



8-2001

An improved method for text summarization using lexical chains

Charles Ray Byler

Follow this and additional works at: https://trace.tennessee.edu/utk_graddiss

Recommended Citation

Byler, Charles Ray, "An improved method for text summarization using lexical chains. " PhD diss., University of Tennessee, 2001.
https://trace.tennessee.edu/utk_graddiss/8476

This Dissertation is brought to you for free and open access by the Graduate School at TRACE: Tennessee Research and Creative Exchange. It has been accepted for inclusion in Doctoral Dissertations by an authorized administrator of TRACE: Tennessee Research and Creative Exchange. For more information, please contact trace@utk.edu.

To the Graduate Council:

I am submitting herewith a dissertation written by Charles Ray Byler entitled "An improved method for text summarization using lexical chains." I have examined the final electronic copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Computer Science.

Michael D. Vose, Major Professor

We have read this dissertation and recommend its acceptance:

Bethany K. Dumas, Bruce J. MacLennan, Bradley T. Vander Zanden

Accepted for the Council:

Carolyn R. Hodges

Vice Provost and Dean of the Graduate School

(Original signatures are on file with official student records.)

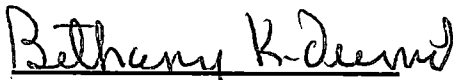
To the Graduate Council:

I am submitting herewith a dissertation written by Charles Ray Byler entitled "An Improved Method for Text Summarization Using Lexical Chains." I have examined the final copy of this dissertation for form and content and recommend that it be accepted in partial fulfillment of the requirements for the degree of Doctor of Philosophy, with a major in Computer Science.




Michael D. Vose, Major Professor

We have read this dissertation
and recommend its acceptance:



Bethany K. Dumas

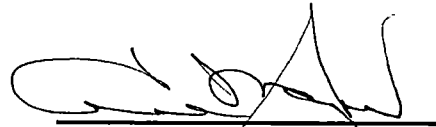


Bruce J. MacLennan



Bradley T. Vander Zanden

Accepted for the Council:



Interim Vice Provost and
Dean of The Graduate School

**An Improved Method for Text
Summarization Using Lexical Chains**

A Dissertation

Presented for the

Doctor of Philosophy Degree

The University of Tennessee, Knoxville

Charles Ray Byler

August 2001

Copyright © Charles Ray Byler, 2001

All rights reserved

Dedication

This work is dedicated to my grandparents, James David and Eva Marie Byler, without whom life would not have been possible. As my grandfather would have said:

Ring the bells that still can ring. Forget your perfect offering. There is a crack in everything. That's how the light gets in.

Leonard Cohen

Acknowledgments

First, I would like to thank my advisor, Dr. Michael Vose, for guiding and prodding me. Next, I would like to thank Dr. Bruce MacLennan, Dr. Brad Vander Zanden, and Dr. Bethany Dumas for agreeing to be on my committee and for all their advice and assistance.

I would like to thank my *guinea pigs* Mark Aubrey, Terri Brogdon, Dr. Jeff Case, Mary Gibson, Matt Hecht, Dr. Jennifer Koella, Faith McCullough, Sandra McLeod, Steve Moulton, Bill Norton, Brian Park, David Spakes, and Mike Vaughn for volunteering their time for comparing and judging program outputs.

I would like to acknowledge the contributions made by the NEC Research Institute, Princeton, NJ, *citeseer.nj.nec.com*, the best site on the web for obtaining free copies of research papers.

This work could not have been completed without the tools provided by the free software community. Text editing was done with vi and joe. Spellchecking was done with ispell. \LaTeX (for Unix systems) and Mik \TeX (for MS-DOS systems) handled all document formatting automatically. All \LaTeX and Bib \TeX packages are available from CTAN (Comprehensive Tex Archive Network) at *www.ctan.org*. Mik \TeX is available from *www.miktex.de*. \LaTeX templates are available from *www.utk.edu/~thesis/latex.htm*. All figures were created using the xfig package, available from *www.xfig.org*. Xfig was used to produce encapsulated postscript files which were then incorporated into the \LaTeX source using the epsfig package. Headers were created with the fancyhdr package written by Piet van Oostrum. The Harvard

citation and bibliography style are part of the Harvard package. The bibliography itself was created using the BIB_TE_X program. Page numbers were automatically matched to figure references with the varioref package. An excellent, free book on L_AT_EX, *The Not So Short Introduction to L_AT_EX2_ε*, by Tobias Oetiker is available from www-h.eng.cam.ac.uk.

On the commercial software side, least squares error solutions were generated using the DataFit program from Oakdale Engineering. The Microsoft Word97 AutoSummarize tool and the Sinope text summarizer from Carp Technologies were used for comparison purposes only.

Abstract

This work is directed toward the creation of a system for automatically summarizing documents by extracting selected sentences. Several heuristics including position, cue words, and title words are used in conjunction with lexical chain information to create a salience function that is used to rank sentences for extraction. Compiler technology, including the Flex and Bison tools, is used to create the AutoExtract summarizer that extracts and combines this information from the raw text. The WordNet database is used for the creation of the lexical chains. The AutoExtract summarizer performed better than the Microsoft Word97 AutoSummarize tool and the Sinope commercial summarizer in tests against ideal extracts and in tests judged by humans.

Contents

1	Introduction	1
1.1	What Is Automatic Text Summarization?	1
1.2	Problem Statement	3
1.3	Why Is Automatic Text Summarization Important?	4
1.4	Where Do We Begin?	5
1.5	General Approach	9
2	Background	11
2.1	Everyday Examples of Summarization	11
2.2	An Illustrative Example of Text Summarization	13
2.3	The Phases of Text Summarization	15
2.4	Significant Text Summarization Methods	18
2.5	Condensation Strategies	19
2.6	Characterizing Text Summarization	20
2.6.1	Surface-Level Approaches	22

2.6.2	Corpus-based Approaches	22
2.6.3	Entity-Level Approaches	23
2.6.4	Discourse-Level Approaches	24
3	Historical Development of Text Summarization	28
3.1	The Early Years	29
3.1.1	Luhn's Summarizing System	31
3.1.2	The TRW study	35
3.1.3	ADAM - Automatic Document Abstracting Method	40
3.1.4	Semantic Networks	42
3.2	Systems Based on Cognitive Science	43
3.2.1	SAM and PAM	45
3.2.2	FRUMP (Fast Reading Understanding and Memory Program)	50
3.2.3	CYRUS	58
3.2.4	SUSY - a Summarizing System for Scientific texts	59
3.2.5	SCISOR - System for Conceptual Information Summarization, Organization, and Retrieval	60
3.2.6	PAULINE	61
3.2.7	TIPSTER	62
4	State of the Art Text Summarization	65
4.1	Commercial Systems	68
4.1.1	Microsoft Summarizer	70

4.1.2	National Research Council of Canada - Extractor	72
4.1.3	Carp Technologies - Sinope (formerly Sumatra)	72
4.2	New Extraction Ideas	74
4.2.1	Extracting Paragraphs Instead of Sentences	75
4.2.2	SweSum	76
4.3	The Trainable Document Summarizer by Kupiec	78
4.4	Modern Systems Using Surface Level Features	80
4.4.1	Location Feature	81
4.4.2	Cue Phrase Feature	82
4.4.3	Sentence Length Feature	83
4.4.4	Thematic Word Feature	83
4.4.5	Uppercase Word Feature	84
4.4.6	Title Feature	84
4.5	Using Concept Counting Instead of Word Counting	85
4.6	Capsule Overviews	85
4.7	Systems Using Lexical Chains	86
4.7.1	Morris and Hirst	88
4.7.2	Barzilay	88
4.7.3	Green	91
4.7.4	Andersen	93
4.8	WordNet	94
4.9	TextTiling	97

4.10	Discourse Structures	98
4.10.1	Using Rhetorical Structures to Make Relevance Judgments . .	98
4.10.2	Pruning Rhetorical Structures to Produce Summaries	99
4.11	Combination Methods	100
4.11.1	SUMMARIST	100
4.11.2	SIMPR - Structured Information Management: Processing and Retrieval	102
4.12	Restricted Data Set Solutions	102
4.12.1	STREAK	103
4.12.2	SUMMONS - SUMMARizing Online NewS articles	104
5	Closely Related Fields	106
5.1	Human Abstracting	106
5.2	Information Extraction	110
5.3	Library Science	111
6	An Improved Method for Text Summarization	112
6.1	Problem Re-statement	113
6.2	Learning Algorithm	113
6.3	Why combine factors?	114
6.4	Which parameters should we use in our salience function?	115
6.4.1	Location Information	116
6.4.2	Cue words - Bonus and Stigma Words	116

6.4.3	High-Content Word Counting	119
6.4.4	Title Words	120
6.4.5	Length Method	121
6.4.6	Uppercase Word Feature	121
6.5	What do human abstractors have to tell us?	121
6.6	How do we incorporate discourse knowledge?	122
6.7	How do we generate output?	123
7	Implementation - AutoExtract System Design	124
7.1	WordNet	126
7.2	Organization of AutoExtract Summarizer	128
7.2.1	Lexer	128
7.2.2	Parser	129
7.2.3	Lexical Chain Creation	133
7.2.4	Sentence Recognition	138
7.2.5	Paragraph Recognition	140
7.2.6	Sentence Scoring	140
7.3	Training and Optimization	142
7.4	AutoExtract Summarization by Example	145
8	Conclusions	200
8.1	Evaluation	200
8.2	Discussion of Overlap Test	202

8.2.1	XSRchainScore	206
8.2.2	MSRchainScore	206
8.2.3	uwordScore	207
8.2.4	bonusScore	207
8.2.5	lastSentScore	207
8.2.6	paraTitleScore	208
8.2.7	lastParaScore	208
8.2.8	firstSentScore	208
8.2.9	SRchainScore	208
8.2.10	stigmaScore	209
8.2.11	globalTitleScore	209
8.3	Extract Comparison Using Human Volunteers	209
8.4	Summary of Contributions	215
8.5	Future Research	219
8.5.1	Corpus Creation	219
8.5.2	Speed-up	219
8.5.3	Type of Document	220
8.5.4	Type of Sentence	220
8.5.5	Continuous Range for Bonus Words	220
8.5.6	Better Chaining	221
8.5.7	Better Saliency Function	222
8.5.8	Compound Words	222

8.5.9	Include Bonus Words in Chaining	223
	Bibliography	224
	Appendices	231
A	The Gettysburg Address	232
A.1	Original Text	232
A.2	Summary Produced by my AutoExtract Summarizer	232
A.2.1	3 Sentences	232
A.3	Summaries Produced by the Sinope Summarizer	233
A.3.1	3 Sentences	233
A.3.2	25 to 32 Percent Summary - 2 sentences	233
A.4	Summary Produced by Microsoft Word97	233
A.4.1	25 to 32 Percent Summary - 3 sentences	234
B	Newspaper Texts Used for Final Testing	235
B.1	Countryside Alliance Fights Trespass Law	235
B.1.1	Original Text	235
B.1.2	Optimal Extract - 35.29% of original sentences	237
B.2	Lyell "Did Not Report Heseltine Concern"	237
B.2.1	Original Text	237
B.2.2	Optimal Extract - 31.58% of original sentences	239
B.3	Major Is Ready For Long Fight With EC	240

B.3.1	Original Text	240
B.3.2	Optimal Extract - 33.33% of original sentences	241
B.4	International - Senate To Stage Public Inquiry On Whitewater	242
B.4.1	Original Text	242
B.4.2	Optimal Extract - 31.58% of original sentences	244
B.5	International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision	244
B.5.1	Original Text	244
B.5.2	Optimal Extract - 38.89% of original sentences	246
B.6	"Thatcher Circle" In Pergau Row	247
B.6.1	Original Text	247
B.6.2	Optimal Extract - 33.33% of original sentences	248
C	AutoExtract Summaries	250
C.1	Countryside Alliance Fights Trespass Law	250
C.2	Lyell "Did Not Report Heseltine Concern"	251
C.3	Major Is Ready For Long Fight With EC	252
C.4	International - Senate To Stage Public Inquiry On Whitewater	253
C.5	International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision	254
C.6	"Thatcher Circle" In Pergau Row	255
D	Sinope Summaries	257

D.1	Countryside Alliance Fights Trespass Law	257
D.2	Lyell "Did Not Report Heseltine Concern"	258
D.3	Major Is Ready For Long Fight With EC	259
D.4	International - Senate To Stage Public Inquiry On Whitewater	260
D.5	International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision	260
D.6	"Thatcher Circle" In Pergau Row	261
E	Microsoft Word97 Summaries	263
E.1	Countryside Alliance Fights Trespass Law	263
E.2	Lyell "Did Not Report Heseltine Concern"	264
E.3	Major Is Ready For Long Fight With EC	264
E.4	International - Senate To Stage Public Inquiry On Whitewater	265
E.5	International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision	266
E.6	"Thatcher Circle" In Pergau Row	266
F	Bonus, Stigma, and Common Words	268
F.1	Bonus Words	268
F.2	Stigma	269
F.3	Common Words - Stop Words - Noise	269
G	Glossary	272

CONTENTS

xvi

Vita

280

List of Tables

4.1	WordNet 1.6 Database Statistics.	95
4.2	WordNet 1.6 Pointer and Search Types.	96
8.1	Comparison of text summarization programs.	212
8.2	Human Extract Comparison For Zechner Texts.	212
8.3	Human Extract Comparison For Gettysburg Address.	213
8.4	Saliency Function Input Parameters.	218

List of Figures

3.1	The First Computer Generated Summary by H.P. Luhn.	36
3.2	SAM Example Script.	47
3.3	SAM Restaurant Frame.	48
3.4	SAM Restaurant Script.	49
3.5	Sample news article processed by FRUMP.	54
3.6	Frump Example One.	57
3.7	Frump Example One - With New Script.	57
3.8	CYRUS Example.	64
4.1	Gettysburg Address - Microsoft Word97 AutoSummarize Output. . .	71
4.2	Example Text with Capsule Overview.	87
4.3	WordNet Lookup Example.	90
4.4	Hypernym/hyponym chain relating elm to rose.	92
4.5	Green's example of text tagged with chain numbers.	95
6.1	Derivation of bonus, stigma, and stop words.	118

7.1	Overall Organization of Summarizer.	187
7.2	Simple Bison grammar used to create parse tree.	188
7.3	Internal Organization of TreeNode.	189
7.4	Processing LWORD, UWORD, or TERM.	190
7.5	Queue Processing.	191
7.6	Internal Organization of ChainNode.	192
7.7	Lexical chain stack organization.	193
7.8	Find Strong Relation.	194
7.9	Find Medium Strength Relation.	195
7.10	Internal Organization of HashNode.	196
7.11	Internal Organization of SortNode.	197
7.12	Partial Comma-Delimited Training Data.	198
7.13	DataFit Least Squares Solution.	199
8.1	Std Dev Graph of Human Extract Comparison For Zechner Texts. . .	212
8.2	Std Dev Graph of Human Extract Comparison For Gettysburg Address.	213

Chapter 1

Introduction

“When I use a word,” Humpty Dumpty said, in rather a scornful tone,
“it means just what I choose it to mean - neither more nor less.”
Lewis Carroll, Through the Looking Glass, Ch. VI

This chapter defines automatic text summarization, gives a precise problem statement, details why text summarization is important, and, in general terms, shows how we plan to proceed.

1.1 What Is Automatic Text Summarization?

As with artificial intelligence, there are many different definitions of automatic text summarization, and those definitions depend a great deal on the time period and the author's interests. In the 1950's definitions would have stressed extraction of text. In the 1960's and 1970's the definition would have shifted to emphasize abstraction instead of extraction. Later definitions shifted back to emphasize text extraction.

For our purposes the following working definition seems the most useful.

Text summarization is the process of extracting the more important pieces of information from the text as a whole to produce an accurate and complete abridged version.

The complete automatic text summarization, or abstraction, problem could be described as an *AI-complete* problem. That is, a complete solution to the automatic summarization problem would be equivalent to a solution to the general problem of creating artificial intelligence. I say this because human abstraction is the standard for text summarization, and human abstraction requires the *understanding* that is characteristic of intelligence.

Since I cannot hope to accomplish such a grand undertaking, I am focusing on a more feasible subset of the problem, automatic text summarization by sentence extraction. Creating this type of summary *only* requires the identification of the more important qualities of the text and returning those sentences that best exemplify those qualities. Although this type of summary is not necessarily coherent, it suffices for most text summarization requirements. Most automatic summarization systems today produce summaries using this tried and true extraction method. The main criticism of this method is that the extracted text sometimes lacks coherence.

(Paice 1990)

1.2 Problem Statement

The basic problem of text summarization by extraction is this: We have a text, a collection of sentences, and from this collection we wish to select or extract a subset with some quality that *best* represents the entire collection. But what criteria do we use to select for this quality? As the melancholy Dane remarked, "Aye, there's the rub."

The nature of this quality, and how best to determine it, has been the subject of continuing research since at least 1958 when Luhn (1958) proposed that text summarization could be accomplished by using the frequency of *significant* words to rank and select sentences. How then does our work solve the problem? How is our work unique (i.e. how does it differ from everything that has gone before)? How is our work better (or at least different) than what has been done previously? Finally, how does our work compare to the state of the art in automatic text summarization and does it advance the state of the art significantly? Answering these questions will be the primary focus of our work.

This work is unique in the parameter set that I have chosen to represent the quality needed for extraction. I will spell out what these parameters are, why they were chosen, how they represent the quality that I am seeking, how the salience function that combined them was optimized, and how they differ from the parameter sets chosen by earlier researchers. This work is unique by virtue of the nature and speed of the algorithms constructed. A clear delineation of these algorithms, what

makes them fast, and how they differ from those of previous researchers is covered in detail. This work is also unique in the way that sentences are scored, making it easy to select the desired degree of compression, without affecting the run time.

1.3 Why Is Automatic Text Summarization Important?

Since at least the invention of the printing press we have been faced with the ever increasing problem of information overload. Daily we are faced with the problem of sorting, collating, and dealing with, if not comprehending, an avalanche of information. Moreover, the success of the World Wide Web has only intensified this problem. There simply isn't enough time to read and digest everything. Yet, we are required to make decisions based on what information we do manage to read and process. This is the primary reason that professional papers have abstracts. But abstracts of professional papers are at best only a partial solution, since they address only a small subset of the information tsunami. We need abstracts or summaries for all texts. These abstracts or summaries could also be used as a means of organizing information and as an index for retrieving the full text. Because they are so very fast, computers are ideal candidates for attacking this automatic text summarization problem. Their speed can be used to offset the size of the problem.

The rationale for creating automatic summaries can be *summarized* as follows:

1. quick search - An abstract or summary can be used to search quickly for specific ideas or determine whether we need to read the full text or not.

2. save time - When necessary we can save time by just reading the summary.
(e.g. Cliff Notes. Let he who is without sin cast the first stone.)
3. guide - The summary can be used to guide our reading by hiding irrelevant details and highlighting important relations. In fact, in reading courses we are taught that we should read the summary first as preparation for the main reading task.

1.4 Where Do We Begin?

I'm rather like a mosquito in a nudist camp:
I know what I ought to do, but I don't know where to begin.
Stephen Bayne

We are all familiar with natural language processing software that can parse sentences and extract their *meaning*, at least in some fashion. But the meaning these programs extract is little more than the naming of sentence components (e.g. verbs, nouns, adjectives, clauses). There is no *understanding* of the text as a whole, no appreciation of context. Parsing then may be a useful tool to aid the main text summarization effort, but cannot be the foundation since, by our definition, text summarization must exhibit qualities of the text as a whole. For this reason, no researcher today uses parsing as the basis for automatic text summarization.

We are also familiar with so-called artificial intelligence (AI) programs such as Eliza (Weizenbaum 1976) which appear to carry on limited conversations. These

programs, however, quickly can be shown to be shams, and the programs clearly *understand* less than a parrot mimicking human speech. Parlour tricks of this sort provide no useful clues for understanding text summarization, and this is the reason that no researchers use them.

Are there then any programs that can really *understand* a complete text? Moreover, what exactly do we mean by *understanding a text*? And is understanding of the text really necessary to produce an abstract or summary? These questions were at the heart of the debate in the drive to produce text summarization software, and are still being debated today. Research since 1958, however, has shown that those text summarization algorithms that appear to understand the text as a whole by capturing some sort of rhetorical structure generate the best summaries (when judged by human summarizers).

The central problem of understanding is that everything is related to everything. Thus, to completely understand one thing you must understand everything. Just as the gravitational pull of the Earth on an object decreases as a function of the square of the distance of separation, but never quite goes to zero, so the relevance of *extraneous* items may approach zero but never reach it. The key to text summarization then is deciding where to place the dividing line. We must place the *best* sentences on one side of the dividing line, and the remainder on the other side. This is the function of the *relevance* or *saliency* functions of modern text summarization systems.

However, as humans have long noticed, *the whole may be greater than the sum of*

the parts. There may be qualities of the subset as a whole that we are missing if we just focus on finding the best sentences to go into that subset. It's possible that a summary composed of *bad* sentences might be better than a summary composed of *good* sentences, because of the properties of the collection as a whole. Thus, ideally during the training phase we would have to compute the total scores of every possible subset to find the parameters and constants that would give the best subset. Then, during the testing phase, we would have to use a large number of humans to render a statistical judgement for each of the possible subsets, then see where our results fell. Of course, the limitations of time, money, and other resources don't permit this. It is for this reason that we use heuristics and limit ourselves to selecting just the best sentences to create the best summary.

As outlined in the next chapter, researchers have attempted to create a solution to the problem of automatic computer text summarization since at least 1958. No solution to date, however, is completely adequate, and there is considerable room for improvement. Summarizing systems today don't perform nearly as well as humans for a variety of reasons, but primarily because humans know much more and can organize material faster and more flexibly. Still we may gain insight into summarization techniques by studying the techniques of human summarizers. Moreover, the great speed of computers coupled with adequate performance makes them suitable for dealing with masses of information in selected areas. **The bottom line, however, is that the summaries produced will be judged by humans, must be usable by humans, and should be validated by humans. That is what we**

do in the last chapter.

