$ is the set of all function nodes which are x's neighbours. The message reflects the evidence about the distribution of x from all functions which have x as an argument, except for the function corresponding to the receiving node f.

A message $\mu_{f \to x}$ from function f to variable x is computed as follows:

$$\mu_{f \to x} = \max_{N(f) \setminus x} (f(x_1, \ldots, x_m) + \sum_{x' \in N(f) \setminus x} \mu_{x' \to f}) \tag{4.3}$$

$N(f)$ is the set of all variable nodes which are f's neighbours. The message reflects the evidence about the distribution of x from function f and its neighbours other than x.

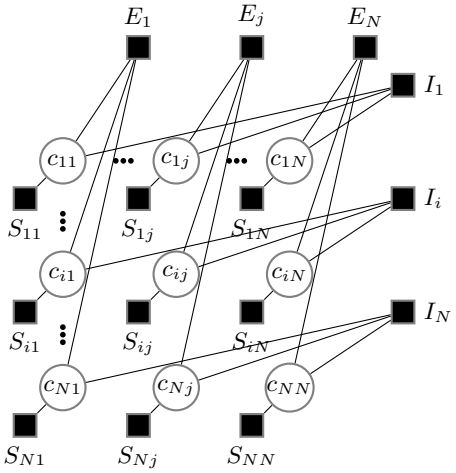
A common message-passing schedule on cyclic factor graphs is *flooding*: iteratively passing all variable-to-function messages, then all function-to-variable messages. Upon convergence, the summary message reflecting final beliefs about the maximizing configuration of variables is computed as a sum of all incoming function-to-variable messages.

4.1.2 Affinity Propagation for clustering

The *APS* algorithm described in this chapter is a modification of the original Affinity Propagation algorithm intended for exemplar-based clustering [Frey and Dueck, 2007, Givoni and Frey, 2009]. This section describes the binary variable formulation proposed by Givoni and Frey, and lays the groundwork for deriving the new update messages (Section 4.2).

Affinity Propagation for exemplar-based clustering is formulated as follows: to cluster N data points, one must specify a matrix of pairwise similarities $\{\mathcal{SIM}(i,j)\}_{i,j \in \{1,\ldots,N\}, i \neq j}$ and a vector of self-similarities (so-called *preferences*) $\mathcal{SIM}(j,j)$ which reflect *a priori* beliefs in how likely each data point is to be selected as an exemplar. Preference values

Figure 4.2: Factor graph for affinity propagation.



occupy the diagonal of the similarity matrix. The algorithm then assigns each data point to an exemplar so as to maximize net similarity – the sum of similarities between all points and their respective exemplars; this is expressed by Equation 4.7. Figure 4.2 shows a schematic factor graph for this problem, with N^2 binary variables. $c_{ij} = 1$ iff point j is an exemplar for point i. Function nodes E_j enforce a *coherence constraint*: a data point cannot exemplify another point unless it is an exemplar for itself:

$$E_j(c_{1j}, \ldots, c_{Nj}) = \begin{cases} -\infty & \text{if } c_{jj} = 0 \ \wedge \ c_{ij} = 1 \\ & \text{for some } i \neq j \\ 0 & \text{otherwise} \end{cases} \tag{4.4}$$

An I node encodes a *single-cluster constraint*: each data point must belong to exactly one exemplar – and therefore to one cluster:

$$I_i(c_{i1}, \ldots, c_{iN}) = \begin{cases} -\infty & \text{if } \sum_j c_{ij} \neq 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.5}$$

An S node encodes user-defined similarities between data points and candidate exemplars

($\mathcal{SIM}(i,j)$ is the similarity between points i and j):

$$S_{ij}(c_{ij}) = \begin{cases} \mathcal{SIM}(i,j) & \text{if } c_{ij} = 1 \\ 0 & \text{otherwise} \end{cases} \tag{4.6}$$

Equation 4.7 shows the objective function which we want to maximize: a sum of similarities between data points and their exemplars, subject to the two constraints (coherence and single-cluster per point).

$$\begin{aligned} S(c_{11}, \ldots, c_{NN}) = & \sum_{i,j} S_{i,j}(c_{ij}) + \sum_{i} I_i(c_{i1}, \ldots, c_{iN}) \\ & + \sum_{j} E_j(c_{1j}, \ldots, c_{Nj}) \end{aligned} \tag{4.7}$$

According to Equation 4.3, the computation of a single factor-to-variable message involves maximizing over 2^n configurations. E and I, however, are binary constraints and evaluate to $-\infty$ for most configurations. This drastically reduces the number of configurations which can maximize the message values. Given this simple fact, Givoni and Frey [2009] show how to reduce the necessary update messages to only two types of scalar ones: *availabilities* (α) and *responsibilities* (ρ).[1]

A responsibility message ρ_{ij}, sent from a variable node c_{ij} to function node E_j, reflects the evidence of how likely j is to be an exemplar for i given all other potential exemplars:

$$\rho_{ij} = \mathcal{SIM}(i,j) - \max_{k \neq j}(\mathcal{SIM}(i,k) + \alpha_{ik}) \tag{4.8}$$

An availability message α_{ij}, sent from a function node E_j to a variable node c_{ij}, reflects

[1] Normally, each iteration of the algorithm sends five types of two-valued messages: to and from functions E and I and a message from functions S. Fortunately, the messages sent to and from E factors to the variable nodes subsume the three other message types and it is not necessary to compute them explicitly. See [Givoni and Frey, 2009, p.195] for details.

how likely point j is to be an exemplar for i given the evidence from all other data points:

$$\alpha_{ij} = \begin{cases} \sum_{k \neq j} \max[\rho_{kj}, 0] & \text{if } i = j \\ \min[0, \rho_{jj} + \sum_{k \notin \{i,j\}} \max[\rho_{kj}, 0]] & \text{if } i \neq j \end{cases} \tag{4.9}$$

Let $\gamma_{ij}(l)$ be the message value corresponding to setting variable c_{ij} to l, $l \in \{0, 1\}$. Instead of sending two-valued messages (corresponding to the two possible values of the binary variables), we can send the difference for the two possible configurations: $\gamma_{ij} = \gamma_{ij}(1) - \gamma_{ij}(0)$ – effectively, a log-likelihood ratio.

The algorithm converges when the set of points labelled as exemplars remains unchanged for a pre-determined number of iterations. When the algorithm terminates, messages to each variable are added together. A positive final message indicates that the most likely value of a variable c_{ij} is 1 (point j is an exemplar for i), a negative message indicates that it is 0 (j is not an exemplar for i).

4.2 Affinity Propagation for flat segmentation

Affinity Propagation is a generic algorithm for clustering data points - proteins, locations on the map, images or sentences. In this section we explain how to adapt this generic algorithm for flat topical segmentation. In this dissertation, we work only with textual data, so we feel that it is more appropriate to refer to exemplars as *segment centres.* A *segment centre* is a sentence or a paragraph that best exemplifies the contents of its segment.[2]

In this setting, the data points are atomic units of text to be segmented (usually sentences or paragraphs. For clarity, we refer to them as sentences throughout this chapter, but paragraphs or arbitrary spans of text are all valid options). There are two inputs into the segmentation algorithm. Firstly, a matrix of similarities between all data points in the document (sentence or paragraphs) which may be sparse. Secondly, a vector or *preferences* - values that specify *a priori* likelihood of how likely each data point is to be chosen as a

[2]The term *centre* makes no reference to the position of the sentence with respect to the segment around it. Any sentence in a segment can be its centre, including the opening or the closing sentence. It is a *centre* only because of its content, not because of its position.

Figure 4.3: Example of valid configuration of hidden variables $\{c_{ij}\}$ for clustering.

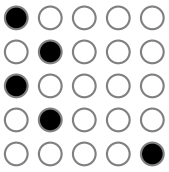
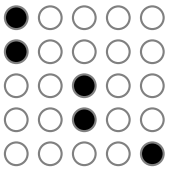


Figure 4.4: Example of valid configuration of hidden variables $\{c_{ij}\}$ for segmentation.



segment centre. There must be one preference value for each data point, the vector may be uniform. The algorithm produces two outputs: segment boundaries and segment centres for each identified segment. A segment centre is a data point that best describes the contents of that segment. Given a document, we want to assign each sentence to a segment centre so as to maximize net similarity.

The new formulation relies on the same underlying factor graph (Figure 4.2). A binary variable node c_{ij} is set to 1 iff sentence j is the segment centre for sentence i. When clustering is the objective, a cluster may consist of points coming from anywhere in the data sequence. When segmentation is the objective, a segment must consist of a solid block of points around the segment centre. Figures 4.3 and 4.4 show, for a toy problem with 5 data points, possible valid configurations of variables $\{c_{ij}\}$ for clustering and for segmentation respectively.

To formalize this new linearity requirement, we elaborate Equation 4.4 into Equation 4.10. E_j evaluates to $-\infty$ in three cases. Case 1 is the original coherence constraint. Case 2 states that no point k may be in the segment with a centre in j, if k lies before the start of the segment (the sequence $c_{(s-1)j} = 0$, $c_{sj} = 1$ necessarily corresponds to the start of the segment). Case 3 handles analogously the end of the segment.

$$E_j = \begin{cases} -\infty & \textbf{1}.\ \text{if } c_{jj} = 0 \ \wedge\ c_{ij} = 1 \text{ for some } i \neq j \\ & \textbf{2}.\ \text{if } c_{jj} = 1 \ \wedge\ c_{sj} = 1 \ \wedge\ c_{(s-1)j} = 0 \\ & \wedge\, c_{kj} = 1 \text{ for some } s \leq j,\ k < s-1 \\ & \textbf{3}.\ \text{if } c_{jj} = 1 \ \wedge\ c_{ej} = 1 \ \wedge\ c_{(e+1)j} = 0 \\ & \wedge\, c_{kj} = 1 \text{ for some } e \geq j,\ k > e+1 \\ 0 & \text{otherwise} \end{cases} \tag{4.10}$$

The *E* function nodes are the only changed part of the factor graph, so we only must re-derive α messages (availabilities) sent from factors *E* to variable nodes. A function-to-variable message is computed as shown in Equation 4.11 (elaborated Equation 4.3), and the only incoming messages to E nodes are responsibilities (ρ messages):

$$\mu_{f\to x} = \max_{N(f)\setminus x}(f(x_1,\ldots,x_m) + \sum_{x'\in N(f)\setminus x} \mu_{x'\to f}) = \tag{4.11}$$
$$\max_{c_{ij},\ i\neq j}((E_j(c_{1j},\ldots,c_{Nj}) + \sum_{c_{ij},\ i\neq j} \rho_{ij}(c_{ij})))$$

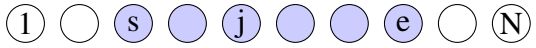
We need to compute the message values for the two possible settings of binary variables – denoted as $\alpha_{ij}(1)$ and $\alpha_{ij}(0)$ – and propagate the difference $\alpha_{ij} = \alpha_{ij}(1) - \alpha_{ij}(0)$.

Consider the case of factor E_j sending an α message to the variable node c_{jj} (i.e., $i = j$). If $c_{jj} = 0$ then point j is not its own segment centre and the only valid configuration is to set all other c_{ij} to 0:

$$\alpha_{jj}(0) = \max_{c_{ij}, i\neq j}(E_j(c_{1j},\ldots,c_{Nj}) + \sum_{c_{ij}, i\neq j} \rho_{ij}(c_{ij})) \tag{4.12}$$
$$= \sum_{i\neq j} \rho_{ij}(0)$$

To compute $\alpha_{ij}(1)$ (point j is its own segment centre), we only must maximize over configurations which will not correspond to cases 2 and 3 in Equation 4.10 (other assignments are trivially non-optimal because they would evaluate E_j to $-\infty$). Let the start of a segment be s, $1 \leq s \leq j$ and the end of the segment be e, $j \leq e \leq N$. We only need to consider configurations such that all points between s and e are in the segment while all others are

not. The following picture shows a valid configuration.[3]



To compute the message $\alpha_{ij}(1), i = j$, we have:

$$\alpha_{jj}(1) = \max_{s=1}^{j}[\sum_{k=1}^{s-1}\rho_{kj}(0) + \sum_{k=s}^{j-1}\rho_{kj}(1)] + \max_{e=j}^{N}[\sum_{k=j+1}^{e}\rho_{kj}(1) + \sum_{k=e+1}^{N}\rho_{kj}(0)] \tag{4.13}$$

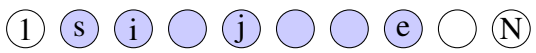
Subtracting Equation 4.12 from Equation 4.13, we get:

$$\alpha_{jj} = \alpha_{jj}(1) - \alpha_{jj}(0) = \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}) + \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}) \tag{4.14}$$

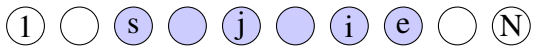
Now, consider the case of factor E_j sending an α message to a variable node c_{ij} other than segment centre j (i.e., $i \neq j$). Two subcases are possible: point i may lie before the segment centre j ($i < j$), or it may lie after the segment centre ($i > j$).

The configurations which may maximize $\alpha_{ij}(1)$ (the message value for setting the hidden variable to 1) necessarily conform to two conditions: point j is labelled as a segment centre (c_{jj} = 1) and all points lying between i and j are in the segment. This corresponds to Equation 4.15 for $i < j$ and to Equation 4.16 for $i > j$. Pictorial examples of corresponding valid configurations precede the equations.

[3]Variables c_{ij} set to 1 are shown as shaded circles, to 0 – as white circles. Normally, variables form a column in the factor graph; we transpose them to save space.

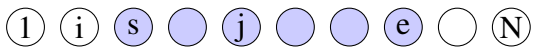


$$\alpha_{ij,\ i<j}(1) = \max_{s=1}^{i}[\sum_{k=1}^{s-1}\rho_{kj}(0) + \sum_{k=s}^{i-1}\rho_{kj}(1)] + \sum_{k=i+1}^{j}\rho_{kj}(1) + \max_{e=j}^{N}[\sum_{k=j+1}^{e}\rho_{kj}(1) + \sum_{k=e+1}^{N}\rho_{kj}(0)] \tag{4.15}$$

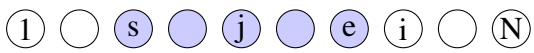


$$\alpha_{ij,\ i>j}(1) = \max_{s=1}^{j}[\sum_{k=1}^{s-1}\rho_{kj}(0) + \sum_{k=s}^{j-1}\rho_{kj}(1)] + \sum_{k=j}^{i-1}\rho_{kj}(1) + \max_{e=i}^{N}[\sum_{k=i+1}^{e}\rho_{kj}(1) + \sum_{k=e+1}^{N}\rho_{kj}(0)] \tag{4.16}$$

To compute the message value for setting the hidden variable c_{ij} to 0, we again distinguish between $i < j$ and $i > j$ and consider whether c_{jj} = 1 or c_{jj} = 0 (point j is / is not a segment centre). For c_{jj} = 0 the only optimal configuration is c_{ij} = 0 for all $i \neq j$. For c_{jj} = 1 the set of possible optimal configurations is determined by the position of point i with respect to point j. Following the same logic as in the previous cases we get Equation 4.17 for $i < j$ and Equation 4.18 for $i > j$.



$$\alpha_{ij}(0) = \max(\sum_{k\notin i,j}\rho_{kj}(0), \sum_{k=1}^{i-1}\rho_{kj}(0) + \max_{s=i+1}^{j}[\sum_{k=i+1}^{s-1}\rho_{kj}(0) + \sum_{k=s}^{j-1}\rho_{kj}(1)] + \rho_{jj}(1) + \max_{e=j}^{N}[\sum_{k=j+1}^{e}\rho_{kj}(1) + \sum_{k=e+1}^{N}\rho_{kj}(0)]) \tag{4.17}$$



$$\alpha_{ij}(0) = \max(\sum_{k \notin i,j} \rho_{kj}(0), \tag{4.18}$$
$$\max_{s=1}^{j}[\sum_{k=1}^{s-1} \rho_{kj}(0) + \sum_{k=s}^{j-1} \rho_{kj}(1)] +$$
$$\rho_{jj}(1) + \max_{e=j}^{i-1}[\sum_{k=j+1}^{e} \rho_{kj}(1) + \sum_{k=e+1}^{i-1} \rho_{kj}(0)]$$
$$\sum_{k=i+1}^{N} \rho_{kj}(0))$$

We omit the details of subtracting Equation 4.17 from 4.15 and Equation 4.18 from 4.16. The final update rules for both $i < j$ and $i > j$ appear in Algorithm 1, where we summarize the whole process.

The equations look cumbersome, but they are trivial to compute. Every summand corresponds to finding the most likely start or end of the segment, taking into account *fixed* information. When computing messages for any given sender node, we can remember the maximizing values for neighbouring recipient nodes. For example, after computing the availability message from factor E_j to c_{ij}, we must only consider one more responsibility value when computing the message from E_j to variable $c_{(i+1)j}$. The cost of computing a message is thus negligible.

When the matrix is fully specified, each iteration requires passing $2N^2$ messages, so the algorithm runs in $O(N^2)$ time and requires $O(N^2)$ memory (to store the similarities, the availabilities and the responsibilities). When performing segmentation, however, the user generally has some idea about the average or maximum segment length. In such more realistic cases, the input matrix of similarities is sparse – it is constructed by sliding a window of size M. M usually needs to be at least twice the maximum segment length or thrice the average segment length. Each iteration, then, involves sending $2MN$ messages and the storage requirements are also $O(MN)$.

As is common in loopy belief propagation algorithms, both availability and responsibility messages are dampened to avoid overshooting and oscillating. The dampening factor is λ where $0.5 \leq \lambda < 1$. In our experiments we found that the value of λ does not significantly

influence the results. We used $\lambda = 0.9$ in most experiments.

$$newMsg = \lambda * oldMsg + (1 - \lambda)newMsg \tag{4.19}$$

The *APS* algorithm is unsupervised. It only benefits from a small development set to fine-tune a few parameters: preference values and the dampening factor. *APS* does not require (nor allow) specifying the number of segments beforehand. The granularity of segmentation is adjusted through preference values; this reflect how likely each sentence is to be selected as a segment centre. (This translates into the cost of adding a segment.)

Because each message only requires the knowledge about one column or row of the matrix, the algorithm can be easily parallelized.

In our experiments, the system runs until convergence or for a maximum of 1000 iterations. Convergence is defined as a situation where the set of segment centres remains unchanged for 100 iterations.

Algorithm 1 Affinity Propagation for Segmentation

1: **input**: 1) a set of pairwise similarities $\{\mathcal{SIM}(i,j)\}_{(i,j)\in\{1,\ldots,N\}^2}$, $\mathcal{SIM}(i,j) \in \mathbb{R}$; 2) a set of preferences (self-similarities) $\{\mathcal{SIM}(i,i)\}_{i\in\{1,\ldots,N\}}$ indicating *a priori* likelihood of point i being a segment centre

2: **initialization**: $\forall i,j : \alpha_{ij} = 0$ (set all availabilities to 0)

3: **repeat**

4: iteratively update responsibilities (ρ) and availabilities (α)

5:
$$\forall i,j : \rho_{ij} = \mathcal{SIM}(i,j) - \max_{k\neq j}(\mathcal{SIM}(i,k) + \alpha_{ik})$$

6:
$$\forall i,j : \alpha_{ij} = \begin{cases} \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}) + \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}) & \text{if } i = j \\ \min[\max_{s=1}^{i}\sum_{k=s}^{i-1}\rho_{kj} + \sum_{k=i+1}^{j}\rho_{kj} + \max_{e=j}^{N}\sum_{k=j+1}^{e}\rho_{kj}, & \\ \quad \max_{s=1}^{i}\sum_{k=s}^{i-1}\rho_{kj} + \min_{s=i+1}^{j}\sum_{k=i+1}^{s-1}\rho_{kj}] & \text{if } i < j \\ \min[\max_{s=1}^{j}\sum_{k=s}^{j-1}\rho_{kj} + \sum_{k=j}^{i-1}\rho_{kj} + \max_{e=i}^{N}\sum_{k=i+1}^{e}\rho_{kj}, & \\ \quad \min_{e=j}^{i-1}\sum_{k=e+1}^{i-1}\rho_{kj} + \max_{e=i}^{N}\sum_{k=i+1}^{e}\rho_{kj}] & \text{if } i > j \end{cases}$$

7: **until** convergence

8: compute the final configuration of variables: $\forall i,j$ j is the centre for i iff $\rho_{ij} + \alpha_{ij} > 0$

9: **output**: segment centre assignments

4.3 Experimental settings

This section reports empirical evaluation of the flat *APS* segmenter. The development and implementation of the hierarchical segmenter are described in the next chapter.

Datasets. We evaluate the performance of the flat *APS* algorithm on four datasets. The first dataset is the *Moonstone* dataset for flat segmentation described in Chapter 3. The second dataset consists of 85 novels collected from the Project Gutenberg website.[4] The other two are demanding benchmark datasets for topical segmentation. We decided to use additional data in order to have a more stereoscopic picture of the performance of our segmenter.

The *Moonstone* dataset is described in detail in the previous chapter (recall that it consists of 20 chapters from *The Moonstone* segmented by 4-6 people per chapter).

The second dataset consists of 85 works of fiction downloaded from Project Gutenberg. The segment boundaries correspond to chapter breaks or to breaks between individual stories. They were inserted automatically using HTML markup in the downloaded files.

The third dataset was compiled by Malioutov and Barzilay [2006]. It consists of manually transcribed and segmented lectures on Artificial Intelligence. This dataset is particularly interesting to us because transcribed speech is likely to have low word repetition rate – similarly to literature. This property makes segmentation of both genres challenging and fascinating at the same time.

The last dataset consists of 227 chapters from medical textbooks [Eisenstein and Barzilay, 2008]. In this dataset, the gold standard segment boundaries correspond to section breaks specified by the authors. While medical textbooks are not an example of informal writing, the dataset was included because it is a rather demanding benchmark.

The datasets exhibit different characteristics. The lectures, the novels and *the Moonstone* dataset are challenging because they are less cohesive than the medical textbooks. The textbooks are cognitively more difficult to process and the authors rely on repetition of terminology to facilitate comprehension. Since lexical repetition is the main source of information for text segmentation, we expect a higher performance on this dataset. Transcribed speech, on the other hand, is considerably less cohesive. The lecturer makes an effort to speak in "plain language" and to be comprehensible, relying less on terminology.

[4] `http://http://www.gutenberg.org/`

The use of pronouns is very common, as is the use of examples.

For different reasons, the flat *Moonstone* dataset is also very challenging. Repeated use of the same words is not common in fiction, which is a problem, since all three segmenters rely only on lexical information. Additionally, since this is a novel, the text is full of dialogues and small digressions. Additionally, this part of the novel is an embedded narrative which shifts back and forth between the time of narration and the main story line (i.e., the character named Betteredge is telling the story of *The Moonstone*). All these characteristics make the dataset very challenging.

Baselines. We compare the performance of *APS* with two recent segmenters: the *Minimum Cut Segmenter* [Malioutov and Barzilay, 2006] and the *Bayesian Segmenter* [Eisenstein and Barzilay, 2008]. The authors have made Java implementations publicly available. For the *Minimum Cut Segmenter*, we select the best parameters using the script included with that distribution. The Bayesian segmenter automatically estimates all necessary parameters from the data.

All results are reported using a form of cross validation. *The Moonstone* and the lectures datasets are quite small, so we use 5-fold cross validation. For the clinical dataset and the fiction one we use ten-fold cross validation. On each iteration, the best parameters are chosen automatically using 3 development files, while the rest of the files in the fold are used for testing. The folds are created so that in the end of the experiment all available files have been used for testing, but at most $3 \times n$ randomly chosen files are used for training (since there is no need for more data to set the parameters).

Preprocessing and the choice of similarity metric. As described in Section 4.2, the *APS* algorithm takes as inputs a matrix of pairwise similarities between sentences in the document and also, for each sentence, a preference value.

We estimated topical similarity between sentences using two methods: cosine similarity using a simple bag-of-words representation and also the model of textual similarity described in [Bär et al., 2012]. This model corresponds to roughly the best system in the SemEval 2012 Textual Similarity competition.[5] The model is available as a part of the *DKPro Similarity* framework [Bär et al., 2013]. Further in this work we refer to this similarity measure as *STS-2012* baseline (because it is used as the baseline for the next SemEval Textual Similarity track).

[5]The system described in [Bär et al., 2012] scored best in two out of three evaluations.

The *STS-2012* baseline consists of a log linear regression model trained on the SemEval 2012 training data. It combines an assortment of measures of textual similarity to come up with its judgements. The metrics include n-gram overlap, word similarity measures based on ontologies, word similarity estimated using Distributional Thesaurus [Lin, 1998], Explicit Semantic Analysis [Gabrilovich and Markovitch, 2007] and stylistic similarity. We chose to use this relatively complicated metric because of its competitive performance in the SemEval 2012 competition. However, the system was not designed to measure topical similarity per se, and certainly not between many sentences coming from the same source document.[6]

By default, The STS-2012 baseline outputs values between 1 and 5. These were normalized to be between 0 and 1.

STS-2012 similarity could not computed on the clinical and the fiction datasets. In the former case, the dataset is only available in pre-processed form and the *DKPro Similarity* software could not handle it. The fiction dataset was skipped because of its size. In general, computing *STS-2012* similarity was very time-consuming (days for each dataset). The files in the fiction dataset are the largest and after a week of computing we only managed to get a small part of the dataset processed. That is why we decided to abandon these experiments.

We also use a simple cosine similarity metric to estimate topical similarity between sentences. It the same metric that is used by the *Minimum Cut Segmenter*. Using cosine similarity allows fair comparison of the segmentation algorithms. To compute cosine similarity, each sentence is represented as a vector of token-type frequencies. Following [Malioutov and Barzilay, 2006], the frequency vectors are smoothed by adding counts of words from the adjacent sentences and then weighted using a tf.idf metric (for details, see *ibid.*) When experimenting with *the Moonstone* dataset we use the fiction dataset as a corpus for tf.idf. In the AI dataset, the tf.idf is computed using segments (effectively, each document is treated a corpus). For the clinical and fiction datasets, only the data in the dataset itself is used for document-based tf.idf.