Computer summarization is most appropriate in processing large amounts of text. However, using a computer program to process our text requires that the text already be available in machine-readable form. In other words, the information must already be formatted. Hence, automatic summaries are often used in areas where text is already formatted, for example, in information retrieval systems, where large numbers of documents must be represented in condensed form. Ultimately, writers may be forced to write in a formatted fashion to ease the burden on summarizers, both automatic and human.

In professional environments, however, we often deal with specialized texts that respond to text summarization systems and strategies differently than systems that were designed for the general public (e.g. FRUMP). Thus, one of our first decisions must be whether to take a general or a specialized approach. That is, do we build a system that deals with specialized texts such as computer science papers or more general texts such as short stories or news stories? On the plus side, restricted, specialized environments have indexing and extracting tools that may make it easier to automate summarizing. **Still a general solution has greater utility to the public as a whole, and this is the approach I will take.**

1.5 General Approach

We know that we must review the work of researchers in the text summarization area, but are there other groups that we can learn from? For example, what do ordinary humans have to teach us about text summarization? If an ordinary human were asked to read a document then recap the main points in 25 words or less, he would undoubtedly try to understand the document as a whole then express its core meaning. How then does one understand a document as a whole?

When we read a text we assume that it is coherent, that is, that each sentence relates in some way to the previous one, and that the whole text has some structure related to its overall meaning. Thus, from ordinary humans we learn that we understand by capturing this structure. We then fill in any gaps in the structure with our world-knowledge, our common sense. Sadly, without access to a project such as Cyc (Guha & Lenat 1990), we cannot rely on world knowledge to fill in gaps in any summarizer. However, we do have several methods for capturing the overall rhetorical structure of the document. From researchers we learn that this rhetorical structure can be captured through the use of thematic words and lexical chains. From professional text summarizers we learn that positional information, like a skeleton, provides additional information about the structure of a document. **If we are to learn from humans, then automatic text summarization by extraction must somehow capture the structure of a document, then use this structure to select the subset of sentences to extract.** The next log-

ical question then is what parameters can be used to compute and capture this structure?

Chapter 2

Background

You can only find truth with logic
if you have already found truth without it.
G. K. Chesterton

In this chapter we examine text summarization from several different perspectives: everyday examples, an in-depth example, types, phases, approaches, and strategies. This is done to provide the reader with an overall appreciation of the techniques and terminology of the text summarization field. This provides background material for the next chapter where I trace the historical development of text summarization with attention to the major ideas and strategies that have been developed.

2.1 Everyday Examples of Summarization

Some everyday examples of summarization are:

- abridgments (e.g. Lamb's Shakespeare for children)
- biographies (e.g. resumes, obituaries)
- bulletins (e.g. weather forecasts, stock market reports)
- digests (e.g. TV guide)
- histories (e.g. H.G. Wells' *Outline of History*)
- minutes of a meeting
- news headlines (e.g. CNN headline news)
- outlines (e.g. Cliff Notes)
- movie previews
- reviews (book, CD, movie, and so forth)
- sound bites (e.g. political ads)
- synopses (e.g. soap opera listings)

The gold standard for judging all of these types of text summarization, of course, is provided by humans. As revealed by the following quote, humans are extremely capable summarizers with an incredible capacity for reducing information to its essence.

Vanity of vanities, saith the Preacher, vanity of vanities; all is vanity.
attributed to King Solomon in Ecclesiastes 1:2

Thus, did Solomon summarize all of human existence. This quote illustrates that complete summarization requires complete understanding, something that machines may never achieve. The goal of our present attempt is, of course, far more modest.

2.2 An Illustrative Example of Text Summarization

The problem of understanding text touches on the way that humans recognize situations, store information, make plans, and represent the world around them. Consider what happens in your head when you read the following quote:

“I’m at your orders; forgive me.” Dick devoured the troubled little face with his eyes. There was triumph in them, because he could not conceive that Maisie should refuse sooner or later to love him, since he loved her. ¹

Computer programs that automatically parse sentences are roughly 90-95% accurate in dividing and subdividing sentences into nouns, pronouns, verbs, and clauses. Were we to turn one loose on the previous quote, however, we quickly would discover that any parser by itself is insufficient for any sort of understanding. It doesn’t have any world knowledge about people and relationships (common sense). Nor is a grammar sufficient for organizing and showing the structure in the text. Templates (or scripts) popularized in the 1970s, attempted to encode this world knowledge or common sense about the way people behave in different situations. (The Cyc

¹The Light that Failed, Rudyard Kipling, Chapter 5

project in Texas is a modern variation on this scheme. (Guha & Lenat 1990)) However, the quote above is still far richer than what can be expressed with parsers and templates, as anyone realizes who has read Rudyard Kipling's *The Light that Failed*.

If, however, you haven't read the work, what can you learn from this one quote? You learn that there are two characters involved, Maisie and Dick, that Dick loves Maisie, and that Dick believes that Maisie loves him. We don't really know why he would believe such a thing since we have no real insight into his character. Moreover, we know nothing of Maisie's character. Not only are we ignorant of the facts themselves, but we are ignorant of which facts are important and which are irrelevant. Not only don't we know what the quote means, we don't even know what we need to know! We lack common sense in this arena because we lack context.

Dick's character was revealed in every previous thing written about him, and each time he appeared we added something to our knowledge of him and his role in the novel. Understanding this quote then depends in a very complex way on everything that came before. Moreover, our anticipation of what might happen as we read created a feedback loop that guided our reading. As we read we encode information, and this encoding guides our understanding of text. It is this understanding that an ordinary human would use to generate a summary of a text.

As we read we constantly reassess everything that we have already read. We make guesses about what is important and what we can afford to forget. And many studies cite this guessing as a fundamental principle of intelligence. Not only do we understand and store items in some complex, unknown way, but we

constantly reprocess our representation as we continue reading. At present, this level of performance is far beyond any mere computer, no matter how large or fast. It is beyond computers because of our lack of understanding of the original process, our lack of understanding of how to simulate that process through programming, and our lack of understanding of the resources required. Of course, no one really expects a computer to understand a novel by Rudyard Kipling.

The question of what we need to know leads us to observe the curious way in which information is encoded, sifted, stored, and recalled by the reader. We see that context is critical for the real understanding that characterizes human generated summaries. **A central problem for automatic text summarization then is how to capture this context. We will see that capturing discourse structure² through the use of lexical chains is one of the better solutions to this problem.**

2.3 The Phases of Text Summarization

Understanding how researchers have broken text summarization into different phases also will give us a better appreciation of the overall problem. We find that most authors divide text summarization into either two or three different phases. For example, Hovy & Lin (1997) break the process into three phases:

- Analysis.

²Discourse structure refers to text type and structural organization, structural entities.

The purpose of analysis is to identify the main topics in the text and there are a number of ways of doing this. For example, word counting at the concept or thematic level (more advanced than simple word counting) or identification of cue phrases (also known as meta-linguistic markers, bonus words, or meta-discourse markers) can be used to find the topic. This is the approach that many summarization systems have taken. This approach is fast, easy to compute, and generally has yielded good results.

An example of concept counting would be counting notebooks, laptops, and workstations under the generic concept *computer*. This is quite similar to the idea of grouping words in synonym sets (synsets) that we will see used in building lexical chains later, and linking synsets tends to focus on concepts instead of words.

Another method of identifying topics is rhetorical parsing, building an RST (Rhetorical Structure Tree) where the main nodes of the tree are used to identify topics. (Marcu 1997*b*) Summaries are then produced by extracting those sentences that correspond to major nodes. This approach, however, is complicated and computationally intensive (i.e. slow).

- Transformation: transforming the text into an intermediate representation.
- Synthesis.

The final phase, synthesis, refers to generating output. According to Hovy & Lin (1997) the two major ways of generating text are text extraction and text

abstraction. In text extraction complete pieces (e.g. sentences) of the original text are extracted and written to the summary. In text abstraction the original text is processed in some way (e.g. parsing followed by linguistic analysis) then the results are interpreted to produce a shorter text. Abstraction requires some sort of natural language generator as in Dalianis (1999). **Abstraction is far more difficult, error-prone, and slower than extraction, which is why we won't use it.**

Instead of Hovy's three phases, Jones (1993) condenses summarization into a two-step process:

- Building a representation from the source text.
- Creating an intermediate representation from the source then generating or synthesizing output.

Either way, we first must decide what information to capture in our source representation. In capturing source representation, there are three levels of processing to consider. Source representation can be captured with surface-level processing such as cue words, entity-level processing such as concept counting, or discourse level processing such as lexical chains. Any combination of these approaches could be chosen as the basis for our source representation.

2.4 Significant Text Summarization Methods

Two recurring methods that we will see used by researchers in the next chapter are lexical aggregation and syntactic aggregation.

Lexical Aggregation The lexical aggregation method was first described by Dalianis & Hovy (1996). In this method two concepts are replaced by a single concept. For example, “buying and selling” might become “business.” The added processing to accomplish this, of course, slows the overall summarization process and requires the encoding of a certain amount of world knowledge. It would be difficult to implement in a fast system.

Syntactic Aggregation Syntactic aggregation was also first described by Dalianis & Hovy (1996). Here the focus is on removing redundancy in a text. For example, “Mary walks and John walks” might become “Mary and John walk.” This is related to the fusion concept described by Hovy & Lin (1997). This method is not nearly as useful in practice as it might appear to be in this example, and again requires a great deal of processing.

Both of these methods can create a more readable summary, but are difficult to implement and time consuming to run. For these reasons, these methods will not be used in our summarization algorithm.

2.5 Condensation Strategies

Within the context of the two or three different phases of text summarization that I mentioned earlier, there are various condensation strategies that can be applied to the source text. Most researchers agree that there are three major ones to consider: (Jones 1998)

- omitting information - or conversely selecting what to retain
- aggregating information - compressing sentences into fewer words
- generalization - combining several sentences into one sentence

These condensation strategies could be applied during any of the text summarization phases. For example, during the analysis phase, we could combine the separate terms “buying” and “selling” into the single term, “business.”

One way of implementing generalization is by inferring a general concept from a sequence of actions (e.g. shopping for clothing might be inferred from a sequence of events such as going to the mall, trying on clothing, and so on.).

In the synthesis phase, multiple descriptions might be combined into a single one (e.g. “the largest planet in the solar system” could be replaced by “Jupiter”).

These condensation strategies make for a more *readable* summary, but are too complicated and computationally intensive for producing summaries in anything approaching real-time, and I will not be using them.

2.6 Characterizing Text Summarization

We also can gain a better appreciation of text summarization by looking at how different authors have characterized it.

Indicative versus informative Many researchers divide summaries into two broad categories, indicative and informative - quick categorization versus content processing. Indicative summaries are used to indicate the major topics in the source text (i.e. provide a map to the user). Informative summaries cover all the concepts in the source text as much as possible given the compression rate.

Top-Down versus Bottom-Up Text summarization systems also may be classified based on the approach used, for example, top down versus bottom up.

The top-down approach is user-driven. That is, the desires of the user, the user's needs or queries, drive the text summarization. The use of templates (e.g. FRUMP) is an example of a top-down approach.

The bottom-up approach is text-driven. That is, the bottom-up approach focuses on gathering everything that is *important*, where importance is measured by some sort of metric applied to a representation of the whole text. An example of a bottom-up approach would be using word frequency or some sort of lexical chains or graph.

Historical Categories Summarization schemes also may be broadly categorized into the classical approaches of the 1950's and 1960's, corpus-based approaches, those exploiting discourse structure, and knowledge-rich approaches.

Level of Processing Finally, text summarization may be characterized by the depth of processing:

- surface level processing
 - term frequency
 - location
 - presence of title words in sentence
 - cue words (bonus words)
- entity level processing
 - vocabulary overlap (similarity)
 - distance between units (proximity)
 - single thesaural relations such as synonymy
 - concept counting
- discourse level processing
 - lexical chains
 - document format

- topic threads
- rhetorical structure e.g. argumentation or narrative structure

2.6.1 Surface-Level Approaches

The surface-level approaches lie at the lowest level of processing. These approaches represent the source text in terms of shallow features such as word frequency which are then selectively combined in some sort of *relevance* or *salience* function that is used to select information for extraction. Surface level features include:

- thematic word frequencies - Luhn's approach
- position information - location in document or paragraph
- title words - title or heading words in the text or sometimes words from a user's query
- cue words - words such as "in summary," "our investigation," "in conclusion", "the paper describes", "important", and "in particular"

2.6.2 Corpus-based Approaches

Corpus-based methods are primarily based on the surface-level approaches mentioned above. Kupiec, Pedersen & Chen (1995) describe the use of a Bayesian classifier to extract sentences; this classifier is trained on feature-vectors constructed from sentences from the text that are labeled as extract-worthy (based on comparison with abstracts). (This is similar to training an artificial neural network). They

found that (on their data set) location information was the best single feature; the best mix was a combination of location, cue phrase (bonus word), and sentence length. **We will see constant reinforcement for this use of location and bonus words through the years.** This is similar to my approach, in that my approach is also a data classification scheme. My approach, however, makes use of discourse structure and uses regression analysis (least squares) to fit the function to the data.

Myaeng & Jang (1997), described a variant of this same method that they applied to technical texts in Korean. The authors found that the combination of cue words, sentence location, and the presence of title words in a sentence produced the best results. **We will see that the use of title words also shows up again and again.**

Aone, Gorfinsky, Larsen & Okurowski (1997), using term-based statistics in a variation of the Kupiec et al. (1995) approach, showed that different ways of aggregating terms could have a positive impact on summarization performance. But, as we have already noted, aggregation is time consuming and complicated.

2.6.3 Entity-Level Approaches

Entity-level approaches dig deeper into the source material than surface-level approaches and build internal representations of the text. These approaches often use graphs to represent patterns of connectivity and show what is important. Relationships between entities may include:

- similarity (e.g. vocabulary overlap. This is the approach used by Hearst (1994) for text segmentation.)
- proximity - distance between text units
- single thesaural relationships - synonymy, hypernymy, part-of-speech relations
- syntactic relations - parse tree relationships
- logical relations - agreement, contradiction, entailment, and consistency

Boguraev & Kennedy (1997) used an entity-level approach based on text cohesion. They selected and combined phrases based on syntactic parsing and resolution of relationships between the terms.

There are also summarization systems that deal with structured data that have no corresponding full-text source. These approaches also are considered to be entity-level approaches.

2.6.4 Discourse-Level Approaches

The deepest level of processing is reached with discourse-level approaches. Discourse-level approaches recognize that a text is not a mere collection of unrelated sentences. Rather, sentences in a text are about the same thing and connected to each other. These approaches attempt to create models of the overall structure of the text. These models may include:

- document format - e.g. html, ASCII text

- rhetorical structure - e.g. argumentation or narrative structure
- topic threads - as they are revealed in the text

Discourse structure is meant to capture the structure, the focus, the patterns of the text, but there is no clear, clean-cut definition as to exactly what this structure is or what constitutes it. So, naturally there is no one way of capturing this structure. Anything that captures this focus or these patterns may be said to be capturing discourse structure.

Mann & Thompson (1988) tried to be a little more exact when they elaborated their Rhetorical Structure Theory (RST). They spoke of rhetorical relations, which are relations between two non-overlapping text spans called the nucleus and the satellite. It was this relation that was said to capture discourse structure. This relation, however, could be almost any relationship that exists between words or groups of words.

For example, we could say that Hearst's TextTiling algorithm captures discourse structure since it uses a similarity measure to relate non-overlapping text tiles. Lexical chains (since they use lexical relationships such as synonymy to link words and groups of words (i.e. chains)) would also qualify as capturing discourse structure.

In a text, we call a sequence of words which have some sort of lexically cohesive structure a lexical chain.³ Such a chain may span short distances in the text, or it

³That is, a lexical chain is just a chain of words where the words have some sort of lexical relationship. Usually, that relationship is defined using the relationships that you find in a dictionary, lexicon, or thesaurus (e.g. synonymy).

may thread its way throughout the entire text. These lexical chains tend to indicate portions of a text that form a semantic unit. The core ideas about lexical chains have remained the same for all approaches, with researchers varying their implementation by different degrees. It is this fact, that the core ideas have remained the same, that allows us cite the work of previous researchers.

Lexical chains have been used for information retrieval by Stairmand (1996), for correction of malapropisms by Hirst & St-Onge (1995), and for text summarization by Barzilay & Elhadad (1997). Barzilay & Elhadad (1997) chose to insert terms into lexical chains based on relationships such as synonymy and hypernymy, then the strength of these chains was used to select sentences for the summary.

The overall method that I use for the creation of lexical chains is quite similar to the approach of Barzilay & Elhadad (1997). However, I do not sum the chain metrics to create an overall chain score that is used to select chains or the sentences that they correspond to. Instead, in my approach the three separate chain scores⁴ that I compute are kept separate, then used as three of the eleven input parameters to my salience function. To the best of my knowledge, I am the only one who does it this way.

I wanted to keep the three types of chain scores separate because I wanted to test the rank ordering that past researchers have used. That is, it has always been assumed that the extra-strong relation was most important (and weighted the

⁴extra-strong chain relation (word repetition), strong chain relation (e.g. synonymy), and medium-strength chain relation (e.g. link in an extended hypernym chain)

highest), and the medium-strength relation was the least important and weighted the least. As the results in the last chapter show, I was right to be skeptical of this ordering.

Chapter 3

Historical Development of Text Summarization

As far as the laws of mathematics refer to reality, they are not certain,
and as far as they are certain, they do not refer to reality.

Albert Einstein

In this chapter we trace the historical development of automatic text summarization from its beginnings in 1958 to the present day.

Chronologically, automatic text summarization systems can be divided into three general phases:

1. The early years from about 1958 to 1975. The approaches during this period were mainly linguistic, using surface features.
2. Systems following the emergence of cognitive science. The psychological ap-

proaches of the late 70's and early 80's were part of this period.

3. The hybrid systems of the 1990s.

3.1 The Early Years

By tracing the development of systems over time, we can clearly understand the basic principles of those early systems and how they evolved to what we use today. Moreover, the early systems are of more than historical interest because the ideas and fundamental principles of those systems are still valid. For example, summarization has always been concerned with the selection of the more important statements in a text. Also, many of the basic methods developed early in text summarization work are still in use today. For example, Microsoft Word97 AutoSummarize primarily relies on term frequency, a method first invented by Luhn (1958).

When judging the systems of this period, we need to consider what researchers of that time had to work with. When researchers first thought of automatically generating abstracts and summaries, they were dealing with punch cards, batch processing, and poor quality printouts. Computer software couldn't cope with all the characters used to print technical papers, so papers had to be cleaned up before processing. This cleanup also involved deleting all drawings and other graphics. Finally, the printed output was usually all in uppercase letters.

These systems were designed to produce abstracts or extracts of technical and scientific documents, but standards were fairly low, since anything was better than

nothing. Selected sentences were simply extracted from the source text and copied into the extract or abstract, and very little was known about the actual process of summarization. Instead, researchers relied on guesses based on practical experience. Moreover, both processing power and memory capacity were limited, and the programming languages of that time were designed for numerical data not character strings. Finally, monitors were not yet available, a deck of cards was the primary input, and a line printer was the primary output.

For these reasons, summarization algorithms were characterized by simple surface-level approaches, for example exploiting thematic features such as term frequency. (Luhn 1958, Rath, Resnick & Savage 1961) The first entity-level approaches based on syntactic analysis didn't appear until the early 1960s (Climenson, Hardwick & Jacobson 1961), and the use of location features was not developed until somewhat later.¹ (Edmundson 1968)

During this period Pollock & Zamora (1975) demonstrated the increased importance of cue phrases and first introduced the use of stigma words.² They described the first commercial application of automatic text summarization. This was an abstracting program at Chemical Abstracts Service which used cue-phrases specific to the field of chemistry. The system used both positive (bonus word) and negative (stigma word) to select sentences, which were then aggregated based on shallow linguistic analyses.

¹The location method is based on the intuition that headings, sentences at the beginning and end of the text, and text formatted in bold contains important information.

²Stigma words are those words that characterize sentences that we want to exclude from our summary.

Several major efforts in automatic text summarization took place during this period. The behavior of human abstractors was studied for clues as to how they made decisions about what to include and what to toss out. Efforts were made to create quality standards to be used for evaluating summaries, and the general principles of automatic summarizing were formulated. Finally, the first attempts were made to make the summary more coherent and readable.

During this period Mathis (1972) formulated the first automatic summarization task list:

1. read
2. analyze
3. apply selection and transformation rules to produce summary
4. format output
5. print

3.1.1 Luhn's Summarizing System

The computer text summarization field really began in 1958 when H. P. Luhn first suggested "the creation of computer abstracts" using the thematic term frequency method that we still use today. His intuition was that the more frequent non-common words represented the more important concepts in the text (e.g. "the" is common).

The "significance" factor of a sentence is derived from an analysis of its words. It is here proposed that the frequency of word occurrence in an article furnishes a useful measurement of word significance. It is further proposed that the relative position within a sentence of words having given values of significance furnishes a useful measurement for determining the significance of sentences. The significance factor of a sentence will therefore be based on a combination of these two measurements. (Luhn 1958)

Luhn (1958) observed that the frequency of words in a text that were important to a summary followed a Bell curve, which was to be expected. But, what was interesting was that the lower minimum of the curve corresponded to words that had the highest overall frequency in the text. He called these words "noise" and proposed that they be eliminated when processing the text by means of a "common-word list." These common words (e.g. "a", "an", "the") are the type of words included in present-day "stop-word lists." (See Figure 6.1 on page 118.) Word-frequency was used to select important statements for extraction and these extracted statements were then combined to produce the abstract or summary. This approach provides the baseline for judging all later systems. Edmundson (1968) later showed that this method by itself would give you only about 36% of the important sentences in a text.

Figure 3.1 on page 36 shows the first auto-abstract produced by Luhn (1958), using his method of sentence extraction. This is also the first time that weighting items based on their importance is mentioned. The final weights that determined the inclusion of the sentences in the summary are in brackets following the sentences.

Because of the problems of processing text that were mentioned above, Luhn

selected papers that didn't need any pre-processing. His procedure for generating automatic summaries was:

1. Punch text onto cards then transfer to magnetic tape.
2. Read text word by word and delete *Common* words using table look-up.

The remaining *content* words were associated with any punctuation that preceded or followed them, and their location in the original document was recorded.

3. Sort content words into alphabetical order.
4. Consolidate words of similar spelling by a rough approximation to a stemming algorithm (i.e. substitute the word stem for the word).

Successive pairs of word tokens were compared letter by letter. Tokens which represented less than 7 letter non-matches were assumed to be of the same word-type(i.e. to represent the same concept). This was one of the first attempts at stemming (reducing words to their stems). The frequency of occurrence of each word-type was then determined and low frequency word-types were deleted. The remaining word-types were considered to be significant.

We aren't sure why Luhn performed his proto-stemming operation, but we know why we do word stemming today. The reason is that there will be far more word matches if we use word stems rather than the words themselves. Moreover, high-frequency stem groups should tend to point to the main topics of the document. These topics are the structure, the context that we need

to understand the document as a whole. (In all stemming operations we first convert the word to lower case to provide even more matching possibilities.)

5. Sort remaining content words into location order.

6. Determine sentence values.

Sentences were divided into substrings, defined as significant words separated by no more than four non-significant words. Significant words separated from other significant words by more than four words were considered isolated and deleted. For each substring, a value was calculated by simply dividing the number of representative words by the total number of words. Sentences reaching a certain threshold, or a predetermined number of sentences, were selected for extraction to form the abstract.

7. Print summary.

Luhn's approach reduced text to sequences of words and discourse structure was never directly considered. (In retrospect, however, Luhn's method is very close to the "extra-strength" relation used in the construction of lexical chains, which do carry some discourse structure information.) However, because it is simple and clear-cut, researchers have continued to use Luhn's approach. His basic method of sentence selection was replaced in later research by more sophisticated techniques, but his basic idea for term frequency is still being used in commercial systems today (e.g. Microsoft Word97 AutoSummarize).

Luhn's extracts could function as abstracts, but Luhn's selection criteria were too simple for producing really usable summaries. I believe this was because he never considered discourse structure, that is information from the text as a whole.

3.1.2 The TRW study

The TRW (Thompson Ramo Wooldridge Inc.) study of automatic abstracting (Edmundson 1961, Edmundson 1963, Edmundson 1968, Wyllys 1968) was a watershed in the study of automatic text summarization. It was meant to simultaneously create a method for indicative abstracting and create a system that could be extended to handle efficiently new texts, new methods, and new selection criteria. It was doubly important because it considered all of the then known parameters of text summarization.

The TRW group considered five evaluation methods:

- comparison with *ideal* abstracts.
- retrieval tests based on abstracts
- statistical correlations
- intuitive value judgments.
- generation of test questions, based on document content, but answered from the abstracts by a sample population.

As noted earlier, the core of human abstracting is deciding relevance: what

Source: The Scientific American. Vol.196, No.2,68-94, February 1958
Title: Messengers of the Nervous System
Author: Amodeo S. Marazzi

Editor's sub-heading: The internal communication of the body is mediated by chemicals as well as by nerve impulses. Study of their interaction has developed important leads to the understanding and therapy of mental illness.

Auto-Abstract

It seems reasonable to credit the single-celled organisms also with a system of chemical communication by diffusion of stimulating substances through the cell, and these correspond to the chemical messengers (e.g. hormones) that carry stimuli from cell to cell in the more complex organisms. (7.0)

Finally, in the vertebrate animals there are special glands (e.g. the adrenals) for producing chemical messengers, and the nervous and chemical communication systems are intertwined: for instance, release of adrenaline by the adrenal gland is subject to control both by nerve impulses and by chemicals brought to the gland by the blood. (6.4)

The experiments clearly demonstrated that acetylcholine (and related substances) and adrenaline (and its relatives) exert opposing actions which maintain a balanced regulation of the transmissions of nerve impulses. (6.3)

It is reasonable to suppose that the tranquilizing drugs counteract the inhibitory effect of excessive adrenaline or serotonin or some related inhibitor in the human nervous system. (7.3)

Figure 3.1: The First Computer Generated Summary by H.P. Luhn.

should be put into the abstract and what not. This question was tackled by the TRW researchers and others (Rath et al. 1961, Resnick 1961) by looking at how consistent human abstractors were in sentence selection. Unfortunately, the answer was that abstractors were not very consistent over time. Moreover, the variation among different abstractors was even more pronounced. Yet, although human abstractors were only moderately consistent, their abstracts were still judged adequate.

Probably the most important contribution of the TRW study was that it demonstrated the value of combining summarization methods and parameters. The study showed the potential of four methods of sentence selection: the cue, key, title, and location methods.

The cue method uses word lists classified as bonus words (positive value), stigma words (negative weight), or null words (irrelevant to sentence selection). This method springs from the intuition that the relevance of a sentence is affected by the presence or absence of certain words. These cue word lists were computed statistically, using man-made abstracts as a reference. Figure 6.1 on page 118 shows how they could have been derived.

The key method that the TRW study used is really just another name for the thematic word frequency method first proposed by Luhn.

The title method weights sentences based on the number of title and subtitle words that appear in it, with main title words having a higher weight than subtitle words.

The location method is based on the observation of human abstractors that

important sentences are located right after headings that announce them and at the beginnings and endings of documents, sections, and paragraphs. Unfortunately, this position information has been shown to be genre dependent. (Kupiec et al. 1995, Hovy & Lin 1997, Teufel & Moens 1997)

Finally, the values produced by these four methods were summed together with a simple linear function.

The conclusions that the TRW group drew from their development experience are still of interest.

The most serious disadvantage of current computer-produced abstracts is that they consist of individual sentences of the original text, extracted according to one or more criteria. Not only do the extraction criteria require further research, but the resulting set of individual sentences presents problems of disjointness, incompleteness, redundancy, and the like. The ultimate goal of research in automatic abstracting is to enable a computer program to "read" a document and "write" an abstract of it in conventional prose style, but the path to this goal is full of unconquered obstacles. (Wyllys 1968)

Edmundson (1968) selected 200 chemistry papers and compared the abstracts produced by professional abstractors with the outputs produced by four different extraction methods. Edmundson (1968) came to the conclusion that more information was needed to select the right sentences for abstracts. He showed that position-based indicators could account for up to 53% of the important sentences in a summary. Later he showed that augmenting the use of term frequency with other features, including cue phrases (e.g. "significant," "impossible," "hardly"), title and heading words, and sentence location gave even better results.

His results showed that the best combination of factors for producing automatic extracts (when evaluated against human-made extracts) was achieved using a combination of parameters that omitted key words. Since the key word method when used alone also had the lowest correlation coefficient of all the four methods, Edmundson (1968) concluded that:

On the basis of these data it was decided to omit the Key method as a component in the preferred extracting system. These data confirm the hypothesis set forth previously (see [4]) that Key words, although important for indexing, may not be as useful for extracting. This decision has important consequences for an extracting system since avoiding frequency-counting the entire text results in considerable simplification and shorter running time for the computer program.

Today, however, we have much faster computers than Edmundson, so the computational overhead for indexing key words is much less. Moreover, today's methods (e.g. using the Free Software Foundation's Flex lexer) are much more efficient than the tools available to Edmundson. Also key words are a key component in the calculation of lexical chains, which we can use to capture discourse structure. Finally, Edmundson does not state the key-word weight computation formula anywhere in his paper, so we don't know how efficient his algorithm was for determining weights. Moreover, later studies have reconfirmed the importance of word frequency (key method) in selecting sentences for extraction.

A later study by Hovy & Lin (1997) showed that the position of important sentences in a text is genre dependent. Hovy & Lin (1997) found that in newspaper articles announcing computer products, the title is most important, followed by the

first sentence of the second paragraph, then the first sentence of the third paragraph. In Wall Street Journal articles, however, the order was: title, first sentence of the first paragraph, then second sentence of the first paragraph. **As other researchers have reported, the position method is a gamble where you may win big or lose big.**

It is now beyond question that future automatic abstracting methods must take into account syntactic and semantic characteristics of the language and the text: they cannot rely simply on gross statistical evidence. (Edmundson 1968)

Research to accomplish these suggestions is still under way in 2001.

The TRW study was meant to create an extensible system. The study, however, limited itself to cue, key, title, and location methods and the relative weights were integrated with a linear function. However, this approach failed to include any discourse structure. We will also be using the cue (bonus), key (thematic term frequency), title, and location parameters in our method, and our approach also uses a linear function to compute the salience function.

3.1.3 ADAM - Automatic Document Abstracting Method

The ADAM system for automatic document abstracting relied on two key components, a dictionary called the word control list (WCL) and a set of rules with associated functions for each WCL entry. The system was designed to produce abstracts based on the following criteria: (Mathis 1972)

1. exclude negative items, unless they are the sole results
2. exclude numbers, except for actual results
3. exclude rare characters and abbreviations
4. exclude nonessentials such as preliminary remarks, equations, footnotes, references, tables, quotations, charts, figures, graphs, catalog data, examples, explanations, speculations, opinions, and comparisons
5. include work objectives, results, and conclusions
6. include methods only if they are the main purpose of the paper
7. use the same terminology as in the original document
8. make abstract size approximately 10% of original document

ADAM primarily relied on the cue method (bonus word). Cue words or phrases were listed in the WCL along with a semantic weight and a syntactic value that was used for deciding whether the current sentence was to be deleted or retained. The WCL listed both (positive) phrases like "our work" and negative phrases such as "obvious" that supported the deletion of the sentence that contained them. The WCL contained about 700 entries, with 14 different semantic weights which matched functions that were used to process data. The syntactic markers, which usually corresponded to word classes, were organized the same way, but also included deletion markers.

Since the creators of ADAM wanted abstracts to be as coherent as possible, selected sentences were transformed to make them more readable. If a sentence to be included in the abstract required an antecedent (For example sentences with “hence” require an antecedent.), then the three preceding sentences were considered for inclusion in the abstract. If a related sentence could not be found, the selected sentence was rewritten to make it stand alone. If this was not possible, it was deleted.

As we said, the ADAM system relied primarily on the cue method. It also suffered from a lack of discourse knowledge and a lack of extensibility. Nevertheless, it adds support for the cue (bonus word) parameter that we will incorporate into our method.

3.1.4 Semantic Networks

The research of Skorohodko (1971) (Skorohodko 1981) was based on the intuition that automatic abstracting should consider the characteristics of the entire text, where those characteristics were defined by a semantic network. Two sentences were said to be semantically related if at least one noun occurred in both sentences, if the sentences contained two words which had been predefined as being semantically related (e.g. smoke and fire), or if the sentences contained two words which were related by the text itself. (This is close to the “similarity” measure that is used in TextTiling. See Section 4.9 on page 97.)

Skorohodko showed how to associate a weighted graph with a text, where nodes

corresponded to individual sentences; weighted links between nodes reflected the semantic overlap between the words of the corresponding sentences. On the basis of the graph, he assigned an importance score to each sentence that was based on the number of arcs that were incident to the node in question, the total number of nodes in the graph, and the number of sentences in the longest connected fragment of text formed after the removal of the selected sentence. (This idea of weighting nodes based on number of connections has been used in many different approaches.)

This approach does make use of limited discourse knowledge. However, it makes no use of surface features such as cue or location information. Moreover, it doesn't seem to be much different from TextTiling, which I found to be inadequate for use in text summarization. (See TextTiling in Section 4.9 on page 97.)

3.2 Systems Based on Cognitive Science

The 1980s saw an explosion of text summarization methods, especially entity-level approaches based on artificial intelligence such as the use of scripts (DeJong 1978, Lehnert 1981, Lehnert & Ringle 1982), logic and production rules (Fum, Guida & Tasso 1985*b*), semantic networks (Reimer & Hahn 1988), as well as hybrid approaches (Rau, Jacobs & Zernik 1989, Aretoulaki 1994). All of these approaches borrowed heavily from the "script" (or template) ideas popularized by psychology during that period. Templates or scripts were used to guide the information extraction process. During this same time period the demand for summarization systems

increased dramatically as large-scale databases and computer networks came into use and many of these new systems were used to summarize news reports.

McKeown, Robin & Kukich (1995), described techniques for efficiently packing information into sentences in two different summarizers: STREAK, which generated basketball game summaries by creating then revising, and PLANDOC, which used discourse planning to summarize the network planning activity of telephone engineers. The methods these summarizers used were based on analysis of human summaries from these two restricted domains.

While early work in automatic summarization focused on the practical aspects of abstraction, researchers during this period (inspired by cognitive science and psychology) did their best to incorporate a model of the human thought process. They primarily used cognitive schemata (singular schema), templates, or scripts to follow this model. Schemata include prior knowledge (context) about the organization of events, and organize incoming data according to these schemata (or templates). This expectation-driven understanding paves the way for summarization because it allows us to guess whether something is important or irrelevant. As soon as new information is entered into the schema, the schema structure itself tells us whether it is important or not. We know, for example, that in a description of a kidnapping, that both the victim and the ransom are important.

The more important instantiations of this expectation-driven summarization idea were SAM, PAM, FRUMP, CYRUS, and SUSY. (SUSY focused on specialized scientific texts.)

The work on SAM, PAM, and CYRUS showed that the template (script, schemata) idea is not extensible. It cannot be expanded to a general solution to text summarization, and has only limited success in restricted environments. Moreover, attempting to expand the repertoire of scripts slows the system unacceptably. (DeJong 1978, DeJong 1982, Fum, Guida & Tasso 1982, Fum, Guida & Tasso 1984, Fum, Guida & Tasso 1985*a*, Fum et al. 1985*b*)

The central problem with the template approach to expectation-driven summarization is that it is not extensible. These approaches try to fit every new text into a limited number of schemata. Though some approaches do allow the dynamic creation of new scripts, these scripts never fit perfectly. Moreover, the more scripts that the program creates, the larger the search area, and the slower it runs.

3.2.1 SAM and PAM

SAM and PAM (Peat 1988) were created in the 1970's by Roger Schank and his colleagues at Yale University as part of a new wave of programs designed to understand natural language. These programs made use of such devices as extended scripts, plans, themes, and goals designed to organize situations that characters might find themselves in, and the actions they might carry out. The belief was that humans employ these organizational schemes to anticipate what comes next and to look for certain actions in order to understand a story, just as our expectations guide us when we read. These researchers felt that a breakthrough in text summarization

would only come from insight gained from human problem solving.

As an example, suppose a character in a story walks into a restaurant. We would expect that he would sit down at a table, be visited by the waiter (or waitress), order a meal, eat it, pay for his food, then leave. SAM could contain this sort of information in a script like the one in Figure 3.2 on the next page.

Guided by such scripts SAM would parse and match sentences in a story to those in the script. SAM would also make use of the script to fill in any missing information as the examples in Figure 3.3 on page 48, Figure 3.4 on page 49 show.

Consider this example:

John went to a restaurant. He sat down. He got mad. He left.

Computer paraphrase:

John was hungry. He decided to go to a restaurant. He went to one. He sat down in a chair. A waiter did not go to the table. John became upset. He decided he was going to leave the restaurant. He left it. (Peat 1988)

This paraphrase, which was generated in English, Chinese, Russian, Dutch, and Spanish, extracted meaning from the simple story then used the appropriate script to expand this information and fill in the missing gaps.

SAM's inference that John left the restaurant because he wasn't served may not

Customer Goes To Restaurant

Players: customer, server, cashier

Props: restaurant, table, menu, food, check, payment, tip

Events:

1 customer goes to restaurant

2 customer goes to table

3 server brings menu

4 customer orders food

5 server brings food

6 customer eats

7 food server brings check

8 customer leaves tip for server

9 customer gives payment to cashier

10 customer leaves restaurant

Header: Event 1

Main Concept: Event 6

Figure 3.2: SAM Example Script.

Generic RESTAURANT Frame

Specialization of: Business-Establishment

Types:

range: (Cafeteria, Seat-yourself Wait-To-Be-Seated)

default: Wait-To-Be-Seated

if-needed: IF plastic-orange-counter THEN Fast-Food,
IF stack-of-trays THEN Cafeteria,
IF wait-for-waitress-sign or reservations-made
THEN Wait-To-Be-Seated,
OTHERWISE Seat-Yourself

Location:

range: an ADDRESS

if-needed: (Look at MENU)

Name:

if-needed: (Look at the MENU)

Food-Style:

range: (Burgers, Chinese,
American, Seafood, French)

default: American

if-added: (Update Alternatives of Restaurant)

Times-Of-Operation:

range: a Time-Of-Day

default: open evenings except Mondays

Payment-Form:

range: (Cash, Credit-Card, Check, Washing-Dishes-Script)

Event Sequence:

default: Eat-at-Restaurant Script

Alternatives:

range: all restaurants with same Food-Style

if-needed: (Find all Restaurants with same Food-Style)

Figure 3.3: SAM Restaurant Frame.

EAT-AT-RESTAURANT Script

Props: (Restaurant, Monday, Food, Menu, Tables, Chairs)

Roles: (Hungry-Persons, Wait-Persons, Chef-Persons)

Points-of-View: Hungry-Persons

Time-Of-Occurrence: Time-Of-Operation-Of-Restaurant)

Place-Of-Occurrence: (Location of Restaurant)

Event-Sequence:

first: Enter-Restaurant Script

then: if (Wait-To-Be-Seated-Sign or Reservations)

 then Get-Maitre-de's-Attention Script

then: Please-Be-Seated Script

then: Order-Food-Script

then: Eat-Food-Script unless (Long-Wait)

 when

 Exit-Restaurant-Angry Script

 then: if (Food-Quality was better than Palatable)

 then Compliments-To-The-Chef Script

 then: Pay-For-It-Script

finally: Leave-Restaurant Script

Figure 3.4: SAM Restaurant Script.

be correct but it is still a good guess.

PAM, which followed SAM, made use of themes and more general goals in its scripts. These scripts were still specific but allowed goals to be fulfilled in several different ways. If John were hungry he could still go into a restaurant, but he could also go to a grocery store, buy food, then return home and cook it.

All of these approaches were at least partially successful in understanding a very simple story, but a disadvantage of these approaches was the processing time required. Moreover, as additional scripts are added to the system to allow it to handle new situations the processing time really begins to slow down. This is one of the reasons that De Jong (next section) decided to work with the more restricted sketchy scripts and avoid parsing the whole text.

3.2.2 FRUMP (Fast Reading Understanding and Memory Program)

The most popular of the script based approaches was FRUMP. FRUMP was designed in the late 1970s by Gerald Francis de Jong (DeJong 1978, DeJong 1982) as a Yale University thesis project. It was designed to understand and create summaries of stories as they came in over the UPI wire service. FRUMP, however, relied heavily on domain dependent scripts and certainly would never have been able to understand the novels of Rudyard Kipling, but it could do more than appear to comprehend simple sentences.

FRUMP used a data structure called a "sketchy script" to organize its world knowledge. Sketchy scripts represented what might occur in certain situations such

as a demonstration, an earthquake, a labor strike, etc. Given a newspaper article, FRUMP selected the most appropriate sketchy script based on clues found in the article. Typically, these clues were words that consistently indicated the right sketchy script — words like Richter scale, death toll, and magnitude for earthquakes, or visit, dignitary, and meeting for a diplomatic visit. A summary could be created based on what had been activated in the sketchy script. (This script-based approach is quite similar to the present-day Cyc AI project in Texas. (Guha & Lenat 1990))

FRUMP's pragmatic and semantic knowledge was encoded in a knowledge base that was then used to predict general events that were likely to be reported. The text analyzer then tried to find instances of those expected events in the input text. Thus, FRUMP's interpretation was expectation-driven. On the basis of whether or not predicted events were found, FRUMP then reassessed its interpretation of the situation and made new predictions.

FRUMP used general knowledge about car accidents, earthquakes, diplomatic visits, and so on to make sense of the stories it read. De Jong called these knowledge outlines or templates "sketchy scripts" to distinguish them from the more complete scripts that were used in systems like SAM. SAM's more comprehensive scripts were simply too time consuming to use. De Jong, argued that since humans tend to skim over news items, all that was needed was a *rough* outline of a stylized situation. This sketchy script prompted FRUMP to expect certain situations and look for the major actors and consequences. A sketchy script contained only the important events that might occur in a situation, whereas other types of scripts were more comprehensive.

FRUMP had scripts for earthquakes, demonstrations, explosions, and other events.

The demonstration script, for example, expected the following events:

1. demonstrators arrive
2. demonstrators march
3. police arrive
4. demonstrators communicate with target
5. demonstrators attack the target
6. demonstrators attack the police
7. police attack the demonstrators
8. police arrest the demonstrators

But, here we must revisit the classic chicken and egg or bootstrapping problem. If FRUMP needed a script to make sense of an article, how did it decide which script was appropriate for that article before reading that article? To understand the text it had to use a script, yet how could it select that script unless it already understood the text? Here it was helped by the format of the news stories. In newspaper stories reporters usually give the main outlines of their story within the first few sentences. FRUMP would read the opening paragraph and if it couldn't select a script it rejected the story. (This worked because newspaper columnists write their stories so that they can be cut at any point to fit. The first paragraph is

the most important; the last paragraph is the least important.) Once a particular script was chosen, FRUMP used its script assistant to scan the story. However, as it scanned further, it was possible that something unexpected would occur and FRUMP, therefore, had the capacity to change to other scripts.

Unlike earlier systems FRUMP did not work by mechanically reading every word of the text, instead it performed a rapid scan, ignoring some sentences and jumping back to others. The system consisted of two parts or modules:

- a substantiator - this did the actual reading
- a predictor - this controlled the scanning and predicted the course of the story

For an earthquake (See Figure 3.5 on the following page.), we might use a structure that contained time, place, strength, injuries, and damage. Things not directly mentioned in the earthquake script, such as the color of the buildings, were considered unimportant. Some items in the script were critical, such as the strength of the quake and the number of victims. Other items, such as the reporting body, were less important.

For each expected event there was a description containing:

- constraints on script variables (e.g. demonstrators must be human)
- constraints between script variables (e.g. demonstrators and police must be in the same location)

Mount Vernon, Ill. (UPI) – A small earthquake shook several Southern Illinois counties Monday night, the National Earthquake Information Service in Golden, CO reported.