The similarity between sentence vectors s_1 and s_2 is computed as follows:

$$cos(s_1, s_2) = \frac{s_1 \bullet s_2}{||s_1|| \times ||s_2||} \tag{4.20}$$

[6]We also tried using several WordNet-based text similarity measures as implemented in the *DKPro Similarity* package. The results were rather poor so we did not pursue that direction.

The representation used by the Bayesian segmenter is too different to be incorporated into our model directly, but ultimately it is based on the distribution of unigrams in documents. This is close enough to our representation to allow fair comparison.

The fiction dataset and *the Moonstone* datasets are examples of literary language, which is known to exhibit less lexical cohesion. Because of this, the sparsity problem when computing bag-of-word representation is aggravated. Additionally, in both of these corpora, the documents are long. Even some chapters from *the Moonstone* are lengthy, and the novels in the fiction corpus are prohibitively long. That is why – when working on these two dataset – we work at the paragraph level: the similarity is measured not between sentences but between paragraphs. We use this representation with all three segmenters.

The *Bayesian Segmenter* estimates all the necessary parameters from the corpus (it has access to all available data on each iteration, not just the data from the current fold).

The *APS* and the *Minimum Cut Segmenter* both contain scripts to automatically select best parameters given a set of development files. These scripts are run on the development part of each fold.

For the *APS* algorithm *per se* we needed to set three parameters: the size of the sliding window for similarity computations, the dampening factor λ and the preference values.

Both the *APS* and the Minimum Cut segmenters also estimate the best parameters to compute the similarity matrix: a type of tf.idf measure to use and the values for smoothing word counts in sentences.

When running the *Minimum Cut Segmenter* and *APS* on matrices computed using *STS-2012* baseline, there is no need to set any of the parameters used for the computation of lexical similarity (tf.idf parameters, smoothing parameters, etc.) We use the matrix supplied by *DKPro Similarity* software and pass it to the java code for *Minimum Cut Segmenter*, effectively bypassing its script for selecting best parameters. Although this setup eliminates the need to set most parameters, there are still several parameters for which we use default values. It is possible that this may negatively affect the performance of the segmenter, though most likely the difference is negligible.

Evaluation metric. We estimate the performance of all three segmenters using two metrics: *windowDiff* and *S*. Both metrics are described in detail in Section 3.1.4 of the Chapter 3. The formulas are repeated here for convenience:

$$winDiff = \frac{1}{N-k}\sum_{i=1}^{N-k}(|ref--hyp| \neq 0) \tag{4.21}$$

$$S(bs_a, bs_b, n) = \frac{1 - |boundary\_distance(bs_a, bs_b, n)|}{pb(D)} \tag{4.22}$$

Recall that the *windowDiff* is a penalty measure – higher values correspond to more mistakes. On the other hand *S* is a similarity metric. Therefore, the higher the value of *S*, the better. *windowDiff* is computed using the code native to *APS* for the results of all three segmenters. To compute *S* we rely on the publicly available package SegEval (`http://nlp.chrisfournier.ca/software/`).

For the *Moonstone* data set we have 4-6 annotations per chapter. Both *S* and *windowDiff* are intended for pairwise comparisons: that is for comparing one hypothetical segmentation against one gold standard segmentation. Therefore, when computing the *windowDiff* value we created a gold-standard consisting of the opinion expressed by the majority of the annotators and compare the segmenters against that. For the *S* metric we report micro-averages between the segmenters and each annotation available for each chapter.

4.4 Experimental results and discussion

Table 4.1 compares the performance of the three segmenters using *windowDiff* values. The last two columns in the table (*MinCutSeg-STS* and *APS-STS*) correspond to the results of running the two similarity-based segmenters using the matrices computed using the *STS-2012* baseline.

We tested statistical significance of the results using the two-tailed t-test and 95% confidence cut-off. The values that are better than the rest with statistical significance are in bold.

On the AI lectures dataset the best result is obtained using the *Bayesian Segmenter*, closely followed by *APS* and the *Minimum Cut Segmenter*. However, the differences are not statistically significant.

	BayesSeg	MinCutSeg	*APS*	MinCutSeg-STS	*APS*-STS
AI	0.400 ($\pm$0.023)	0.431 ($\pm$0.020)	0.424 ($\pm$0.014)	0.451 ($\pm$0.023)	0.428 ($\pm$0.03)
Clinical	**0.332** (**$\pm$0.027**)	0.382 ($\pm$0.025)	0.402 ($\pm$0.042)	-	-
Moonstone	0.462 ($\pm$0.035)	0.470 ($\pm$0.095)	0.437 ($\pm$0.091)	0.441 ($\pm$0.051)	0.479 ($\pm$0.041)
Fiction	0.378 ($\pm$0.04)	0.381 ($\pm$0.076)	**0.316** (0.068)	-	-

Table 4.1: *windowDiff* values for the segmentations of the three datasets using the Bayesian segmenter, the Minimum Cut segmenter and *APS*. The values in parentheses are standard deviations across all folds.

On the clinical dataset the *Bayesian Segmenter* is once again the best one, this time with statistical significance.

On the *Moonstone* dataset *APS* produces best segmentations but the difference between its result and the second-best segmentation (*MinCutSeg-STS*) is not statistically significant.

On the fiction dataset the segmentations created by *APS* are better than the rest with 95% confidence.

It is rather disappointing that using the complicated *STS-2012* similarity metric improves the results only for the *Minimum Cut Segmenter* when running on *Moonstone* dataset.

Table 4.2 reports the results using *S* values. The values output by the metric have a disadvantage of being very close to one another and very optimistic. However, the overall picture largely mirrors the evaluation using *windowDiff* in Table 4.1. The performance of the segmenters is particularly close on the AI Lectures dataset. The *Bayesian Segmenter* is the best segmenter for the clinical dataset, while *APS* performs best on the fiction and the *Moonstone* datasets. Because the values output by *S* are so close in range in most cases the differences are statistically significant (with the exception of AI dataset).

All datasets are challenging and the baselines are very competitive, so drawing definitive conclusions is difficult. Still, we can be fairly confident that *APS* performs at least as well as the other two segmenters. It also has certain advantages.

One important difference between *APS* and the other segmenters is that *APS* does not require the number of segments as an input parameter. This is very helpful, because such

	BayesSeg	MinCutSeg	*APS*	MinCutSeg-STS	*APS*-STS
AI	0.932 (±0.007)	0.929 (±0.005)	0.931 (±0.004)	0.925 (±0.008)	0.932 (±0.001)
Clinical	**0.944** (±0.003)	0.929 (±0.003)	0.931 (±0.005)	-	-
Moonstone	0.897 (±0.007)	0.896 (±0.008)	**0.904** (±0.004)	0.901 (±0.001)	0.905 (±0.003)
Fiction	0.961 (±0.003)	0.958 (±0.003)	**0.971** (±0.005)	-	-

Table 4.2: *S* values for the segmentations of the three datasets using the Bayesian segmenter, the Minimum Cut segmenter and *APS*.

information is generally unavailable in any realistic deployment setting. The parameters are fine-tuned to maximize WindowDiff values, so this results in high-precision, low-recall segment assignments; that is because WindowDiff favours missing boundaries over near-hits.

APS also outputs segment centres, thus providing some information about a segment's topic. In Chapter 5 we evaluate how well segment centres summarize the contents of a given segment. The results suggest that while they cannot truly serve as a summary, they provide a good idea of what the segment is about. Figure 4.6 shows segment centres identified for Chapter 2 of *Moonstone*. For comparison, Figure 4.5 shows a one-level outline created by one of the annotators.

On the literary datasets *APS* performs slightly better than the other segmenters but not by much. We hypothesize that one of the reasons is that *APS* relies on the presence of descriptive segment centres which are not necessarily present for large, coarse-grained segments such as chapters in novels. It is possible for *APS* to have an advantage performing fine-grained segmentation.

4.5 Conclusions

In this Chapter, we have described *APS*, a similarity-based algorithm for topical segmentation. We have derived, implemented and evaluated a segmenter for flat segmentation.

Figure 4.5: Example of flat segmentation of Chapter 2 of *The Moonstone* by Wilkie Collins. Annotator 2.

1. Paragraphs 1-4: Betteredge describes his mistress. In number 3 and 4 he still focuses on the mistress but begins to shift the focus to himself as a transition to the next episode.
2. Paragraphs 5-11: Betteredge describes his marriage. Number 11 shifts focus to his daughter Penelope
3. Paragraphs 11-14: Betteredge describes his promotion to Stewart
4. Paragraph 15: Betteredge realizes that he has been telling the story of his own life rather than the moonstone.

Figure 4.6: Segment centres identified by *APS* for Chapter 2 of *The Moonstone* by Wilkie Collins.

1. Segm. centre 2: I have omitted to state that I went with the bride to the bride's husband's house and lands down here. "Sir John," she says, "I can't do without Gabriel Betteredge." "My lady," says Sir John, "I can't do without him, either." That was his way with her – and that was how I went into his service. It was all one to me where I went, so long as my mistress and I were together.
2. Segm. centre 6: "I have been turning Selina Goby over in my mind," I said, "and I think, my lady, it will be cheaper to marry her than to keep her."
3. Segm. centre 11: As for me, I went on with my business as bailiff year after year up to Christmas 1847, when there came a change in my life. On that day, my lady invited herself to a cup of tea alone with me in my cottage. She remarked that, reckoning from the year when I started as page-boy in the time of the old lord, I had been more than fifty years in her service, and she put into my hands a beautiful waistcoat of wool that she had worked herself, to keep me warm in the bitter winter weather.

In the majority of cases it performs on par, or better than two very demanding baseline systems.

The segmenter takes as input a possibly sparse matrix of similarities between sentences or paragraphs in the input document. It output segment boundaries and segment centres – data points that best describe the contents of each segment.

We evaluated *APS* on four datasets, using two variants of similarity matrices and compared the results against two demanding baselines. Our evaluation suggests that apart from the clinical dataset, *APS* performs similarly or better than the other segmenters. This is especially the case with the literary datasets.

In the following chapter we extend our model of topical segmentation to build trees instead of sequences of segments.

Chapter 5

Hierarchical segmentation using Affinity Propagation

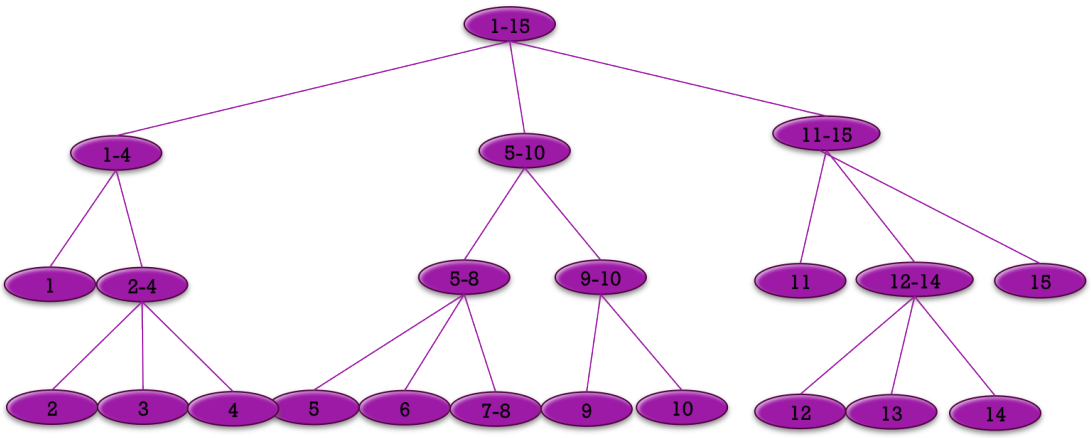
5.1 Introduction

In chapter 4 we derived *APS*, an algorithm for flat topical segmentation of text based on Affinity Propagation for clustering [Givoni and Frey, 2009]. *APS* finds the most pronounced topical shifts in a document and assigns each segment a central point – a sentence or a paragraph that best captures what the segment is about.

However, topic in documents fluctuates in many subtle ways and modelling a document as a sequence of several rough topical segments is perhaps not the most informative way. Section 2.3 contains a brief review of computational approaches to modelling topical and discourse structure. [Mann and Thompson, 1988, Marcu, 2000, Hernault et al., 2010, Feng and Hirst, 2012] model the structure of discourse as trees, while Wolf and Gibson [2006] model it as a graph. In this work, we take the former approach and attempt to take our topical segmenter a step further, by building topical trees that represents how the topical structure of documents changes.

We have been using Chapter 2 of the novel *The Moonstone* as an example throughout this dissertation. The complete chapter is available in Appendix A. The *APS* algorithm divided this chapter of the novel into three segments (with the segments centres shown in Figure 4.6 in the preceding chapter). In this chapter we will show how to build a topical

Figure 5.1: Example of a topical tree for Chapter 2 of *The Moonstone* by Wilkie Collins.



tree, similar to the one shown in Figure 5.1. The tree in the Figure 5.1 corresponds to the outline created by a student annotator shown in Figure 5.2. Both are repeated here from Chapter 1 for the reader's convenience.

Figure 5.2: Example segmentation of Chapter 2 of *The Moonstone* by Wilkie Collins. Annotator 2.

1. Paragraphs 1-4: the single life of the narrator (a note: paragraph one sets the time the story begins)
    (a) paragraph 1: the diamond's location
    (b) paragraphs 2-4: how he came to be in his career
        i. paragraph 2: how he met the lady
        ii. paragraph 3: how he became hired by the lady
        iii. how he came to be promoted to bailiff
2. paragraphs 5-10: life with a love interest before and after marriage
    (a) paragraphs 5-8: motivations for finding a wife
        i. paragraph 5: loneliness
        ii. paragraph 6: economics and love
        iii. paragraphs 7-8: the lady's approval
            A. paragraph 7: asking for approval
            B. paragraph 8: receiving approval
    (b) paragraphs 9-10: experiences of marrying and being married to a woman
        i. paragraph 9: apprehensions about marriage
        ii. paragraph 10: mediocrity in married life
3. paragraphs 11-15: life as a widower and into old age
    (a) paragraph 11: death of the narrator's wife and first mention of daughter Penelope
    (b) paragraphs 12-14: convincing the narrator to alter his occupation
        i. paragraph 12: Lady lulls him into security with kindness
        ii. paragraph 13: narrator realises her plan, and his pride fights against it
        iii. paragraph 14: narrator is convinced after reading Crusoe he will feel better about it tomorrow
    (c) paragraph 15: the difficulty of writing a story without accidentally writing about oneself (a note: only technically in his old age, not really about his old age)

In order to find a tree-like topical structure we derive a hierarchical version of *APS*, to which we further refer as Hierarchical Affinity Propagation for Segmentation or *HAPS*. Givoni et al. [2011], Givoni [2012] propose a hierarchical version of Affinity Propagation for Clustering. We use that graphical model as a basis for our hierarchical segmentation algorithm. Similarly to *APS*, *HAPS* takes as input similarities between data points (sentences or paragraphs) and finds segments by identifying segment centres – data points that best describe all other data points in a segment. The only difference is that *HAPS* finds a hierarchy of segment centres: segment centres at any level l must come from the set of segment centres identified at a previous, more fine-grained level $l - 1$.

This chapter is structured as follows. Section 5.2 formulates the underlying *HAPS* model. Section 5.2.1 contains step-by-step derivation of the new update messages. Section 5.3 describes the settings for the experimental evaluation of *HAPS*. And Section 5.4 reports results. In Section 5.6 we discuss some of the weaknesses of this approach and potential avenues for its improvement.

The readers not interested in how the formulae are derived can safely skip section 5.2.1 describing the derivation of the algorithm.

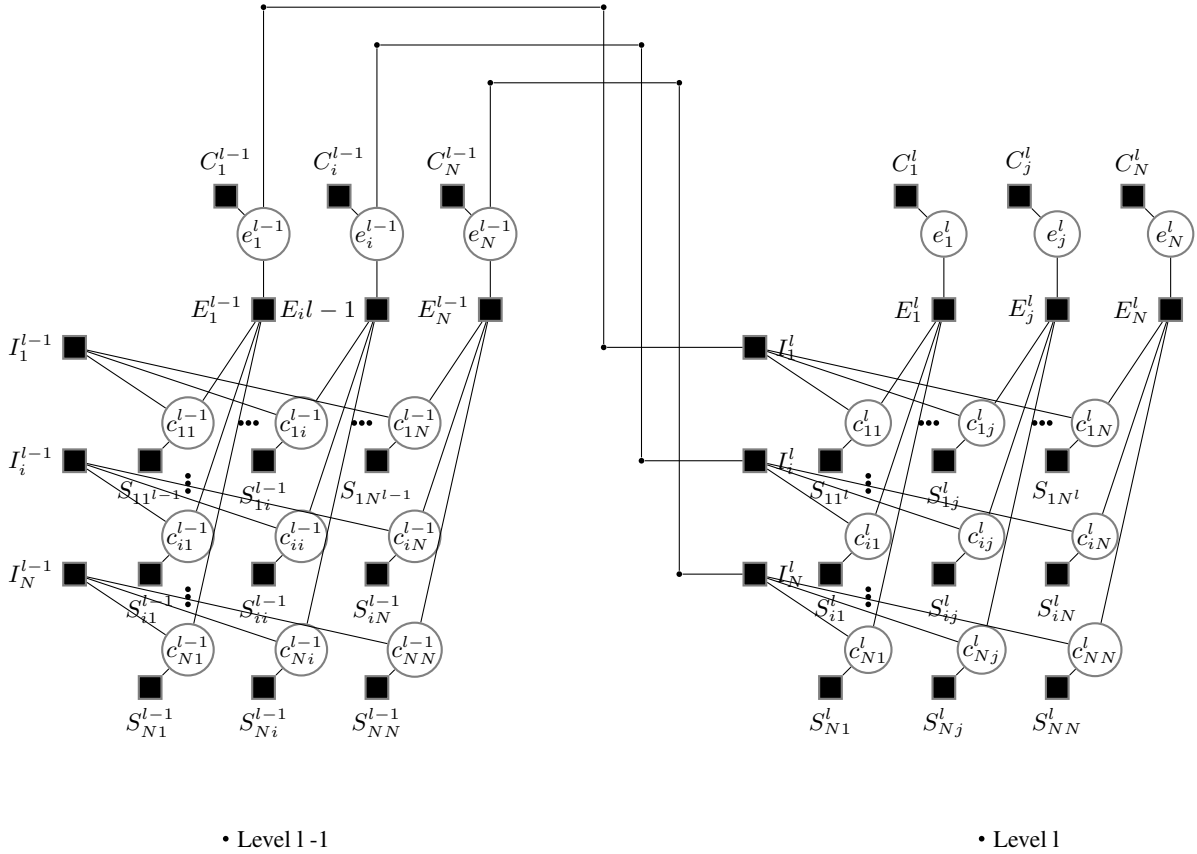
5.2 Formulating hierarchical Affinity Propagation for segmentation (*HAPS*)

We will start by defining a factor graph model for *HAPS*. The model described here closely follows the model for hierarchical clustering proposed by [Givoni, 2012], in particular the model HAP-B. For the sake of completeness, we repeat a lot of the definitions from that work in this dissertation. However, we skip lengthy derivations that are identical to the HAP-B model (namely the derivation of the new responsibility messages) and refer the reader to [Givoni, 2012, p. 96-120, 160-172] for details.

Our objective is as follows. Given a sequence of N data point (sentences or paragraphs) we wish to build a topical tree where top levels capture the most pronounced fluctuations of topic in a document, and lower levels capture more fine-grained ones. Formally, we wish to find a hierarchy of segments such that segments higher up in the tree consist of smaller segments found at the lower levels of the tree. At each level of the tree, all data points will be assigned to segment centres. However, we impose a constraint that segment centres at the level $l, l > 1$ must be chosen from the set of of segment centres identified at the preceding level $l - 1$.

The inputs for *HAPS* are quite similar to the inputs for *APS*. Assuming that one wishes to segment a sequence of N data points into a hierarchy of L levels, one needs to supply *HAPS* with L matrices of pairwise similarities between these points, $\{\mathcal{SIM}^l(i,j)\}_{i,j\in\{1,\ldots,N\}}$. Each matrix $\{\mathcal{SIM}^l(i,j)\}$ defines similarities for a corresponding level $l \in \{1,\ldots,L\}$. It is possible (and indeed this is what we do in this work) to have a single matrix of similarities shared between levels, but the model allows additional flexibility. The matrices do not

Figure 5.3: Factor graph for Hierarchical Affinity Propagation for Segmentation, *HAPS*.



need to be fully specified, nor do they need to be symmetric.

The second type of input are preferences, vectors of self-similarities which reflect *a priori* beliefs of how likely each data point is to be selected as a segment centre. For a model with L layers one needs to supply L vectors $P^l = p_1^l, ...p_N^l$ where each value p_i^l reflects *a priori* belief of how likely the point i is to be chosen as a segment centre in the layer l. Because preferences are also the only parameter that controls the granularity of segmentation, these values need to differ between levels, with higher values corresponding to more segments (suitable for bottom levels) and lower values corresponding to fewer segments (suitable for top-level segments). In this work, we set preferences to the same value *within* levels, effectively making all sentence equally likely candidates to be segment centres. However, more intelligent settings may improve the quality of segmentations.

Figure 5.3 shows a factor graph corresponding to the *HAPS* model. It looks quite similar to the model for *APS* (Chapter 4, Figure 4.2) with several important differences. The factor graph contains layers corresponding to the levels of segmentation. In Figure 5.3 the level l on the right-hand side of the picture corresponds to a higher, more coarse-grained level in the topical tree. The level on the left, $l-1$, is a more fine-grained level. The bottom node in this notation is always $l = 1$. Two adjacent levels l and $l-1$ are connected by edges between factor nodes I_i^l at level l and corresponding variable nodes e_i^{l-1} at the lower level $l-1$. Please note that in our new notation the superscripts on both factor and variable nodes denote the layer from which node comes.

Recall that the dark rectangular nodes are factor (or function) nodes. They encode a function on variables that are connected to them. The light circles are variable nodes. In the *HAPS* model, all variable are binary. By running the max-sum algorithm, we can obtain a configuration that maximizes that sum of all of the component functions (encoded as factor nodes).

There are four types of factor nodes in Figure 5.3 : C, E, I, S. These functions reoccur at each level. Factors S and C encode similarities and preferences. Factors I and E are constraint functions: they evaluate either to $-\infty$ (for invalid configurations) or to 0 (for valid ones).

Let us look closer at Figure 5.3. Just like in *APS*, variable nodes c_{ij}^l are binary. If $c_{ij}^l = 1$ then the point i is assigned to the segment centred around point j in the level l of the hierarchy. Conversely, if $c_{ij}^l = 0$ then j is not the segment centre for the point i in the

level l.

There are two main differences between *HAPS* and the flat *APS* model. First of all, factor nodes I_i^l at each level are connected to the variable nodes e_i^{l-1} in the previous level, carrying information across levels. Secondly, there are new variable nodes, e_j^l, and the nodes e_j^l are in turn connected to new factor nodes C_j^l. The new variables e_j^l are binary. They explicitly capture if the point j is chosen to be a segment centre in the level l. Therefore $e_j^l = 1$ iff there is a segment around j in the level l. Otherwise $e_j^l = 0$.

Factor nodes C_j^l explicitly capture preference values at the level l. $C_j^l(e_j^l = 1) = p^l(j)$ (the preference value of data point j in level l) and $C_j^l(e_j^l = 0) = 0$. Recall that in the flat *APS* model, preferences were captured by the S_{jj} factor nodes on the diagonal of the model. In the *HAPS* model, for all $\forall j, l$, $S_{jj}^l = 0$ and preferences are captured by the C_j^l factors.

Now let us describe in detail the factor nodes.

Similarly to *APS*, S_{ij}^l factor nodes encode user-defined similarities between data points and candidate segment centres. ($\mathcal{SIM}^l(i,j)$ is the similarity between points i and j in the level l of segmentation hierarchy):

$$S_{ij}^l(c_{ij}^l) = \begin{cases} \mathcal{SIM}^l(i,j) & \text{if } c_{ij}^l = 1 \\ 0 & \text{otherwise} \end{cases} \tag{5.1}$$

Factor nodes $C_j^l(e_j^l)$ are new in this model. They encode preferences (self-similarities) for data points at each level l: if a data point j is chosen to be a segment centre at a given level, then $C_j^l(e_j^l)$ evaluates to p_j^l (preference of j at the level l), otherwise its value is 0.

$$C_j^l(e_j^l) = \begin{cases} p_j^l & \text{if } e_j^l = 1 \\ 0 & \text{otherwise} \end{cases} \tag{5.2}$$

In the flat *APS* model an I factor node encodes a *single-segment constraint*: each data point must belong to exactly one segment (and therefore be assigned to exactly one segment centre). In the *HAPS* model I factors also encode relationships between levels of the hierarchy. At any given level $l > 1$, a data point j may be chosen to be a segment centre if and only if it was chosen to be a segment centre in the previous level:

$$I_i^{l>1}(c_{i1}^l, \ldots, c_{iN}^l, e_i^{l-1}) = \begin{cases} 0 & \text{if } \sum_j c_{ij}^l = 1, c_{ii}^l \leq e_i^{l-1} \\ -\infty & \text{otherwise} \end{cases} \tag{5.3}$$

At the bottom of the hierarchy, $l = 1$, the constraint I is identical to the flat *APS*:

$$I_i^{l=1}(c_{i1}^1, \ldots, c_{iN}^1) = \begin{cases} 0 & \text{if } \sum_j c_{ij}^1 = 1 \\ -\infty & \text{otherwise} \end{cases} \tag{5.4}$$

The E factor nodes encode the segmentation constraint (as opposed to clustering) and the coherence constraint. *The coherence constraint* states that a if any point i chooses point j as a segment centre then j must be its own segment centre as well ($e_j^l = c_{jj}^l = \max_i c_{ij}^l$). *The segmentation constraint* states that if a point j is chosen to be a segment centre for a segment that starts at s and ends at e, then all points between s and e must be included in the segment (cases 1 and 2).

$$E_j^l = \begin{cases} 0 & e_j^l = c_{jj}^l = \max\limits_i c_{ij}^l \text{ and neither case 1 nor 2 applies} \\ & \textbf{1}. \text{ if } c_{jj}^l = 1 \ \wedge \ c_{sj} = 1 \ \wedge \ c_{(s-1)j} = 0 \\ & \wedge\, c_{kj}^l = 1 \text{ for some } s \leq j, k < s-1 \\ & \quad \text{(all points between start of the segment and its centre must be part of the segment)} \\ & \textbf{2}. \text{ if } c_{jj}^l = 1 \ \wedge \ c_{ej} = 1 \ \wedge \ c_{(e+1)j} = 0 \\ & \wedge\, c_{kj}^l = 1 \text{ for some } e \geq j, k > e+1 \\ & \quad \text{(all points between the segment centre and the end of the segment must be} \\ & \quad \text{part of the segment)} \\ -\infty & \text{otherwise} \end{cases} \tag{5.5}$$

Therefore, given L levels of segmentation, L similarity matrices $\{\mathcal{SIM}^l(i,j)\}$ and L vectors of preferences P^l, the objective function that we are trying to maximize is given in Equation 5.6:

$$\max_{\{c_{ij}^l\},\{e_j^l\}} S(\{c_{ij}^l\},\{e_j^l\}) = \sum_{i,j,l} S_{i,j}^l(c_{ij}^l) + \sum_{i,l} I_i^l(c_{i1}^l,\ldots,c_{iN}^l,e_i^{l-1}) + \sum_{j,l} E_j^l(c_{1j}^l,\ldots,c_{Nj}^l,e_j^l) + \sum_{j,l} C_j^l(e_j^l) \tag{5.6}$$

Intuitively, we are trying to find a configuration of variables that maximizes the sum of similarities of all data points and their respective segment centres for all layers, plus the sum of preferences for all segment centres subject to the constraints of coherence, segmentation and hierarchical structure.