Spokesman Don Finley said the quake measured 3.2 on the Richter scale, "probably not enough to do any damage or cause any injuries." The quake occurred about 7:48 p.m. CST and was centered about 30 miles east of Mount Vernon, Finley said. It was felt in Richland, Clay, Jasper, Effington and Marion Counties.

Small earthquakes are common in the area, Finley said.

Sketchy script: \$earthquake

Summary: There was an earthquake in Illinois with a 3.2 Richter scale reading.

Figure 3.5: Sample news article processed by FRUMP.

- causation relations (e.g. any deaths in a vehicle accident must be due to the crash itself.).

Once scripts were available, the next problem was to activate the right script.

There were three different mechanisms for doing this:

1. explicit references - e.g. earthquake
2. implicit references - Police arresting demonstrators implies a demonstration event.
3. Event-induced activation took place if the key request of a sketchy script was detected in a text. The arrest script would be activated as soon as FRUMP found that someone had been arrested by the police.

To activate scripts FRUMP had to analyze the data, but full parsing wasn't necessary. Instead there was a bottom-up substantiator. It tried to verify from input, expectations derived from the current sketchy script.

In another example, a news item might appear to be about a kidnapping, so the kidnapping script might be chosen. The Predictor would then expect to read about a victim, a ransom note, and a large sum of money. The Predictor would then ask the Substantiator to look for the victim's name. The Substantiator would then scan the text, a piece at a time, and send that information back to the predictor which then would search for the ransom note. If, however, the Substantiator couldn't find any information in the story about a kidnapping victim, then the Predictor

might have made an error. The story might not be about a kidnapping at all, but about a robbery. The system would then try a different script. Once the Predictor's requirements were met the scanning would stop and FRUMP would print out a summary in English, French, Spanish, Russian, and Chinese. The whole process took less than half a minute and, since UPI stories only arrived about every five to seven minutes, the system could produce its multi-lingual summaries in real time.

De Jong's system contained some 48 scripts, and could handle about half of the stories that came over the wire service. Many of the stories in the other half were not about events covered by FRUMP's scripts or FRUMP was misled by the first few sentences and selected the wrong script. As a result, it missed the point of the story, drew incorrect conclusions, or correlated unrelated events. FRUMP'S overall success rate was poor, only about 5 percent of stories coming over the wire service were processed correctly. This average would have been better if there were a larger library of scripts, but that would have slowed the system significantly.

In Figure 3.6 on the next page FRUMP missed the point of the story because it didn't use the right sketchy script. With a new script FRUMP did better. See Figure 3.7 on the following page.

De Jong later worked on improvements in story comprehension. In one of these approaches the computer was provided with general world knowledge which it improved as it continued to read stories. In place of scripts, dynamic schema were used that were continually updated. Faced with a new situation, the computer made use of its general knowledge about people's goals and behavior and how the world works,

FRUMP

UPI Story March 26, 1979

INPUT:

JERUSALEM (UPI) A BOMB EXPLODED IN THE WALLED OLD CITY OF JERUSALEM MONDAY JUST ABOUT THE SAME TIME THE LEADERS OF EGYPT AND ISRAEL SIGNED A PEACE TREATY IN WASHINGTON.

A POLICE SPOKESWOMAN SAID THERE WERE "SOME" CASUALTIES BUT NO FURTHER DETAILS WERE IMMEDIATELY AVAILABLE.

THE BOMBING CAME DESPITE INCREASED SECURITY PRECAUTIONS BY ISRAELI ARMY TROOPS AND BORDER GUARDS AGAINST POSSIBLE ATTACKS BY ARABS OPPOSED TO THE TREATY.

BUT ARABS IN THE WEST BANK AND GAZA STRIP DECLARED MONDAY A DAY OF MOURNING, SHUTTING THEIR SHOPS FROM NABLUS TO GAZA TO PROTEST WHAT THEY SEE AS A BETRAYAL OF THEIR CAUSE.

SELECTED SKETCHY SCRIPT \$ AGREE

CPU TIME FOR UNDERSTANDING = 2763 MILLISECONDS

ENGLISH SUMMARY:

EGYPT AND ISRAEL HAVE AGREED TO A TREATY

Figure 3.6: Frump Example One.

SELECTED SKETCHY SCRIPT: \$ EXPLOSION

CPU TIME FOR UNDERSTANDING = 5250 MILLISECONDS

ENGLISH SUMMARY:

AN EXPLOSION IN JERUSALEM HAS INJURED SEVERAL PEOPLE.

Figure 3.7: Frump Example One - With New Script.

learned from that particular story, and then attempted to generalize. For example, when it was presented with a story about kidnapping and extortion it made use of its knowledge about theft, bargaining, and human goals and attempted to construct schema for "kidnapping" and "extortion." As further stories were read the system noted similarities and differences and further refined those new schema.

Because we are looking for a general solution, there is nothing in FRUMP's design that we can use for our present project. Scripts, schemes, templates and the like are not extensible, too brittle to fit all situations, too difficult to construct, too time consuming to implement, too slow to operate, and not successful enough as a general solution to warrant consideration right now. Templates, however, might be useful as an adjunct to a general method, if we were operating in a restricted domain. For example, if we were just dealing with articles about earthquakes, we might attempt to fit the article to a limited number of templates, then if it failed revert to the general solution.

3.2.3 CYRUS

CYRUS, developed at Yale University during the same time period as SAM, PAM, and FRUMP, used various heuristics to help it answer questions in an intelligent way. Its knowledge base was built up from information passed on to it from FRUMP and it dealt only with the lives of former Secretaries of State Cyrus Vance and Edmund Muskie. CYRUS is interesting because it attempted to make sense of its input,

that is to *understand* it, and its answers generally made sense. CYRUS answered questions about Cyrus Vance as shown in Figure 3.8 on page 64.

While FRUMP, SAM, PAM, and CYRUS moved beyond the level of understanding single sentences to the comprehension of complete paragraphs, they still could not really deal with stories, reports, and extended conversations.

3.2.4 SUSY - a Summarizing System for Scientific texts

Systems such as SUSY (Fum et al. 1982, Fum et al. 1984, Fum et al. 1985a, Fum et al. 1985b), TOPIC (Reimer & Hahn 1988), and SCISOR (Jacobs & Rau 1990) were similar to FRUMP. However, each experimented with different types of knowledge representation structures.

The SUSY system used the research findings of Kintsch (1974) (Kintsch & van Dijk 1978) as its basis. Summaries were tailored to the interests and goals of the user. Unfortunately, only parts of the system were ever implemented. The overall design, however, can be described based on the description supplied by Fum et al. (1982).

SUSY was intended to understand and summarize specialized scientific texts such as scientific essays. The user selected the input schema, the summary schema, and the text that SUSY would use. These schemata were retrieved from a library of basic text and summary structures, then adapted if necessary, as the user described how the input text was structured and what the summary should look like. Also, if

necessary, a totally new schema could be defined. As we have seen previously, these schemata provide an expectation-driven guide to the analysis of the text.

SUSY contained a parser that constructed a propositional representation of the text. (Kintsch 1974, Kintsch & van Dijk 1978). The natural language generator then took the propositions produced by the parser, that had not been pruned during importance ranking, and retrieved the corresponding linguistic elements from the original text. From these it compiled the summary text based on sentence models that it contained.

3.2.5 SCISOR - System for Conceptual Information Summarization, Organization, and Retrieval

SCISOR summarized newspaper stories using what the author described as "an event knowledge representation scheme." Jacobs & Rau (1990) presented it as an alternative to a traditional information retrieval system and SCISOR was designed to output conceptual structures rather than summaries.

SCISOR contained three abstraction levels that were inspired by ideas about human memory: event knowledge, abstract knowledge and semantic knowledge. SCISOR stored the conceptual representation of a story as an episodic network, where concepts were members of classes such as companies, offers, or mergers.

The association of stories with classes was used for retrieving the stories. Questions were analyzed in the same way as news articles. From its memory, SCISOR generated summaries in response to user queries. That is, the user's question deter-

mined the topic. The concepts that the question activated in the knowledge base were candidates for inclusion in the summary. After retrieval, the representations of the question and the story were matched. In this way, the answer and the most relevant articles were found.

This system is interesting because it was designed to produce summaries from the large-scale knowledge base in its memory. For example, from many different reports about corporate mergers in its memory, SCISOR could construct a single summary of all the events.

Supposedly, SCISOR could answer simple questions in English. A future goal was for SCISOR to produce summaries, but I haven't found any reports that that feature actually was implemented.

Without any feedback from actual summarization studies, it's difficult to see what, if anything, should be incorporated from SCISOR into our summarizer.

3.2.6 PAULINE

There are, as we all know, many ways of saying the same thing. As Shakespeare remarked, a man plays many parts in life. We adapt ourselves and our speech to our audience when we talk or write. Summarization systems before PAULINE, however, made no attempt to tailor their summaries to their audience as humans do. PAULINE (Hovy 1988), however, demonstrated that summarization could be tailored to the individual user, and was capable of producing more than 100 different

texts from one semantic representation. PAULINE started with a collection of goals, that described the effect that it was supposed to have on its audience, for example, change future behavior. In addition, the user provided PAULINE with conversational topics.

What makes PAULINE interesting are the relevance criteria that were used to produce an "utterance." PAULINE knew different scenarios and tailored this utterance to the listener and situation.

PAULINE showed promise in tailoring output to individual users but demonstrated little in terms of actual summary or abstract production.

3.2.7 TIPSTER

The TIPSTER Text Program was a government project led by the Defense Advanced Research Project Agency (DARPA). Its goal was to advance the state of the art in different text technologies through the cooperation of academic researchers and commercial developers. The tools that were developed were meant to be deployed throughout the academic and commercial communities. The program formally ended in the Fall of 1998.

TIPSTER focused on three main programs: document detection, locating documents containing information the user wanted; information extraction, locating specific information in a document; and text summarization.

The first phase of research from 1991 until 1994 focused on document detection. The MUC (Message Understanding Conferences) and TREC (Text Retrieval

Conferences) were part of this initiative.

Phase II from April 1994 until September 1996 focused on developing and defining an architecture for the system. The MET (Multilingual Entity Task) developed Chinese and Japanese document collections during this period.

Finally, during phase III from October 96 until the Fall of 1998, text summarization was added as a fundamental task area.

Before the TIPSTER program, North America and Europe combined had fewer than ten summarization research efforts, and three of those were commercial enterprises: Lexis-Nexis, Oracle, and Microsoft. None of the commercial systems produced satisfactory results, and there were no commonly accepted evaluation measures.

Unfortunately, I could not discover any TIPSTER tools or techniques that were useful for my text summarization effort.

USER: When was the last time you were in Egypt?
CYRUS: On Dec. 10, 1978.
USER: Why did you go there?
CYRUS: To negotiate the Camp David accords.
USER: Who did you talk to there?
CYRUS: With Anwar Sadat.
USER: Has your wife ever met Mrs. Begin?
CYRUS: Yes, most recently at a state dinner in Israel in Jan. 1980.
USER: Who have you discussed SALT with?
CYRUS: Carter, Brezhnev, Gromyko, other American and Russian diplomats, and Mustafa Khalil.

Figure 3.8: CYRUS Example.

Chapter 4

State of the Art Text

Summarization

We think in generalities, but we live in detail.
Alfred North Whitehead

In this chapter we cover the major ideas and methods that are in current use in automatic text summarization.

During the current period (late 1990s, 2000s) all the major approaches to text summarization are being explored very aggressively. However, most recent work has focused almost exclusively on extracts rather than abstracts, along with renewed interest in earlier surface-level approaches. This provides added reason for us to take an extraction approach to text summarization, and reinforces the decision to include surface-level parameters in our

approach (e.g. bonus words, thematic term frequency).

Historically, we have seen that summaries can be built on a deep semantic analysis of the source text or on shallow linguistic parameters such as:

- thematic term frequency

For example, early summarization systems (Luhn 1958) directly exploited linguistic source information, based on the intuition that the more frequent content words represented the more important concepts in the text.

- cue phrases

The cue phrase method is another linguistic method. This method uses meta-linguistic markers (e.g. “in conclusion,” “the paper describes”) to select important phrases (Edmundson 1968). This method is based on the hypothesis that such phrases provide a “rhetorical” context for identifying important sentences.

Work by Edmundson (1968), Paice (1981), Kupiec et al. (1995), and Teufel & Moens (1997) all showed that keywords or cue-words were a good indicator of importance. The recall of important sentences based on cue-words ranged from 40 to 50%. Cue-words as a major part of a hybrid system showed up in the work of Aone et al. (1997) and Myaeng & Jang (1997).

- location information

The location method is based on observation that headings, paragraphs at the beginning and end of the text, sentences at the beginning and end of

paragraphs, and text formatted in bold, contain important information for the summary (Hovy & Lin 1997).

The techniques cited above rely on shallow text clues and are easily computed. As reported in Paice (1990), location and cue phrases by themselves produce better results than the word frequency method, and also can be accurately computed. Recently, Kupiec et al. (1995) and Teufel & Moens (1997) used learning to combine several shallow heuristics (cue phrase, location, sentence length, word frequency and title), using a corpus of research papers with manually produced abstracts for training.

The most severe limitation on abstraction by location and cue phrase is its dependence on the text genre. The number and type of rhetorical markers changes drastically from *Scientific American* articles to *Business Week* articles. Ono, Sumita & Miike (1994) report large differences in accuracy when building a discourse representation from technical texts and one from newspaper texts. Techniques that rely on cue phrases or positional information can be a high risk gamble. Your results may vary from very poor to very good.

Methods that rely more on content do not suffer from this all or nothing syndrome. For example, word frequency is a good indicator of words that represent important concepts, and this is true of most texts independently of their style. Simple word frequency by itself, however, is insufficient. It contains information on the words in the text, but noth-

ing on the connection between the words. Adding information about the relations between words can significantly increase the quality of source abstraction (e.g. with lexical chains).

The limitations of using frequency information by itself can be seen in the following example from Barzilay & Elhadad (1997).

1. Dr. Kenny has invented an anesthetic machine . This device controls the rate at which an anaesthetic is pumped into the blood.
2. Dr. Kenny has invented an anesthetic machine. The Doctor spent two years on this research.

The words “Dr. Kenny” and “machine” appear the same number of times in both sentences. The first sentence, however, is about the machine, while the second sentence is about Dr. Kenny.

To capture the meaning of sentences we must somehow capture this difference. Halliday & Hasan (1976) showed that we can do so by capturing the cohesion of the sentence parts. This is part of the rationale for using lexical chains.

4.1 Commercial Systems

The state-of-the-art in commercial text summarization can be seen in systems such as British Telecom’s online program ProSum (from TranSend, cost 25p per use), Oracle’s Context (constructed for data mining), Inxight’s summarizer used for filtering

in AltaVista Discovery, Microsoft Word's AutoSummarize (under the tools menu), and Sinope from Carp Technologies. These systems leave a lot to be desired, but commercial systems do tell us which ideas businessmen think worthy of investment. Since they are commercial products, however, not much is published about their internal workings.

Most of these commercial summarizers rely on simple extraction of sentences to produce summaries and can be classified into two general categories: domain dependent approaches that use knowledge of the specific domain and text structure (e.g. financial, medical), and domain independent approaches. Domain independent approaches employ various statistical and linguistic techniques to identify key sentences in the document. The main problem with text extractors, as we have noted, is that they often produce incoherent or misleading summaries (e.g. the problem of anaphoric resolution). Text abstractors try to overcome these limitations by parsing the original text then finding new, shorter concepts to describe it. Rather than simply extracting sentences, these systems automatically transform the extracted information so that it is both concise and coherent. **Text abstractors, however, are far more complicated to implement, and much slower than text extractors.** The slowness may be offset in future by faster computers, but the complications will not.

4.1.1 Microsoft Summarizer

Independent tests show that Microsoft's Word97 AutoSummarize tool extracts about 32% of relevant information when the summary size is 20% of the original. (Dalianis 2000)

Gore (1997) provided the following information about the inner workings of the summarizer in *Slate*, the on-line magazine produced by Microsoft. AutoSummarize can summarize texts to a fixed number of sentences, words, or percentage of the original. According to the article, AutoSummarize just weights sentences using simple term frequencies, with a stop-word list to remove words such as "a" and "the." It then computes "averages" for each sentence by adding the scores of its words then dividing by the number of words in the sentence. That's the official explanation. However, the summarize option appears to depend heavily on the first sentence in each paragraph. **Nevertheless, this information reinforces the importance of term frequency for text summarization.**

When Microsoft Word97 AutoSummarize was turned loose on Lincoln's *Gettysburg Address* it produced the results shown in Figure 4.1 on the following page. These results are usable as a summary.

For comparison, the full text of *The Gettysburg Address* is contained in Appendix A on page 232, along with summaries produced by my AutoExtract program and the Sinope commercial summarizer.

25 Percent Summary

Fourscore and seven years ago our fathers brought forth upon this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal. Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure. The brave men, living and dead, who struggled here, have consecrated it far above our poor power to add or detract.

10 Percent Summary

Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure.

Figure 4.1: Gettysburg Address - Microsoft Word97 AutoSummarize Output.

4.1.2 National Research Council of Canada - Extractor

Extractor, developed by the Interactive Information Group of the National Research Council of Canada (NRC), takes an input file then generates lists of key words and key sentences as output. It is available as a Microsoft Internet Explorer plugin, a stand-alone executable, or as a DLL (Microsoft dynamic link library (meaning dynamically loadable library)). The Extractor web site lists several commercial products that use it. One of the more popular packages listed as using this summarization engine is the Copernic Summarizer (\$69.95). No published reports of Extractor's effectiveness as a summarizer could be found, and there is no information available about its internal organization.

4.1.3 Carp Technologies - Sinope (formerly Sumatra)

Carp Technologies was forced to change the name of its summarizer from Sumatra to Sinope because the Sumatra name was already in use by another company. Sinope is intended to be a general purpose system that is domain independent. Instead of relying on statistical techniques it supposedly uses a natural language processing approach that involves parsing, semantic analysis, and text generation. (Lie 1998)

Unlike Microsoft, Carp Technologies provides quite a bit of information about the internal workings of Sinope. Sinope first uses pre-processing to segment the text into paragraphs, sentences, and words. It then looks up each word in a lexicon.

The regular processing portion then proceeds in three phases:

- First, it parses the text and builds a semantic structure from it.

During this phase the parser uses rewrite rules to restate sentences in fewer words. (Lie, Hulstijn, op den Akker & Nijholt 1997).

- Sinope then uses the output from the previous phase to build a semantic structure that contains the meaning representation.

Semantic analysis views text as a collection of relations between objects. For example, "Jeff hit Ron with a rock.", obviously denotes some kind of relation between Jeff, Ron and a rock. The goal of semantic analysis is to identify and store these relations in a tree structure. Whenever the semantic analyzer identifies a relation it stores it in a tree node. If the node already exists, it's just reused. These tree nodes are assigned importance values based on their number of connections. (Remember we saw this basic graphing approach earlier.)

Sinope then uses heuristics such as position information to add to the importance score of a node. We know, for example, that the first and last sentences in a paragraph are usually more important than other sentences. Sinope also uses cue words or bonus words to boost the importance of a node. The author notes that a neural network might be ideal for finding or learning the right weights to attach to these different factors. After assigning values, the tree structure is pruned to omit information. (Actually, certain nodes are selected for the extract and the rest are ignored.)

- Finally, new text is generated from this semantic structure. During this phase, sentences are compacted and combined.

This system was developed in Java as a M. Sc. Thesis project at the University of Twente. It was then known as Sumatra. The author states that the system was evaluated by using final exam texts from the Dutch grammar school in summarizing. The author also states that it extracts about 50% of relevant information elements when the summary size is set to 25% of the original. This entry appears to be an improvement over Microsoft's. Sinope (Sumatra) is the most promising of the approaches we have seen so far. My major criticism is that it is written in Java (far too slow for the real world). (We will see in the last chapter, when we run actual comparison tests, that Sinope doesn't do nearly as well in practice. So, recoding would make it faster, but not better.)

4.2 New Extraction Ideas

Extraction from the original document is still the easiest path to systems that do something approximating summarization. One somewhat novel approach, however, is to extract entire paragraphs instead of sentences. Another approach is to train extraction systems on human abstracts or relevance judgments.

4.2.1 Extracting Paragraphs Instead of Sentences

Entire paragraphs provide more context and coherence than a simple group of sentences. Thus, extracting entire paragraphs would seem to be an improvement. Moreover, this method avoids many of the problems of dangling references.

Paragraph extraction has been proposed by Mitra, Singhal & Buckley (1997) and Salton, Singhal, Mitra & Buckley (1997). They assume that every text can be represented as a vector of weighted terms. They then compute pairwise similarity coefficients for these vectors. This shows the similarity between pairs of texts. Since this similarity is based on vocabulary overlap, large similarity measures demonstrate that the vocabulary of two paragraphs overlaps in a meaningful way. (See Section 4.9 on page 97. This too is similar to the "similarity" measure used in TextTiling.)

The text is represented as a tree where the nodes in the tree correspond to paragraphs or segments. Hearst & Plaunt (1993) (Hearst 1994) define a segment as an internally linked contiguous piece of text, that is largely disconnected from adjacent text. Bushy nodes are segments or nodes that have many links to other nodes. A highly bushy node has a vocabulary overlap with many other paragraphs. It is, therefore, likely to discuss topics covered in several paragraphs and is a good candidate for extracting. Bushy nodes are selected such that they follow each other in the document, thereby increasing coherence.

The authors used the standard technique of measuring overlap with an ideal (human) extract to evaluate their work. The overlap, however, was low, as it often

is with real human summarizers. They concluded that their method had the same performance as simply extracting the first paragraph of the document. (Brandow, Mitze & Rau 1995) Summarization by paragraph extraction, the authors concluded, may not be worthwhile because of the wide variation in documents.

Paragraph extraction by itself is inadequate. Moreover, this approach incorporates very little discourse information. However, our summarizer does incorporate the idea that original sentence order must be preserved in the summary.

4.2.2 SweSum

SweSum is one of the new summarizers that creates summaries by text extraction. The author states that SweSum is built on a combination of statistical methods, linguistic methods, and heuristics. It uses a 700,000 word dictionary to get the group that the word belongs to and the stem of that word. The author estimates that SweSum extracts about 70% of the relevant information when the summary size is about 30% of the original and SweSum was tested on Swedish HTML-tagged newspaper text. (Dalianis 2000) SweSum's approach to word stemming, however, is really little more than the term frequency idea of Luhn. Moreover, it fails to incorporate any discourse information.

Dalianis (2000) uses a simple linear function to combine the following parameters without any special weighting.

- title words - Title words in a sentence boosts the sentence score.
- term frequency - This appears to be the same as the classical term frequency that Luhn first used in 1958. The dictionary is used here for stem lookup.
- position - Since the text is newspaper text, which is written with each succeeding paragraph less important than the previous one, sentences are scored accordingly. The first sentence ranks highest, the last sentence ranks lowest.
- query signature - The words of the user's query function as cue words or bonus words for boosting sentence scores.
- numerical data - Sentences with numerical data score higher than ones without. A simple 1 or 0 score is used here.

The author tested and apparently discarded several other parameters including sentence length, quotations, proper names, lexical connectivity, and position based on the first sentence of paragraphs.

I also will be using title words, term frequency, and position information in my summarizer, and I will be doing word stemming using the WordNet lexical database. However, I saw no valid reason for boosting the scores of sentences with numerical data. There will be no user queries in my summarizer, so that parameter is irrelevant.

4.3 The Trainable Document Summarizer by Kupiec

This system creates summaries by sentence extraction based on the following weighted heuristics: (Kupiec et al. 1995)

- sentence length cut-off feature

Sentences containing less than an arbitrary number of words are simply deleted.

- fixed-phrase feature (or cue words)

Sentences containing certain cue words receive more consideration for inclusion.

Section header words (i.e. subtitles) are treated like cue words.

- paragraph feature

This is the same as the location/position parameter used since Edmundson.

- thematic word feature

The more frequent words are defined as thematic words. This is really just another name for Luhn's term frequency. Interestingly, the length of the words themselves was a parameter.

- uppercase word feature

Upper-case words, with some exceptions, are treated as thematic words.

For training, Kupiec et al. (1995) used a corpus of 188 document and abstract pairs from some 21 scientific and technical publications. These abstracts were produced by professional abstractors and the sentences in the abstracts were aligned

(i.e. matched up with) sentences in the original documents to get a similarity measure. About 80% of the abstract sentences were direct sentence matches to the original text. Sentences in the original text that matched sentences in the abstract were used to create the training extract.

Kupiec et al. (1995) trained his program on this corpus before using it to extract sentences. That is, he approached extraction as a statistical classification problem. Given a training set of documents with hand-selected document extracts, he applied a function that estimated the probability that a given sentence should be included in the extract. New extracts could then be generated by ranking sentences according to this probability and selecting a user-specified number of the top scoring ones. This assumes that features are statistically independent. Of course, the features (parameters) are not really statistically independent, but that is a good first order approximation.

A statistical evaluation, which computed the performance for each of the heuristics separately against the corpus, showed that the summarizer produced a recall between 0.2 and 0.338. This is not terribly impressive.

The best single feature for extracting sentences that matched the handcrafted summary was the "Paragraph Feature." The best feature combination was Paragraph + Fixed-Phrase + Sentence-Length, which yielded a recall of 0.44. Since the number of sentences produced by the system always equaled the number of sentences in the corresponding manual summary, precision and recall were identical.

Rath et al. (1961) showed in an extraction study of humans that four different human judges had only a 25% overlap, and for a given judge over time only 55%. Not only don't human summarizers agree with each other, they agree with themselves only about half the time. Thus, a recall of 0.44 is respectable.

The thematic word feature (or term frequency) and the uppercase feature gave the poorest performance. This is similar to the results of Edmundson (1968). This may have been due to the type of material selected for summarization. In scientific and technical material important information is often located at the beginning or end of the document, which would boost position scores at the expense of other scores.

In our approach the number of sentences in the automatic summary will also equal the number in the manual summary. So recall, precision, and overlap measurements will be identical for our system. The idea of optimizing parameter constants based on the training corpus is the same approach that we will take.

4.4 Modern Systems Using Surface Level Features

Most of these features are the same as those used by the TRW study in the 1960's. (See Table 8.4 on page 218 to see which researchers use which feature.)

4.4.1 Location Feature

The location or position feature (using paragraph features) is based on the observation of professional abstractors that paragraphs at the beginning and end of a document, and sentences at the beginning and end of a paragraph, are more likely to contain material useful for a summary. Therefore, these sentences should be good candidates for extraction. Just how good is this heuristic?

Baxendale (1958) conducted experiments that showed that in 85% of 200 individual paragraphs that he tested, topic sentences occurred in the initial position and in 7% in the final position. In 1968 Edmundson demonstrated a 52% recall and precision rate of relevant information when the position method was used in combination with the title method. He also cited the position method as the best individual method for selecting sentences for extraction (Edmundson 1968). However, in 1980 Donlan's experiments showed that only 13% of the paragraphs of writers of that time started with topic sentences (Donlan 1980). Kupiec's study in 1995 demonstrated a 33% recall and precision rate with the location method, and also cited it as the best individual method (Kupiec et al. 1995). Teufel & Moens (1997) demonstrated a 32% recall and precision rate when it was used alone. Their studies also showed that the recall rate increased by 10% when combined with the cue method.

The indications are that the positions of the more important pieces of information in documents may have shifted with time, but that position

may still be a clear indicator of importance. Clearly the location method should be considered as one of our parameters for choosing sentences to extract. However, the same objections apply here as in the Cue Phrase Method. We should also remember that studies have shown that position information can be genre dependent.

4.4.2 Cue Phrase Feature

The cue phrase feature (bonus word, fixed-phrase feature) selects, or weights, sentences containing certain fixed phrases. Kupiec et al. (1995) used twenty-six indicator phrases such as "In conclusion." He also treated section headings as fixed phrases. Teufel & Moens (1997) expanded this idea to a list of 1670 positive and negative phrases. Teufel sorted these cue phrases into 5 classes based on how likely a sentence containing a phrase would be extracted for the summary.¹

Cue phrases have been shown to be valuable in deciding the importance of information to include in the summary, however, they are only one tool. The idea of dividing cue phrases into classes (i.e. providing different weights to different cue words) is very interesting. However, this approach by itself fails to provide discourse information.

In our approach we will use 140 bonus words (cue phrases, cue words).

See Appendix F on page 268 for a complete listing.

¹Another name for cue-phrases is "meta-discourse markers." This term should not be confused with "discourse markers" which are words used to signal coherence relations. For example, "although," "still," and "despite" are discourse markers.

4.4.3 Sentence Length Feature

The sentence length feature (sentence length is used as a cut-off feature) scores all sentences under a certain length threshold as 0 and all above the threshold as 1. **Most studies show this method to be ineffective, and we are not using it.**

4.4.4 Thematic Word Feature

The thematic word feature attempts to identify key words that characterize a particular document. Kupiec et al. (1995) used the more frequent “content” words. Teufel & Moens (1997) focused on high frequency words that weren’t deleted by the stop word list. Teufel’s idea was that sentences that contained clusters of thematic words should be more characteristic of the document as a whole than other sentences. She chose the ten top-scoring words as thematic words. Sentence scores were then computed as the sum of the thematic words in the sentence divided by the number of words in the sentence. This is basically the same approach that Microsoft says that it uses in Word97 AutoSummarize.

This is really a rehash of the frequency method that Luhn pioneered in 1958. It provides only limited discourse structure information. Moreover, Teufel’s method requires processing the entire collection first to create the content word list. This would be extremely slow.

4.4.5 Uppercase Word Feature

The uppercase word feature of Kupiec et al. (1995) stressed the importance of proper names.

4.4.6 Title Feature

The title method weights a sentence based on the occurrence of title words (excluding certain stop-words) This method was used by Teufel & Moens (1997), Myaeng & Jang (1997), and Dalianis (2000).

The same objections apply here as in the Cue Phrase Method. Nevertheless, this parameter has demonstrated that it can be valuable for selecting sentences and we are including it.

The same objections apply here as in the Cue Phrase Method. However, since databases such as WordNet don't include proper names, this feature might be ideal for capturing such information.

We have seen that Edmundson (1968) manually adjusted the weights of the heuristics that he used for summarization. Kupiec et al. (1995) and Teufel & Moens (1997), however, trained their programs on a corpus before using them to extract sentences. They approached extraction as a statistical classification problem. Given a training set of documents with hand-selected document extracts, they derived a function whose output corresponded to the probability that a given sentence would be included in the extract. New extracts could then be generated by ranking sen-

tences according to this probability and selecting a user-specified number of the top scoring ones. (This is very similar to a ANN (Artificial Neural Network) learning approach.)

We are also approaching text extraction as a statistical classification problem. Our salience function will be optimized to provide the best least squares error for selecting target sentences from a training corpus, then applied to a separate testing corpus for evaluation.

4.5 Using Concept Counting Instead of Word Counting

Words have always been counted in text summarization to quantify the importance of the concepts (referents) represented by those words. What we are really interested in, of course, are the concepts to which those words refer. If we were to read several statements about trees, then trees must be important as a topic, must be a topical referent.

This idea has not actually been implemented. However, the primary objection is the limited discourse knowledge that it would incorporate. Moreover, since it has never been implemented, we have no real idea of the computational overhead.

4.6 Capsule Overviews

Boguraev & Kennedy (1997) use something they call "capsule overviews." A capsule overview is a condensed representation of the document, derived by using data

reduction on the original text (i.e. compression). This is really closer to an indexing system than summarizing. The authors identify something called “topic stamps,” important phrases in the document, then use this set of topic stamps to create an overview of the document. Figure 4.2 on the next page shows an example of the creation of a capsule overview.

Since it is really an indexing system, this approach fails to provide the qualities needed for a good summarization system.

4.7 Systems Using Lexical Chains

Lexical chaining refers to the process of placing the words of a document into chains of similar meaning, or some other lexical relationship. The technique has been used successfully in a number of text summarization and text retrieval applications. The first such text summarization model using lexical chains was presented in the work of Morris & Hirst (1991). Their approach (which built on the earlier work of Halliday & Hasan (1976) which stressed the role of lexical cohesion in text coherence) was to form chains of lexical items. These items were not just terms, but also sequences of related words. As such, lexical chains provide a representation of the lexical cohesive structure of the whole text. Lexical chains have also been used for information retrieval (Stairmand 1996) and for correction of malapropisms (Hirst & St-Onge 1995).

Two factors can have a negative impact on the creation of lexical chains: syn-

"One day, everything Bill Gates has sold you up now, whether it's Windows95 or Windows97, will become obsolete," declares Gilbert Amelio, the boss at Apple Computer. "Gates is vulnerable at that point, and we want to make sure we're ready to come forward with a superior answer."

Bill Gates vulnerable? Apple would swoop in and take Microsoft's customers? Ridiculous! Impossible! In the last fiscal year, Apple lost \$816 million; Microsoft made \$2.2 billion. Microsoft has a market value thirty times that of Apple.

Outlandish and grandiose as Amelio's idea sounds, it makes sense for Apple to think in such big, bold terms. Apple is in a position where standing pat almost certainly means slow death.

It's a bit like a patient with a probably terminal disease deciding to take a chance on an untested but promising new drug. A bold strategy is the least risky strategy. As things stand, customers and outside software developers alike are deserting the company. Apple needs something dramatic to persuade them to stay aboard. A radical redesign of the desktop computer might do the trick. If they think the redesign has merit, they may feel compelled to get on the bandwagon lest it leave them behind.

Capsule Overview of the Segment

... APPLE would swoop in ...
... take MICROSOFT's customers?
... APPLE lost \$816 million;
... MICROSOFT made \$2.2 billion.
... MICROSOFT has a market value ...
... APPLE is in a position ...
... APPLE needs something dramatic ...

Figure 4.2: Example Text with Capsule Overview.

onymy (many words referring to the same concept e.g. dog and hound) and polysemy (many concepts having the same word e.g. bank). Words that are synonyms of one another may not be considered related when really are (e.g. dog, canine). Polysemy has the opposite effect. Sentences that use the same word in different senses are considered related when they aren't (e.g. river bank, savings bank). Using WordNet synsets takes care of the first problem. The second problem is more complex.

4.7.1 Morris and Hirst

Morris & Hirst (1991), using a method based on pointers into Roget's Thesaurus, demonstrated that the structure of the lexical chains in a document corresponds to the structure of the document itself (i.e. discourse structure). In this study, the researchers did not require the same word in a chain to always have the same sense (the problem of polysemy noted above). This is confusing and would seem to lead to poor chain construction. Morris, however, did not actually implement this algorithm, because there was no machine-readable version of Roget's Thesaurus at that time. Hirst & St-Onge (1995) redesigned this technique to work with WordNet, and showed that chains were easily computed and could be used for the detection and correction of spelling errors.

4.7.2 Barzilay

Barzilay & Elhadad (1997) presented a somewhat different approach to creating lexical chains. Their algorithm constructs all possible interpretations of the source

text using lexical chains, then selects the chain with the best score. (Actually the algorithm operates on text segments, then merges the results.) In their approach, terms are placed into lexical chains based on relationships such as synonymy (the state or quality of being synonymous) and hypernymy ("is-a" relationship); these chains are then used to weight sentences for selection.

Barzilay & Elhadad (1997) used WordNet, instead of Roget's Thesaurus, as the knowledge base. WordNet is a lexical database which captures all senses of a word (in synsets) and contains semantic information about the relations between words. See Figure 4.3 on the following page for an example of WordNet lookup. The numbers in front of each sense are the file offsets. WordNet does not contain any proper nouns. Thus, all words are looked up in lower case form.

Barzilay created summaries of 10% and 20% of the original. At 10% the precision was 61% and the recall was 67%. At 20% the precision was 47% and the recall was 64%. You will note that precision and recall decreased as the summary size increased. My summaries are about 35% in size and my precision and recall were both 60%. Barzilay gives the Microsoft Word97 AutoSummarize tool, when producing a summary of 20%, a precision of 32% and a recall of 39%. At 35% of the size of the original document I found 47% for both precision and recall for Microsoft.

In Barzilay's approach (and in many others), summarization proceeds in three steps:

- The original text is segmented.

output produced by "wn apple -hyphen -o"

Synonyms/Hypernyms (Ordered by Frequency) of noun apple

2 senses of apple

Sense 1

{05775435} apple

=> {05745250} edible fruit

=> {05745048} produce, green goods, green groceries, garden truck

=> {05622219} foodstuff, food product

=> {00011575} food, nutrient

=> {00010572} substance, matter

=> {00009457} object, physical object

=> {00001740} entity, something

=> {09423815} fruit

=> {07991950} reproductive structure

=> {09381783} plant organ

=> {09381264} plant part

=> {00010123} natural object

=> {00009457} object, physical object

=> {00001740} entity, something

=> {09426892} pome, false fruit

=> {09423815} fruit

=> {07991950} reproductive structure

=> {09381783} plant organ

=> {09381264} plant part

=> {00010123} natural object

=> {00009457} object, physical object

=> {00001740} entity, something

Sense 2

{08933828} apple, orchard apple tree, Malus pumila

=> {08933472} apple tree

=> {08951617} fruit tree

=> {09401109} angiospermous tree, flowering tree

=> {09396070} tree

=> {09395329} woody plant, ligneous plant

=> {09378438} vascular plant, tracheophyte

=> {00008864} plant, flora, plant life

=> {00002086} life form, organism, being, living thing

=> {00001740} entity, something

Figure 4.3: WordNet Lookup Example.

- Lexical chains are constructed.
- The strongest chains are identified and the sentences they identify are extracted from the text.

Constructing all possible chains is far too time consuming for any real world operation. This approach literally takes hours to run.

An example of the type of lexical chains that we will use in our summarizer is shown in Figure 4.4 on the next page.

4.7.3 Green

While Barzilay's approach was centered on producing the best possible lexical chains, then summarizing the document, Green (1996) focused on summarization, with lexical chains merely being a means to an end. Green's approach is much more suited to the real world, since he is willing to accept less than optimum lexical chains in the interest of speed.

Green (1996) like Barzilay uses the WordNet database to build three kinds of relations between words:

- extra strong - word repetition.
- strong - synonyms, antonyms, connected by single, IS-A, or INCLUDES relations
- regular - there is an allowable path in the WordNet graph between synsets that contain the two words. An allowable path has restrictions on changes in

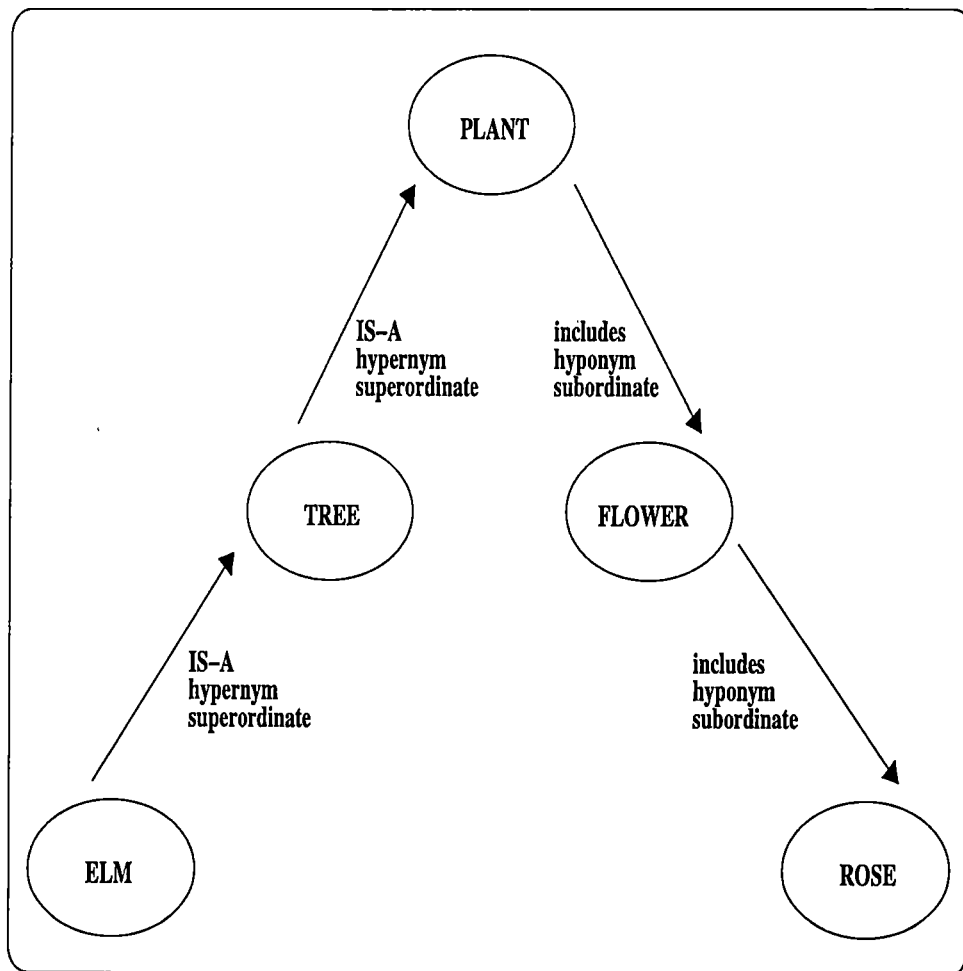


Figure 4.4: Hypernym/hyponym chain relating elm to rose.

direction, and is shorter than some arbitrary maximum length.

Both Barzilay and Green initially represent each word as a list of all the synsets that contain it. As chaining proceeds and relations are built between words, synsets that aren't part of the current chaining operation are discarded. As a result, the sense of the lexical chain is progressively narrowed. Green states that compared to traditional document processing tasks in IR (e.g. keyword extraction), the chaining process is relatively slow. He states that chaining a database of 30,000 newspaper articles takes about 5 hours, while a traditional IR system would take about 15 minutes. Of course, the output of a traditional IR system is not useful for text summarization. An example of Green's chaining technique is shown in Figure 4.5 on page 95.

Green refers to the importance of a chain as its density, where this density is based on the fraction of content words (non stop words) in the paragraph that are also in the chain. This seems to me to be throwing away the real lexical chain information and substituting a version of the old thematic word frequency idea.

Green (1996) estimated that about 47% of the relevant information was extracted when the summary size was 20% of the original.

4.7.4 Andersen

The work of Andersen (2000) wasn't concerned with summarization, but with how to form longer chains faster. His was a modified greedy approach where he performed

a breadth-first search to find the lexical relation with the shortest path first. He restricted his searches to hypernyms/hyponyms, holonyms/meronyms, and antonyms. He found that including the other lexical relations dramatically increased the search time without producing any better chains. He restricted searches to plus or minus ten sentences, and concluded that this restriction was crucial to the algorithms success.

4.8 WordNet

The total of all unique noun, verb, adjective, and adverb strings in WordNet is actually greater than the 121962 shown in Table 4.1 on the next page. Many strings, however, appear in more than one syntactic category. The number in the table is the number of unique strings when all syntactic categories are combined.

WordNet allows you to perform 28 different types of searches on four different parts of speech (noun, verb, adjective, adverb). I use the search types in bold font in Table 4.2 on page 96 in my AutoExtract summarizer.

Lexical chaining provides a solid foundation, but makes no use of the invaluable information present in cue words, topics, and location. A simplified chaining procedure is at the heart of my AutoExtract summarizer. See Subsection 7.2.3 on page 133 for a description.

Although no one is **pushing**¹² virtual-reality **headgear**¹⁶ as a **substitute**¹ for **parents**¹, many technical ad **campaigns**¹³ are promoting cellular **phones**²², **faxes**²², **computers**¹ and pagers to **working**¹ **parents**¹ as a way of bridging **separations**¹⁷ from their **kids**¹. A recent **promotion**¹³ by AT&T and **Residence**² **Inns**⁷ in the United **States**⁶, for **example**³, suggests that **business**³ **travelers**¹ with **young**¹ children use **video**³ and audio **tapes**²², **voice**³ **mail**³, videophones and E-mail to **stay**³ connected, including **kissing**²³ the **kids**¹ **good night**²¹ by **phone**²².