5.2.1 Update messages for *HAPS*

This section contains step-by-step derivation of the new update messages for *HAPS*. We skip the derivation of the new responsibility messages because the process is lengthy and identical to the HAP-B model by [Givoni, 2012, pp. 168-172].

Figure 5.4 shows a close-up of all message types in the graph.

Recall that a message from a variable to a function is computed as the sum of all incoming messages, except for the message from the recipient node:

$$\mu_{x\to f} = \sum_{f'\in N(x)\setminus f} \mu_{f'\to x} \tag{5.7}$$

$N(x)$ is the set of all function nodes which are x's neighbours. The message reflects the evidence about the distribution of x from all functions which have x as an argument, except for the function corresponding to the receiving node f.

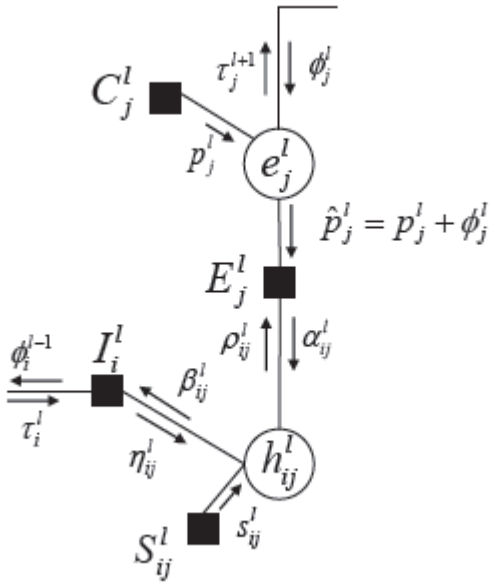
A message $\mu_{f\to x}$ from function f to variable x is computed as follows:

$$\mu_{f\to x} = \max_{N(f)\setminus x} (f(x_1,\ldots,x_m) + \sum_{x'\in N(f)\setminus x} \mu_{x'\to f}) \tag{5.8}$$

$N(f)$ is the set of all variable nodes which are f's neighbours. The message reflects the evidence about the distribution of x from the function f and its neighbours other than x.

Looking at Figure 5.4 we can see there are ten types of messages: $\alpha, \rho, \phi, \tau, p, \hat{p}, \mu, \eta, \beta$

Figure 5.4: Message types sent in the *HAPS* model.



and s. Messages $\alpha, \rho, p, \hat{p}, \mu, \eta, \beta$ and s are intra-level ones while τ and ϕ carry information across levels.

α_{ij}^l and ρ_{ij}^l messages circulate between variable nodes c_{ij}^l and function nodes E_j^l. Messages β_{ij}^l and η_{ij}^l are sent between variable nodes c_{ij}^l and function nodes I_i^l within level l. There are also messages μ_j^l and $\hat{p}_{ij}^l$ sent between E_j^l and e_j^l. While we use the values of these messages in our computations, they do not need to be sent explicitely as they are subsumed by other messages (see below). Messages s_{ij}^l and p_j^l evaluate either to 0 or to the value of either $\mathcal{SIM}^l(i,j)$ or p_j^l respectively (similarity between points i and j or preference value for point j in the level l).

The inter-layer messages are ϕ and τ.

ϕ_i^{l-1} is sent from the factor node I_i^l to the variable node e_i^{l-1}in the *preceding* level $l-1$. For clarity, we refer to the messages going from level l to $l-1$ as ϕ^{l-1} and we refer to the messages incoming to level l as ϕ^l. In essence, ϕ_i^{l-1} are messages that 'tell' lower levels about the distribution of variables from upper levels.

τ_j^{l+1} is sent from a variable node e_j^l in the level l to the factor node I_i^{l+1} of the *following* level $l+1$. With τ messages, we refer to the upward-going messages as τ_j^{l+1} and to the

incoming messages from level $l-1$ - as τ_j^l. Effectively, these messages carry the information about the distribution of variables and the constraints of the lower levels to the upper levels.

Let us consider messages $\hat{p}_j^l$ sent from the variable node e_j^l to the factor node E_j^l. According to Equation 5.7, a message sent from a variable node to a factor node is computed as the sum of all incoming messages (for each possible value of the node) except for the message from the recipient node. Similarly to the derivation of APS in the previous chapter, instead of sending two separate messages (corresponding to two possible values of a binary variable) we send the difference, which corresponds to log-likelihood ratio. Recall that we denote the messages corresponding to setting a binary variable to 1 as $msg(1)$, a message for setting the variable to 0 as $msg(0)$. The difference is denoted as msg.

$$\hat{p}_j^l = \hat{p}_j^l(1) - \hat{p}_j^l(0) \tag{5.9}$$

The value $\hat{p}_j^l(1)$ corresponds to setting $e_j^l = 1$, which means that data point j is a segment centre at the level l. This means that the message ϕ_j^l coming in from the next level is unconstrained (since data points that are chosen as segment centres in level l may or may not be segment centres in level $l+1$, according to constrain I_j^{l+1} (Equation 5.3)). It also means that the value of the message from factor $C_j^l = p_j^l$ – the preference value for this data point.

$$\hat{p}_j^l(1) = p_j^l + \phi_j^l(1) \tag{5.10}$$

The value $\hat{p}_j^l(0)$ corresponds to setting $e_j^l = 0$, which means that data point j is not a segment centre.

$$\hat{p}_j^l(0) = p_j^l * 0 + \phi_j^l(0) = \phi_j^l(0) \tag{5.11}$$

Taking the difference between 5.10 and 5.11 we get:

$$\boxed{\hat{p}_j^l = p_j^l + \phi_j^l} \tag{5.12}$$

Now let us consider message μ_j^l going from the factor node E_j^l to the variable node e_j^l. According to Equation 5.8, a message from a factor node to a variable node corresponds to the maximum value of the sender function plus the sum of all incoming messages.

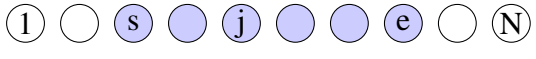
The case where $e_j^l = 0$ means that means that data point j is not segment centre and therefore it cannot exemplify any other data points. Otherwise, the constraint E_j is violated and the function evaluates to $-\infty$. The only valid configuration is to set all $c_{ij}^l = 0, \forall i$:

The picture graphically shows a valid configuration. However, keep in mind that the value of μ_j depends on the column j, not the row j. The picture is only horizontal to simplify the layout of this dissertation.

$$\mu_j^l(0) = \rho_{jj}(0) + \sum_{i \neq j} \rho_{ij}(0) \tag{5.13}$$

In Equation 5.13 we factor out $\rho_{jj}(0)$ only so as to simplify the following subtraction, otherwise it could have been under the summation sign.

If $e_j^l = 1$, it means that data point j is a segment centre and there is a segment starting at some point $s \leq j$ and ending at some point $e \geq j$:



$$\mu_j^l(1) = \rho_{jj}^l(1) + \max_{s=1}^{j}[\sum_{k=1}^{s-1}\rho_{kj}(0) + \sum_{k=s}^{j-1}\rho_{kj}(1)] + \max_{e=j}^{N}[\sum_{k=j+1}^{e}\rho_{kj}(1) + \sum_{k=e+1}^{N}\rho_{kj}(0)] \tag{5.14}$$

Subtracting Equation 5.13 from Equation 5.14 we get:

$$\boxed{\mu_j^l = \rho_{jj}^l + \max_{s=1}^{j}[\sum_{k=s}^{j-1}\rho_{kj}] + \max_{e=j}^{N}[\sum_{k=j+1}^{e}\rho_{kj}(1)]} \tag{5.15}$$

Next we derive upward going τ_j^{l+1} messages sent from variable nodes e_j^l at level l to function nodes I_j^{l+1} at level $l+1$.

If $e_j^l = 1$ (corresponding to $\tau j^{l+1}(1)$) then:

$$\tau j^{l+1}(1) = p_j^l(1) + \mu_j^l(1) \tag{5.16}$$

$$\tau j^{l+1}(0) = p_j^l(0) + \mu_j^l(0) \tag{5.17}$$

Subtracting Equation 5.17 from Equation 5.16 we get:

$$\tau j^{l+1} = p_j^l + \mu_j^l \tag{5.18}$$

Substituting 5.15 in Equation 5.18 we get:

$$\boxed{\tau_j^{l+1} = p^l(j) + \rho_{jj}^l + \max_{s=1}^{j}(\sum_{k=s}^{j-1} \rho_{kj}^l) \; + \; \max_{e=j}^{N}(\sum_{k=j+1}^{e} \rho_{kj}^l)} \tag{5.19}$$

Please note that there is no need to send messages μ explicitly as the necessary information is incorporated directly in terms of ρ messages in Equation 5.19.

Next, we turn our attention to the downward-going ϕ_i^{l-1} messages, sent from the factor node I_i^l at the level l to variable nodes e_j^{l-1} at the level $l-1$. According to the constraint I, only data points selected to be segment centres at the lower level may be selected to be segment centres at the current level. Additionally, I_i^l requires that each data point i be assigned to exactly 1 segment centre j in each level l (translating into the fact that for any row i, $\sum c_{ij} = 1$).

$\phi_i^{l-1}(1)$ corresponds to setting $e_j^{l-1} = 1$. This means that at this level, l, we only need to satisfy the one-out-of-all constraint (since if i is a segment centre at the level $l-1$ it may or may not be a segment centre at the level l):

$$\phi_i^{l-1}(1) = \max_k(\beta_{ik}^l(1) + \sum_{t \neq k} \beta_{it}^l(0)) \tag{5.20}$$

If, on the other hand, $e_j^{l-1} = 0$, then the data point j cannot be chosen to be a segment centre at the level l ($\beta_{jj}^l = 0$). However, the one-out-of-all constraint must still be satisfied:

$$\phi_i^{l-1}(0) = \beta_{ii}^l(0) + \max_{k \neq i}(\beta_{ik}^l(1) + \sum_{t \neq k} \beta it^l(0)) \tag{5.21}$$

It is easier to take the difference between 5.20 and 5.21 if we re-write 5.20 as a maximum of two cases (if point i is not a segment centre at the level l or if it is one):

$$\phi_i^{l-1}(1) = \max_k(\beta_{ik}^l(1) + \sum_{t\neq k}\beta it^l(0)) = \tag{5.22}$$
$$max[\beta_{ii}^l(0) + \max_{k\neq i}(\beta_{ik}^l(1) + \sum_{t\neq k}\beta it^l(0)), \beta_{ii}^l(1) + \sum_{k\neq i}\beta_{ik}^l(0)]$$

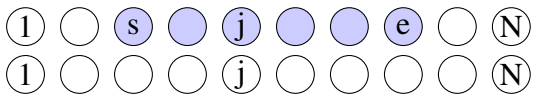
Now we can take the difference between Equations 5.22 and 5.21 using two facts: 1) $\max(x, y) - x = \max(0, y - x)$ and 2) $\beta_{ij}^l = s_{ij}^l + \alpha_{ij}^l$:

$$\phi_i^{l-1} = max[\beta_{ii}^l(0) + \max_{k\neq i}(\beta_{ik}^l(1) + \sum_{t\neq k}\beta it^l(0)), \beta_{ii}^l(1) + \sum_{k\neq i}\beta_{ik}^l(0)] - \tag{5.23}$$
$$- [\beta_{ii}^l(0) + \max_{k\neq i}(\beta_{ik}^l(1) + \sum_{t\neq k}\beta it^l(0))] =$$
$$= \max[0, \alpha_{ii} - \max_{k\neq i}(s_{ij}^l + \alpha_{ij}^l)]$$

$$\boxed{\phi_i^{l-1} = \max[0, \alpha_{ii} - \max_{k\neq i}(s_{ik}^l + \alpha_{ik}^l)]} \tag{5.24}$$

Next we will derive the availability messages α_{ij}^l sent within level l from factor node E_j^l to variable node c_{ij}^l. The derivation of these messages is almost identical to that in the flat version of APS. The only difference is that factor nodes E_j^l have one more incoming message that needs to be taken into account, namely $\hat{p}_j^l$. Just as before, we need to consider three cases: 1) $i = j$ 2) $i < j$ and 3) $i > j$.

If $i = j$ we need to take the difference between the following configurations (which corresponds to taking the difference between Equations 5.25 and 5.26):



$$\alpha^l_{jj}(1) = \hat{p}^l_j(1) + \max_{s=1}^{j}[\sum_{k=1}^{s-1}\rho_{kj}(0) + \sum_{k=s}^{j-1}\rho_{kj}(1)] + \max_{e=j}^{N}[\sum_{k=j+1}^{e}\rho_{kj}(1) + \sum_{k=e+1}^{N}\rho_{kj}(0)] \tag{5.25}$$

Note that we use the value of $\hat{p}^l_j$ corresponding to setting $e^l_j = 1$ because if there is a segment centred around j, e^l_j must be 1, according to the constraint E. Conversely, if there is no segment, then $e^l_j = 0$ and all other variables (c^l_{ij}) must be set to 0 as well:

$$\alpha^l_{jj}(0) = \hat{p}^l_j(0) + \sum_{i\neq j}\rho_{ij}(0) \tag{5.26}$$

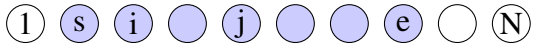
Taking the difference and substituting $\hat{p}^l_j = p^l_j + \phi^l_j$ we get:

$$\alpha^l_{jj} = \hat{p}^l_j + \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}) \; + \; \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}) \tag{5.27}$$

$$\boxed{\alpha^l_{jj} = p^l_j + \phi^l_j + \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}) \; + \; \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj})}$$

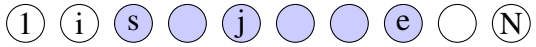
Please note that by using the equality $\hat{p}^l_j = p^l_j + \phi^l_j$ we eliminate the need to explicitly compute or send $\hat{p}$ messages.

Next, let us consider the case where $i < j$. If $c^l_{ij} = 1$, then there must a segment centred around j. This means that $c^l_{jj} = 1$ and $e^l_j = 1$:



$$\alpha_{ij,\ i<j}(1) = \hat{p}^l_j(1) + \max_{s=1}^{i}[\sum_{k=1}^{s-1}\rho^l_{kj}(0) + \sum_{k=s}^{i-1}\rho^l_{kj}(1)] + \sum_{k=i+1}^{j-1}\rho^l_{kj}(1) + \rho^l_{jj}(1) + \quad (5.28)$$
$$+ \max_{e=j}^{N}[\sum_{k=j+1}^{e}\rho^l_{kj}(1) + \sum_{k=e+1}^{N}\rho^l_{kj}(0)]$$

If, on the other hand, $c^l_{ij} = 0$ then there are two possibilities. There may be a segment around j but it starts after i (Equation 5.29) or there may be no segment around j (Equation 5.30). The former case correspond to:



$$\alpha^l_{ij}(0) = \hat{p}^l_j(0) + \sum_{k=1}^{i-1}\rho_{kj}(0) + \max_{s=i+1}^{j}[\sum_{k=i+1}^{s-1}\rho_{kj}(0) + \sum_{k=s}^{j-1}\rho_{kj}(1)] + \quad (5.29)$$
$$\rho_{jj}(1) + \max_{e=j}^{N}[\sum_{k=j+1}^{e}\rho_{kj}(1) + \sum_{k=e+1}^{N}\rho_{kj}(0)]$$

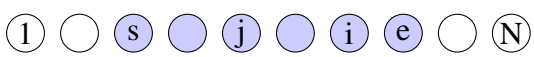
In the second case (if there is no segment centred around j at this level) we have:

$$\alpha^l_{jj}(0) = \hat{p}^l_j(0) + \sum_{i\neq j}\rho_{ij}(0) + \rho_{jj}(0) \quad (5.30)$$

Taking the difference we get an equation much like that in the flat APS:

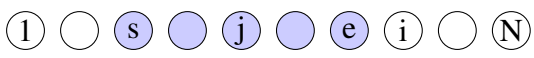
$$\alpha^l_{ij,i<j} = \min[(\max_{s=1}^{i} \sum_{k=s}^{i-1} \rho^l_{kj} + \sum_{k=i+1}^{j} \rho^l_{kj} + \max_{e=j}^{N} \sum_{k=j+1}^{e} \rho^l_{kj}) + p^l_j + \phi^l_j, \quad \max_{s=1}^{i} \sum_{k=s}^{i-1} \rho_{kj} + \min_{s=i+1}^{j} \sum_{k=i+1}^{s-1} \rho_{kj}] \tag{5.31}$$

Next we turn to the case where $i > j$. If $c^l_{ij} = 1$ then j must be a segment centre and all points between j and i must belong to the segment:



$$\alpha^l_{ij,\ i>j}(1) = \max_{s=1}^{j}[\sum_{k=1}^{s-1} \rho_{kj}(0) + \sum_{k=s}^{j-1} \rho_{kj}(1)] + \sum_{k=j}^{i-1} \rho_{kj}(1) + \max_{e=i}^{N}[\sum_{k=i+1}^{e} \rho_{kj}(1) + \sum_{k=e+1}^{N} \rho_{kj}(0)] + \hat{\rho}^l_j(1) \tag{5.32}$$

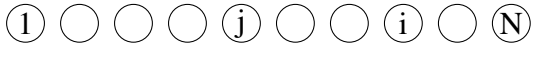
If $c^l_{ij} = 0$ there may be a segment around j which ends before i:



$$\alpha^l_{ij}(0) = max^j_{s=1}[\sum_{k=1}^{s-1} \rho^l_{kj}(0) + \sum_{k=s}^{j-1} \rho^l_{kj}(1)] + \rho^l_{jj}(1) + \max_{e=j}^{i-1}[\sum_{k=j+1}^{e} \rho^l_{kj}(1) + \sum_{k=e+1}^{i-1} \rho^l_{kj}(0)] + \tag{5.33}$$

$$+ \sum_{k=i+1}^{N} \rho_{kj}(0) + \hat{\rho}^l_j(1)$$

Alternatively, there may be no segment around j:



$$\alpha_{ij}^{l}(0) = \sum_{k \notin i,j} \rho_{kj}^{l}(0) + +\hat{\rho}_{j}^{l}(0) \tag{5.34}$$

Taking the difference we get:

$$\boxed{\begin{aligned}\alpha_{ij,i>j}^{l} = \min[&(\max_{s=1}^{j} \sum_{k=s}^{j-1} \rho_{kj}^{l} + \sum_{k=j}^{i-1} \rho_{kj}^{l} + \max_{e=i}^{N} \sum_{k=i+1}^{e} \rho_{kj}^{l}) + p_{j}^{l} + \phi_{j}^{l}, \\ &\min_{e=j}^{i-1} \sum_{k=e+1}^{i-1} \rho_{kj}^{l} + \max_{e=i}^{N} \sum_{k=i+1}^{e} \rho_{kj}^{l}]\end{aligned}} \tag{5.35}$$

The top level of the hierarchy, $l = L$ is not connected to another higher level and there are no incoming ϕ messages. Because of that the availability messages at the level $l = L$ are identical to those of *APS*.

The responsibility messages in *HAPS* are identical to the responsibility messages in HAP [Givoni, 2012, pp. 168-171]. The derivation of these messages follows the same logic as all other messages explained here. It relies on the fact that $\rho_{ij}^{l} = s_{ij}^{l} + \eta_{ij}^{l}$ and $\beta_{ij}^{l} = s_{ij}^{l} + \alpha_{ij}^{l}$. We will not repeat these lengthy derivations here. The final equations are as follows:

If $i = j$ then:

$$\boxed{\rho_{ii}^{l,l>1} = \min(0, \tau_{i}^{l}) - \max_{k \neq i}(s_{ik}^{l} + \alpha_{ik}^{l})} \tag{5.36}$$

If $i \neq j$ then:

$$\boxed{\rho_{ij}^{l,l>1} = s_{ij}^{l} + \min[\max(0, -\tau_{i}^{l}) - \alpha_{ii}^{l}, -\max_{k \notin i,j}(s_{ik}^{l} + \alpha_{ik}^{l})]} \tag{5.37}$$

It should be noted that at the bottom level of the hierarchy, $l = 1$, the I factors are not connected to e variables from the previous level (as there is no previous level). Therefore,

at the level $l = 1$ the responsibility messages are identical to those of *APS*:

$$\boxed{\rho_{ij}^{l,l=1} = s_{ij}^{1} - \max_{k \neq j}(s_{ik}^{1} + \alpha_{ik})} \tag{5.38}$$

The pseudo-code for *HAPS* is shown in Algorithm 2.

The computational properties of *HAPS* are similar to those of the flat version. On each iteration, we need to send $L \times M \times N$ messages where L is the number of levels in the hierarchy, $M \leq N$ is the size of the sliding window used to compute similarities and N is the total number of data points (sentences or paragraphs). So the computational complexity is $O(MNL)$. The algorithm is easy to parallelize because the computation of messages for each column (or row) is independent of other columns (or rows).

The messages still need to be dampened to avoid overshooting and oscillation:

$$newMsg = \lambda * oldMsg + (1 - \lambda)newMsg \tag{5.39}$$

We used a dampening factor λ, $0.5 \leq \lambda < 1$. In practice, *HAPS* appears to be rather insensitive to the values of λ. We used $\lambda = 0.95$ for most experiments.

In our experiments we have tried two message-passing schedules: the flooding schedule described in the previous chapter and also a different schedule suggested by [Givoni, 2012]. When using the flooding schedule, for each level of the hierarchy, we iteratively send ϕ, ρ, α and τ messages. When using Givoni's schedule, the intra-level messages are sent 10 times within each level and only then the upward going τ message are sent once. The procedure is propagated all the way up the hierarchy and is repeated going downward. During preliminary experiments we found that the results were identical for both schedules but the latter schedule took significantly longer to converge. Because of that all results are reported using the flooding schedule.

Algorithm 2 Hierarchical Affinity Propagation for Segmentation

1: **input**: 1) a set of pairwise similarities $\{\mathcal{SIM}(i,j)\}_{(i,j)\in\{1,\ldots,N\}^2}$, $\mathcal{SIM}(i,j) \in \mathbb{R}$; 2) a set of preferences (self-similarities) $\{\mathcal{SIM}(i,i)\}_{i\in\{1,\ldots,N\}}$ indicating *a priori* likelihood of point i being a segment centre

2: **initialization**: $\forall i,j : \alpha_{ij} = 0$ (set all availabilities to 0)

3: **repeat**

4: iteratively update responsibilities (ρ) and availabilities (α)

5:
$$\forall i,l : \phi_i^{l-1} = \max[0, \alpha_{ii} - \max_{k\neq i}(s_{ik}^l + \alpha_{ik}^l)]$$

6:
$$\forall i,j,l : \rho_{ij}^l = \begin{cases} \min(0, \tau_i^l) - \max_{k\neq i}(s_{ik}^l + \alpha_{ik}^l) & \text{if } i = j \\ s_{ij}^l + \min[\max(0, -\tau_i^l) - \alpha_{ii}^l, -\max_{k\not\in i,j}(s_{ik}^l + \alpha_{ik}^l)] & \text{if } i \neq j \end{cases}$$

7:
$$\forall i,j,l : \alpha_{ij}^l = \begin{cases} p_j^l + \phi_j^l + \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}^l) \; + \; \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}^l) & \text{if } i = j \\ \alpha_{ij,i<j}^l = \min[(\max_{s=1}^{i}\sum_{k=s}^{i-1}\rho_{kj}^l + \sum_{k=i+1}^{j}\rho_{kj}^l + \max_{e=j}^{N}\sum_{k=j+1}^{e}\rho_{kj}^l) + p_j^l + \phi_j^l, & \\ \qquad \max_{s=1}^{i}\sum_{k=s}^{i-1}\rho_{kj}^l + \min_{s=i+1}^{j}\sum_{k=i+1}^{s-1}\rho_{kj}^l] & \text{if } i < j \\ \min[(\max_{s=1}^{j}\sum_{k=s}^{j-1}\rho_{kj}^l + \sum_{k=j}^{i-1}\rho_{kj}^l + \max_{e=i}^{N}\sum_{k=i+1}^{e}\rho_{kj}^l) + p_j^l + \phi_j^l, & \\ \qquad \min_{e=j}^{i-1}\sum_{k=e+1}^{i-1}\rho_{kj}^l + \max_{e=i}^{N}\sum_{k=i+1}^{e}\rho_{kj}^l] & \end{cases}$$

8:
$$\forall j,l : \tau_j^{l+1} = p^l(j) + \rho_{jj}^l + \max_{s=1}^{j}(\sum_{k=s}^{j-1}\rho_{kj}^l) \; + \; \max_{e=j}^{N}(\sum_{k=j+1}^{e}\rho_{kj}^l)$$

9: **until** convergence

10: compute the final configuration of variables: $\forall i,j \; j$ is the segment centre for i iff $\rho_{ij} + \alpha_{ij} > 0$

11: **output**: segment centre assignments

5.3 Experimental setting

In this section we describe an experimental evaluation of *HAPS*.