More **advice**³ from **advertisers**¹: **Business**³ **travelers**¹ can dine with their **kids**¹ by **speaker**¹-phone or “tuck them in” by cordless **phone**²². Separately, a **management**¹⁰ **newsletter**²⁴ recommends faxing your **child**¹ when you have to **break**¹⁷ a **promise**³ to be **home**² or **giving**¹² a **young**¹ **child**¹ a beeper to make him **feel**²³ more secure when **left**⁵ alone.

Figure 4.5: Green’s example of text tagged with chain numbers.

Table 4.1: WordNet 1.6 Database Statistics.

POS	Unique Strings	SynSets	Total Senses
Noun	94474	66025	116317
Verb	10319	12127	22066
Adjective	20170	17915	29881
Adverb	4546	3575	5677
Totals	121962	99642	173941

Table 4.2: WordNet 1.6 Pointer and Search Types.

Pointer Type	Search
OVERVIEW	Show all synsets for word
SYNS	Find synonyms
ANTPTR	Antonyms
HYPERPTR	Hypernyms generalization of a word - fruit is a hypernym of orange opposite of hyponym
HYPOPTR	Hyponyms specification of a word - fig is a hyponym of fruit opposite of hypernym
HMERONYM	Hierarchical meronym search whole in a part-whole relation opposite of holonym
MERONYM	All meronyms
ISMEMBERPTR	Member meronym
ISSTUFFPTR	Substance meronym
ISPARTPTR	Part meronym
HHOLONYM	Hierarchical holonym search part in part-whole relation opposite of meronym
HOLONYM	All holonyms
HASMEMBERPTR	Member holonym
HASSTUFFPTR	Substance holonym
HASPARTPTR	Part holonym
VERBGROUP	verb group
CAUSETO	cause of another action
ENTAILPTR	entailment - implies another action
PPLPTR	participle of verb
SEEALSOPTTR	also see - refer to another expression that contains the word
PERTPTR	pertains to noun or derived from adjective
ATTRIBUTE	attribute - honest is an attribute of honesty
FREQ	polysemy
FRAMES	verb example sentences and generic frames
COORDS	noun coordinates
RELATIVES	group related senses
WNGREP	find keywords by substring
SIMPTR	similar - another adjective close in meaning

4.9 TextTiling

Hearst (1994) states that TextTiling is primarily concerned with identifying coherent passages in text by finding “topic boundaries” by computing the similarity of adjacent blocks of texts. It approximates the topical structure of a document by using patterns of lexical connectivity to find coherent blocks or tiles. Supposedly the tiles correspond well to human judgments of the boundaries of major topics of articles in science magazines.

Barzilay uses Hearst’s TextTiling algorithm for breaking the text into blocks, creates the chains for the blocks, then merges all the chains together. The only rationale I can see for this is speed. Producing chains for the entire text will generate chains that represent the entire text rather than some sub-block. Moreover, chain merging is less than perfect. This approach adds the two extra steps of TextTiling and chain merging. It only saves on time if your overall chaining algorithm is very slow, which Barzilay’s is. Barzilay’s approach, analogous to the quantum-mechanical paths of photons through an aperture, holds all possible chains until the final decision is made as to which to keep. This approach may produce superior lexical chains, but is far too slow to use in any summarization system in the real world.

I investigated the use of TextTiling in creating my lexical chains, but found it to be time consuming and counterproductive. You can produce better, faster chains without it. I viewed TextTiling as duplicating the

work that was already being done by lexical chains. TextTiling might be useful if you were interested in the properties of the tiles themselves. For example, we might be able to tile such that different tiles corresponded to different viewpoints within a document. Summarization based on the tiles themselves then might be used to preserve all viewpoints.

4.10 Discourse Structures

4.10.1 Using Rhetorical Structures to Make Relevance Judgments

Marcu (1997*b*) showed that a strong correlation exists between the main nodes (nuclei) of the RST (Rhetorical Structure Tree) of a text and those text units that humans rate as important.

He uses a rhetorical parsing algorithm (Marcu 1997*a*), based on a corpus analysis of more than 450 discourse markers and almost 8000 text fragments, to create the RST. The summarization program then selects from the RST (produced by the parser) the more important text units. The closer nodes are to the root, the more important are the units associated with them. The longer the summary, the farther the selected units will be from the root.

This approach incorporates discourse information, but fails to make use of cue, title, or location information. Moreover, we have already decided to rely on lexical chains for our structure information, because they are much easier to compute than RST's.

4.10.2 Pruning Rhetorical Structures to Produce Summaries

Ono et al. (1994) developed an automatic abstract generation system for Japanese based on the extraction of rhetorical structures (Mann & Thompson 1988). They argued that a rhetorical structure provides a natural order of importance among sentences in the text, and that it could be used to determine which sentences should be extracted to produce a summary. He further argued that summaries which reproduce the more important items of the rhetorical structure are more coherent, since the structure of the text is preserved.

Ono used surface connectives (discourse markers) (the Japanese equivalents of "but," "for example," etc.) to detect discourse structures. He then pruned the resulting binary tree to derive summaries of the desired length. As soon as sentences were selected, the program arranged the sentences and the connectives into the output text.

Ono used abstracts of 30 editorial articles from the Asahi Shinbun newspaper and 42 technical papers from the Toshiba Review to evaluate his system. The abstracts of the newspaper articles contained the most important sentence (as determined by three human judges) in the document 60% of the time, and matched 41% of the key sentences selected by human judges. For the technical articles his abstract contained the most important sentence 74% of the time, and matched 51% of the key sentences. Ono explained that the performance was better on technical texts because explicit connectives are more frequent there. Thus, the system could build

a more complete rhetorical structure when using technical texts.

This method, however, makes no use of the invaluable information present in topics or location and is overly dependent in its emphasis on connectives, which explains its mediocre performance on newspaper articles. Moreover, this system was designed to process Japanese text. How such a system would perform on English text is difficult to predict.

4.11 Combination Methods

The two systems discussed in the following section are both examples of combination methods. The first system, SUMMARIST, combines the lexical semantics of the WordNet database with statistical information retrieval methods.

The second system, SIMPR, system is based on the knowledge of professional indexers combined with surface linguistic constraints developed in corpus linguistics. SIMPR's goal is to index and retrieve text passages from large text databases (e.g. technical documentation).

4.11.1 SUMMARIST

SUMMARIST (Hovy & Lin 1997) is a system designed to produce both extracts and abstracts of general English texts. Producing abstracts requires the additional stages of topic fusion and text generation that are not needed for producing extracts.

SUMMARIST does summarization in three phases:

- topic identification - This phase relies on “concept counting,” which is slightly more advanced than Luhn’s “term frequency.”
- interpretation - This is the *concept-based* topic fusion phase, and relies on statistical techniques from IR (information retrieval) such as word clustering.
- generation - There are three alternatives during this phase:
 - keyword list
 - phrase template generator
 - sentence generator

SUMMARIST was meant to overcome the shortcomings of approaches limited to only one scientific background. It integrates methods from several disciplines, including classical statistical information retrieval and symbolic knowledge processing with resources such as WordNet. The system architecture relies heavily on WordNet.

SUMMARIST is significant in that the system uses no parser or grammar. Methods, however, must be added by hand, then carefully fitted to achieve any success. Also its “term indicator phrases” are really just cue phrases by another name, and concept counting is not much more worthwhile than Luhn’s thematic word frequency.

4.11.2 SIMPR - Structured Information Management: Processing and Retrieval

SIMPR (Gibb & Smart 1990, Karetnyk, Karlsson & Smart 1991, Gibb 1993) is aimed at large technical databases that may contain thousands of documents. SIMPR represents the text components of these documents with indexing terms or phrases that a reader of the document might use in an information search. This index is in alphabetical order and the entries are derived from noun phrases from the source document that are provided by an automatic indexing procedure. Processing proceeds in three steps:

- preprocessing
- language analysis
- indexing

Because this is really much more an indexing system than a summarizing system, it is not really suitable as a summarization solution.

4.12 Restricted Data Set Solutions

The next two systems were designed to work with structured knowledge or data sources. Their operation is similar to that of a man telling a story that he already knows by heart.

STREAK is designed to deal with basketball games, and the summary style of basketball game reports. It focuses on individual games, but supposedly puts those games into perspective. SUMMONS is designed to deal with news stories about terrorism, and summarizes knowledge about terrorism events culled from several sources.

4.12.1 STREAK

The STREAK system described by McKeown et al. (1995) condenses basketball game reports by combining the information from box scores and the lead sentences of the reports. Supposedly, this makes scores more meaningful and easier to absorb.

Sentences in basketball game reports from newswire services are usually from 21 to 46 words in length, and are complex and densely packed. Essential facts such as game results, however, are always contained in the lead sentence with other optional information such as historical information placed throughout the text. This optional information accounts for over 40% of the content of the lead sentence.

STREAK is composed of five modules:

1. fact generator
2. sentence planner
3. lexicalizer
4. sentence reviser
5. SURGE - a lexical and syntactic surface generation package

STREAK produces summaries in two stages. It first generates a draft that contains the information that must be included. Then it revises the draft to add optional items such as historical information.

The fact generator produces a list of essential facts that must be included and a list of facts that may be added. The sentence planner takes the network that contains all the essential facts and maps it onto a semantic tree which represents the overall sentence structure. The lexicalizer then maps this semantic tree onto a skeletal syntactic tree.

The sentence reviser encodes the sentence meaning in the semantic tree, then outputs a final draft that incorporates both the essential facts of the first draft and as many additional facts as can be added subject to domain, linguistic, and space constraints.

4.12.2 SUMMONS - SUMMARizing Online NewS articles

SUMMONS (McKeown & Radev 1995) is a prototype system for creating summaries of a series of news articles about the same event. The system was designed to use MUC-4 (the fourth Message Understanding Conference) templates to generate output. In the actual implementation, however, the authors supplemented the system with handcrafted templates. Summaries consist of one paragraph that describes one event or a series of events and how they changed over time.

The system contains a contradiction operator to handle the different opinions of various wire services, and a refinement operator to insert additional information

such as the number of victims of a terrorist act. The final text uses lexical cues such as “however,” “exactly,” and “finally” to tie things together.

STREAK and SUMMONS because they operate on severely restricted data sets have little to tell us about general solutions. Moreover, their ideas about information gathering (e.g. using location as a cue) are little more than a rehash of early research work. *However*, the idea of using lexical cues to make the final text more readable is certainly worth considering.

Chapter 5

Closely Related Fields

It doesn't matter if a cat is black or white,
so long as it catches mice.

*Chinese Leader Deng Xiaoping*¹

This chapter covers related fields that have had a direct impact on automatic text summarization. We cover abstract creation by professional human summarizers and related information from the fields of Information Extraction (IE) and library science.

5.1 Human Abstracting

There have been several other research fields that have had a direct impact on automatic text summarization. First are the psychological studies of human summarization in the laboratory by Kintsch & van Dijk (1978) and vanDijk (1979). The

¹*Time Magazine, 6 January 1986*

subjects in these experiments apparently used conceptual structures (schemata) for text comprehension, and interpreted texts based on previous experience (episodic memory). These experiments revealed that humans appear to create a hierarchical discourse organization, that provides cues for memory retrieval. Humans then restore missing information through an inference-based reconstruction process (i.e. common sense). The subjects in this experiment reacted in much the same way as those in other psychology experiments, that is, they tended to inject their own comments, opinions, and attitudes into summaries. This indicates that memory is reconstructive, relying on some incomplete core representation.

Professional abstractors also have something to teach us. They are taught to use prescribed programs for generating effective abstracts. Cremmins (1996) found that professional abstractors, perhaps because of information overload, don't attempt to fully *understand* the text, but use surface-level features such as headings, key phrases, and position in paragraphs. They also use discourse features such as overall text structure to organize abstracts. Moreover, they continually revise and edit abstracts.

Liddy (1991) studied a corpus of 276 abstracts and found that their structure was based on components such as background, purpose, methodology, results, and conclusions. Endres-Niggemeyer, Maier & Sigel (1995) (Endres-Niggemeyer 1998) in their studies of professional abstractors also found that professional abstractors exploit discourse structure taking a top-down strategy.

Although humans constitute our *gold* standard for judging summaries, the gold of

the human abstractors was somewhat tarnished. Marcu (1997c) found that overall, human abstractors only agree on which sentences should be in a summary about 71% of the time. Rath et al. (1961) showed in a study that extracts created by four different human judges only had a 25% overlap, and for the same judge over time only 55%. Thus, not only don't human summarizers agree with each other, they agree with themselves only about half the time.

Nevertheless, we need to consider carefully the parameters that human abstractors use, with an eye to including them in our own method. Here are some of the factors that human summarizers consider to be important in creating a summary:

- Title

The title often tells us what the article is about and helps the summarizer anticipate the content. Edmundson (1968) was able to produce about 41% of the relevant sentences in a summary by using the words in the title and headings. This method also appears as a parameter in several later hybrid systems. **Title and sub-title (heading) words are one of the parameters of our summarization formula.**

- Format

The format will often tell you what sort of text you are reading (e.g. newspaper article). You can then decide how to read the text and what kind of information to look for. **Since, our summarizer is general-purpose, this sort of information doesn't help us very much.**

- Illustrations

Pictures and photos often emphasize certain portions of the text. Unfortunately, there is no way our text-based summarizer can make use of this information.

- Text Organization

Most text is organized in some logical way (e.g. episode, description, timeline, problem solving, comparison, or cause and effect). If we know which technique is being used, we will know how to organize the incoming data. (This, of course, was the central idea behind the template methods of the 1970's.) This text organization could be expressed in some sort of rhetorical structure (e.g. tree).

Lexical chains superimpose a hierarchical structure upon a text that has been shown to be equivalent to this rhetorical structure. The higher a word appears in a lexical chain structure, the more abstract the concept that it represents. (See Figure 4.4 on page 92.), and abstraction is the name of the game. That is, text summarization is all about abstraction. For example, apple and orange are both examples of fruit, and beets and turnips are both vegetables. Continuing on, fruits and vegetables are both examples of produce. Thus, a paragraph with all these items mentioned might be construed to be *about* produce. This power of abstraction, this focusing on topics, is part of our motivation for using hypernyms and holonyms in lexical chains.

- Discourse Markers

The term “discourse markers” as used by professional text summarizers refers to words that indicate how one sentence relates to another, and may be used to break text into discourse segments. (There is an intersection between discourse markers and cue phrases but they are not the same.) Both ordinary humans and professional abstractors use these discourse markers to understand the rhetorical structure of a text. However, we are relying on lexical chains for this structure information instead of discourse markers.

- Context

Humans use the context within the text itself as a clue to the meaning of unfamiliar words. We are relying on lexical chains to provide us with limited context information.

5.2 Information Extraction

The symbolic techniques using parsers, grammars, and semantic representations used in many text summarization systems don't scale well. This has caused some researchers to look at the statistical techniques of Information Retrieval (IR). However, IR and statistical techniques (e.g. word counting, word clustering) by themselves cannot be used to create a true abstract because these techniques only operate at the surface level, not at any concept level.

Closely related to IR is IE (information extraction). IE is concerned with identifying and extracting relevant information from texts. While IR focuses on recovering

a subset of documents that match a user's query, IE focuses on recovering a data subset from one text. Information-extraction-based summarization systems generate extracts that focus on specifics defined by the interrogatives: who, what, when, where, and why.

There are no books or journals devoted to information extraction, and apparently the best sources of IE information are the Message Understanding Conference Proceedings (MUC-3 in 1991, MUC-4 in 1992, MUC-5 in 1993, and MUC-6 in 1995).

We are already planning on using the statistical techniques of word counting in addition to lexical chains.

5.3 Library Science

There is also a fairly large body of work on abstracting from researchers working in library science (Borko & Bernier 1975). This work distinguishes between different types of abstracts, with the primary division being between indicative abstracts and informative abstracts. Indicative abstracts tell what an article is about, and informative abstracts include major results from the article and can be read in place of it.

At present, I cannot find any techniques from library science that I can apply to my project, but this is an area that might be useful in the future.

Chapter 6

An Improved Method for Text Summarization

The problems are solved, not by giving new information,
but by arranging what we have always known.

*Ludwig Wittgenstein,*¹

In this chapter I restate the basic problem, then cover in detail how I plan to solve the problem of automatic text summarization, and why I chose the methods and approaches that I did. I cover the parameters that I plan to use and the ones that I plan to discard. A chart showing which parameters I used for text summarization (compared to others) is shown in Table 8.4 on page 218.

¹*Philosophical Investigations, 1953*

6.1 Problem Re-statement

The basic problem of text summarization by extraction is this: We have a collection of sentences, and from this collection we wish to select or extract a subset with some quality that *best* represents the entire collection. But what criteria do we use to determine and select for this quality?

6.2 Learning Algorithm

One of my thoughts about systems detailed in previous chapters was that they could be improved by adding a learning algorithm. Once, I began to implement my own system I began to fully appreciate the reason for not doing so. Learning algorithms can add a great deal of complexity and greatly increase the time required not just for training, but more importantly the run-time of the algorithm. **This is probably a reason that commercial systems don't include a learning algorithm, and this increased run-time is the reason that I don't include one.** Moreover, a learning algorithm only makes sense if you somehow know the difference between good and bad, for example, if you had a training corpus where every sentence had been rated by humans, or if you had human feedback that somehow told you how to modify your output to better fit the audience or material that was being processed.

6.3 Why combine factors?

Our examination of the single concept approaches that past researchers have used shows us that each has merit, but by themselves are insufficient to produce adequate summaries. Also the work of human abstractors shows us that some knowledge of the overall structure is necessary to produce knowledgeable and coherent summaries. The work of researchers on RST and discourse analysis shows promise in producing that knowledge.

We have seen that Edmundson (1968) manually adjusted the weights of the heuristics that he used for summarization. Kupiec et al. (1995) and Teufel & Moens (1997), however, trained their programs on a corpus before using them to extract sentences. That is, they approached extraction as a statistical classification problem. Given a training set of documents with hand-selected document extracts, they applied a function that estimated the probability that a given sentence should be included in the extract. New extracts could then be generated by ranking sentences according to this probability and selecting a user-specified number of the top scoring sentences. This approach assumes that features are statistically independent. Of course, the features (parameters) are not really statistically independent, but that is a good first order approximation.

Work during the early period of text summarization (Luhn 1958, Edmundson 1968, Pollock & Zamora 1975) demonstrates that term frequency is a valuable fundamental tool for text summarization work. However, this must be augmented with

information from cue phrases (e.g. “significant,” “impossible”), title and heading words, and the location of sentences within the paragraph.

Kupiec et al. (1995) found that on their data, location was the best individual feature to use for selecting sentences to extract. In their feature mix, location, cue phrase, and sentence length was the best combination.

Myaeng & Jang (1997), described a variant of the above method which was applied to technical texts in Korean. The authors found that using a combination of cue words, sentence location, and presence of title words in a sentence led to the best results.

Pollock & Zamora (1975) described an abstracting program at Chemical Abstracts Service which relied on the use of cue-phrases specific to chemistry sub-domains. These cue-phrases were divided into positive (bonus word) and negative (stigma word) tests for the selection of sentences.

All of these studies clearly show that a single feature is insufficient for selecting sentences, and that a feature mix is clearly warranted. Which features (parameters) then do we choose for our mix?

6.4 Which parameters should we use in our salience function?

Based on the research cited above and detailed in previous chapters, we have decided that the following parameters are the most useful grouping

of input parameters for our extraction function.

6.4.1 Location Information

The rationale for the location parameter comes from the observations of professional summarizers that boundary information is more important than other information in summarizing a document. That is, paragraphs at the beginning and end of a document, and sentences at the beginnings and ends of paragraphs, are more likely to contain important material than other areas. Teufel & Moens (1997) has shown that the location parameter is very effective in selecting good sentences for extraction.

6.4.2 Cue words - Bonus and Stigma Words

all animals are created equal,
but some animals are more equal than others.
George Orwell

Once we consider the idea that words in a document may be partitioned into common words and uncommon, high content words, it follows naturally that they may be further sub-partitioned such that some high-content words are more equal than others. That is, some bonus words, as some researchers refer to them, or cue phrases, or meta-discourse markers as others call them, such as “argue,” “propose,” “develop,” and “attempt” carry more weight than other high content words. Likewise, some stigma words such as inadequate, inconclusive, and insufficient tip the

scale in the other direction (i.e. carry a negative weight). We see then that bonus and stigma word parameters should be considered as amplifications of the basic word-frequency parameter.