It was more difficult to evaluate the *HAPS* segmenter than the flat *APS* segmenter. Hierarchical segmentation is far less explored than one-level topical segmentation and because of this it was difficult to find either comparable systems or evaluation metrics or benchmark datasets. In the experiments described in this section we did our best to evaluate the segmenter and to appropriately position it within the context of related research given what is available in the public domain.

In order to see how sensible and useful are the topical trees produced by the *HAPS* algorithm, we ran the system on two datasets annotated with hierarchical structure. The only comparable hierarchical segmenter [Eisenstein, 2009] is unfortunately not publicly available at the time of writing. To gauge the performance of *HAPS* we compared the trees produced by *HAPS* to those obtained by running two flat segmenters (*Minimum Cut Segmenter* and *Bayesian Segmenter*) iteratively. Both *Minimum Cut Segmenter* and *Bayesian Segmenter* require 'knowing' the number of segments at each level in advance – which is not a very realistic requirement for hierarchical segmentation: it is hard to know that number even for a single level, let alone for each subsegment at each level. *HAPS* does not require this information (nor can it make any use of it if available), but through the use of preferences offers a degree of abstraction when choosing the granularity of segmentations.

We compared the performance of *HAPS* with two baselines (none of them a comparable hierarchical segmenter) using two evaluation metrics. The baselines consist of running the *Minimum Cut Segmenter* and the *Bayesian Segmenter* iteratively (first, to find the top level segment boundaries, then for each identified segment, we treat it as a separate document and find its own internal 'segments'). Normally, both the *Minimum Cut Segmenter* and the *Bayesian Segmenter* have been designed for situations when the number of segments is known in advance. It is possible to run the *Bayesian Segmenter* without this parameter (though this is not the intended usage), however *Minimum Cut Segmenter* cannot be run without specifying the number of segments in advance. To allow fair comparison (since *HAPS* does not have access to this information) we ran the *Bayesian Segmenter* with and without specifying the number of segments and the *Minimum Cut Segmenter* only with the number of segments given. *HAPS* did not have access to the information about the number

of segments.

The results of the experiments suggest that *HAPS* performs comparably or slightly better than the baseline segmenters with the number of segments specified. In the situation when the *Bayesian Segmenter* does not have access to the information about the number of segments, *HAPS* performs significantly better. However, it should be kept in mind that the evaluation metrics are severely limited in their perceptiveness.

Datasets. We have used two datasets to evaluate the performance of the *HAPS* segmenter. The first one is the hierarchical *Moonstone* dataset described in detail in Section 3.2. Recall that the dataset consists of nine chapters from *The Moonstone* annotated by 3-6 people each. The average number of annotations per chapter is 4.8. The annotators were requested to build topical trees corresponding to each chapter, briefly describing the contents of each segment. The resulting structure is similar to a hierarchical outline that can be mapped back onto the text. An example annotation for Chapter 2 of *the Moonstone* can be found in Figure 5.2 in the beginning of this Chapter. It is also important to remember that this dataset is segmented at the paragraph level. Therefore, in our experiments we also work with paragraphs.

The hierarchical *Moonstone* dataset is a very challenging dataset. The annotators themselves remarked that the task was at times difficult and that they occasionally had trouble finalizing their decisions. Just like the flat *Moonstone* dataset it is a literary one, with many digressions and a low rate of lexical repetition.

The second dataset is the hierarchical *Wikipedia* dataset compiled by Carroll [2010]. The dataset consists of 66 Wikipedia articles. The hierarchical annotations were automatically derived from structure of the corresponding webpage for each article. The dataset is also rather challenging. For one thing, the structure contains more levels than the hierarchical *Moonstone* corpus. It was mentioned in Section 3.2.3 that the average depth of segmentation in the *Moonstone* dataset is 3.00 levels ($std.dev. = 0.65$). In the *Wikipedia* dataset the average depth of the topical tree is 3.85 levels ($std.dev. = 0.43$). On the other hand, the trees in the *Moonstone* corpus are broader, while in the *Wikipedia* dataset there often are long and narrow branches, or occasionally one comes across the top level nodes consisting of one or two sentences (this often happens with the first introductory sentences of the articles). The *Wikipedia* corpus is segmented at the level of individual sentences.

To simplify fine-tuning of the parameters and the interpretation of results, we trimmed

the trees in the Wikipedia corpus to three levels. That is, we only load the three top (i.e., the most prominent) levels of segmentation. We also run *HAPS* and the baseline segmenters to produce three-tier trees.

Baselines. To the best of our knowledge, there are no publicly available implementations of hierarchical segmenters. A recent segmenter explicitly built to establish hierarchal structure [Eisenstein, 2009] is no longer available at the time of writing this dissertation.

To see if there is any use in explicitly producing hierarchical topical trees, we have compared the performance of *HAPS* against the results obtained by iteratively running single-level segmenters. For example, to iteratively produce a hierarchical segmentation, we would run a flat segmenter requesting it to produce the gold-standard number of segments in the coarsest level of the reference segmentation, then take the obtained segments and segment each one one more time, again requesting the segmenter to produce the reference number of boundaries for each subsegment. This procedure would be repeated $l - 1$ times to produce a tree with l levels.

The flat segmenters chosen for these experiments are *Minimum Cut Segmenter* and *Bayesian Segmenter*. It should be noted that both baseline segmenters were effectively given a lot more information than *HAPS*: at each level they were 'told' how many segments and subsegments there were in the reference segmentation. In many cases at lower levels of the tree, providing this information determines the answer. This is especially true about the leaves of the topical trees in the *Moonstone* dataset. In that dataset, described at the level of paragraph, the segmentations in the bottom level are very fine-grained, oftentimes each paragraph is described as a separate segment. In such cases, providing the number of segments at the bottom level gives a very significant advantage to the segmenter.

HAPS, on the other hand, neither needs nor accepts the number of segments as a parameter. Instead, the values of the preferences specify a more abstract parameter: the granularity of segmentation at each level.

In order to allow fair comparison, we compared the trees produced by *HAPS* to the trees produced by the *Bayesian Segmenter* under two different conditions: with and without knowing the number of segments in advance. It should be noted that the *Bayesian Segmenter* was designed and tested in a situation when it had access to this information. It is not possible to start the *Minimum Cut Segmenter* without specifying the number of segments, so all trees produced by that segmenter were produced having access to the in-

formation about the number of segments at each level.

Evaluation metrics. Despite our hopes to evaluate *HAPS* with either B metric or S metric [Fournier and Inkpen, 2012, Fournier, 2013a], the hierarchical versions of these metrics are not yet publicly available.

Because of this, we evaluate *HAPS* and the baselines using two metrics: 1) *windowDiff* computed for each layer of the tree in isolation and 2) the *evalHDS* metric for evaluating hierarchical segmentations proposed by Carroll [2010]. It is available as a part of EvalHDS software.[1]

Recall that *windowDiff* is computed as follows:

$$winDiff = \frac{1}{N-k}\sum_{i=1}^{N-k}(|ref - hypo| \neq 0) \tag{5.40}$$

Effectively, *windowDiff* is computed by sliding a window of size k throughout the reference and segmentation sequences in parallel and counting the number of windows where the number of reference (ref) and hypothetical ($hypo$) boundaries in the windows are not equal. The number of erroneous windows is normalized by $N - k$ where N is the number of the sequence to be segmented.

*evalHDS*metric is computed by spreading the error over all levels in the segmentation tree. It is designed to give higher weight to the top levels of the tree and lower weight to the levels towards the bottom:

$$E_{windowDiff} = \frac{1}{|R|}\sum_{i} c_i windowDiff(R_i, H_i) \tag{5.41}$$

Here *R* is the set of reference boundaries and *H* is the set of hypothetical ones, ordered by their prominence (the level in the tree). The following equation must also hold:

$$R_i = \{b_j : b_j \in R \wedge j \leq i\} \tag{5.42}$$

The software allows using either *windowDiff* or *Pk* as the fundamental error metric inside the Equation 5.41. In our experiments we use the *windowDiff* version because it is a stricter measure than *Pk*.

[1] `http://www.idiom.ucsd.edu/~lucien/segmentation/EvalHDS.tgz`

Both *windowDiff* and *evalHDS* are less than ideal metrics for evaluating hierarchical segmentations because in both cases each level of the tree is viewed and evaluated in complete isolation from its neighbours. It would be far more insightful if we could evaluate segmenters taking into account all levels of the tree. It would be good to distinguish the errors of omission (the segmenter completely missed a reference boundary) from the error of substitution (a segmenter missed a boundary in the correct level, but placed it in an adjacent one). The hierarchical version of the S metric promised such analysis but, unfortunately, it is not yet publicly available.[2]

In the *Moonstone* dataset, each chapter is annotated by 3-6 people. It is not obvious how to combine all available annotations into a gold standard for each chapter. For flat datasets, it is customary to use the majority opinion as a judgement. However, it is less than ideal in this situation. As we have pointed out in Section 3.2.3, people disagree less about the presence/absence of a segment boundary than they disagree about its prominence - what level it should be at. Because of this, we felt that taking the majority opinion for each level did not make much sense. When evaluating the performance of *HAPS* and the baselines on the *Moonstone* dataset we compared the hypothetical segmentations against each available annotation and used the average value of the metric across all annotations to select the best parameters. The results reported in the next section are computed in the same manner.

We did our best to obtain a realistic picture of the results, but each metric has its shortcomings. We compared topical trees using *windowDiff* and *evalHDS* [Carroll, 2010]. Both metrics are penalties: the higher the values, the worse the hypothetical segmentation. *evalHDS* computes *windowDiff* for each level of the tree in isolation and weighs the errors according to their prominence in the tree. We computed *evalHDS* using the publicly available Python implementation [Carroll, 2010]. When working with the *Moonstone* dataset, we realized that the software produces very low values, almost too good to be true. That is because the bottommost annotations are very fine-grained. Sometimes each paragraph corresponds to a separate segment. This causes problems for the software. So, when we report *evalHDS* values for the *Moonstone* dataset, we only consider two top levels of the

[2]We chose not to use single-level versions of either *S* or *B* in the same manner as we used *windowDiff*. Using these versions would not have circumvented the disadvantage of viewing each level in isolation from its neighbours. Neither of the metrics has been used very widely and the results would have been difficult to position within the community without providing much additional insight.

tree, disregarding the leaves. We also remove the "too good to be true" outliers, though the "bad" tail is left intact. We applied the same procedure to all three segmenters, only for the *Moonstone* dataset.

When computing *windowDiff*, we treated each level of the tree as a separate segmentation and compared each hypothetical level against a corresponding level in the reference segmentation.

To ensure that evaluations are well-defined at all levels, we propagated the more pronounced reference breaks to lower levels (in both annotations and in the results). This was done to ensure that the whole sequence is segmented at each level. Otherwise *windowDiff* is not well-defined. Conceptually this means that if there is a topical shift of noticeable magnitude (e.g. at the top level), there must be at least a shift of less pronounced magnitude (e.g., an an intermediate level).

Experimental design. We ran *HAPS* and the two iterative baselines on both available hierarchical datasets. To simplify the analysis and because the average depth of manually created trees in the *Moonstone* dataset is 3.00, we ran all segmenters so as to build three-tier trees. An example tree produced by *HAPS* is shown in Figure 5.5. The text in this Figure corresponds to the segment centres identified by *HAPS*. Of course, other segmenters do not identify segment centres. The Figure shows only the top two levels of the tree since the bottom level of the tree contains six segments and, correspondingly, six paragraph-length segment centres.

Bayesian Segmenter computes its own internal representation of documents. The representations used by both *HAPS* and *Minimum Cut Segmenter* are matrices of similarities between sentence or paragraphs in the input document. The default similarity metric is cosine similarity (for details see the previous Chapter). We also ran both *HAPS* and *Minimum Cut Segmenter* using *STS-2012* similarity. It is computed in exactly the same manner as for the flat *APS* (see the previous Chapter).

All segmenters are unsupervised and do not require training. However, *Minimum Cut Segmenter* and *HAPS* benefit from a small development set to fine-tune the segmentation parameters. We evaluate the segmenters using a procedure similar to cross validation. For n-fold cross-validation the corpus is randomly divided into n folds. Each fold contains at most 3 development files and the rest of the fold is used for testing. The best parameters are automatically selected using the development part of the fold. For *HAPS* the parameters

Figure 5.5: Example topical tree produced by *HAPS* for Chapter 2 of *The Moonstone*. Two top levels.

1. Paragraphs 0-8. Segment centre:
   The woman I fixed my eye on, was the woman who kept house for me at my cottage. Her name was Selina Goby. I agree with the late William Cobbett about picking a wife. See that she chews her food well and sets her foot down firmly on the ground when she walks, and you're all right. Selina Goby was all right in both these respects, which was one reason for marrying her. I had another reason, likewise, entirely of my own discovering. Selina, being a single woman, made me pay so much a week for her board and services. Selina, being my wife, couldn't charge for her board, and would have to give me her services for nothing. That was the point of view I looked at it from. Economy with a dash of love. I put it to my mistress, as in duty bound, just as I had put it to myself.

   (a) Paragraphs 0-3. Segment centre:
   I have omitted to state that I went with the bride to the bride's husband's house and lands down here. "Sir John," she says, "I can't do without Gabriel Betteredge." "My lady," says Sir John, "I can't do without him, either." That was his way with herand that was how I went into his service. It was all one to me where I went, so long as my mistress and I were together.

   (b) Paragraphs 4-8. Segment centre:
   The woman I fixed my eye on, was the woman who kept house for me at my cottage. Her name was Selina Goby. [...]

2. Paragraphs 9-14. Segment centre:
   I received this magnificent present quite at a loss to find words to thank my mistress with for the honour she had done me. To my great astonishment, it turned out, however, that the waistcoat was not an honour, but a bribe. My lady had discovered that I was getting old before I had discovered it myself, and she had come to my cottage to wheedle me (if I may use such an expression) into giving up my hard out-of-door work as bailiff, and taking my ease for the rest of my days as steward in the house. I made as good a fight of it against the indignity of taking my ease as I could. But my mistress knew the weak side of me; she put it as a favour to herself. The dispute between us ended, after that, in my wiping my eyes, like an old fool, with my new woollen waistcoat, and saying I would think about it.

are fine-tuned using exhaustive searching and *Minimum Cut Segmenter* chooses the best parameters using a version of A* search.[3]

The parameters for *HAPS* can be roughly divided into two groups: parameters for computing the similarity matrix (e.g., smoothing parameters and the type of $tf.idf$ metric to use - corpus or segment-based) and the segmentation parameters: dampening factor λ, preference values for each level of the tree and the size of the sliding window for the similarity matrix. For the first group of parameters we use the experiments from Chapter 4 as a guide and reduce the search space by computing similarity matrices with the same parameters as we used in the experiments in Section 5.4. *HAPS* segmenter turned out rather insensitive to the value of λ as long as $\lambda \geq 0.8$. We used the value $\lambda = 0.95$ in all experiments.

For preference values, preferences at the top of the tree must have lower values than at the bottom of the tree because the lower the value of preferences, the fewer segments will be identified at that level. The segmenter is quite sensitive to the fluctuations in preference values for fine-grained segmentations (the values between the minimum and the median value of similarities in the input matrix). However, it becomes less sensitive for near-zero or negative preferences. This happens because *HAPS* and *APS* reply on the existence of informative segment centres to identify segment boundaries. If such representative data points cannot be found, then the segmenter is forced to choose "in the dark" – it must assign data points to segment centres that are all equally bad at representing them. Effectively, this translates into the fact that it is rather difficult to force *HAPS* to produce fewer segments than it finds segment centres for past a certain point. In all our experiments, *HAPS* converged to a solution.

We used the regular flooding schedule of message passing in all experiments on hierarchical segmentation.[4]

[3]Because of the naive search strategy that *HAPS* uses it effectively can only search a very small hypothesis space, much smaller than the *Minimum Cut Segmenter*. This in one of the first items to correct in our future work.

[4]In the preliminary experiments we also tried the schedule proposed by Givoni [2012]: sending 10 iterations of intra-level messages before sending upward going messages to the next level once and repeating the same procedure going downward. But the results appeared to be the same, so we did not use this schedule in the final experiments.

	Level	*Moonstone windowDiff*	*Wikipedia windowDiff*	*Moonstone evalHDS*	*Wikipedia evalHDS*
HAPS	3 (top)	0.337 (± 0.060)	0.421 (± 0.060)	0.353	0.450
	2 (middle)	0.422 (± 0.060)	0.447 (± 0.070)	(± 0.072)	(± 0.015)
	1 (bottom)	0.556 (± 0.070)	0.617 (± 0.080)		
MinCutSeg-iter.	3 (top)	0.375	0.440 (± 0.075)	0.377	0.444
	2 (middle)	0.541	0.424 (± 0.064)	(± 0.002)	(± 0.002)
segm. known	1 (bottom)	0.601	0.471 (± 0.057)		
BayesSeg-iter.	3 (top)	0.353 (± 0.071)	0.391 (± 0.070)	0.367	0.370
	2 (middle)	0.406 (± 0.053)	0.344 (± 0.033)	(± 0.089)	(± 0.019)
segm. known	1 (bottom)	0.504 (± 0.064)	0.354 (± 0.033)		
BayesSeg-iter.	3 (top)	0.600 (± 0.071)	0.637 (± 0.070)	0.453	0.437
	2 (middle)	0.447 (± 0.053)	0.877 (± 0.033)	(± 0.089)	(± 0.022)
segm. unknown	1 (bottom)	0.545 (± 0.064)	0.952 (± 0.033)		

Table 5.1: Evaluation of *HAPS* and iterative versions of *Minimum Cut Segmenter* and *Bayesian Segmenter* using *windowDiff* per level and *evalHDS*

5.4 Experimental results and discussion

Table 5.1 shows the results of comparing *HAPS* with two baseline segmenters using *windowDiff* and *evalHDS*. (In this Table the results are reported for the versions of *HAPS* and *Minimum Cut Segmenter* that use simple cosine similarity matrices. The results using *STS-2012* matrices are reported in Table 5.2.)

HAPS was run without knowing the number of segments. The *Minimum Cut Segmenter* required that the exact number be specified. The *Bayesian Segmenter* was tested with and without that parameter. Therefore, rows 3 and 4 in Table 5.1 correspond to baselines considerably more informed than *HAPS*. This is especially true about the bottom levels where sometimes knowing the exact number of segments unambiguously determines the only possible segmentation.

The results suggest that *HAPS* performs well on the *Moonstone* dataset even when compared to more informed baselines. This applies to both metrics, *windowDiff* and *evalHDS*. *Bayesian Segmenter* performs slightly better at the bottom levels of the tree when it has the information about the exact number of segments. We hypothesize that the advantage may be due to this additional information, especially when segmenting already small segments at level 1 into a predefined number of segments. Another explanation may be that when using *windowDiff* as the evaluation metric, *HAPS* was fine-tuned so as to maximize the value of *windowDiff* at the top level, effectively disregarding lower levels of segmentation.

On the Wikipedia dataset all segmenters perform worse. The informed version of *Bayesian Segmenter* performs best, but it is interesting to note a significant drop in per-

	Level	*Moonstone windowDiff*	*Wikipedia windowDiff*	*Moonstone evalHDS*	*Wikipedia evalHDS*
HAPS-STS	3 (top)	0.347 (± 0.05)	0.418 (±0.05)	0.359	0.500
	2 (middle)	0.532(± 0.012)	0.458 (±0.057)	(± 0.078)	(±0.023)
	1 (bottom)	0.584 (± 0.031)	0.743 (±0.101)		
MinCutSeg-iter.-STS	3 (top)	0.401	0.441(± 0.071)	0.423	0.489
	2 (middle)	0.513	0.480 (±0.051)	(± 0.046)	(±0.034)
segm. known	1 (bottom)	0.584	0.479 (± 0.023)		

Table 5.2: Evaluation of *HAPS* and iterative version of *Minimum Cut Segmenter* using *STS-2012* matrices as input. The table reports *windowDiff* per level and *evalHDS* and standard deviation.

formance when the number of segments is not specified.

Overall *HAPS* appears to perform better than, or comparably to, the more informed baselines, and much better than the baseline not given information about the number of segments. The results of *HAPS* are different with 95% confidence when compared to the uninformed *Bayesian Segmenter* baseline, but the differences with the more informed baselines are not significant.

Looking at Table 5.2, it is somewhat surprising that using a more elaborate metric of textual similarity, *STS-2012*, results in a palpable drop of performance. This is not entirely surprising because the metric was not designed for topical segmentation per se. Because it is rather complex, it is hard to pin point what exactly causes this drop in performance. However, we hypothesize that this may be due to the fact that overall values of *STS-2012* similarity are significantly higher than those of cosine similarity. This is because the metric leverages considerably more information about each sentence. In a way, everything is 'more similar' when looking at it from the lens of *STS-2012*, especially considering that in our case we are comparing sentences and paragraphs coming from the same source document. For example, because the sentences come from the same source, stylistically they all are likely to be very similar and stylistic similarity is a factor for *STS-2012*. This probably results in overestimating topical similarity in many cases and ultimately, in worse segmentations.

5.5 Evaluating informativeness of the segment centres

Unlike the baselines segmenters, *HAPS* also identifies segment centres for each segment and sub-segment. A segment centre provides a succinct summary of all other sentences or paragraphs in the segment. We decided to investigate if segment centres identified by *HAPS* could be used as summaries. To this end, we compared the segment centres from the second (middle) level of the topical trees to automatically created summaries built using a recent automatic summarizer *CohSum* [Smith et al., 2012]. Thorough evaluation of summaries is notoriously difficult and the results are preliminary. However, they suggest that *HAPS* segment centres are rather indicative of the contents of the document.

To compare the quality of the segment centres identified by *HAPS* and that of the summaries created by *CohSum* we used the *ROUGE* metric [Lin, 2004].

Data. In this experiment we used two datasets. The first dataset is the hierarchical *Moonstone* dataset. The second dataset consists of 20 chapters from several novels by Jane Austen (further *Austen* corpus). We used chapters from the novels by the same author so that preference values selected on a hold-out set would generalize well for the rest of the corpus.

ROUGE software operates by comparing automatically built summaries against one or more manually written gold-standard ones. Therefore, we needed manually written summaries for both the *Moonstone* and the *Austen* corpora. These gold-standard summaries were downloaded from the SparkNotes website for each individual chapter.[5]

Evaluation metric. The basis of comparing the segment centres and the *CohSum* summaries was the publicly available Perl package *ROUGE*. *ROUGE* is the *de facto* evaluation standard for research on automatic text summarization. *ROUGE* scores are computed by measuring several variations of n-gram overlap between the gold-standard summaries and the candidate summaries produced by automatic summarizers. Effectively, a *ROUGE* score indicates how much lexical overlap there is between the candidate summary and the gold-standard summaries.

The *ROUGE* metric is popular due to its simplicity of use and availability for intermediate evaluations. However, it has several disadvantages. First and foremost, lexical overlap is a rather distant approximation of the information content. While *ROUGE* scores have

[5] www.sparknotes.com/

been shown to correlate with human judgements for news, there exists no such evidence for evaluating summaries of literature. Alternative methods of comparing summaries created out of segment centres with the *CohSum* summaries could be asking human judges to read the summaries, the chapters and to rate them on several counts. However, this would be a rather expensive exercise. For now, we will restrict ourselves to using *ROUGE* and we will explore more thorough methods of evaluating summaries in future work.

In our experiments we used ROUGE-1 and ROUGE-L variants of the *ROUGE* metric. We selected these variants because they are routinely used in the annual competitions of text summarization systems – Text Analysis Conferences (TAC).

Baselines. We compared segment-centres based summaries to the summaries created by the *CohSum* system. We chose this system because it is one of the very few recent summarizers designed for single-document, not multi-document summarization and also because of its competitive performance.

CohSum attempts to identify the most informative sentences in a document by taking into account not only lexical information, but also the information about coreference links between sentences. Sentences are treated as nodes in a graph, while coreferential expressions are treated as in-coming links. The system uses a version of PageRank algorithm [Brin and Page, 1998] to identify sentences that are central to the document.

Experimental design. In this experiment we used segment centres from the middle level in the topical trees. This choice is due to the fact that the bottom layer contains to many segment centres and that the top layer may contain too few. This approximately corresponds to the 10% compression rate that we used for the *CohSum* system. The system does not require fine-tuning parameters though it may benefit from a corpus to improve its estimation of inter-sentence similarities. We used the corpus based on Mary Shelley's *Frankenstein* that was provided with the distribution. It is possible that a larger corpus would improve the results.

Results. Table 5.3 shows the results of comparing *CohSum* summaries to the summaries composed of segment centres identified by *HAPS*. While keeping in mind why *ROUGE* is not a conclusive metric for evaluating summaries, it appears that the two types of summaries are rather comparable. Both ROUGE-1 and ROUGE-L scores are slightly higher for segment-based summaries of the *Moonstone* corpus. The situation is reversed on the *Austen* corpus. However, the differences are very small. We interpret this as a sign that

	Moonstone corpus		Austen corpus	
	ROUGE-1	ROUGE-L	ROUGE-1	ROUGE-L
Segment centres	0.341 (0.312, 0.370)	0.321 (0.298, 0.346)	0.291 (0.272, 0.311)	0.301 (0.293, 0.330)
CohSum summaries	0.294 (0.243, 0.334)	0.269 (0.226, 0.306)	0.305 (0.290, 0.320)	0.307 (0.287, 0.327)

Table 5.3: *HAPS* segment centres compared to *CohSum* summaries: ROUGE scores and 95% confidence intervals

segment centres identified by *HAPS* are rather indicative of the contents of the input documents.

5.6 Conclusions

In this Chapter we presented an algorithm for hierarchical segmentation of documents. Given a sequence of n data points (sentences or paragraphs), *HAPS* takes as inputs the value L (the desired number of levels in the resultant tree), a matrix of pairwise similarities between data points and a preference value for each data point at each level. It outputs segment boundaries and segment centres for each level of the tree.

We could not find another publicly available hierarchical segmenter to compare with *HAPS*. Therefore, we compared the system with the results of running flat segmenters iteratively. The baselines that we compared *HAPS* to were designed for situations when the number of segments to be found is known in advance – a rather unrealistic requirement for building hierarchical structures. *HAPS* does not require this information and through the use of preferences offers a level of abstraction when it comes to granularity of segmentation. We think that this is a significant advantage.

Because the baselines were designed to "know" the desired number of segments in advance, we compared *HAPS* to the trees built with and, when possible, without that information. The topical trees created by *HAPS* are comparable or slightly better than the trees built by considerably more informed baselines. The *HAPS* trees are significantly better than the trees created by the *Bayesian Segmenter* when it had no access to the information about the number of segments at each level of the tree.

An additional advantage of using *HAPS* is that it also outputs segment centres. This

effectively turns the topical trees into loose outlines, though it is rather likely that oftentimes they may be not very coherent.

A Java implementation of *HAPS* will be made publicly available (it is currently available upon request).

The *HAPS* segmenter also has several shortcomings and there is a lot of room for improvement. One concern is the fact that the objective function is very centred on the segment centres and on maximizing similarity between them and the data points found in each segment. This function does not take into account intra-segment similarity between data points as such, nor does it pay any attention to dissimilarity between segments. In future work, it would be good to modify the objective function so as to take that information into account. This may be especially useful for situations when the data points correspond to short sentences, *e.g.,* utterances in dialogues. In such situations, it may be difficult to find sentences that are descriptive of the whole segment in isolation if no other information is taken into account.

Other possible improvements are on-going. For now, during the fine-tuning stage *HAPS* selects the best parameters by exhaustively searching the hypothesis space. This is rather crude, as only a very small search space can be covered in this manner. It is almost certain that the performance would be improved through using a more intelligent way to set parameters. We plan to do this in the near future.

From the definition of *HAPS* in Section 5.2, it can be seen that *HAPS* offers enough flexibility to use different similarities for different levels of segmentation. In our current experiments, we have not explored this capability and used the same similarity matrix for all levels. It may be more intelligent to measure similarity differently for coarse-grained and for fine-grained segmentations.

In a similar vein, we have not explored the flexibility of *HAPS* with regards to the preference values. For each data point, at each level, the preference value specifies *a priori* belief about how likely that data point is to be chosen as a segment centre at that level. It would make sense to give higher preference values to sentences that are longer, or, possibly, to sentences that have high concentration of character mentions, to name just a couple of heuristics. Choosing good candidate segment centres may be a research area in and of itself. We have not touched upon it in this work, but we plan to do so in the future.

A problem that was outside the scope of this dissertation is evaluation of hierarchical

segmenters. In this research, we did our best to obtain an adequate picture of the *HAPS* performance and to compare it to the baselines. However, the metrics that we have used are not for tree-like structures by their nature, though they have been adapted to be used on trees and not only on sequences. Neither *windowDiff* nor *evalHDS* takes into account similarities and differences across different levels of the tree. Given our analysis of the hierarchical *Moonstone* corpus, such distinctions should be taken into account since even for manual segmentations a lion share of disagreements is not about the presence/absence or about the precise location of a boundary, but about the prominence of a boundary in a topical tree.

To the best of our knowledge, apart from [Eisenstein, 2009] *HAPS* is the only publicly available hierarchical segmenter. We think that hierarchical segmentation is a step up from linear segmentation in terms of informativeness and usability.

Chapter 6

Coreferential similarity

6.1 Introduction

In Chapters 4 and 5 we have described a flat and a hierarchical model for topical segmentation based on Affinity Propagation. Both algorithms rely on an already pre-computed matrix of similarities between sentences. So far we have used two methods of measuring to what extent two passages are "about the same thing": cosine similarity using a bag-of-words representation and a more competitive measure of similarity, that used in the winning system in the Semeval Semantic Textual Similarity track in 2012 (we refer to it as *STS-2012* similarity). Both metrics use explicit surface information about tokens in the text to measure how similar two sentences are. In this chapter, we try to improve upon these methods of measuring similarity by taking into account referential expressions.

Most work on topical segmentation today revolves around the idea that if the topic of discourse changes, so does the vocabulary [Youmans, 1991]. Indeed, by and large, introducing new concepts almost necessarily requires that the concepts be named and described. However, how densely the concepts are explicitly mentioned, how often the mentions are repeated depends to a large degree on the genre and on the cognitive complexity of the document. In scientific papers or legal documents clarity is paramount and the author goes to lengths stating things explicitly and avoiding ambiguity. However, the less complicated the document, the less it is necessary to explicitly repeat terminology. In literature, for example, word repetition is not only uncommon, it is usually a sign of bad writing. In casual conversations, the topic can easily be never mentioned explicitly. How can we identify

Figure 6.1: An excerpt from *The Moonstone*, Chapter 5.

[He]$_{ent1}$ came back with a character that closed the doors of all his$_{ent1}$ family against him$_{ent1}$, my lady (then just married) taking the lead, and declaring (with Sir John's approval, of course) that [her brother]$_{ent1}$ should never enter any house of hers. There was more than one slur on [the Colonel]$_{ent1}$ that made people shy of [him]$_{ent1}$; but the blot of [the Diamond]$_{ent2}$ is all I need mention here.

It was said [he]$_{ent1}$ had got possession of [his Indian jewel]$_{ent2}$ by means which, bold as [he]$_{ent1}$ was, [he]$_{ent1}$ didn't dare acknowledge. [He]$_{ent1}$ never attempted to sell [it]$_{ent2}$–not being in need of money, and not (to give [him]$_{ent1}$ his due again) making money an object. [He]$_{ent1}$ never gave [it]$_{ent2}$ away; [he]$_{ent1}$ never even showed [it]$_{ent2}$ to any living soul. Some said [he]$_{ent1}$ was afraid of [its]$_{ent2}$ getting [him]$_{ent1}$ into a difficulty with the military authorities; others (very ignorant indeed of the real nature of [the man]$_{ent1}$) said [he]$_{ent1}$ was afraid, if [he]$_{ent1}$ showed [it]$_{ent2}$, of its costing [him]$_{ent1}$ [his]$_{ent1}$ life.

topical shifts in a document whose author does not "hold the reader's hand"?

It turns out that lexical cohesion (or, put simply, word repetition) is only one of several devices of cohesion [Halliday and Hasan, 1976, p. 29]. Other possibilities are reference, substitution, ellipsis and conjunction. In this chapter we attempt to leverage referential cohesion in order to estimate to what extent the topic remains constant between two sentences.

Figure 6.1 contains a short excerpt from *The Moonstone*. This example is annotated for the presence of the two main entities that are the focus of the conversation – Colonel John Herncastle and the Moonstone. The name of the former is never mentioned, he is referred to mostly as *he*, *the Colonel*, *the brother* and *the man*. The text is highly cohesive and talks almost exclusively about the Colonel and the jewel, yet there is almost no lexical repetition. Both *APS* and *HAPS* use only lexical information and would not be able to establish and leverage topical similarity between sentences in this example (the *Minimum Cut Segmenter* and the *Bayesian Segmenter* would have the same problem). The same is true of the second entity: the Moonstone is referred to as *the Diamond*, *the jewel* and *it*, and so a segmenter relying only on lexical repetition would not be able to make much progress.

In general, a speaker/writer uses explicit vocabulary only to the extent that is necessary

for the message to get across and avoids unnecessary redundancy.[1] If we replaced every instance of anaphoric reference to *ent1* – Colonel John Herncastle – by the explicit mention, the result (shown in Figure 6.2) is annoying and is in fact difficult to understand.

Figure 6.2: An excerpt from *The Moonstone*, Chapter 5 with all references to *Colonel John Herncastle* made replaced by *Colonel John Herncastle*.

Colonel John Herncastle came back with a character that closed the doors of all *Colonel John Herncastle*'s family against *Colonel John Herncastle*, my lady (then just married) taking the lead, and declaring (with Sir John's approval, of course) that *Colonel John Herncastle* should never enter any house of hers. There was more than one slur on *Colonel John Herncastle* that made people shy of *Colonel John Herncastle*; but the blot of *the Moonstone* is all I need mention here.

It was said *Colonel John Herncastle* had got possession of *the Moonstone* by means which, bold as *Colonel John Herncastle* was, *Colonel John Herncastle* didn't dare acknowledge. *Colonel John Herncastle* never attempted to sell *the Moonstone* –not being in need of money, and not (to give *Colonel John Herncastle* his due again) making money an object. *Colonel John Herncastle* never gave *the Moonstone* away; *Colonel John Herncastle* never even showed *the Moonstone* to any living soul. Some said *Colonel John Herncastle* was afraid of *the Moonstone* getting *Colonel John Herncastle* into a difficulty with the military authorities; others (very ignorant indeed of the real nature of *Colonel John Herncastle*) said *Colonel John Herncastle* was afraid, if *Colonel John Herncastle* showed *the Moonstone* , of its costing *Colonel John Herncastle* his life.

Lexical repetition is lower in texts that are cognitively less dense – fiction, every day conversations, emails, etc. In technical texts, on the other hand, lexical repetition is necessary to ensure comprehension of concepts which are relatively involved. Since this research project is concerned with literary data, the question arises how we can measure topical similarity between sentences/paragraphs if lexical repetition does not seem to be too promising a source. How can we make use of cohesion devices other than repetition, for example the devices used in Figure 6.1?

An obvious answer would be to run an anaphora resolution system, and hopefully resolve that *he* and *the Colonel* refer to *John Herncastle* and *it* and *the jewel* refer to the Moonstone. However, not only are anaphora resolution systems not very reliable yet, but

[1] See [Hoey, 1991] for a discussion of how lexical density relates to cognitive complexity.

also it would be somewhat of an overkill. Anaphora resolution is generally a rather complicated task requiring a lot of pre-processing. On the other hand, topical segmentation is a pre-processing task itself and running a complex system just to aid in pre-processing is inefficient. Additionally, it is known that anaphoric connections are restricted by the topical structure of a text [Kehler, 2002, p. 143–180], making the problem circular.

So, how can we make use of the pronoun *he* in Figure 6.1 without resolving it? Well, we can safely assume that if an anaphoric expression is present (be it pronominal or nominal), it must refer to something. In other words, it almost necessarily signals a reference to a previously mentioned entity – or to an entity somehow clear from the context.[2] Therefore, for the purposes of establishing topical continuity, we do not need to resolve *he* in Figure 6.1. By the virtue of simply encountering it we know that it refers to a nearby entity.

On the other hand, the noun phrase *the Colonel* also refers to something, and so does *the man*. How can we know whether the antecedent of a referring expression happens to occur in the preceding sentence, or further behind? It turns out that the syntactic type of an anaphoric expression can tell us something about how accessible its referent is. In general, the less information is encoded in an anaphoric reference, the more accessible the referent is [Givón, 1983]. For example, the referent of *he* should be more accessible than the referent of a proper noun phrase *the Diamond*. The notion of accessibility is complex and is influenced by the context, prominence and the number of potential antecedents, and several other factors. However, in written text, the distance between the referent and the antecedent is of primary importance. If we make a somewhat simplistic assumption of the other qualities being held equal, we can reasonably expect that the form of an anaphoric expression is correlated with how far its antecedent is. By using this information, we can model topical similarity of text using anaphoric expressions without actually resolving them.

The idea that the type of referring expression tells a lot about the accessibility of its antecedent dates back to Givón [1981]. He postulated that the more informative the referring expression is, the less accessible the antecedent will be. Figure 6.4 shows the list of expressions from the least to the most informative. Projecting this information onto our

[2]There are, naturally, cases when anaphoric expressions refer to entities clear from the context and never mentioned explicitly. There also are the issues of cataphora and pronouns used non-referentially. However, since we are mostly concerned with processing written text, we will assume these cases are a minority and will not deal with these issues here, however interesting they are.

task, we can say that the more informative the expression is, the less continuity there will be in the topic.

In this Chapter we show how such information can be used to improve the quality of text segmentation. We extract NPs and classify them by informativeness. This is achieved with the help of a syntactic parser, but a lighter form of processing might be sufficient, perhaps even if it is only captured personal pronouns. Using this information, we augment and correct a matrix of lexical similarities between sentences, a structure used by *APS*, *HAPS* and the *Minimum Cut Segmenter*.

We evaluate the results of using coreferential similarity for both flat and hierarchical segmentation for *APS*, *HAPS* and the *Minimum Cut Segmenter*. The results suggest that the new matrix never hurts, and in several case improves, the performance of the segmenter, especially for the novel. We also check whether this metric would still be useful if instead of the traditionally used lexical similarity we used a similarity metric which took synonymy into account. In this case, the margin of improvement is lower, but still the coreferential similarity metric never hurts the performance and often improves it.

This chapter is structured as follows. Section 6.2 briefly reviews similarity metrics used for the purposes of topical segmentation and introduces Talmy Givón's theory of topic accessibility, which is the basis of the similarity metric proposed here. Section 6.3 describes our similarity metric and how we compute it. Section 6.4 shows the details of the experiments, while Section 6.5 discusses the results. We conclude in Section 6.6 with a discussion of how our metric can be improved and simplified.

6.2 Background: accessibility of antecedents and coreferential distance

This section reviews metrics of textual similarity used for the purposes of topical segmentation and introduces the relationship between topic accessibility and coreferential distance.

Cosine similarity metric described in Chapter 4 and used by *APS*, *HAPS* and the *Minimum Cut Segmenter* is by far the most popular way to gauge topical similarity between sentences. While cosine similarity between vectors is easy to compute, it is hardly a reliable metric of topical similarity. Several researchers have used *lexical chains* – first introduced

by Halliday and Hasan [1976] – to improve the performance of topical segmenters. It can be thought of as a set of related words following the head word introducing the chain. The intuition behind using lexical chains for text segmentation is that the beginning and end of a chain tend to correspond with the beginning and end of a topically cohesive segment. One version of TextTiling [Hearst, 1997] measures similarity through lexical chains that are manually constructed using Roget's Thesaurus. Okumura and Honda [1994] apply automatically created lexical chains to segment a small set of documents in Japanese. More recently, Marathe [2010] tried to build lexical chains using distributional semantics and to apply the method to text segmentation.

Other proposals to move beyond word repetition in topical segmentation include the use of bigram overlap in Reynar [1999], information about collocations in [Jobbins and Evett, 1998], LSA [Landauer and Dumais, 1997] in [Choi et al., 2001, Olney and Cai, 2005] and WordNet in [Scaiano et al., 2010].

Overall, a number of approaches have been proposed to measure cohesion between sentences, that is to say, to describe to what extent a pair of sentences is "about the same thing". Most of them have a common denominator: they use explicit lexical information, sometimes augmented by semantic relations from thesauri or ontologies.

Lexical resources, such as ontologies and knowledge-bases, may help improve the quality of segmentations, but such resources are not always available. They also may cause problems with precision. More importantly, however, they do not solve a more fundamental problem: a text may be highly cohesive and coherent without being tightly bound by either lexical cohesion or synonymy. Figure 6.3 shows a fragment of the play *Waiting for Godot* by Samuel Beckett. In this example, the focal entity – a person named Godot is only explicitly mentioned by his name once. The conversation is tightly centred around this person, yet precisely because the focus is so clear he is referred to mostly through the usage of pronouns. How could we leverage this information?

Simplifying matters somewhat, we can say that noun phrases (further *NPs*) containing a lot of information (such as lengthy definite noun phrases or proper names) tend to refer to entities that are not very accessible from the recent context while short " light" NPs (*e.g.,* personal pronouns) tend to have antecedents that are readily accessible [Sanders and Gernsbacher, 2004, Ariel, 2004].

Givón [1981] proposes a theory of functional domains. Fundamentally, the theory sug-

Figure 6.3: A fragment of Samuel Beckett's play *Waiting for Godot* (the focal entity, Godot, is mentioned explicitly only once)

VLADIMIR: Let's wait and see what he says.
ESTRAGON: Who?
VLADIMIR: Godot.
ESTRAGON: Good idea.
VLADIMIR: Let's wait till we know exactly how we stand.
ESTRAGON: On the other hand it might be better to strike the iron before it freezes.
VLADIMIR: I'm curious to hear what he has to offer. Then we'll take it or leave it.
ESTRAGON: What exactly did we ask him for?
VLADIMIR: Were you not there?
ESTRAGON: I can't have been listening.
VLADIMIR: Oh . . . Nothing very definite.
ESTRAGON: A kind of prayer.
VLADIMIR: Precisely.
ESTRAGON: A vague supplication.
VLADIMIR: Exactly.
ESTRAGON: And what did he reply?
VLADIMIR: That he'd see.
ESTRAGON: That he couldn't promise anything.
VLADIMIR: That he'd have to think it over.
ESTRAGON: In the quiet of his home.
VLADIMIR: Consult his family.
ESTRAGON: His friends.
VLADIMIR: His agents.
ESTRAGON: His correspondents.
VLADIMIR: His books.
ESTRAGON: His bank account.
VLADIMIR: Before taking a decision.

gests that various aspects of language can be viewed as functional domains, meaning that there exist several dimensions (so called *clines*) of explicit linguistic coding that affect a particular aspect of a language. The points on each dimension have a functional relationship with that aspect of the language.

This somewhat abstract linguistic definition can be easily explained by a particular example of a functional domain – topical continuity/discontinuity. One aspect (or a cline, in Givón's terminology) of this functional domain is topic accessibility.[3] A number of coding devices affect this property. They are listed in Figure 6.4, ordered from the devices used to mark the most continuous topics to those marking the least continuous ones.

The order in Figure 6.4 is governed by a simple principle: the more accessible the topic is, the less information is used to code it.

The author argues that the continuum is applicable in many languages. He also mentions that while the exact values of the phenomenon in question are difficult to predict or even estimate, their relative order can be predicted with certainty (even if some devices are

[3]There is a volume devoted to this particular cline, Givón [1983].

Figure 6.4: Linguistic coding devices which signal topic accessibility Givón [1981]

Most continuous (least surprising)

1. zero anaphora
2. unstressed pronouns
3. right-dislocated definite noun phrases (NPs)
4. neutral-ordered definite NPs
5. left-dislocated definite NPs
6. Y-moved NP's
7. cleft/focus constructions
8. referential indefinite NPs

Least continuous (most surprising)

unavailable in some languages).

In a similar vein, Ariel [2014] groups non-initial noun phrases into expressions with *low accessibility* (definite NPs and proper names), those with *intermediate accessibility* (personal and demonstrative pronouns) and those with *high accessibility* (pronouns).

In this work we attempt to leverage this information for the purposes of text segmentation. We hypothesize that upon encountering a referring expression (further *RE*) we can estimate how likely it is that it refers to something in the recent context. If a *RE* is a personal pronoun, *e.g., he*, we may be fairly certain that this sentence continues the thread of discussion about the 'he' clear from the context. If we encounter a lengthy modified NP, it is quite likely that it re-introduces a concept that was out of focus for a while.

Subsequent research on the topic has substantially deepened the role that each of the devices plays in topic continuity (and other phenomena), *e.g.,* [Bestgen and Costermans, 1994, Vonk et al., 1992, Sanford et al., 2004]. However, the ordering in Figure 6.4 provides a good starting point for studying the role of syntactic coding in topic accessibility. We use it as an inspiration to propose a syntactically motivated measure of topical similarity in running text.

In order to leverage this information for text segmentation, we extract noun phrases from sentences and classify them roughly into categories shown in Figure 6.4. When computing similarity between sentences, we take into account how many instances of each RE type a sentence contains and consider those counts in our calculations (see Section

6.3). This new matrix, the matrix of coreferential similarities is added to a more conventional matrix of lexical similarity. With this augmented model, we segment fiction and spoken lecture transcripts, the two types of data where low rates of lexical cohesion preclude achieving segmentation of good quality using only surface information about token types.

6.3 Estimating coreferential similarity

Most of the recent segmenters [Du et al., 2013, Eisenstein and Barzilay, 2008, Kazantseva and Szpakowicz, 2011, Malioutov and Barzilay, 2006] use only surface information about tokens encountered in the document. Probabilistic segmenters [Du et al., 2013, Eisenstein and Barzilay, 2008] model documents as a sequence of tokens generated by an underlying graphical model. Similarity-based segmenters, such as *APS* and *HAPS* or the *Minimum Cut Segmenter* create a vector space representation. They compute similarity between sentences using bag-of-words vectors and then measure cosine similarity between the vectors.

Continuing to work with the similarity-based representation, we augment a conventional matrix of lexical similarity with information about referential expressions to see if this information is helpful for topical segmentation.

The proposed similarity metric relies on the idea that in order to measure how many concepts are shared between two sentences we do not need to fully resolve anaphoric expressions but only to map them to sentences that contains their most recent antecedent (without actually naming the antecedents).

In order to do that, we parse the documents using the Connexor parser [Tapanainen and Järvinen, 1997] and extract all noun phrases with their constituents. Next, we attempt to classify the NPs into categories that would roughly correspond to those listed in Figure 6.4 and to Ariel's categories [Ariel, 2014].

Since fiction is mostly concerned with describing people and societal relations and problems, in our categorizations we distinguish between *animate* and *inanimate* entities. A manual study by Brown [1983] suggests that the average referential distance for animate and inanimate entities differs widely within the same document. Therefore, it makes sense to distinguish between these two types. In datasets other than the *Moonstone* and fiction (novels from Project Gutenberg) this distinction is not so important because the discussion

Figure 6.5: Categories of noun phrases taken into account when computing coreferential similarity

1. animate personal pronouns (*he, she, they*)
2. inanimate pronouns (*it*)
3. demonstrative pronouns (*that, those*)
4. animate proper names (*John Herncastle*)
5. inanimate proper names (*London*)
6. animate definite noun phrases (*the man*)
7. inanimate definite noun phrases (*the jewel*)
8. animate indefinite noun phrases (*a man*)
9. inanimate indefinite noun phrases (*a jewel*)

rarely focuses on people.

So finally we classify each identified noun phrase into one of the categories specified in Figure 6.5. The list is not exhaustive and in some cases a noun phrase may belong to more than one type (in practice, however, a NP is always assigned a single type based on the implementation).

Equation 6.1 shows the formula used to estimate coreferential similarity between two sentences S_i and S_j. We settled on this variant after some preliminary experimentation. Coreferential similarity estimated this way has an interpretation similar to lexical similarity measured using the cosine measure. The numerator estimates how many concepts are shared between sentences S_i and S_j by counting referential expressions of each type and weighting the number of occurrences so as to favour the least informative REs (*e.g.,* pronouns). The denominator normalizes this value by the dot product of the lexical vectors of both sentences. Since the matrix of similarities obtained this way is eventually added to the underlying matrix of lexical similarities, we tried to come up with a measure that has an intuitive interpretation and naturally combines with the basic lexical similarity.

$$coref\_sim(S_i, S_j) = \left(\frac{\sum_{t=0}^{|T|} count_t^{S_j} \times weight_t}{|S_1| \times |S_2|}\right)^{(j-i-1)\times decayFactor} \tag{6.1}$$

T is the set of of all types of referring expressions which we consider – those given in Figure 6.5. $count_t^{S_j}$ is the number of times an expression of type t appears in the most recent sentence, S_j. Note that we only consider the referring expressions in the most recent

sentence because a referring expression, by its nature, must refers to something previously mentioned. The "tightness" of the link is controlled by setting $weight_t$ for each expression type t. $weight_t$ effectively specifies how likely it is that the antecedent for an expression of a type t appears in sentence s_i. The values of the weights are set experimentally on the holdout data. They can almost certainly be further fine-tuned. Intuitively, the settings of the weights reflect the logic behind Givón's theory. Consider an example vector of weights for expressions, where a higher weight corresponds to a more accessible antecedent (for animate and inanimate entities respectively).

> <personal_pronouns_anim: 4, demonstr_pronouns_anim: 2,
> proper_names_anim: 1, def_np_anim: 0.5, indef_np_anim: 0,
> pronouns_inanim: 2, demonstr_pronouns_inanim: 2, proper_names_inanim: 0,
> def_np_inanim: 0, indef_np_inanim: 0>

The actual values of the weights are provided in the Appendix C.

The denominator of Equation 6.1 normalizes the value by the product of the lengths of sentences S_1 and S_2. The exponent $(j-i-1) \times decayFactor$ is responsible for decreasing similarity as the distance between sentence S_i and S_j increases. The decay factor, $0 < decayFactor < 1$, is set experimentally, and $j-i$ is the distance between sentences S_i and S_j, $i < j$.

Figure 6.6 contains a walk-through example of computing referential similarity between two sentences.

The coreferential similarity as defined by Equation 6.1 is rather limited. The first limitation is the range: it can only measure similarity between nearby sentences or paragraphs, because it only makes sense between the closest occurrences of an antecedent and a subsequent referring expression. For example, it does not make sense to measure coreferential similarity between sentences that are several paragraphs apart. Even if they indeed talk about the same entities, the topic has most likely been re-introduced several times in between. That is why we only compute coreferential similarity for sentences no more than $decayWindow$ sentences apart. The value of $decayWindow$ is usually between 2 and 6 and it is set experimentally on the holdout set for each corpus.

Another issue with coreferential similarity is that it is a non-transitive metric (as well as non-symmetric).

Figure 6.6: An example of computing coreferential similarity

$$coref\_sim(S_i, S_j) = \left(\frac{\sum_{t=0}^{|T|} count_t^{S_j} \times weight_t}{|S_1| \times |S_2|}\right)^{(j-i-1)\times decayFactor}$$

S1: "At the sands, of course!" says Nancy, with a toss of her head.
S2: "She had another of her fainting fits this morning, and she asked to go out and get a breath of fresh air."