When we stop and think about it, what are bonus words (cue words)? (See Figure 6.1 on the following page.) They are those words that *in an average document* appear more often in those sentences that we want to extract. Stigma words are those words that *in an average document* appear more often in those sentences that we don't want to extract. Finally, stop words (common words) are those words that are so common throughout all sentences in all documents that they are, as Luhn termed them, nothing more than noise. But aren't thematic term words (the meaningful high frequency words in our document) then the same as bonus words? In the words of Yogi Berra, "Yes, no, and maybe." Thematic term words characterize the document that we are summarizing, whereas bonus words characterize the mythical *average document*. They may or may not overlap. For example, the word "thus" may appear only once in the document that we are summarizing, but we know from statistical evidence that "thus" functions very well as a bonus word for selecting sentences. **Thus, the statistical qualities of the present document and the statistical qualities of the *average document* are both important for selecting sentences for extraction.**

The TRW study used this bonus word method. ADAM made use of this method. Kupiec et al. (1995) used 26 indicator phrases, two-word phrases such as "this letter" or "In conclusion." The cue phrase method of Teufel & Moens (1997) expanded this

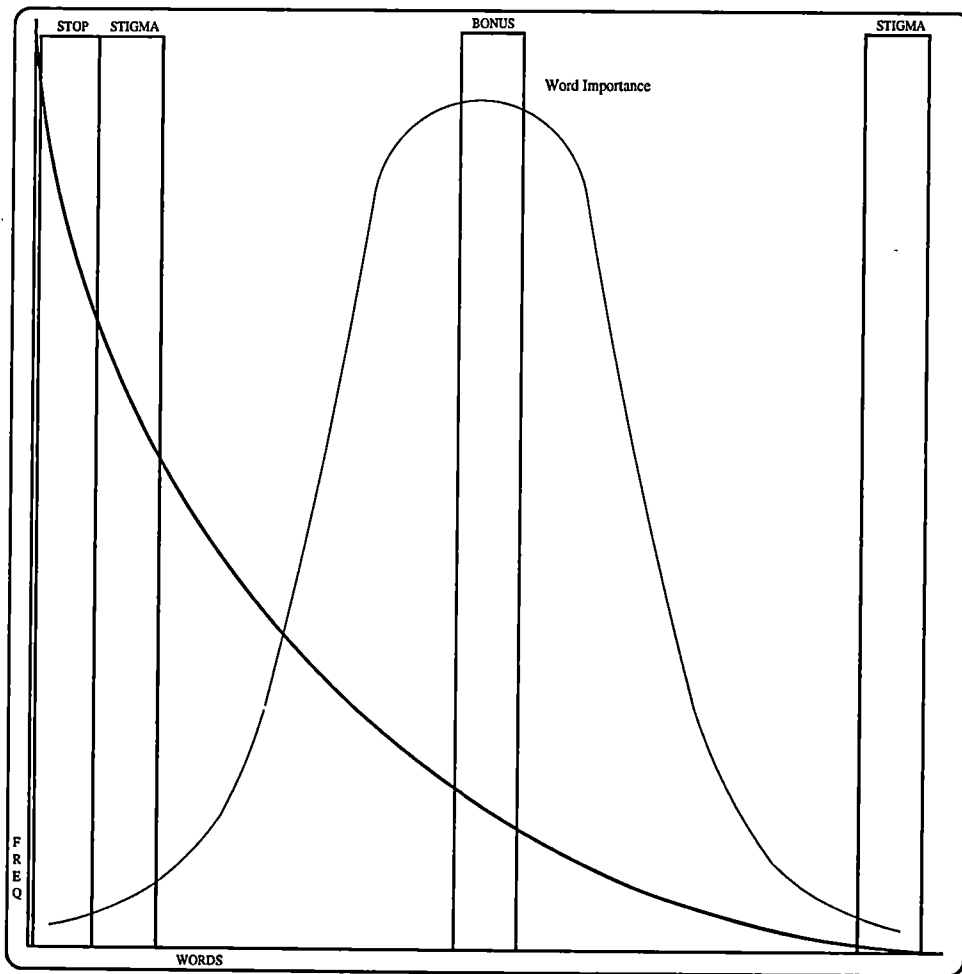


Figure 6.1: Derivation of bonus, stigma, and stop words.

idea to a list of 1670 positive and negative cues, indicator phrases, and combined expressions.

We developed lists of 140 bonus or cue words, 16 stigma words, and 386 common or stop words. These lists were compiled by finding the intersection of all the other lists of other researchers that I could find. These words (in both upper and lower case form) are compiled into the lexer portion of the AutoExtract summarizer. I outline a better procedure for compiling bonus words in the last chapter.

6.4.3 High-Content Word Counting

Based on his study of text summarization, and more importantly his own insights, Luhn (1958) proposed using the word frequency of high-content words to select sentences. It isn't clear that he understood that these high-content words were important because they captured some of the rhetorical structure of the document, nevertheless, he realized that they were important. Teufel & Moens (1997) proposed using the "Thematic Word Method," and researchers using lexical chains (such as Barzilay & Elhadad (1997)) place the greatest emphasis on "extra-strong" relations between words. All three approaches are in essence the same. All of them try to identify key words that are characteristic of the whole document, then use the frequency of those words in a sentence to weight that sentence for extraction. All make use of some sort of stop list or list of common words to exclude. **This recurring theme through the years clearly demonstrates that the word frequency**

of high-content words is one of the more important parameters that we must consider.

6.4.4 Title Words

The title parameter method, cited by Teufel & Moens (1997) and professional summarizers as being very effective, incorporates both positional information and key word information. Our *common-sense* tells us that, as an instant summary, titles must be important. Professional text summarizers tell us that titles are important for their positional information. Furthermore, consideration of thematic words and cue phrases tells us that non-common title words should have as much weight as cue phrases. Thus, the occurrence of main title words and subheading title words in a sentence should be used in weighting that sentence for selection.

This method is used by Teufel & Moens (1997) and in the TRW study. In these versions of the title method, sentences containing words that also occur in the title are assigned a higher weight than sentences that share words with subtitles, or sentences that don't share any words. Common words are assumed to have no value (e.g. "a," "an," "the").

In my version of the title method, there are separate scores (parameters) for counting title words and sub-title words in a sentence. These counts are then normalized as inputs to the salience function. For an account of how the parameter weights were arrived at, see the last two chapters.

6.4.5 Length Method

We learn as much from failure as we do from success. Thus, we must be aware of ideas that researchers have tried and discarded. One such method is the sentence length method. In this method sentences above a certain threshold length receive a score of 1. All those below the cutoff receive a score of 0. Most researchers have concluded that the sentence length method is extremely ineffective for selecting good sentences, and we saw no valid reason for considering it. It would appear that its main utility is in discarding titles and sub-titles from the summary itself.

6.4.6 Uppercase Word Feature

The uppercase word feature of Kupiec et al. (1995) stresses the importance of proper names and acronyms. Moreover, since no uppercase (i.e. proper names) are included in WordNet, we probably need to capture this feature ourselves.

6.5 What do human abstractors have to tell us?

In our earlier discussion of ordinary humans, we noted that ordinary humans, unlike professional abstractors, try to understand the text before generating a summary or recap of the topic. Cremmins (1996) found that professional abstractors, however, do not attempt to fully *understand* the text, but use surface-level features such as headings, key phrases, and position in paragraphs. They also use discourse features such as overall text structure to organize abstracts. Endres-Niggemeyer

et al. (1995) (Endres-Niggemeyer 1998) in her studies of professional abstractors found that professional abstractors take a top-down strategy, exploiting discourse structure. They build topic sentences, consider beginnings and ends of units as relevant, and exploit in-text summaries. This behavior is undoubtedly due to the information overload that professional abstractors face; they simply don't have time to understand everything they read.

The behavior of humans tells us that we need to exploit discourse structure (e.g. using lexical chains) and that we must take into account title information, cue phrases, location information, in-text summaries, and document outlines.

6.6 How do we incorporate discourse knowledge?

A number of studies also have shown that lexical chains can provide the discourse knowledge that we need to generate text summaries. Moreover, lexical chains may be the most efficient way to grab this discourse information from the source text. The approach of Barzilay & Elhadad (1997), however, is far too slow to be used in any system that ordinary people might use. The approach of Green (1996) is far more efficient, but is still too overly complicated in the types of paths that are allowed between words. The work of Andersen (2000) shows that a greatly simplified approach to the creation of lexical chains is more than satisfactory. The lexical chains that I will create will be divided into three strength levels

(as almost all other researchers have done), however, the calculation of the weakest type of chain is greatly simplified. See the next chapter for a description of this calculation.

6.7 How do we generate output?

We are faced with a fundamental question in generating output. Do we generate the output text by abstraction or extraction. There is no question but that the majority of people would like to see the text abstraction produced by human abstractors. However, this presents far more difficulties than text extraction, which is why most modern researchers have opted for extraction. Text extraction also frees us of the need for a natural language generator. **Extraction from the original document is still the easiest path to systems that do something approximating human summarization, and this is the approach I will take.**

Chapter 7

Implementation - AutoExtract

System Design

It is not the going out of port, but the coming in,
that determines the success of a voyage.

Henry Ward Beecher

*This chapter covers the actual implementation of my AutoExtract summarizer,
the pieces and how they fit together.*

This work attempts to create an improved system for automatically summarizing general source ASCII text by extracting selected sentences. Parameters that will be used in conjunction with lexical chain information to create a salience function that is used to rank sentences for extraction are position (3 parameters), cue words, content word counting, title words, sentence length, and uppercase words. Compiler technology, including the Flex and Bison tools (Levine, Mason & Brown 1992), is

used to extract and combine this information from the raw text. The WordNet database is used for the creation of the lexical chains. The best fit for the parameters of the salience function is obtained by using least-squares error fitting.

My original idea was to use an ANN (artificial neural net) to fit the parameter training data to the salience function. Although this approach might produce better results in the long run, I decided not to pursue it for the following reasons:

- Past researchers have achieved good results with simple linear functions and two of the valid functions that DataFit found for my training data were simple linear functions.
- The training time for a neural network for 12 parameters for all the sentences in 50 documents would be extremely long. Backpropagation is the standard training method, and it is extremely slow.
- Once trained, there is no easy way to extract or create a simple equivalent function to the neural net. That means that each time the summarizer runs, we would have to run 11 input parameters through the neural net to calculate the output score. This would be much slower than just using a simple linear function, and one of my primary goals is speed. I want my summarizer to complete processing in a matter of seconds.
- I needed to *see* the constants for each of the input parameters to judge their importance in the overall scheme. It would be difficult to see inside a neural net to make such judgements.

Figure 7.1 on page 187 shows the two major components of the AutoExtract summarizer, the lexer and the parser. Flex is used to process the file containing the token definitions to produce the lexer portion. Bison is used to process the file containing the grammar and auxiliary code to produce the parser portion.

7.1 WordNet

Since my summarizer is based on the use of lexical chains, and those chains are formed using the WordNet database, we need a general understanding of WordNet organization before proceeding.

WordNet is a lexical database of English words organized along the lines of a thesaurus. In WordNet, words are represented by synsets or synonym sets. A synset then is a set of words and we the file offsets of those words to represent them. For example, car might be represented by the synset {car, auto, automobile, machine, motorcar }. Synsets in WordNet can be tracked and retrieved by their file offsets such as [09032214]. I store the file offsets of the synsets of a word as an array of long integers (i.e. longs).

We can also think of WordNet as having tree structures in it for hypernyms¹, hyponyms², holonyms³, and meronyms⁴. A hypernym is a container where the

¹B is a hypernym of A if A is a (kind of) B. Vehicle is a hypernym of car. Flower is a hypernym of pansy.

²A word with a narrower meaning whose meaning is included in the meaning of a more general term. Pansy is a hyponym of flower. Rose is a hyponym of flower. Car is a hyponym of vehicle.

³The whole of which the meronym is a part. B is a holonym of A if A is a part of B. Car is holonym of motor.

⁴Part of, substance of, or member of something. A is a meronym of B if A is a part of B. Motor is a

hyponym is a member. Likewise a holonym is a container with the meronym as a member. A hypernym can have many hyponyms and each of those hyponyms can have their own hyponyms, so you have a tree branching down from the original hypernym. Likewise you can have a tree branching down from a holonym. When you do extended searches in WordNet you get a portion of this tree returned as a linked list. When you perform a tree type hypernym search you get back either a printout like Figure 4.3 on page 90 or a linked list of data structures that contains a more complete listing of all hypernyms of the word, and the hypernyms of those words, etc. From the diagram you can see that apple IS-A fruit (i.e. fruit is a hypernym of apple) which IS-A reproductive structure which IS-A plant organ which IS-A plant part which IS-A natural object which IS-A object which IS-A entity. In front of each hypernym in the figure is listed the file offset for the synset that contains that word or term. These file offsets (representing synsets) are what we use to characterize words and build relations between them.⁵

meronym of car.

⁵WordNet, available for free from *ftp.cogsci.princeton.edu*, contains at least 57,000 noun word forms organized into 48,800 synsets. The first thing you must do is download the bug list. (I really don't understand why the patches have not already been applied.) Patch the source code before you make and install the program. At the same site you can download *Introduction to WordNet: An On-line Lexical Database* (Miller, Beckwith, Fellbaum, Gross & Miller 1990). This and four other publications are contained in the file *5papers.pdf*. These papers are useful for gaining an overall appreciation of WordNet, but are totally worthless for interfacing with the C code. There are various books available on WordNet, but they are likewise worthless for interfacing with the code. The only useful information for the programmer is contained in the Unix man pages and function prototype definitions. Unfortunately, much of the information in the man pages is just plain wrong. For example, in many places the man pages tell you that search pointers are returned in the ptrlist in a SynsetPtr structure. However, this is only true if you are doing a tree-like search as in a hypernym tree type search. Regular searches return nothing in the ptrlist. Instead, in regular searches the information is returned in the ptroff array of longs.

7.2 Organization of AutoExtract Summarizer

7.2.1 Lexer

The first part of the system is the lexer. The lexer assumes that the text is single spaced and that paragraphs are separated by blank lines or indented. The lexer's job is to scan the text and convert it into a stream of unique numeric tokens that is sent to the parser. It does this by searching the text for matches for each of the token definitions listed below.

- LWORD – words that begin with a lower case letter
- UWORD – words that begin with an upper case letter
- TERM – hyphenated words and words with special characters
- BONUS – special words that indicate that the sentence is more important than other sentences
- STIGMA – special words that indicate that the sentence is less important than other sentences
- COMMON – noise, common words such as “a,” “an,” “the” that are used in the tree structure but not in lexical chaining
- punctuation – periods, commas, colons, semicolons, exclamation marks
- NUMBER – defined to include both integers and dotted-decimal

- indent tokens - paragraph indentation
- blank token - whitespace at the beginning of a line

Whenever a definition is matched, the lexer immediately returns the numeric token that represents that item to the parser. The parser has access to both the token and the string that corresponds to it. Non-matches (e.g. excess tabs, spaces, unmatched characters, and newlines) are thrown away. Anything thrown away is echoed to the screen for debugging purposes.⁶

7.2.2 Parser

Parse Tree

The second piece, the parser, is the major portion of the summarizer. The parser uses a simple grammar to arrange words into sentences, titles, and paragraphs. (See Figure 7.2 on page 188 for the Bison grammar that is used to create the parse tree.) During this phase, word location, sentence location, paragraph location, title word attributes, header word attributes, uppercase word attributes, BONUS word, and STIGMA word attributes are stored in the parse tree that is based on the grammar.⁷

⁶Special definitions and tokens had to be created to cover cases such as sentences ending with symbols after the normal sentence terminator such as "!"". A useful trick here is to define things that you want to ignore as COMMON words, for example "...". Common words are placed in the parse tree by the parser, but receive no processing.

⁷A text is a sequence of one or more paragraphs that may or may not be followed by a blankTokenList. A paragraph is a blankTokenList followed by one or more sentences. (A title is just a paragraph with a single sentence.) A sentence is one or more words that may or may not be followed by a period, exclamation, or question mark. A blankTokenList is a one or more blank or indent tokens. A blank token is any combination of zero or more spaces or tabs followed by a newline, but only when the combination occurs at the beginning of a line of text. An indent token is two spaces, but only when they occur at the beginning of a line of text. A word is a lower-case word (LWORD), a word that begins with an upper-case

The internal structure of the treeNodes used in the parse tree is shown in Figure 7.3 on page 189.

Breaking the text up this way is necessary, because sentences and paragraphs may require special processing. For example we must recognize titles, first and last paragraphs, and the first and last sentence of each paragraph. Moreover, sentences are used as temporary queues during lexical chain construction.

I originally thought to use a part-of-speech tagger, and a shallow parser, along with a text segmentation algorithm derived from Hearst (1994). This is the method that Barzilay & Elhadad (1997) use. However, the WordNet utility package already contains methods for finding word stems. Also, my algorithm for finding WordNet stems (See Figure 7.4 on page 190.) obviates any need for a part-of-speech tagger or shallow parser.

Every word in the parse tree has eleven separate real numbers associated with it: XSRchainScore, SRchainScore, MSRchainScore, uwordScore, bonusScore, lastSentScore, paraTitleScore, lastParaScore, firstSentScore, SRchainScore, and stigmaScore.

All of the words that are sent to parser are placed in the parse tree, but not all receive scores. For example, COMMON words and NUMBERS are not scored.

letter (UWORD), a word from our BONUS word list, a word from our stop word list (COMMON word i.e. noise), a word from our STIGMA word list, a TERM (e.g. "lower-case"), or a NUMBER.

Chain Stack

Only LWORDS, UWORDS, and TERMS receive further processing at this point. The morphological stems of these words are stored in a frequency hash table for later lookup.

Whenever an LWORD, UWORD, or TERM is processed, it is first converted to lower case (since all entries in WordNet are in lower case) then WordNet's stemmer (morphstr) is used to find the morphological noun stem of the word, if it exists (e.g. the stem of feet is foot). In addition, if it is a UWORD the treeNode is marked accordingly. This stem is stored in the chain node, along with the original word, and is used for all subsequent lookup operations for lexical chaining (e.g. for hypernyms). This is done in a cascaded fashion. (See Figure 7.4 on page 190.)

First we try to find the noun stem, if not found then the verb stem, if not found then the adjective stem, if not found then the adverb stem. Thus, our lexical chainer will give first priority to nouns, then to verbs, then adjectives, then adverbs. **Unlike most researchers who only use nouns, all WordNet parts of speech appear in our chains.** Both the original word and the morphological stem are stored in a chainNode. At the same time, the parser attempts to form a noun compound with the current word and the previous word. If successful, this new combination takes the place of both old stems. This is done because previous researchers have shown that noun compounds are more meaningful in overall chain structure than simple nouns.

The order of processing (noun stem, verb stem, adjective stem, then adverb stem as shown in Figure 7.4 on page 190) is based on intuition about the relative importance of the different categories. This intuition, of course, is reinforced by the work of previous researchers, and by the relative sizes of the different databases, with the noun database in WordNet being the largest. Given the time, however, we should run an experiment where each possible ordering is tried to verify which would give the best long term results.

ChainNodes are then placed into a chainQueue, which is processed when a sentence is recognized. Processing is done based on sentences since the processing for SRchainScore and MSRchainScore is done based on the distance between sentences. When the chainQueue is processed, nodes are processed one by one and placed into a chainStack using the algorithm in Figure 7.5 on page 191.

Bonus words, COMMON words (stop words) and STIGMA words are recognized and the nodes marked and placed in the parse tree, but they are not considered for chaining. Most researchers don't consider COMMON words for chaining, and it was decided not to chain BONUS words or STIGMA words because they were already being used to boost the sentence score. In retrospect, we should probably have tested the chaining procedure both with and without BONUS words, then made a decision. Numbers are simply stored in the tree without any special processing.

ChainNodes are used to store information such as word frequency, chain strength, synonym file offsets from WordNet, pointers to hypernym and holonym lists, and structure information such as next node in chain and next chain. The internal

structure of a chainNode is shown in Figure 7.6 on page 192.

The stems of each of these words is then looked up in WordNet and the synsets (file offsets) for each word are stored with that word. The chaining procedure then tries to find a lexical chain relationship between the synsets of a word in the queue and the words in the chain stack.

7.2.3 Lexical Chain Creation

A lexical chain is nothing more than a list of words that have been shown to have some sort of lexical relationship (e.g. word A could be a synonym of word B). Lexical chains, however, have been shown to contain as much structure information as rhetorical structures. This is because they allow you to focus on the topics in the text. For example, apple and orange are both hyponyms of fruit, and beets and turnips are both hyponyms of vegetables. Continuing on, fruits and vegetables are both hyponyms of produce. Thus, a paragraph with all these items mentioned might be construed to be *about* the topic of produce. This power of abstraction, or topic identification, is the major reason for using hypernyms and holonyms in lexical chains. (See Figure 4.4 on page 92.)

Initially, each word to be chained is represented as an array of the file offsets of all the synsets that contain it. As chaining proceeds and relations are built between words, synsets (offsets) that aren't part of the current chaining operation are discarded, that is we keep the intersection of the sets of file offsets. As a result, the sense of the lexical chain is progressively narrowed.

Chain nodes are connected together in linked lists to form chains. The chains are then linked together in a chain stack. (See Figure 7.7 on page 193.)

I used three types of lexical chain relations (as most researchers have done in the past): extra-strong relations, strong relations, and medium strength relations. Extra-strong chains are formed throughout the document (thematic word frequency).

Since all nodes in a chain must contain the same set of synsets we only need to compare with the first node in the chain to make a decision about adding. And that means that we only have to store synset offsets in the first node of the chain. Whenever an addition is made to a chain that chain is moved to the top of the chain stack, and all searches for matches begin at the top of the stack. (The extra-strength relation XSRchainScore doesn't actually add to the chain, so it doesn't modify the stack structure.) Since I use a greedy algorithm for adding to a chain (i.e. I add to the first chain that presents a connection.), the more recently a chain has had an addition the more likely it is to be considered for a new addition, since we search from the top of the stack and it will be nearer the top. This ensures that close connections are preferred over distant connections. Also whenever a new addition is made to a chain, we keep the intersection of the synsets of the new node and the synsets of the chain as a whole (stored in the first node). Thus, the sense of the chain, the topic, is progressively narrowed.

I used a greedy algorithm for forming chains because I was much more interested in speed than I was in creating perfect chains. I was willing to accept less than perfect

chains in order to approach the speed of commercial summarizers. My unoptimized algorithm took about 30 seconds per test, whereas the commercial versions took about 5 seconds. However, my summarizer performed better than the commercial versions on both the overlap test and the human evaluation test, and was much faster than other lexical chain implementations. (Some chaining algorithms take literally hours to run.) Moreover, I have a number of optimizations for speed that I can make that should allow me to approach 5 seconds for the test texts in the appendices.

Forming chains paragraph by paragraph to create paragraph chains or tile by tile (as Barzilay & Elhadad (1997) did) then merging to form an overall chain structure is simply too slow. Moreover, the results are not as complete or accurate as direct chaining all entries. It might be argued that considering the entire document for chaining is not scalable since the entire chain stack structure must be checked for each new chain created, but this is not true. The frequency portion of extra strong chains (really thematic word frequency) is handled automatically by hash table insertion and lookup when tokens first arrive, and is extremely fast. True, the entire chain stack structure must be checked for word matches, but this is still fairly rapid. Moreover, in future versions this check could be eliminated completely by simply storing pointers in the treeNodes that point into the hash table that tracks frequencies. Moreover, strong chain relations are always limited to a distance of plus or minus seven sentences from the current sentence, and medium strength chain relations are always limited to plus or minus four sentences. So the formation

of strong and medium-strength chains is independent of document size.