Expression counts:	Weights:
personal_pronouns_anim: 2 (she, she)	4
demonstr_pronouns_anim: 0	2
proper_names_anim: 1	1
def_np_anim: 0	0.5
indef_np_anim: 0	0
pronouns_inanim: 0	2
demonstr_pronouns_inanim: 1	2
proper_names_inanim: 0	0
def_np_inanim: 2 (this morning, fainting fits)	0
indef_np_inanim: 1 (a breath)	0

$$coref\_sim(S_2, S_1) = \frac{2 \times 4 + 1 \times 1 + 1 \times 2}{21 \times 22}^{(2-1-1)\times 0.5} = 0.0234$$

Consider the following pair of example:

1. *1A:* He read a book. *1B:* The book is good. *1C:* Books in general are good.

2. *2A:* He read that book. *2B:* He liked it. *2C:* It was about the War of 1812.

In the first example, the lexical similarity component (cosine similarity or STS similarity) will be able to establish that 1A is like 1B, 1B is like 1C and that 1A is like 1C. On the other hand, the coreferential similarity component cannot directly establish it for the example 2 because there is no way to guarantee that *it* in 2C refers to the same entity as in 2B. We attempt to circumvent this problem by using $decayFactor$ and gradually decreasing similarity between a pair of sentences the farther away they are.[4]

[4]We also tried adjusting *coreferential similarity* using the values of lexical similarity for the same pairs of sentences, *e.g.,* $\hat{coref\_sim}(s_i, s_j) = coref\_sim(s_i, s_j) \times lexical\_sim(s_i, s_j)$. However, experimentally this heuristic led to no improvements.

Measuring cosine similarity between paragraphs. The metric was specifically developed to improve the quality of segmentations on literary data. However, the two literary corpora used in this work – *the Moonstone* corpora and the fiction corpus – both are segmented at the paragraph level instead of the sentence level because of their excessive length. Coreferential similarity between paragraphs is computed slightly differently:

$$coref\_sim(p_i, p_j) = \left(\frac{\sum_{t=0}^{|T|} count_t^{p_j} \times weight_t}{|p_1| \times |p_2|}\right)^{(j-i)\times decayFactor} \quad (6.2)$$

In this case, $count_t^{p_j}$ refers to the number of occurrences of expression of type t in the first $paragraphCutOff$ sentences of the paragraph p_j, instead of the whole paragraph. The rationale behind this heuristic is that the referring expressions in the opening sentences of the paragraph are likely to refer to entities from the previous paragraph while expressions in the middle or the end of the paragraph are likely to refer to entities that were introduced inside the paragraph.

The values of $coref\_sim$ are usually quite small and the information it uses is rather one-sided. That is why we use it in addition to, not instead of, the lexical similarity. In our experiments, we first compute lexical similarity between sentences (or paragraphs) and then modify the lexical matrix by adding the matrix of coreferential similarity to it.

6.4 Experimental evaluation

In order to test the effectiveness of coreferential similarity metric we performed a set of experiments that compared how much it improves the quality of topical segmentations. To this end, we ran *APS*, *HAPS* and MinCutSeg with and without adding coreferential similarity to lexical similarity and compared the results. We chose these segmenters for comparison because $coreferential\_similarity$ can only be naturally incorporated into a similarity-based segmenter (the second baseline segmenter, *Bayesian Segmenter*, is a based on a generative model and could not be used in these experiments.)

Data. For flat segmentation we used the flat *Moonstone* dataset and the AI lectures dataset. Unfortunately we could not include the fictions dataset and the clinical dataset in these experiments.

For the fiction dataset, the sheer size of novels made parsing them and then manipu-

lating the resulting XML files too time-consuming. For the time being, the only implementation of co-referential similarity is that based on the Connexor parser [Tapanainen and Järvinen, 1997]. The parser outputs results in XML files. It cannot be directly incorporated into the processing pipeline because of the licensing restrictions and, therefore, files have to be written to the disk and then reloaded on a different machine. For novels this simply takes too long. In the future, we plan to implement coreferential similarity using the Stanford Core NLP suite,[5] eliminating the need to deal with files containing parses of the complete document.

We also had to exclude the clinical dataset from these experiments. The problem with this dataset is that it is only available in an already pre-processed form: the documents are lemmatized, turned to lower case and stripped of punctuation. The Connexor parser cannot handle this input.

For hierarchical segmentation, we used the hierarchical Moonstone corpus as well as the hierarchal Wikipedia corpus.

Metrics. For evaluation, use used the same metrics as when evaluating the corresponding type of segmenter. In the flat segmentation experiments, the segmenters were evaluated using *windowDiff*. In the hierarchical segmentation experiments, we report *windowDiff* per level and the values of *evalHDS*.

Experimental setup. Each document was parsed with the Connexor parser and the noun phrases were extracted and labelled using the output of the parser. The reason for using Connexor parser is that it provides high-quality partial parses even when it cannot parse the full sentence – which is important in our setting because we do not need a complete parse tree.

The drawback of using the Connexor parser is that it is proprietary. In the future work, we plan to adapt the code to use one of the publicly available parsers or even a lighter form of processing, avoiding parsing altogether.

In order to capture named entities, we used Stanford Named Entity Recognizer, a part of the Stanford CoreNLP suite [Finkel et al., 2005]. We also used the coreference module from that software to tag NPs as animate or inanimate [Raghunathan et al., 2010].

Segmenters and settings. In order to see if coreferential similarity improves the quality of segmentations we ran *HAPS*, *APS* and the *Minimum Cut Segmenter* with and without

[5] `http://nlp.stanford.edu/software/corenlp.shtml`

augmenting the basic matrix of lexical similarity with the information about coreferential similarity. We also used two different flavours of the underlying matrices of lexical similarity: simple cosine similarity and *STS-2012* similarity matrices.

In the experiments on flat segmentation, we used the best parameters obtained in Chapter 4 as a guide to narrow down the search space.

In addition to selecting the parameters for lexical similarity and for the *APS* segmenter, we also needed to set the values of parameters for coreferential similarity. Those are $decayWindow$, $decayFactor$ and $weight_t$, $t \in \{T\}$ where {T} is the set of all types of referential expressions under consideration (the list of expressions in Figure 6.5). To set these parameters we held out 2 files from each dataset to manually set them following the logic behind Figure 6.4: the less information a certain RE contains, the higher its weight. The weights were set separately for animate and inanimate referential expressions.

When running the *Minimum Cut Segmenter*, we use a matrix identical to the one used by *APS* (or, for the experiments in hierarchical segmentation, the matrix identical to the one used by *HAPS*). Because of this, we did not need (and could not use) the script for finding the best parameters included with that segmenter. Providing a precomputed matrix eliminated the need to select most parameters. Still there were several which we set to default values. It is possible that more careful fine-tuning would improve the results slightly.

In the hierarchical segmentation experiments the procedure was quite similar. For the hierarchical *Moonstone* dataset, we used the same parameters for coreferential similarity as were used for the flat *Moonstone* dataset. The parameters for coreferential similarity for *Wikipedia* dataset were set on a holdout set of two articles.

The results are reported using cross-validation. Just as before, each dataset was randomly split into folds and within each fold at most three files were used for finding the best values of parameters (parameters for lexical similarity and also parameters for *APS* and *HAPS*). The rest of the files in the fold were used for testing. The AI lectures and the *Moonstone* datasets are quite small, so we used 5-fold cross-validation on these datasets. The Wikipedia dataset is larger, so we use 10-fold cross-validation. The hierarchical *Moonstone* dataset is the smallest of all and we used only 4 folds.

	AI Lectures	*Moonstone*
APS	0.420 (± 0.014)	0.441 (± 0.075)
APS-$coref\_sim$	0.411 (± 0.025)	0.391 (± 0.060)
APS-STS	0.428 (± 0.049)	0.479 (± 0.041)
APS-STS-$coref\_sim$	0.429 (± 0.020)	0.478 (± 0.035)
MCSeg	0.431 (± 0.045)	0.470 (± 0.095)
MCSeg-$coref\_sim$	0.410 (± 0.060)	0.413 (± 0.030)
MCSeg-STS	0.451 (± 0.023)	0.441 (± 0.051)
MCSeg-STS-$coref\_sim$	0.433 (± 0.070)	0.430 (± 0.025)

Table 6.1: Results of comparing *APS* and *Minimum Cut Segmenter* using four different matrix types (*windowDiff* values and standard deviation)

6.5 Experimental results and discussion

Tables 6.1 and 6.2 show the results of experiments for flat and hierarchical segmentations correspondingly.

Table 6.1 presents the results of running *APS* and the *Minimum Cut Segmenter* using four different input matrices each. The first column shows the combination of the name of the segmenter and the specific input matrix. *APS* and *Minimum Cut Segmenter* refer to the cases where both segmenters were run using simple cosine similarity matrices. STS refers to matrices computed using *STS-2012* from the *DKPro Similarity* framework. $coref\_sim$ refers to cosine similarity matrices modified by adding a matrix with coreferential similarities. STS-$coref\_sim$ are matrices computed using *STS-2012* which had coreferential similarity added to them.

In all experiments, we set the weights for different types of referring expressions on two hold-out files. The remainder of the data is divided into five folds. Standard deviation reported in the Tables is computed across folds.

Coreferential similarity improves the results of the cosine matrix for both segmenters, but the improvement on the AI dataset is rather small (1% for *APS* and 2% for *Minimum Cut Segmenter*).

It is interesting to see that in most cases using STS matrices slightly hurts the performance of the segmenters compared to using simple cosine similarity matrices. The only exception is running *Minimum Cut Segmenter* on the *Moonstone* dataset which improves the performance by 3%.

	Wikipedia	*Moonstone*
HAPS	0.45 (± 0.015)	0.353 (± 0.072)
HAPS-$coref\_sim$	0.429 (± 0.071)	0.330 (± 0.07)
HAPS-STS	0.500 (± 0.023)	0.359 (± 0.083)
HAPS-STS-$coref\_sim$	0.462 (± 0.015)	0.359 (± 0.016)
MCSeg	0.444 (± 0.002)	0.377 (± 0.023)
MCSeg-$coref\_sim$	0.443 (± 0.043)	0.365 (± 0.052)
MCSeg-STS	0.489 (± 0.023)	0.423 (± 0.07)
MCSeg-STS-$coref\_sim$	0.489 (± 0.023)	0.423 (± 0.071)

Table 6.2: Results of comparing *HAPS* and hierarchical iterative version of the *Minimum Cut Segmenter* using four different matrix types (*evalHDS* values and standard deviation)

Adding a matrix of coreferential similarities to STS matrices slightly improves the performance on the *Moonstone* dataset and leaves it practically unchanged on the dataset of AI lectures.

It is somewhat surprising that using *STS-2012* for similarity computation does not improve, and occasionally worsens, the results compared to using simple cosine similarity. Coreferential similarity, on the other hand, produces a small but consistent improvement.

Table 6.2 shows the results of running *HAPS* and the iterative version of the *Minimum Cut Segmenter* on two hierarchical datasets: hierarchical *Moonstone* dataset and the Wikipedia dataset. Similarly, each segmenter was run using four different types of input matrices. The results are reported in the table using *evalHDS* metric. Again, using the *STS-2012* similarity metric worsens the performance for both *HAPS* and the *Minimum Cut Segmenter*.

The results of hierarchical segmentation are rather similar to those of the flat ones. Adding coreferential matrix improves the results of *HAPS* by 2% and 3.8% on Wikipedia dataset. On the *Moonstone* dataset the improvement is smaller: 1.7% for the cosine similarity matrix and 0 for the *STS-2012* matrix.

The performance of the *Minimum Cut Segmenter* segmenter varies less. There is about 1% improvement on both Wikipedia and the *Moonstone* datasets using cosine similarity. There is no improvement when using the *STS-2012* matrices.

Overall it appears that adding the matrix of coreferential similarities to the underlying lexical matrix either slightly improves the performance or leaves it unchanged.

The improvements obtained using coreferential similarity are not statistically signifi-

cant, unfortunately.

6.6 Conclusions

In this Chapter we presented a metric for measuring co-referential similarity between sentences or paragraphs. It uses the presence and information about the specific type of referential expressions to gauge whether the topic has changed or remained constant between a pair of sentences (or paragraphs). We estimate coreferential similarity between nearby sentences by counting the number of referential expressions of each type, assuming that referential expressions containing little information (*e.g.,* pronouns) signal more continuity of topic than elaborate referential expressions (*e.g.,* modified NPs). The matrix of coreferential similarities is added to the underlying matrix of lexical similarities. The resulting matrix is used as an input for flat or hierarchical similarity-based segmenters.

The results of adding the matrix of coreferential similarities, unfortunately, does not significantly improve the resulting segmentations. It does improve them a little, however. What is more important, the results are never worsened. The most perceptible improvement is seen on the flat *Moonstone* dataset. This was anticipated, as in fiction the usage of personal pronouns is very common while repetition of words is generally rare.

The improvements are small or none on the AI lectures dataset and on the Wikipedia dataset. We hypothesize that possibly this is due to the fact that these articles are not about people, but about concepts and entities. Hence, the usage of 'it' and 'they' is far more likely than the usage of personal pronouns. However, 'it' is frequently used non-referentially. This makes the occurrence of 'it' not very useful for our purposes.

Two serious drawbacks of the current implementation of coreferential similarity are its complexity and its reliance on the proprietary parser. However, looking at the examples of weights used for different datasets it becomes apparent that the most useful type of coreferential expression is demonstrative and personal pronouns. These expression types can be easily captured without any need for syntactic parsing at all. In the future we plan to experiment with including such a "light" version of coreferential similarity metric with the *HAPS* Java implementation (for now, the implementation of coreferential similarity is not publicly available due to the fact that it uses a proprietary parser). We also plan to port the current implementation to a freely available syntactic parser, most likely the Stanford

Parser [Socher et al., 2013].

It would also be very interesting to test coreferential similarity on different types of data. The *Moonstone* datasets are an example of literary language. However, it is input into the segmenters at the level of paragraphs, not sentences. This, most likely, makes coreferential similarity much less useful because it only takes into account referential expressions in the opening sentences of each paragraph. Additionally, paragraphs are longer than sentences and the values of lexical similarities can be computed more reliably since there is more tokens in each atomic unit of text.

Ideally, we would like to test coreferential similarity on a dataset of dialogues, such as [Gruenstein et al., 2005] or on a literary dataset where the information about paragraphs is not readily available, *e.g.,* [Brooke et al., 2012]. Unfortunately, the former is not free and the latter is not yet publicly available.

Overall, we proposed a metric of topical similarity that is useful for texts where lexical cohesion is low. While it only improves the results slightly, it never hurts the performance. We hope that a simplified version of coreferential similarity metric may still be useful and plan to continue exploring that direction.

Chapter 7

Conclusions and future work

7.1 Contributions

The work described in this dissertation resulted in three main contributions: 1) the *APS* and *HAPS* algorithms for flat and hierarchical topical segmentation and a Java implementation of both, 2) two high-quality corpora for flat and hierarchical segmentation of literature, 3) a proof-of-concept implementation of a new similarity metric for texts with low lexical cohesion.

The first on the list is a model for flat and hierarchical text segmentation. In this model the document is viewed as a tree of segments, where each segment is characterized by a certain degree of topical unity. The nodes near the root of the tree correspond to the most prominent shifts of topic in the document. The nodes lower down, and especially the leaves of the tree, correspond to minor fluctuations of topic.

In order to build such structures, we modified a recent clustering algorithm, Affinity Propagation, so that instead of clustering it performs hierarchical segmentation. Each segment is described by a segment centre – a data point that best describes the contents of the segment. The algorithm assigns each data point – a sentence or a paragraph – to a segment centre in a such way that the sum of similarities between all segment centres and the points that they exemplify is maximized (the quantity is known as net similarity).

Our system, *HAPS*, takes as input a matrix of similarities between data points and, for each data point and each level, a preference value that describes *a priori* belief of how likely that data point is to be chosen as a segment centre at that level in the topical tree.

The choice of the similarity metric as well as that of the preference values is completely decoupled from the algorithm, so any matrix can be used. Compared to generative models, this is a considerable degree of freedom.

Another significant advantage of *HAPS* is that it does not require knowing the number of segments in advance. The baselines used in this work, as well as other publicly available (flat) segmenters need this information as one of the inputs. While this may be roughly estimated for a one-level segmenter, the number of nodes at each level of the topical tree is very difficult to predict in advance. Of course, *HAPS* also requires some guidance with respect to the granularity of segmentation. This is achieved through the use of preferences, but using preferences allows a considerable degree of abstraction. For example, if one wishes to find relatively fine-grained segments (*e.g.,* the leaves of the topical trees) using the median value of similarities is usually a good start. A value corresponding approximately to the minimum similarity value in the input matrix will result in a medium number of segments. A negative value is appropriate for coarse, top-level segmentations. Compared to supplying the exact number of segments at each level, this again offers considerable freedom.

At the time of writing, *HAPS* is the only publicly available hierarchical segmenter.

The second contribution of this research are the flat and the hierarchical *Moonstone* corpora. In each corpus, each chapter is annotated by 3-6 people: a reasonable number of annotations that enables the study of how people segment literature, where they agree and disagree and what they find difficult. This is quite important. While there are several corpora that were created specifically to study topical segmentation [Gruenstein et al., 2005], most experiments use either synthetic data (*e.g.,* [Choi, 2000]) or metadata created by the authors [Eisenstein and Barzilay, 2008, Carroll, 2010]. While this is suitable for some experiments, not all metadata are created equal. For one thing, metadata (such as markup about sections and subsections of a book) allow only one valid way – that of the author – of viewing the structure of the document. Considering the amount of disagreement between the annotators in our *Moonstone* corpora or other annotated data, *e.g.,* [Gruenstein et al., 2005], there is usually more than one way to interpret the structure of a document. Since we should be concerned not so much with mimicking a single annotator as with producing reasonable segmentations, it is important to have access to more than one example of a valid annotation is important. This is essential in hierarchical segmentation for which very few benchmarks exist, most of them created using metadata. The *Moonstone* corpora – the

annotations as well and the raw outlines – are publicly available.

The third contribution of this work is a proof-of-concept implementation of the coreferential similarity metric. Instead of relying on repetition of open-class words, it takes into account the types of referential expressions found in sentences and leverages that information to estimate to what extent the topic has changed between two nearby sentences. The improvement gained by using this similarity metric is not wide, but it is consistent insofar as the metric never worsens the results – an important fact because such one-sidedness is rarely the case, as can be seen from our relatively elaborate baseline similarity metric (*STS-2012*).

Shortcomings. Nobody and nothing is perfect, and we certainly will not argue with that as far as the results of this research are concerned. We have already listed the main shortcomings of each part of this work in the respective chapters, but we will briefly review them here.

Our main concern about *HAPS* is that the objective function it maximizes (net similarity between segment centres and their children) only takes into account similarity between each data point in the segment and its segment centre. The objective function ignores the amount of similarity or dissimilarity between all points assigned to the same segment. Because of this characteristic, *HAPS* is likely to perform worse when informative segment centres cannot be found – for example at coarse-grained levels of segmentation or on corpora where sentences are very short. To some degree, the segmentation constraint compensates: even if there is no good segment centre for a given data point, it will be assigned to a segment centre that is most suitable for its neighbours (unlike clustering where the assignments are independent of the neighbouring ones). In the future we plan to experiment with incorporating more information about intra-segment similarities into the algorithm.

Out of all the parts of this research, the implementation of coreferential similarity would most benefit from future work. As is, it is too complex to justify the small improvement in results that it achieves. In the future we intend to experiment with using a publicly available parser as well as with avoiding parsing altogether. If we could preserve the desirable properties of this metric, such a light version of it would be a welcome addition to segmentation systems.

7.2 Directions for future work

Flat topical segmentation has been a rather popular topic of research. Hierarchical segmentation, on the other hand, is much less explored. In this work we created a specific hierarchical segmenter, but a lot of possibilities remain unexplored.

To start with, it is unclear how deep and how wide topical trees should be. The annotations in the hierarchical *Moonstone* corpus provide a somewhat surprising starting point (broad shallow structures) for that particular genre. There are two sides to this question. Firstly, what kind of structures human annotators prefer? Secondly, what kind of structures is most useful in NLP applications? To answer these questions, it is necessary to collect additional data on hierarchical segmentations as well as conduct rigorous experimentation about applications of hierarchical segmentations.

It is one of the shortcomings of this work that we have not explored the usefulness of topical trees in NLP applications. We have already noted that no other hierarchical segmenters are publicly available. Free discourse-level parsers for large documents are also rather difficult to come by. Perhaps this shortage explains the lack of research on applications of various models of hierarchical structure.

Intuitively, knowing the topical tree of a document should help high-level NLP applications such as text summarization. It is not unreasonable to hypothesize that identifying sub-trees most central to the document can allow an automatic summarizer to explore only part of the input document, maybe resulting in computational gains. Additionally, if the summarizer is extractive, it is likely that sentences extracted from a smaller part of the document will result in a more coherent summary than summaries containing sentences from parts of the document that are far apart (*e.g.,* beginning and end). Along the same lines, topical trees may potentially be used in question-answering systems.

We can explore how automatically detected topical structure (single-level and hierarchical) can be used to improve the quality of anaphora resolution systems. Kehler [2002] discusses the multiple inter-relations between possible anaphoric links and the discourse structure of a document. Intuitively, anaphoric links (especially pronominal ones) should be less likely between segments focused on different topics. The prominence of the boundary is likely to affect just how likely such links are. We will study this relationship and see whether knowing the topical structure of a document can guide an anaphora resolution

system.

It may be also possible to use topical trees to analyze people's writing styles and perhaps to apply them in writing aids or language learning aids. While topical trees are a rather shallow representation, it would be interesting to see how typical topical structures look across different genres. Are there really significant topical shifts between sections of scientific articles? Is it always the case that the opening paragraph of a newswire item contains all the important terms discussed in it? Topical trees may be used as a light-weight form of large-scale studies of discourse structure. They may be also used in a similar manner as a corrective tool.

For now, these are only untested hypotheses. In the nearest future, however, we plan to explore the use of topical trees in summarization and in anaphora resolution.

One more somewhat urgent problem with hierarchical segmentation is the evaluation of topical trees. We did our best to obtain an adequate picture of the performance of *HAPS* but the results are rather preliminary. The main problem is the fact that there is no accepted insightful metric for comparing one topical tree to another in a meaningful way. We have used two metrics: *windowDiff* per level and *evalHDS*. Both have shortcomings and, most important, they view the levels of topical trees in isolation. In the future it would be extremely useful to evaluate the trees in a different manner. Fournier and Inkpen [2012] describe the *S* metric which we use for corpus analysis.[1] The hierarchical version of *S* unfortunately has never been released publicly and it is somewhat under-documented in [Fournier, 2013b]. We plan to include an implementation of hierarchical *S* in the future releases of *HAPS*. *S* suffers from the problem of being overly optimistic (most values are above 0.95) which is due to its normalization factor. However, it has a considerable advantage of being able to break down the errors into omissions/deletions, near hits/misses and inter-level error. It would be very useful to design a metric that has these desirable properties but also has a wider range of values.

We have already mentioned that *HAPS* relies too much on the existence of informative segment centres and its objective function ignores similarity between all data points in each segment (it only takes into account the similarity between each data point and its chosen segment centre). We intend to explore different objective functions and try taking into

[1] Thanks to Chris Fournier for a kind offer to analyse the hierarchical corpus using early versions of code for hierarchical *S*.

account intra-segment similarity. Factor graphs are a very flexible representation and such a modification is very feasible. The difficulty is likely to be in the complexity of update messages.

As far as the current implementation of *HAPS* is concerned, there is a lot of room for ongoing improvements. Perhaps the most urgent is the need to modify the script for fine-tuning input parameters. Currently, both *APS* and *HAPS* choose the best parameters by exhaustively searching the available space (on the developement part of each fold). This of course results in the fact that only a very small space can be searched and at a relatively crude step size. A more intelligent search strategy is highly likely to improve the results. Ideally, we would like to employ a system specifically designed to find the best parameters, such as SMAC [Hutter et al., 2011] and use it with all systems. We also intend to experiment with discriminative setting of the preference values (giving higher preferences to sentences that are more likely to be segment centres).

Overall we have designed and implemented a system for hierarchical segmentation of documents that creates a particular kind of topical trees, but much remains to be done in this area. We plan to continue working on hierarchical segmentation and hope that this work will inspire the interest of fellow researchers.

Bibliography

Apoorv Agarwal, Augusto Corvalan, Jacob Jensen, and Owen Rambow. Social Network Analysis of Alice in Wonderland. In *Proceedings of the Workshop on Computational Linguistics for Literature, NAACL 2012*, Montréal, Canada, June 2012.

James Allan, Jaime Carbonell, George Doddington, Jonathan Yamron, and Yiming Yang. Topic Detection and Tracking Pilot Study: Final Report. In *Proceedings of the 1998 DARPA Broadcast News Transcription and Understandign Workshop*, 1998.

Mira Ariel. Accessibility marking: Discourse functions, discourse profiles, and processing cues. *Discourse Processes*, 36:91–116, 2004.

Mira Ariel. *Accessing Noun-Phrase Antecedents*. Routeledge, London and New York, 2014.

Aristotle. Poetics. In Francis Fergusson and S.H. Butcher, editors, *Aristotle's Poetics*. Hill and Wang, 1961.

Ron Artstein and Massimo Poesio. Inter-Coder Agreement for Computational Linguistics. *Computational Linguistics*, 34(4):555–596, 2008.

Nicholas Asher and Alex Lascarides. *Logics of conversation*. Cambridge University Press, 2003.

Mieke Bal. *Introduction to the Theory of Narrative*. University of Toronto Press, 1985.

Jason Baldridge and Alex Lascarides. Probabilistic Head-Driven Parsing for Discourse Structure. In *Proceedings of the Ninth Conference on Computational Natural Language*

Learning (CoNLL-2005), pages 96–103, Ann Arbor, Michigan, June 2005. Association for Computational Linguistics.

Satanjeev Banerjee and Alexander Rudnicky. Segmenting meetings into agenda items by extracting implicit supervision from human note-taking. In *Proceedings of the 12th international conference on Intelligent user interfaces*, pages 151–159. ACM Press, 2007.

Daniel Bär, Chris Biemann, Iryna Gurevych, and Torsten Zesch. UKP: Computing Semantic Textual Similarity by Combining Multiple Content Similarity Measures. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics - Volume 1: Proceedings of the Main Conference and the Shared Task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation*, SemEval '12, pages 435–440, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

Daniel Bär, Torsten Zesch, and Iryna Gurevych. DKPro Similarity: An Open Source Framework for Text Similarity. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 121–126, Sofia, Bulgaria, August 2013. Association for Computational Linguistics.

Roland Barthes. *S/Z*. Sueil, Paris, 1970.

Doug Beeferman, Adam Berger, and John Lafferty. Statistical Models for Text Segmentation. *Machine Learning*, 34:177–210, February 1999. ISSN 0885-6125. doi: 10.1023/A:1007506220214.

Yves Bestgen and Jean Costermans. Time, space, and action: Exploring the narrative structure and its linguistic marking. *Discourse Processes*, 17(3):421–446, 1994. doi: 10.1080/01638539409544877.

Christopher M. Bishop. *Pattern Recognition and Machine Learning*. Springer, 2006.

David Blei and Pedro Moreno. Topic segmentation with an aspect hidden Markov Model. In *Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 343–348. ACM Press, 2001.

David M. Blei, Andrew Ng, and Michael Jordan. Latent Dirichlet allocation. *Journal of Machine Learning Research*, 3:993–1022, 2003.

Rens Bod, Bernhard Fisseni, Kurjim Aadil, and Benedict Löwe. Objectivity and Reproducibility of Proppian Annotations. In Mark A. Finlayson, editor, *Proceedings of the Third Workshop on Computational Models of Narrative*, pages 17–23, Istanbul, Turkey, May 2012.

Claude Bremond. *Logique de Récie*. Seuil, Paris, 1973.

William F. Brewer and James C. Treyens. Role of schemata in memory for places. *Cognitive Psychology*, 13(2):207 – 230, 1981. ISSN 0010-0285. doi: 10.1016/0010-0285(81)90008-6.

Sergey Brin and Lawrence Page. The Anatomy of a Large-scale Hypertextual Web Search Engine. In *Proceedings of the Seventh International Conference on World Wide Web 7*, WWW7, pages 107–117, Amsterdam, The Netherlands, The Netherlands, 1998. Elsevier Science Publishers B. V.

Julian Brooke, Adam Hammond, and Graeme Hirst. Unsupervised Stylistic Segmentation of Poetry with Change Curves and Extrinsic Features. In *Proceedings of the Workshop on Computational Linguistics for Literature, NAACL 2012*, Montréal, Canada, June 2012.

Elizabeth Brown. Topic Continuity in Written English Narrative. In Talmy Givón, editor, *Topic Continuity in Discourse*, pages 313–343. John Benjamins PublishingCompany, Philadelphia/Amsterdam, 1983.

Rogelio E. Cardona-Rivera, Bradley A. Cassel, Stephen G. Ware, and Michel R. Young. Indexter: A Computational Model of the Event-Indexing Situation Model for Characterizing Narratives. In Mark A. Finlayson, editor, *Proceedings of the Third Workshop on Computational Models of Narrative*, pages 34–43, Istanbul, Turkey, May 2012.

Lynn Carlson, Daniel Marcu, and Mary Ellen Okurowski. Building a discourse tagged corpus in the framework of Rhetorical Structure Theory. In *Proceedings of 2nd SIGDIAL Workshop on Discourse and Dialogue, Eurospeech 2001*, Aalborg, Denmark, 2001.

Lucien Carroll. Evaluating Hierarchical Discourse Segmentation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Asso-*

ciation for Computational Linguistics, pages 993–1001. The Association for Computational Linguistics, 2010. ISBN 978-1-932432-65-7.

Nathanael Chambers and Dan Jurafsky. Unsupervised Learning of Narrative Schemas and their Participants. In Keh-Yih Su, Jian Su, and Janyce Wiebe, editors, *Proceedings of the 47th Annual Meeting of the Association for Computational Linguistics and the 4th International Joint Conference on Natural Language Processing of the AFNLP*, pages 602–610. The Association for Computer Linguistics, 2009. ISBN 978-1-932432-45-9, 978-1-932432-46-6.

Nathanael Chambers and Daniel Jurafsky. Unsupervised Learning of Narrative Event Chains. In Kathleen McKeown, Johanna D. Moore, Simone Teufel, James Allan, and Sadaoki Furui, editors, *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, pages 789–797. The Association for Computer Linguistics, 2008. ISBN 978-1-932432-04-6.

Harr Chen, S. R. K. Branavan, Regina Barzilay, and David R. Karger. Global Models of Document Structure Using Latent Permutations. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 371–379, Stroudsburg, PA, USA, 2009. Association for Computational Linguistics.

Freddy Y. Y. Choi. Advances in Domain Independent Linear Text Segmentation. In *Proceedings of the 1st North American chapter of the Association for Computational Linguistics conference*, pages 26–33, 2000.

Freddy Y. Y. Choi, Peter Wiemer-Hastings, and Johanna Moore. Latent Semantic Analysis for Text Segmentation. In *Proceedings of the 2001 Conference on Empirical Methods in Natural Language Processing*, pages 109–117, Pittsburgh, Pennsylvania, 2001.

Jacob Cohen. A coefficient of agreement for nominal scales . *Educational and Psychological Measurement*, 20:37–46, 1960.

Dan Cristea, Nancy Ide, and Laurent Romary. Veins Theory: A Model of Global Discourse Cohesion and Coherence. In *Proceedings of the 36th Annual Meeting of the Association*

for Computational Linguistics and 17th International Conference on Computational Linguistics - Volume 1, ACL '98, pages 281–285, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics.

R.E. Cullingford. *Script Application: Computer Understanding of Newspaper Stories*. PhD thesis, Dept. of Computer Science, Yale University, 1978.

William Darling and Fei Song. PathSum: A Summarization Framework Based on Hierarchical Topics. In *Proceedings of the Workshop on Text Summarization. Collocated with Canadian Conference on Artificial Intelligence*, St. John's, Canada, 2011.

Pradipto Das and Rohini Srihari. Learning to Summarize using Coherence. In *NIPS Workshop on Applications for Topic Models: Text and Beyond*, Whistler, Canada, 2009.

Giel Dik and Henk Aarts. Behavioral cues to others' motivation and goal pursuits: The perception of effort facilitates goal inference and contagion. *Journal of Experimental Social Psychology*, 43:727–737, 2007.

Lan Du, Wray Buntine, and Mark Johnson. Topic Segmentation with a Structured Topic Model. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 190–200, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

Michael G. Dyer. *In-Depth Understanding: A Computer Model of Integrated Processing for Narrative Comprehension*. MIT Press Cambridge, MA, USA, 1983.

Umberto Eco. *The Role of the Reader: Explorations in the Semiotics of Texts*. Indiana University Press, 1979.

Jacob Eisenstein. Hierarchical Text Segmentation from Multi-Scale Lexical Cohesion. In *HLT-NAACL Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings*, pages 353–361, 2009.

Jacob Eisenstein and Regina Barzilay. Bayesian Unsupervised Topic Segmentation. In *Proceedings of the 2008 Conference on Empirical Methods in Natural Language Processing*, pages 334–343, Honolulu, Hawaii, October 2008.

Micha Elsner. Character-based kernels for novelistic plot structure. In Walter Daelemans, Mirella Lapata, and Lluís Màrquez, editors, *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, pages 634–644. The Association for Computer Linguistics, 2012. ISBN 978-1-937284-19-0.

David K. Elson. *Modelling Narrative Discourse*. PhD thesis, Columbia University, 2012.

David K. Elson, Nicholas Dames, and Kathleen McKeown. Extracting Social Networks from Literary Fiction. In Jan Hajic, Sandra Carberry, and Stephen Clark, editors, *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, pages 138–147. The Association for Computer Linguistics, 2010. ISBN 978-1-932432-66-4, 978-1-932432-67-1.

Bridgitte Endres-Niggemeyer. *Summarizing Information*. Springer-Verlag, Berlin Heidelberg, 1998.

Christiane Fellbaum, editor. *WordNet: An Electronic Lexical Database (Language, Speech, and Communication)*. The MIT Press, May 1998. ISBN 026206197X.

Vanessa Wei Feng and Graeme Hirst. Text-level Discourse Parsing with Rich Linguistic Features. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, pages 60–68, Jeju Island, Korea, July 2012. Association for Computational Linguistics.

Fernando Ferrara. Theory and model for the structural analysis of fiction. *New Literary History*, 5:245–268, 1974.

Jenny Rose Finkel, Trond Grenager, and Christopher Manning. Incorporating Non-local Information into Information Extraction Systems by Gibbs Sampling. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 363–370, Stroudsburg, PA, USA, 2005. Association for Computational Linguistics.

Edward Morgan Foster. *Aspects of the Novel*. Penguin, Harmondsworth, 1963.

Chris Fournier. Evaluating Text Segmentation using Boundary Edit Distance. In *Proceedings of 51st Annual Meeting of the Association for Computational Linguistics*, pages 1702–1712, Stroudsburg, PA, USA, 2013a. Association for Computational Linguistics.

Chris Fournier. Evaluating Text Segmentation. Master's thesis, University of Ottawa, 2013b.

Chris Fournier and Diana Inkpen. Segmentation Similarity and Agreement. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, Montréal, Canada, June 2012.

Brendan J. Frey and Delbert Dueck. Clustering by Passing Messages Between Data Points. *Science*, 315:972–976, 2007.

Evgeniy Gabrilovich and Shaul Markovitch. Computing Semantic Relatedness Using Wikipedia-based Explicit Semantic Analysis. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 1606–1611, San Francisco, CA, USA, 2007. Morgan Kaufmann Publishers Inc.

Michel Galley, Kathleen McKeown, Eric Fosler-Lussier, and Hongyan Jing. Discourse segmentation of multi-party conversation. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 562–569, Stroudsburg, PA, USA, 2003. Association for Computational Linguistics.

Talmy Givón. Typology and Functional Domains. *Studies in Language*, 5(2):163–193, 1981.

Talmy Givón, editor. *Topic Continuity in Discourse*. John Benjamins PublishingCompany, Philadelphia/Amsterdam, 1983.

Inmar E. Givoni and Brendan J. Frey. A Binary Variable Model for Affinity Propagation. *Neural Computation*, 21:1589–1600, 2009.

Inmar E. Givoni, Clement Chung, and Brendan J. Frey. Hierarchical Affinity Propagation. In *Uncertainty in AI, Proceedings of the Twenty-Seventh Conference (2011)*, pages 238–246, 2011.

Inmar Ella Givoni. *Beyond Affinity Propagation: Message Passing Algorithms for Clustering*. PhD thesis, University of Toronto, 2012.

James Glass, Timothy J. Hazen, Scott Cyphers, Igor Malioutov, David Huynh, and Regina Barzilay. Recent Progress in the MIT Spoken Lecture Processing Project. In *Proceedings of Interspeech*, 2007.

Amit Goyal, Ellen Riloff, and Hal Daumé III. Automatically Producing Plot Unit Representations for Narrative Text. In *EMNLP '10 Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, pages 77–86, 2010.

Arthur Graesser, Murray Singer, and Tom Trabasso. Constructing inferences during narrative text comprehension. *Psychological Review*, 101(3):371–395, 1994.

Arthur C. Graesser, DanielleS. McNamara, MaxM. Louwerse, and Zhiqiang Cai. CohMetrix: Analysis of text on cohesion and language. *Behavior Research Methods, Instruments, and Computers*, 36:193–202, 2004.

Algirdas Julien Greimas. Narrative Grammar. Units and Levels. *Comparative Literature*, 86(6), 1971.

Barbara J. Grosz and Candace L. Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, July 1986.

Alexander Gruenstein, John Niekrasz, and Matthew Purver. Meeting Structure Annotation: Data and Tools. In *In Proceedings of the SIGdial Workshop on Discourse and Dialogue*, pages 117–127, 2005.

Aria Haghighi and Lucy Vanderwende. Exploring Content Models for Multi-Document Summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 362–370, Boulder, Colorado, June 2009.

M.A.K Halliday and Ruqaiya Hasan. *Cohesion in English*. Longman, London and New York, 1976.

David Haussler. Convolution kernels on discrete structures. Technical report, Computer Science Department, UC Santa Cruz, 1999.

Timothy J. Hazen. Latent Topic Modeling for Audio Corpus Summarization. In *Proceedings of Interspeech*, pages 913–916. ISCA, 2011.

Marti A. Hearst. TextTiling: segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, 23:33–64, March 1997. ISSN 0891-2017.

Franz Heider and Marianne Simmel. An experimental study of apparent behaviour. *The American Journal of Psychology*, 57:243–259, 1944.

Hugo Hernault, H Prendinger, David A. duVerle, and M. Ishizuka. HILDA: A Discourse Parser Using Support Vector Machine Classification. *Dialogue and Discourse*, 3:1–33, 2010.

Ryan Heuser and Long Le-Khac. Pamphlet 4. A Quantitative Literary History of 2,958 Nineteenth Century British Novels: the Semantic Cohort Method. Stanford Lit Lab. `http://litlab.stanford.edu/?page_id=255`, 2012.

Michael Hoey. *Patterns of Lexis in Text*. Oxford University Press, 1991.

Frank Hutter, Holger H. Hoos, and Kevin Leyton-Brown. Sequential Model-Based Optimization for General Algorithm Configuration. In *Proceedings of Learning and Intelligent Optimization LION-5*, pages 507–523, 2011.

Geert Jacobs and Tom van Hout. Towards a process view of reformulation in press releases. In Jan Renkema, editor, *Discourse, of course*, pages 239–251. John Benjamins, 2009.

Adam Janin, Don Baron, Jane Edwards, D. Ellis, David Gelbart, Nelson Morgan, Barbara Peskin, Thilo Pfau, Elizabeth Shriberg, A.ndreas Stolcke, and Chuck Wooters. The ICSI Meeting Corpus. In *Proceedings of the IEEE International Conference on Acoustics, Speech, and Signal Processing (ICASSP-03)*, volume 1, pages 364–367, April 2003.

Ekaterina Jasinskaja, Ulrike Kölsch, and Jörg Mayer. Nuclear Accent Placement and Other Prosodic Parameters As Cues to Pronoun Resolution. In *Proceedings of the 6th Discourse Anaphora and Anaphor Resolution Conference on Anaphora: Analysis, Algorithms and Applications*, DAARC'07, pages 1–14, Berlin, Heidelberg, 2007. Springer-Verlag.

Amanda C. Jobbins and Lindsay J. Evett. Text Segmentation Using Reiteration and Collocation. In *Proceedings of the 17th International Conference on Computational Linguistics - Volume 1*, COLING '98, pages 614–618, Montréal, Québec, 1998.

Hans Kamp and Uwe Reyle. *From Discourse to Logic*. Kluwer Academic Publishers, 1993.

Anna Kazantseva and Stan Szpakowicz. Summarizing short stories. *Computational Linguistics*, 36(1):71–109, 2010.

Anna Kazantseva and Stan Szpakowicz. Linear Text Segmentation Using Affinity Propagation. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*, pages 284–293, Edinburgh, Scotland, UK., July 2011. Association for Computational Linguistics.

Anna Kazantseva and Stan Szpakowicz. Topical Segmentation: a Study of Human Performance and a New Measure of Quality. In *Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 211–220. The Association for Computational Linguistics, 2012. ISBN 978-1-937284-20-6.

Anna Kazantseva and Stan Szpakowicz. Hierarchical Segmentation of Topic Using Affinity Propagation. In *25th International Conference on Computational Linguistics (COLING 2014)*, pages 37–47, 2014a.

Anna Kazantseva and Stan Szpakowicz. Measuring Lexical Cohesion: Beyond Word Repetition. In *25th International Conference on Computational Linguistics (COLING 2014)*, pages 476–485, 2014b.

Andrew Kehler. *Coherence, Reference and the Theory of Grammar*. CSLI Publications, 2002.

Caryn E. Krakauer and Patrick H. Winston. Story Retrieval and Comparison using Concept Patterns. In Mark A. Finlayson, editor, *Proceedings of the Third Workshop on Computational Models of Narrative*, pages 119–124, Istanbul, Turkey, May 2012.

Klaus Krippendorff. *Content Analysis. An Introduction to Its Methodology.* Sage Publications., 2004.

Frank R. Kschischang, Brendan J. Frey, and Hans-A Loeliger. Factor graphs and the sum-product algorithm. In *IEEE Transactions on Information Theory, Vol 47, No 2*, pages 498–519, February 2001.

John Lafferty, Andrew McCallum, and Fernando Pereira. Conditional Random Fields: Probabilistic Models for Segmenting and Labeling Sequence Data. In *ICML '01 Proceedings of the Eighteenth International Conference on Machine Learning 1*, pages 282–289, 2001.

Thomas K Landauer and Susan T. Dumais. A solution to Plato's problem: The latent semantic analysis theory of acquisition, induction, and representation of knowledge. *Psychological Review*, pages 211–240, 1997.

J. Richards Landis and Garry G. Koch. The Measurement of Observer Agreement for Categorical Data. *Biometrics*, 33(1):159–174, 1977.

Michael Lebowitz. Creating a Story-Telling Universe. In *Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pages 63–65, 1983.

Wendy G. Lehnert. Plot Units: A Narrative Summarization Strategy. In Wendy G. Lehnert and Martin H. Ringle, editors, *Strategies for Natural Language Processing*, pages 375–414, Hillsdale, NJ, 1982. Erlbaum.

Claude Lévi-Strauss. The Structural study of myth. *Journal of American Folklore*, 68: 428–444, 1955.

Boyang Li, Stephen Lee-Urban, Darren Scott Appling, and Mark O. Riedl. Automatically Learning to Tell Stories about Social Situations from the Crowd. In Mark A. Finlayson, editor, *Proceedings of the Third Workshop on Computational Models of Narrative*, pages 142–151, Istanbul, Turkey, May 2012.

Chin-Yew Lin. ROUGE: A Package for Automatic Evaluation of summaries. In *Text Summarization Branches Out, Proceedings of the ACL Workshop*, pages 74–81, 2004.

Dekang Lin. Automatic Retrieval and Clustering of Similar Words. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and 17th*

International Conference on Computational Linguistics - Volume 2, ACL '98, pages 768–774, Stroudsburg, PA, USA, 1998. Association for Computational Linguistics. doi: 10.3115/980691.980696.

Max M. Louwerse, Phillip M. McCarthy, Daniel S. McNamara, and Arthur Graesser. Variation in language and cohesion across written and spoken registers. In *Proceedings of the 26th Annual Meeting of the Cognitive Science Society*, pages 843–848, 2004.

Meena Mahajan, Prajakta Nimbhorkar, and Kasturi Varadarajan. The planar k-means problem is np-hard. In *Proceedings of the 3rd International Workshop on Algorithms and Computation*, WALCOM '09, pages 274–285, Berlin, Heidelberg, 2009. Springer-Verlag. ISBN 978-3-642-00201-4. doi: 10.1007/978-3-642-00202-1_24. URL `http://dx.doi.org/10.1007/978-3-642-00202-1_24`.

Igor Malioutov and Regina Barzilay. Minimum Cut Model for Spoken Lecture Segmentation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, pages 25–32, Sydney, Australia, July 2006.

Inderjeet Mani. *The Imagined Moment. Time, Narrative and Computation*. University of Nebraska Press, 2010.

William C. Mann and Sandra A. Thompson. Rhetorical Structure Theory: A Theory of Text Organization. Technical Report ISI/RS-87-190, Information Sciences Institute, June 1987 1987.

William C. Mann and Sandra A. Thompson. Rhetorical Structure Theory: Toward a functional theory of text organization. *Text*, 8(3):243–281, 1988.

Meghana Marathe. Lexical Chains Using Distributional Measures of Concept Distance. Master's thesis, University of Toronto, 2010.

Daniel Marcu. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, Cambridge, Mass, 2000.

Daniel Marcu. Discourse Structures: Trees or Graphs? Unpublished Web Post. `http://www.isi.edu/~marcu/discourse/Discourse structures.htm`, 2003.

Mitchell P. Marcus, Beatrice Santorini, and Mary Ann Marcinkiewicz. Building a Large Annotated Corpus of English: The Penn Treebank. *Computational Linguistics*, 19(2): 313–330, 1993.

Georey J. Mclachlan and Thriyambakam Krishnan. *The EM Algorithm and Extensions*. Wiley-Interscience, 1 edition, 1996.

Loizos Michael. Similarity of Narratives. In Mark A. Finlayson, editor, *Proceedings of the Third Workshop on Computational Models of Narrative*, pages 105–113, Istanbul, Turkey, May 2012.

Hemant Misra, François Yvon, Olivier Cappé, and Joemon M. Jose. Text segmentation: A topic modelling perspective. *Information Processing and Management*, 47(4):528–544, 2011.

Franco Moretti. Pamphlet 2. Network Theory, Plot Analysis. Stanford Lit Lab. `http://litlab.stanford.edu/?page_id=255`, 2011.

Peter Norvig. Frame Activated Inferences in a Story Understanding Program. In *In Proceedings of the 8th International Joint Conference on Artificial Intelligence*, pages 624–626, 1983.

Peter Norvig. Marker Passing as a Weak Method for Text Inferencing. *Cognitive Science*, 13(4):569–620, 1989.

Manabu Okumura and Takeo Honda. Word Sense Disambiguation and Text Segmentation Based On Lexical Cohesion. In *COLING 1994 Volume 2: The 15th International Conference on Computational Linguistics*, pages 775–761, Kyoto, Japan, 1994.

Andrew Olney and Zhiqiang Cai. An Orthonormal Basis for Topic Segmentation in Tutorial Dialogue. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing – HLT '05*, pages 971–978, Vancouver, Canada, 2005.

Judea Pearl. Reverend Bayes on inference engines: A distributed hierarchical approach. In *Proceedings of the American Association of Artificial Intelligence National Conference on AI*, pages 133–136, Pittsburgh, PA, 1982.

Lev Pevzner and Marti A. Hearst. A Critique and Improvement of an Evaluation Metric for Text Segmentation. *Computational Linguistics*, 28(1):19–36, 2002.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. The Penn Discourse TreeBank 2.0. In *In Proceedings of LREC*, 2008.

Vladimir Propp. *Morphology of the Folk Tale.* Indiana University Press, Bloomington, 2nd edition, 1968.

Matthew Purver. Topic Segmentation. In G. Tur and R de Mori, editors, *Spoken Language Understanding: Systems for extracting Semantic Information from Speech*, pages 291–317. Wiley, Hoboken, NJ, 2011.

Karthik Raghunathan, Heeyoung Lee, Sudarshan Rangarajan, Nathanael Chambers, Mihai Surdeanu, Dan Jurafsky, and Christopher Manning. A Multi-pass Sieve for Coreference Resolution. In *Proceedings of the 2010 Conference on Empirical Methods in Natural Language Processing*, EMNLP '10, pages 492–501, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

David Reitter. Simple signals for complex rhetorics: On rhetorical analysis with rich-feature support vector models. *LDV Forum*, 18(1-2):38–52, 2003.

Jan Renkema. *Introduction to Discourse Studies*. John Benjamins, 2004.

Jeffrey C. Reynar. Statistical Models of Text Segmentation. In *Proceedings of the 37th Annual Meeting of the Association for Computational Linguistics*, pages 357–364, 1999.

Mark O. Riedl and R. Michael Young. Narrative Planning: Balancing Plot and Character. *Journal of Artificial Intelligence Research (JAIR)*, 39:217–268, 2010.

Martin Riedl and Chris Biemann. TopicTiling: A Text Segmentation Algorithm Based on LDA. In *Proceedings of ACL 2012 Student Research Workshop*, ACL '12, pages 37–42, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

Shlomith Rimmon-Kenan. *Narrative Fiction*. New Accents, 1983.

Andrea Rocci. Doing discourse analysis with possible worlds. In Jan Renkema, editor, *Discourse, of course*, pages 15–35. John Benjamins, 2009.

Mihai Rotaru and Diane J. Litman. Exploiting Discourse Structure for Spoken Dialogue Performance Analysis. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 85–93, Stroudsburg, PA, USA, 2006. Association for Computational Linguistics.

Marie Laure Ryan. *Possible Worlds, Artificial Intelligence and Narrative Theory*. Indiana University Press, Bloomington, Indiana, 1991.

Ted J.M. Sanders and Morton Ann Gernsbacher. Accessibility inText and Discourse Processing. *Discourse Processes*, 36:79–89, 2004.

A. J. Sanford, K. Moar, and S. C. Garrod. Proper Names as Controllers of Discourse Focus. *Language and Speech*, 37(2):79–89, January/March 2004.

Martin Scaiano and Diana Inkpen. Getting More from Segmentation Evaluation. In *Proceedings of NAACL-HLT 2012*, Montréal, Canada, June 2012.

Martin Scaiano, Diana Inkpen, Robert Laganière, and Adele Reinhartz. Automatic Text Segmentation for Movie Subtitles. In *Advances in Artificial Intelligence*, volume 6085 of *Lecture Notes in Computer Science*, pages 295–298. Springer, 2010.

William Scott. Reliability of content analysis: The case of nominal scale coding. *Public Opinion Quaterly*, 19(3):321–325, 1955.

Patrick E. Shrout and Joseph L. Fleiss. Intraclass correlations: uses in assessing rater reliability. *Psychological Bulletin*, 86(2):420–428, 1979.

Sidney Siegel and John. N. Jr. Castellan. *Nonparametric statistics for the behavioral sciences*. McGraw Hill, Boston, MA, 1988.

Christian Smith, Henrik Danielsson, and Arne Jönsson. A More Cohesive Summarizer. In *24th International Conference on Computational Linguistics, Proceedings of COLING 2012: Posters*, pages 1161–1170, Mumbai, India, 2012.

Richard Socher, John Bauer, Christopher D. Manning, and Andrew Y. Ng. Parsing with Compositional Vector Grammars. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013*, pages 455–465, 2013.

Fei Song, William M. Darling, Adnan Duric, and Fred W. Kroon. An iterative approach to text segmentation. In *Proceedings of the 33rd European conference on Advances in information retrieval*, ECIR'11, pages 629–640, Berlin, Heidelberg, 2011. Springer-Verlag.

Radu Soricut and Daniel Marcu. Sentence Level Discourse Parsing using Syntactic and Lexical Information. In *Proceeding NAACL '03 Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology - Volume 1*, pages 149–156, 2003.

Rajen Subba and Barbara Di Eugenio. An effective Discourse Parser that uses Rich Linguistic Information. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceeding*, pages 566–574, 2009.

Maria Teresa Taboada. *Building Coherence and Cohesion*. John Benjamins, 2004.

Maria Teresa Taboada. Implicit and explicit coherence relations. In Jan Renkema, editor, *Discourse, of course*, pages 127–143. John Benjamins, 2009.

Pasi Tapanainen and Timo Järvinen. A non-projective dependency parser. *Proceedings of the 5th Conference on Applied Natural Language Processing*, pages 64–71, 1997.

Simone Teufel and Marc Moens. Summarizing Scientific Articles - Experiments with Relevance and Rhetorical Status. *Computational Linguistics*, 28:2002, 2002.

Gian Lorenzo Thione, Martin Van den Berg, Livia Polanyi, and Chris Culy. Hybrid Text Summarization: Combining External Relevance Measures with Structural Analysis. In Marie-Francine Moens and Stan Szpakowicz, editors, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 51–55, Barcelona, Spain, July 2004. Association for Computational Linguistics.

Perry W. Thorndyke. *Cognitive Structures in Human Story Comprehension and Memory*. PhD thesis, Stanford University, 1975.

Tom Trabasso and Linda L. Sperry. Causal relatedness and importance of story events. *Journal of Memory and Language*, 24(5):595–611, 1985.

Tom Trabasso and Nancy L. Stein. Narrating, representing and remembering event sequences. In Tammy Bourg Paul van den Broek and Patricia J. Bauer, editors, *Developmental Spans in Event Comprehension and Representation*, pages 237–269. Erlbaum, Mahwah, New Jersey, 1997.

Vinícius Rodrigues Uzêda, Thiago Alexandre Salgueiro Pardo, and Maria Das Graças Volpe Nunes. A Comprehensive Comparative Evaluation of RST-based Summarization Methods. *ACM Transactions on Speech and Language Processing*, 6(4): 4:1–4:20, May 2010. ISSN 1550-4875.

Teun van Dijk. Discourse and Communication: a new journal to bridge two fields. *Discourse and Communication*, 1(1):5–7, February 2007.

Marc Verhagen, Inderjeet Mani, Roser Sauri, Jessica Littman, Robert Knippen, Seok Bae Jang, Anna Rumshisky, John Phillips, and James Pustejovsky. Automating Temporal Annotation with TARSQI. In *ACL 2005, 43rd Annual Meeting of the Association for Computational Linguistics, Proceedings of the Conference, 25-30 June 2005, University of Michigan, USA*, 2005.

Wietske Vonk, Lettica G.M.M. Hustinx, and Wim H.G. Simons. The use of referential expressions in structuring discourse. *Language and Cognitive Processes*, 7(3-4):301–333, 1992.

Lu Wang and Claire Cardie. Unsupervised Topic Modeling Approaches to Decision Summarization in Spoken Meetings. In *Proceedings of the SIGDIAL 2012 Conference, The 13th Annual Meeting of the Special Interest Group on Discourse and Dialogue, 5-6 July 2012, Seoul National University, Seoul, South Korea*, pages 40–49, 2012.

B. Webber, M. Egg, and Valia Kordoni. Discourse structure and language technology. *Natural Language Engineering*, 18(4):437–490, 2012.

Robert Wilensky. PAM: a program that infers intentions. In *Proceedings of the 5th international joint conference on Artificial intelligence - Volume 1*, IJCAI'77, pages 15–15, San Francisco, CA, USA, 1977. Morgan Kaufmann Publishers Inc.

Florian Wolf and Edward Gibson. *Coherence in Natural Language: Data Structures and Applications*. MIT Press, Cambridge, MA, 2006.

Yaakov Yaari. Segmentation of Expository Texts by Hierarchical Agglomerative Clustering. In *Proceedings of RANLP 97*, 1997.

Gilbert Youmans. A new tool for discourse analysis: The vocabulary-management profile. *Language*, 67(4):763–789, 1991.

R. Young and A. Becker. The role of lexical and grammatical cues in paragraph recognition. Technical report, Michigan: Centre for Research on Language, University of Michigan, 1966.

Appendix A

Full text of Chapter II of *The Moonstone*

THE MOONSTONE

A Romance

by Wilkie Collins

THE STORY

FIRST PERIOD THE LOSS OF THE DIAMOND (1848)

The events related by GABRIEL BETTEREDGE, house-steward in the service of JULIA, LADY VERINDER. CHAPTER II

1. I spoke of my lady a line or two back. Now the Diamond could never have been in

our house, where it was lost, if it had not been made a present of to my lady's daughter; and my lady's daughter would never have been in existence to have the present, if it had not been for my lady who (with pain and travail) produced her into the world. Consequently, if we begin with my lady, we are pretty sure of beginning far enough back. And that, let me tell you, when you have got such a job as mine in hand, is a real comfort at starting.

2. If you know anything of the fashionable world, you have heard tell of the three beautiful Miss Herncastles. Miss Adelaide; Miss Caroline; and Miss Julia–this last being the youngest and the best of the three sisters, in my opinion; and I had opportunities of judging, as you shall presently see. I went into the service of the old lord, their father (thank God, we have got nothing to do with him, in this business of the Diamond; he had the longest tongue and the shortest temper of any man, high or low, I ever met with)–I say, I went into the service of the old lord, as page-boy in waiting on the three honourable young ladies, at the age of fifteen years. There I lived till Miss Julia married the late Sir John Verinder. An excellent man, who only wanted somebody to manage him; and, between ourselves, he found somebody to do it; and what is more, he throve on it and grew fat on it, and lived happy and died easy on it, dating from the day when my lady took him to church to be married, to the day when she relieved him of his last breath, and closed his eyes for ever.

3. I have omitted to state that I went with the bride to the bride's husband's house and lands down here. "Sir John," she says, "I can't do without Gabriel Betteredge." "My lady," says Sir John, "I can't do without him, either." That was his way with her–and that was how I went into his service. It was all one to me where I went, so long as my mistress and I were together.

4. Seeing that my lady took an interest in the out-of-door work, and the farms, and such like, I took an interest in them too–with all the more reason that I was a small farmer's seventh son myself. My lady got me put under the bailiff, and I did my best, and gave satisfaction, and got promotion accordingly. Some years later, on the Monday as it might be, my lady says, "Sir John, your bailiff is a stupid old man. Pension him liberally, and let Gabriel Betteredge have his place." On the Tuesday as it might be, Sir John says, "My lady, the bailiff is pensioned liberally; and Gabriel Betteredge has got his place." You hear more than enough of married people living together miserably. Here is an example to the contrary. Let it be a warning to some of you, and an encouragement to others. In the

meantime, I will go on with my story.

5. Well, there I was in clover, you will say. Placed in a position of trust and honour, with a little cottage of my own to live in, with my rounds on the estate to occupy me in the morning, and my accounts in the afternoon, and my pipe and my ROBINSON CRUSOE in the evening–what more could I possibly want to make me happy? Remember what Adam wanted when he was alone in the Garden of Eden; and if you don't blame it in Adam, don't blame it in me.

6. The woman I fixed my eye on, was the woman who kept house for me at my cottage. Her name was Selina Goby. I agree with the late William Cobbett about picking a wife. See that she chews her food well and sets her foot down firmly on the ground when she walks, and you're all right. Selina Goby was all right in both these respects, which was one reason for marrying her. I had another reason, likewise, entirely of my own discovering. Selina, being a single woman, made me pay so much a week for her board and services. Selina, being my wife, couldn't charge for her board, and would have to give me her services for nothing. That was the point of view I looked at it from. Economy–with a dash of love. I put it to my mistress, as in duty bound, just as I had put it to myself.

7. "I have been turning Selina Goby over in my mind," I said, "and I think, my lady, it will be cheaper to marry her than to keep her."

8. My lady burst out laughing, and said she didn't know which to be most shocked at–my language or my principles. Some joke tickled her, I suppose, of the sort that you can't take unless you are a person of quality. Understanding nothing myself but that I was free to put it next to Selina, I went and put it accordingly. And what did Selina say? Lord! how little you must know of women, if you ask that. Of course she said, Yes.

9. As my time drew nearer, and there got to be talk of my having a new coat for the ceremony, my mind began to misgive me. I have compared notes with other men as to what they felt while they were in my interesting situation; and they have all acknowledged that, about a week before it happened, they privately wished themselves out of it. I went a trifle further than that myself; I actually rose up, as it were, and tried to get out of it. Not for nothing! I was too just a man to expect she would let me off for nothing. Compensation to the woman when the man gets out of it, is one of the laws of England. In obedience to the laws, and after turning it over carefully in my mind, I offered Selina Goby a feather-bed and fifty shillings to be off the bargain. You will hardly believe it, but it is nevertheless

true–she was fool enough to refuse.

10. After that it was all over with me, of course. I got the new coat as cheap as I could, and I went through all the rest of it as cheap as I could. We were not a happy couple, and not a miserable couple. We were six of one and half-a-dozen of the other. How it was I don't understand, but we always seemed to be getting, with the best of motives, in one another's way. When I wanted to go up-stairs, there was my wife coming down; or when my wife wanted to go down, there was I coming up. That is married life, according to my experience of it.

11. After five years of misunderstandings on the stairs, it pleased an all-wise Providence to relieve us of each other by taking my wife. I was left with my little girl Penelope, and with no other child. Shortly afterwards Sir John died, and my lady was left with her little girl, Miss Rachel, and no other child. I have written to very poor purpose of my lady, if you require to be told that my little Penelope was taken care of, under my good mistress's own eye, and was sent to school and taught, and made a sharp girl, and promoted, when old enough, to be Miss Rachel's own maid.

12. As for me, I went on with my business as bailiff year after year up to Christmas 1847, when there came a change in my life. On that day, my lady invited herself to a cup of tea alone with me in my cottage. She remarked that, reckoning from the year when I started as page-boy in the time of the old lord, I had been more than fifty years in her service, and she put into my hands a beautiful waistcoat of wool that she had worked herself, to keep me warm in the bitter winter weather.

13. I received this magnificent present quite at a loss to find words to thank my mistress with for the honour she had done me. To my great astonishment, it turned out, however, that the waistcoat was not an honour, but a bribe. My lady had discovered that I was getting old before I had discovered it myself, and she had come to my cottage to wheedle me (if I may use such an expression) into giving up my hard out-of-door work as bailiff, and taking my ease for the rest of my days as steward in the house. I made as good a fight of it against the indignity of taking my ease as I could. But my mistress knew the weak side of me; she put it as a favour to herself. The dispute between us ended, after that, in my wiping my eyes, like an old fool, with my new woollen waistcoat, and saying I would think about it.

14. The perturbation in my mind, in regard to thinking about it, being truly dreadful after my lady had gone away, I applied the remedy which I have never yet found to fail

me in cases of doubt and emergency. I smoked a pipe and took a turn at ROBINSON CRUSOE. Before I had occupied myself with that extraordinary book five minutes, I came on a comforting bit (page one hundred and fifty-eight), as follows: "To-day we love, what to-morrow we hate." I saw my way clear directly. To-day I was all for continuing to be farm-bailiff; to-morrow, on the authority of ROBINSON CRUSOE, I should be all the other way. Take myself to-morrow while in to-morrow's humour, and the thing was done. My mind being relieved in this manner, I went to sleep that night in the character of Lady Verinder's farm bailiff, and I woke up the next morning in the character of Lady Verinder's house-steward. All quite comfortable, and all through ROBINSON CRUSOE!

15. My daughter Penelope has just looked over my shoulder to see what I have done so far. She remarks that it is beautifully written, and every word of it true. But she points out one objection. She says what I have done so far isn't in the least what I was wanted to do. I am asked to tell the story of the Diamond and, instead of that, I have been telling the story of my own self. Curious, and quite beyond me to account for. I wonder whether the gentlemen who make a business and a living out of writing books, ever find their own selves getting in the way of their subjects, like me? If they do, I can feel for them. In the meantime, here is another false start, and more waste of good writing-paper. What's to be done now? Nothing that I know of, except for you to keep your temper, and for me to begin it all over again for the third time.

Appendix B

Instructions for the flat segmentation experiment

Experiment I

The Guidelines

Objective: the goal of this experiment is to see how well people can find high-level topical structure in literature.

Background. You will be asked to split two book chapters into topically continuous segments. We refer to such segments as episodes. An episode may be an event (such as a conversation or a fight), a speculation by the author or one of the characters, a substantial descriptive segment (e.g., of nature, of a family, etc.), a flashback, etc . What matters is that there is more unity within an episode than between any two adjacent episodes. This unity may be along one or more of the following dimensions: unity of characters (e.g., main protagonists remain the same throughout an episode), unity of time and space (e.g., an episode occurs in one place during a continuous span of time), thematic unity (reflections on a particular topic) or a constant speed of narration. Looking at it another way, an episode boundary should coincide with a perceptible shift of topic. Of course, the topic shifts continually throughout discourse. Shifts between paragraphs are more pronounced than those between sentences, but in most cases they are still relatively minor. Your task is

to identify shifts of focus that occur at the top level, that is the most noticeable ones (i.e., if your segmentation consists of individual paragraphs, you are probably working with details too much. We are interested in what you have to say of the bigger picture.) Please find below a chapter from The Hound of the Baskervilles by A. Conan Doyle and a possible segmentation of it. It is important to understand that this is a subjective process and there are no right and wrong answers. Therefore, you may or may not agree with the example segmentation.

———————————— EXAMPLE of a possible segmentation.

The Hound of the Baskervilles By A. Conan Doyle Chapter 1 Mr. Sherlock Holmes 1 Mr. Sherlock Holmes, who was usually very late in the mornings, save upon those not infrequent occasions when he was up all night, was seated at the breakfast table. I stood upon the hearth-rug and picked up the stick which our visitor had left behind him the night before. It was a fine, thick piece of wood, bulbous-headed, of the sort which is known as a ”Penang lawyer.” Just under the head was a broad silver band nearly an inch across. ”To James Mortimer, M.R.C.S., from his friends of the C.C.H.,” was engraved upon it, with the date ”1884.” It was just such a stick as the old-fashioned family practitioner used to carrydignified, solid, and reassuring.
2 ”Well, Watson, what do you make of it?”
3 Holmes was sitting with his back to me, and I had given him no sign of my occupation.
4 ”How did you know what I was doing? I believe you have eyes in the back of your head.”
5 ”I have, at least, a well-polished, silver-plated coffee-pot in front of me,” said he. ”But, tell me, Watson, what do you make of our visitor’s stick? Since we have been so unfortunate as to miss him and have no notion of his errand, this accidental souvenir becomes of importance. Let me hear you reconstruct the man by an examination of it.”
6 ”I think,” said I, following as far as I could the methods of my companion, ”that Dr. Mortimer is a successful, elderly medical man, well-esteemed since those who know him give him this mark of their appreciation.”
7 ”Good!” said Holmes. ”Excellent!”
8 ”I think also that the probability is in favour of his being a country practitioner who does

a great deal of his visiting on foot."
9 "Why so?"
10 "Because this stick, though originally a very handsome one has been so knocked about that I can hardly imagine a town practitioner carrying it. The thick-iron ferrule is worn down, so it is evident that he has done a great amount of walking with it."
11 "Perfectly sound!" said Holmes.
12 "And then again, there is the 'friends of the C.C.H.' I should guess that to be the Something Hunt, the local hunt to whose members he has possibly given some surgical assistance, and which has made him a small presentation in return."
13 "Really, Watson, you excel yourself," said Holmes, pushing back his chair and lighting a cigarette. "I am bound to say that in all the accounts which you have been so good as to give of my own small achievements you have habitually underrated your own abilities. It may be that you are not yourself luminous, but you are a conductor of light. Some people without possessing genius have a remarkable power of stimulating it. I confess, my dear fellow, that I am very much in your debt."
14 He had never said as much before, and I must admit that his words gave me keen pleasure, for I had often been piqued by his indifference to my admiration and to the attempts which I had made to give publicity to his methods. I was proud, too, to think that I had so far mastered his system as to apply it in a way which earned his approval. He now took the stick from my hands and examined it for a few minutes with his naked eyes. Then with an expression of interest he laid down his cigarette, and carrying the cane to the window, he looked over it again with a convex lens.
15 "Interesting, though elementary," said he as he returned to his favourite corner of the settee. "There are certainly one or two indications upon the stick. It gives us the basis for several deductions."
16 "Has anything escaped me?" I asked with some self-importance. "I trust that there is nothing of consequence which I have overlooked?"
17 "I am afraid, my dear Watson, that most of your conclusions were erroneous. When I said that you stimulated me I meant, to be frank, that in noting your fallacies I was occasionally guided towards the truth. Not that you are entirely wrong in this instance. The man is certainly a country practitioner. And he walks a good deal."
18 "Then I was right."

19 ”To that extent.”
20 ”But that was all.”
21 ”No, no, my dear Watson, not allby no means all. I would suggest, for example, that a presentation to a doctor is more likely to come from a hospital than from a hunt, and that when the initials ’C.C.’ are placed before that hospital the words ’Charing Cross’ very naturally suggest themselves.”
22 ”You may be right.”
23 ”The probability lies in that direction. And if we take this as a working hypothesis we have a fresh basis from which to start our construction of this unknown visitor.”
24 ”Well, then, supposing that ’C.C.H.’ does stand for ’Charing Cross Hospital,’ what further inferences may we draw?”
25 ”Do none suggest themselves? You know my methods. Apply them!”
26 ”I can only think of the obvious conclusion that the man has practised in town before going to the country.”
27 ”I think that we might venture a little farther than this. Look at it in this light. On what occasion would it be most probable that such a presentation would be made? When would his friends unite to give him a pledge of their good will? Obviously at the moment when Dr. Mortimer withdrew from the service of the hospital in order to start in practice for himself. We know there has been a presentation. We believe there has been a change from a town hospital to a country practice. Is it, then, stretching our inference too far to say that the presentation was on the occasion of the change?”
28 ”It certainly seems probable.”
29 ”Now, you will observe that he could not have been on the staff of the hospital, since only a man well-established in a London practice could hold such a position, and such a one would not drift into the country. What was he, then? If he was in the hospital and yet not on the staff he could only have been a house-surgeon or a house-physicianlittle more than a senior student. And he left five years agothe date is on the stick. So your grave, middle-aged family practitioner vanishes into thin air, my dear Watson, and there emerges a young fellow under thirty, amiable, unambitious, absent-minded, and the possessor of a favourite dog, which I should describe roughly as being larger than a terrier and smaller than a mastiff.”
30 I laughed incredulously as Sherlock Holmes leaned back in his settee and blew little

wavering rings of smoke up to the ceiling.

31 ”As to the latter part, I have no means of checking you,” said I, ”but at least it is not difficult to find out a few particulars about the man’s age and professional career.” From my small medical shelf I took down the Medical Directory and turned up the name. There were several Mortimers, but only one who could be our visitor. I read his record aloud.

32 ”Mortimer, James, M.R.C.S., 1882, Grimpen, Dartmoor, Devon. House-surgeon, from 1882 to 1884, at Charing Cross Hospital. Winner of the Jackson prize for Comparative Pathology, with essay entitled ’Is Disease a Reversion?’ Corresponding member of the Swedish Pathological Society. Author of ’Some Freaks of Atavism’ (Lancet 1882). ’Do We Progress?’ (Journal of Psychology, March, 1883). Medical Officer for the parishes of Grimpen, Thorsley, and High Barrow.”

33 ”No mention of that local hunt, Watson,” said Holmes with a mischievous smile, ”but a country doctor, as you very astutely observed. I think that I am fairly justified in my inferences. As to the adjectives, I said, if I remember right, amiable, unambitious, and absent-minded. It is my experience that it is only an amiable man in this world who receives testimonials, only an unambitious one who abandons a London career for the country, and only an absent-minded one who leaves his stick and not his visiting-card after waiting an hour in your room.”

34 ”And the dog?”

35 ”Has been in the habit of carrying this stick behind his master. Being a heavy stick the dog has held it tightly by the middle, and the marks of his teeth are very plainly visible. The dog’s jaw, as shown in the space between these marks, is too broad in my opinion for a terrier and not broad enough for a mastiff. It may have beenyes, by Jove, it is a curly-haired spaniel.”

36 He had risen and paced the room as he spoke. Now he halted in the recess of the window. There was such a ring of conviction in his voice that I glanced up in surprise.

37 ”My dear fellow, how can you possibly be so sure of that?”

38 ”For the very simple reason that I see the dog himself on our very door-step, and there is the ring of its owner. Don’t move, I beg you, Watson. He is a professional brother of yours, and your presence may be of assistance to me. Now is the dramatic moment of fate, Watson, when you hear a step upon the stair which is walking into your life, and you know not whether for good or ill. What does Dr. James Mortimer, the man of science, ask of

Sherlock Holmes, the specialist in crime? Come in!"
39 The appearance of our visitor was a surprise to me, since I had expected a typical country practitioner. He was a very tall, thin man, with a long nose like a beak, which jutted out between two keen, gray eyes, set closely together and sparkling brightly from behind a pair of gold-rimmed glasses. He was clad in a professional but rather slovenly fashion, for his frock-coat was dingy and his trousers frayed. Though young, his long back was already bowed, and he walked with a forward thrust of his head and a general air of peering benevolence. As he entered his eyes fell upon the stick in Holmes's hand, and he ran towards it with an exclamation of joy. "I am so very glad," said he. "I was not sure whether I had left it here or in the Shipping Office. I would not lose that stick for the world."
40 "A presentation, I see," said Holmes.
41 "Yes, sir."
42 "From Charing Cross Hospital?"
43 "From one or two friends there on the occasion of my marriage."
44 "Dear, dear, that's bad!" said Holmes, shaking his head.
45 Dr. Mortimer blinked through his glasses in mild astonishment.
46 "Why was it bad?"
47 "Only that you have disarranged our little deductions. Your marriage, you say?"
48 "Yes, sir. I married, and so left the hospital, and with it all hopes of a consulting practice. It was necessary to make a home of my own."
49 "Come, come, we are not so far wrong, after all," said Holmes. "And now, Dr. James Mortimer "
50 "Mister, sir, Mistera humble M.R.C.S."
51 "And a man of precise mind, evidently."
52 "A dabbler in science, Mr. Holmes, a picker up of shells on the shores of the great unknown ocean. I presume that it is Mr. Sherlock Holmes whom I am addressing and not "
53 "No, this is my friend Dr. Watson."
54 "Glad to meet you, sir. I have heard your name mentioned in connection with that of your friend. You interest me very much, Mr. Holmes. I had hardly expected so dolichocephalic a skull or such well-marked supra-orbital development. Would you have any objection to my running my finger along your parietal fissure? A cast of your skull, sir, until the original is available, would be an ornament to any anthropological museum. It is not my intention

to be fulsome, but I confess that I covet your skull."
55 Sherlock Holmes waved our strange visitor into a chair. "You are an enthusiast in your line of thought, I perceive, sir, as I am in mine," said he. "I observe from your forefinger that you make your own cigarettes. Have no hesitation in lighting one."
56 The man drew out paper and tobacco and twirled the one up in the other with surprising dexterity. He had long, quivering fingers as agile and restless as the antennae of an insect.
57 Holmes was silent, but his little darting glances showed me the interest which he took in our curious companion.
58 "I presume, sir," said he at last, "that it was not merely for the purpose of examining my skull that you have done me the honour to call here last night and again to-day?"
59 "No, sir, no; though I am happy to have had the opportunity of doing that as well. I came to you, Mr. Holmes, because I recognized that I am myself an unpractical man and because I am suddenly confronted with a most serious and extraordinary problem. Recognizing, as I do, that you are the second highest expert in Europe "
60 "Indeed, sir! May I inquire who has the honour to be the first?" asked Holmes with some asperity.
61 "To the man of precisely scientific mind the work of Monsieur Bertillon must always appeal strongly."
62 "Then had you not better consult him?"
63 "I said, sir, to the precisely scientific mind. But as a practical man of affairs it is acknowledged that you stand alone. I trust, sir, that I have not inadvertently "
64 "Just a little," said Holmes. "I think, Dr. Mortimer, you would do wisely if without more ado you would kindly tell me plainly what the exact nature of the problem is in which you demand my assistance."

Segmentation of Chapter 1, The Hound of the Baskervilles by A. Conan Doyle.

Episode 1: paragraphs 1-17: Watson's conjecture

Episode 2: paragraphs 15-38: Holme's conjecture (notes: hard to establish the exact boundary between Episodes 1 and 2. Paragraphs 13-16 serve as a transition)

Episode 3: paragraphs 39-62: real Dr. Mortimer

The Task.

Please print this package before beginning work. You may record your segmentation on the computer, but we would like you to have a paper copy of the chapter with numbered paragraphs before your eyes.

Important: Please record the time when you start the experiment and also when you finish it. If you take breaks, please record the times when you are not working.

Please read carefully the following four chapters from The Moonstone by Wilkie Collins. After you do so, create an outline in a format similar to the one given in the example above. Record spans or paragraphs that, in your opinion, correspond to our definition of an episode. For each span, write a very short description characterizing it (a short sentence or a phrase). If you have trouble characterizing any span, please record that fact (write 'hard to classify', 'transition', or something similar).

In general, if you have any comments as to whether you find certain portions of the text easy or hard to segment, please record them.

Appendix C

Weights used in the experiments with coreferential similarity

Weights applied to the *Moonstone* datasets:

> <personal_pronouns_anim: 4, demonstr_pronouns_anim: 2,
> proper_names_anim: 1, def_np_anim: 0.5, indef_np_anim: 0,
> pronouns_inanim: 2, demonstr_pronouns_inanim: 2, proper_names_inanim: 0,
> def_np_inanim: 0, indef_np_inanim: 0>

Weights applied to the AI lectures and to the WIkipedia datasets:

> <personal_pronouns_anim: 2, demonstr_pronouns_anim: 3,
> proper_names_anim: 0, def_np_anim: 0, indef_np_anim: 0, pronouns_inanim:
> 4, demonstr_pronouns_inanim: 5, proper_names_inanim: 0, def_np_inanim: 0,
> indef_np_inanim: -1>