Why use a distance of seven for strong chain relations and four for medium-strength relations instead of 42 or 95? The easiest answer, of course, is that it has always been done that way. That doesn't mean that that is the right answer, but it does mean that we should take that approach unless we have some good reason for doing otherwise. Moreover, Andersen (2000) concluded that restricting his algorithm to a distance of ten sentences for the medium-strength relation was crucial for its success.

Of course, the seven and four search lengths implicitly assume that the so-called strong relation is more important than the medium-strength relation, and I kept these lengths because I trusted the work of past researchers. With twenty-twenty hindsight, I should have observed the effect of increasing and decreasing the search lengths of both variables. As the test results in the last chapter show, the medium-strength relationship actually may be much more important than the strong relation.

When we start chaining, the chain stack is empty, so there is nothing to relate to the word being processed. In that case, the word is simply placed into a trivial chain and pushed onto the chain stack. Since no relation was used to add that word to the chain, all of its chain scores (XSRchainScore, SRchainScore, and MSRchainScore) contain zero. The chaining procedure maintains separate scores for each word for XSRchainScore (extra strength), SRchainScore (strong), and MSRchainScore (medium strength). We then attempt to add each word in the queue to the chains in the chain stack through one of the following three relations:

Extra-strong relations - XSRchainScore

When a chain queue is processed the entire stack of chain structures is checked for extra-strong relations (i.e. word matches). To find word repetition we simply compare the stems of two words. (See Figure 7.5 on page 191.) If found the word and the word it matches both receive an extra-strong score and it is inserted in the chain at that point. This score goes into the XSRchainScore for that word.

Strong relations - SRchainScore

If an extra-strong relation is not found, then sentences within a distance of plus or minus seven of the current sentence are checked for strong relations – direct synonym matches, direct antonym matches, direct meronym matches, direct hypernym matches, direct holonym matches, and direct hyponym matches. Here a match means that the list of synsets for word A and the list of synsets for word B have at least one synset in common. If found, the word receives a strong score. The word it matches receives whichever is higher its present score or the strong word score. This score goes into the SRchainScore. The intersection of the synsets of the new word and the word it relates to are kept in the new word which then becomes the first word in the chain.

See Figure 7.8 on page 194.

Medium-strength relations - MSRchainScore

If a strong relation is not found, then sentences within a distance of plus or minus four of the current sentence are checked for medium strength relations. The medium strength search routine returns four linked lists from WordNet corresponding to hypernym, hyponym, holonym, and meronym searches. These lists are searched until the first synset match is found. A match means that the list of synsets for word A and the list of synsets for word B have at least one synset in common. If a match is found, the new word receives a medium-strength score that decreases with the number of separation links. The word it matches receives whichever is higher its present score or the medium-strength word score. This score goes into the MSRchainScore. The intersection of the synsets of the new word and the word it relates to are kept in the new word which then becomes the first word in the chain.

See Figure 7.9 on page 195.

Finally, if no match can be found a new chain with one entry with all zero scores is created.

7.2.4 Sentence Recognition

GlobalTitles and paraTitles are recognized when sentences are organized by the parser into paragraphs, since titles are only paragraphs with a single sentence. If recognized, globalTitle words are inserted into the globalTitle hash table (once) and paraTitle words are inserted into the paraTitle hash table (for each new paragraph).

If the paragraph doesn't have a title then the paraTitle hash table is cleared. Actually, only LWORDS, UWORDS, or TERMS are inserted into any hash table. COMMON words, NUMBERS, etc. are thrown away. During this same phase, the first and last sentences in the paragraph are also marked, if they are not titles. If the sentence isn't a global or paragraph title then the words in the sentence are checked for matches with the words in the global and paragraph hash tables. If found, that parameter score for that word is incremented.

There are three separate chained hash tables that use hashNodes. The first table is used to store word frequencies for the stems of LWORDS, UWORDS, and TERMS. These are used in computing the XSRchainScore (extra strong chain score) for lexical chains. There is another hash table used to store words from the main title and still another hash table used to temporarily store sub-title (or paragraph title) words. Chained Hash tables are commonly used in compiler construction and were chosen because they are easy to use and provide fast lookup.

The hash function used for storing word frequencies is adapted directly from the hashpjw function from page 436 of *Compilers, Principles, Techniques, and Tools*, by Aho, Sethi & Ullman (1988). This function adds the characters in the key then mods with the number of entries to produce a number that is used to index into a hash table. This is already one of the better hash functions available from compiler technology, and a prime number is always used for the base size of the hash table, more evenly spread the entries, for even faster lookup and insertion.

The internal structure of a hashNode is shown in Figure 7.10 on page 196.

7.2.5 Paragraph Recognition

As paragraphs are recognized and placed in the parse tree, the first and last sentence of each paragraph is marked. Once the parse tree is finished, it is scanned to mark sentences in the first and last paragraphs of the text, then the sentence scoring procedure is started.

7.2.6 Sentence Scoring

The scoreSentence procedure first allocates and zeroes an array of sortNodes large enough to hold pointers to all the sentences in the document, then a score array large enough to hold the scores of all the sentences, then opens the output file for writing. By default the output file is named "summary.txt," but this can be overridden with command line arguments. Sentence scores for each of the eleven parameters of the salience function (listed below) are zeroed.

- XSRchainScore - extra-strong chain score, same as Luhn's term frequency
- SRchainScore - strong chain score - immediate synonyms, antonyms etc.
- MSRchainScore - medium strong chain score - hypernym, hyponym, meronym, and holonym chain connections
- globalTitleScore - words from global title in sentence
- paraTitleScore - words from paragraph title (header) in sentence
- lastParaScore - is the sentence in the last paragraph

- firstSentScore - is the sentence the first sentence in the paragraph
- lastSentScore - is the sentence the last sentence in the paragraph
- bonusScore - BONUS words (cue words) in sentence
- uwordScore - upper-case words in sentence
- stigmaScore - STIGMA words (negative cue words) in sentence

Next the sentence nodes for each sentence are scanned and the scores for each parameter are added to the sentence total for that parameter. (Only the first node in the sentence contains chain and positional scores.) As each sentence is processed, the maximum document values for each of the parameters is tracked. Thus, each sentence has eleven separate scores that are sums of the eleven separate scores of each word in the sentence. The example summarization of *The Gettysburg Address* in the last section should make this clear.

Once all the sentences are processed, the individual parameter scores for each sentence are normalized by dividing by the max values that we tracked earlier. We want normalized parameter scores so that all training documents will have the same parameter score range.

If we are testing phase, then once the parameter scores are normalized they are summed together using our linear salience function. As sentence scores are calculated, they are placed in the scoreList array. This array of structures (which tracks sentence scores and location) is then sorted (using quicksort) so the highest

score is at the top. Then based on the percentage of sentences that we want for our summary, the top n sentences are selected, then re-sorted into sentence order. The nodes for these sentences are then used to write the summary to the output file.

Quicksort was chosen for sorting because it is easy to code and reasonably fast, being on average $n \log(n)$ in its complexity. (Quicksort in the *worst* case is n^2 in its complexity, so others might prefer heapsort or mergesort where the complexity is always $n \log(n)$.)

The internal structure of a `sortNode` is shown in Figure 7.11 on page 197.

If we are in the training phase then the individual parameter totals for each sentence are written out to a comma-delimited data file that is used for function optimization as detailed in the next section.

7.3 Training and Optimization

Training data was constructed using the same technique that Witbrock & Mittal (1998), Georgantopoulos (1996), and others have used. I first downloaded 50 random articles from Business Week Magazine (www.businessweek.com) from January through May 2001. Each article came with an abstract. The abstract was then used to create an extract by aligning sentences in the abstract with sentences in the main text. Matching sentences were then placed in the extract. Admittedly, this process is far from perfect, since you are using your human judgement to find the *closest* match or matches for a summary sentence in the main text. But the alternative,

finding human volunteers to grade all the sentences in all the texts for inclusion in a summary was just not feasible.

There are several fundamental problems in mapping parameter scores for sentences via a salience function to an overall sentence score that can be used for selecting sentences for extraction. First, we don't know what, if any, *score* the human abstractors assigned or might have assigned to individual sentences. All we have are a group of sentences that were finally selected to be the summary. That is, the computed scores for every selected sentence must be higher than the computed scores for every non-selected sentence. However, any given sentence in the summary might be the highest, or lowest, scored sentence in that summary. Salience function scores must then map to two possible states. These states could be represented by any two output values; I chose 0 and 1000.0 to spread out actual computed results once the salience function was finalized. In retrospect, using 0.0 and 1.0 would have been perfectly adequate.

Secondly, for parameter scores to be usable inside an overall salience function, individual parameter scores for each sentence, for any given text, must range from some *worst* value for selection to some *best* value for selection. We also must be able to insert any arbitrary number of sentence scores at any point. Thus, parameter scores must be real numbers, not integers, and they must range from some minimum value to some maximum value. Range control is accomplished by normalizing all parameter scores so that they range from 0.0 to 1.0. For example, the sentence with the highest XSRchainScore has its XSRchainScore normalized to 1.0. Normalization

is accomplished by dividing the parameter score by the maximum value in the text for that parameter. These normalized parameter scores are then the input parameters to the salience function. Position parameter scores are either 0 or 1.0. Either the sentence is in the desired position or it isn't.

Scores were computed for each of the eleven *independent* parameters for each of the sentences in each of the training texts. If that sentence should have been in the summary, it was assigned an output score of 1000.0; otherwise it was assigned an output score of 0. These data were written to a text file by the parser as comma-delimited data. A partial example is shown in Figure 7.12 on page 198.

I compiled all the normalized parameter values for all the sentences for all the training examples into one comma-delimited ASCII data file, then fed the data into the DataFit program from Oakdale Engineering. DataFit found three least squares error solutions, one non-linear using exponentials, one linear with an added constant, and one linear that only involved constant multipliers for the parameters. I wanted to focus on the parameters, I didn't want the overhead of computing exponentials, and the error for all three was about the same. I, therefore, chose the linear function that only involved constant multipliers for the parameters. That function is shown in Figure 7.13 on page 199. The multipliers from this final score formula were then integrated into the salience function in the parser in preparation for the testing phase.

The DataFit program from Oakdale Engineering is designed to find the function that gives the best fit for a data set, where best fit is defined as minimizing least

squares error. It has both linear and non-linear functions in its database and you can define your own function if you wish. You can direct it to try out all the possible solutions in its repertoire, then give you the ones that work. This is what I did.

Of course, my method of normalizing for optimization is not the only method to consider. My method focuses on normalizing at the parameter level, I could have focused on normalizing at the score level by dividing the sentence score by the sum of all the sentence scores. This latter method, however, presupposes that you know how to combine the different parameter scores to get an overall score; we don't. We know the individual parameter scores; we know the result that we want; we don't know the mapping, weights involved or anything else. Thus, I feel that my way of normalizing is the most *reasonable*, since we are normalizing what we know, the individual parameter scores.

In the future, we might want to investigate the Simplex Method from linear programming or the use of an Artificial Neural Network (ANN) as a means of optimizing the parameter multipliers, but the linear regression performed by the DataFit program seems adequate for now.

7.4 AutoExtract Summarization by Example

There are many parts and procedures in my summarizer and an example is probably the best way to explain its operation. To do this, I modified my parser to print out intermediate steps. A step-by-step summarization of *The Gettysburg Address*

follows. (The full text is shown in Appendix A on page 232.)

The lexer converts the first sentence into the following stream of tokens and their corresponding strings, which are then sent to the parser.

```
UWORD: Fourscore
COMMON: and
LWORD: seven
LWORD: years
LWORD: ago
BONUS: our
LWORD: fathers
LWORD: brought
COMMON: forth
COMMON: upon
BONUS: this
LWORD: continent
COMMON: a
COMMON: new
LWORD: nation
LWORD: conceived
COMMON: in
LWORD: liberty
COMMON: and
LWORD: dedicated
COMMON: to
COMMON: the
LWORD: proposition
BONUS: that
COMMON: all
LWORD: men
COMMON: are
LWORD: created
LWORD: equal
```

There are two different places for words to be stored by the parser, the parse tree that corresponds to the simple Bison grammar and a chain stack for those words that are considered for chaining. The parse tree is the tree that is built from the

words of the document using the simple Bison grammar. The chain stack is a stack structure where each item on the stack is a pointer to a lexical chain. When tokens first arrive at the parser, token information is used to create both a `treeNode` and a `chainNode` (if called for). The `treeNode` is marked with the attribute of the token (e.g. `LWORD`, `UWORD`, `BONUS`, etc.), then the `treeNode` is placed in the parse tree.

`TreeNodes` are used to store all the words and punctuation of the text in this parse tree. A parse tree implementation of a simple grammar is necessary to break the text into titles, sentences, sub-titles, and paragraphs. This is necessary because some functions operate on sentences and use inter-sentence distance as a parameter and because some positional information is based on paragraphs. `TreeNodes` store the original word, the stem of that word (if it is in WordNet), tree structure information (e.g. next tree node in list), and score information for `BONUS` words, `STIGMA` words, title words, and position (e.g. first sentence in paragraph).

Every word in the parse tree has eleven separate scores associated with it: `XSR-chainScore`, `SRchainScore`, `MSRchainScore`, `uwordScore`, `bonusScore`, `lastSentScore`, `paraTitleScore`, `lastParaScore`, `firstSentScore`, `SRchainScore`, and `stigmaScore`.

All of the words that are sent to parser are placed in the parse tree, but not all receive scores. `Treenodes` are always created for `COMMON` words and `NUMBERS` and placed in the parse tree, but all the scores for these words are always zero. Their only uses are for completing sentences in the parse tree and final printout.

Only the words recognized as `UWORDS` get their `UWORD` score, `uwordScore`

, marked (i.e. incremented from 0 to 1). In the first sentence only "Fourscore" was recognized as a UWORD. Only words recognized as BONUS words get their BONUS score incremented from 0 to 1. "our," "this," and "that" were recognized as BONUS words. Only words recognized as STIGMA words get their STIGMA score incremented. There were no STIGMA words recognized in the first sentence.

Only LWORDS, UWORDS, and TERMS receive further processing at this point, that is consideration for chaining. The morphological stems of these words are stored in a frequency hash table for later lookup.

Whenever an LWORD, UWORD, or TERM is processed, (See Figure 7.4 on page 190.) it is first converted to lower case (since all entries in WordNet are in lower case) then WordNet's stemmer (morphstr) is used to find the morphological noun stem of the word, if it exists (e.g. the stem of feet is foot). This stem is used for all subsequent lookup operations for lexical chaining (e.g. for hypernyms).

The chains created from these LWORDS and UWORD are shown below. Word repetition is indicated by F:? for the frequency of the word at this point in chaining. The morphological stem of the word (if found in WordNet) is shown next followed by the original word in parentheses. Finally, the synset list (as file offsets) is shown, but only for the first node in the chain, since only the first node is checked for chaining operations.

In the first sentence, the words "ago," "brought," "conceived," "dedicated," and "created" were not found in the WordNet database. Thus, there are no synsets (actually file offsets corresponding to synsets) listed for them. The only way that

they can participate in chaining is for the word to be repeated (extra-strength chain relation). At this point the chains containing these words are considered trivial chains with zeros in all scores.

The words "continent" and "proposition" were found in WordNet and their synsets are listed. However, they were not related to any other words in the chain stack, and at this point are also considered trivial chains. All of the words placed in the chain stack so far have zero scores for their XSRchainScore, SRchainScore, and MSRchainScore buckets.

The words "fathers," "men," "nation," and "equal" have been related. The intersection of all the synsets that participated in the chaining is kept in the first node of the chain, [7075378 7461763]. Also the words "seven," "liberty," "years," and "Fourscore" have been related. They have only a single synset remaining as the intersection, [9893975].

CHAIN STACK DUMP

```
F:1/ago(ago) []
F:1/bring(brought) []
F:1/continent(continent) [6835820 6836334 6836510 6837221 6837693
6838825 6840014 6840277 6840536 6841750 6841982 6842207 6842401]
F:1/conceive(conceived) []
F:1/dedicate(dedicated) []
F:1/proposition(proposition) [5040541 5059358 5059457 5059598
5059863 5059980 5060175 5060347]
F:1/create(created) []
F:1/father(fathers) [7075378 7461763]
->F:1/nation(nation) []
->F:1/man(men) []
->F:1/equal(equal) []
```

```
F:1/seven(seven) [9893975]
  ->F:1/liberty(liberty) []
  ->F:1/year(years) []
  ->F:1/fourscore(Fourscore) []
```

Now the second sentence is processed.

```
COMMON: Now
COMMON: we
COMMON: are
LWORD: engaged
COMMON: in
COMMON: a
LWORD: great
LWORD: civil
LWORD: war
LWORD: testing
COMMON: whether
BONUS: that
LWORD: nation
COMMON: or
COMMON: any
LWORD: nation
BONUS: so
LWORD: conceived
COMMON: and
BONUS: so
LWORD: dedicated
COMMON: can
LWORD: long
LWORD: endure
```

TreeNodes are created to store each token-word combination in the parse tree and the BONUS words in the tree have their BONUS word score incremented, then the LWORDS are passed on for chaining. We see that new trivial chains have been created for “endure,” “engage,” “conceived,” “great,” and “dedicated” because they

weren't found in WordNet, and a chain with no relations has been created for civil-war, which was created by combining the words civil and war to create a more specific term. That means that the following LWORDS were added to pre-existing chains.

LWORD: testing
LWORD: nation
LWORD: nation
LWORD: long

"nation" was word repetition and increased the frequency of the word "nation" in the next to the last chain. "testing" was added to the chain that contained "continent" through the MSRchain relation. "long" was added to the last chain also through the MSRchain relation.

CHAIN STACK DUMP

F:1/civil-war(civil-war) [626011 851826 853846]
F:1/endure(endure) []
F:1/engage(engaged) []
F:1/ago(ago) []
F:1/bring(brought) []
F:1/test(testing) [6672286]
 ->F:1/continent(continent) []
F:2/conceive(conceived) []
F:2/dedicate(dedicated) []
F:1/proposition(proposition) [5040541 5059358 5059457
 5059598 5059863 5059980 5060175 5060347]
F:1/create(created) []
F:1/father(fathers) [7075378 7461763]
 ->F:3/nation(nation) []
 ->F:1/man(men) []
 ->F:1/equal(equal) []

```

F:1/long(long) [10951137]
  ->F:1/seven(seven) []
  ->F:1/liberty(liberty) []
  ->F:1/year(years) []
  ->F:1/fourscore(Fourscore) []

```

Now the third sentence is processed.

```

COMMON: We
COMMON: are
LWORD: met
COMMON: on
COMMON: a
LWORD: great
LWORD: battlefield
COMMON: of
BONUS: that
LWORD: war

```

Again the BONUS words are marked and their treeNodes are inserted into the tree and the LWORDS are passed on for chaining. Empty chains are formed for "met" and "battlefield," "war" is added to the chain with "civil-war".

CHAIN STACK DUMP

```

F:1/great(great) []
F:1/battlefield(battlefield) [6384327 6274908 6320533 6369402]
F:1/meet(meet) [5543177 278242 5551445 5551573 5551776 5551854 5552012]
F:1/war(war) [609840 610138 610268 610417 617941 619159 621140
  621298 643613 643714 803370 852529 852735 852986 853179
  853664 854360 854569 854887 855068 855316 855623 855981
  856551 856729 856891 857620 857807 858055 858212 858512
  858919 859169 859377 859637 859928 10072161]
  ->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []

```

F:1/ago(ago) []
 F:1/bring(brought) []
 F:1/test(testing) [6672286]
 ->F:1/continent(continent) []
 F:2/conceive(conceived) []
 F:2/dedicate(dedicated) []
 F:1/proposition(proposition) [5040541 5059358 5059457
 5059598 5059863 5059980 5060175 5060347]
 F:1/create(created) []
 F:1/father(fathers) [7075378 7461763]
 ->F:3/nation(nation) []
 ->F:1/man(men) []
 ->F:1/equal(equal) []
 F:1/long(long) [10951137]
 ->F:1/seven(seven) []
 ->F:1/liberty(liberty) []
 ->F:1/year(years) []
 ->F:1/fourscore(Fourscore) []

Now the fourth sentence is processed. BONUS words are again marked and placed in the tree.

COMMON: We
 COMMON: have
 LWORD: come
 COMMON: to
 LWORD: dedicate
 COMMON: a
 LWORD: portion
 COMMON: of
 BONUS: that
 LWORD: field
 COMMON: as
 COMMON: a
 LWORD: final
 TERM: resting-place
 BONUS: for
 BONUS: those
 COMMON: who

COMMON: here
 LWORD: gave
 COMMON: their
 LWORD: lives
 BONUS: that
 BONUS: that
 LWORD: nation
 COMMON: might
 LWORD: live

Then the LWORDS and TERMS are passed on for chaining. "resting-place," "gave," "live," "come," and "dedicate" all form trivial chains. "nation" is again chained by word repetition. "lives" is added to the last chain. "final" is added to the chain with "testing." "field" is added to the chain with "battlefield." "portion" is added to the chain with "proposition."

CHAIN STACK DUMP

```
F:1/resting-place(resting-place) []
F:1/give(gave) [3920674]
F:1/live(live) []
F:1/come(come) []
F:1/great(great) []
F:1/field(field) [6384327 3638970 6282882 6318813 6324781
6335561 6346705]
->F:1/battlefield(battlefield) []
F:1/meet(met) [5543177 278242 5551445 5551573 5551776
5551854 5552012]
F:1/war(war) [609840 610138 610268 610417 617941 619159 621140
621298 643613 643714 803370 852529 852735 852986 853179
853664 854360 854569 854887 855068 855316 855623 855981
856551 856729 856891 857620 857807 858055 858212 858512
858919 859169 859377 859637 859928 10072161]
->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []
```


F:1/ago(ago) []
 F:1/bring(brought) []
 F:1/final(final) [5554348 5549662 5550428]
 ->F:1/test(testing) []
 ->F:1/continent(continent) []
 F:2/conceive(conceived) []
 F:3/dedicate(dedicated) []
 F:1/portion(portion) [18916 9945697 4749333 4867079 9946426
 9946775 9947031 9947198 9947291 9947647]
 ->F:1/proposition(proposition) []
 F:1/create(created) []
 F:1/father(fathers) [7075378 7461763]
 ->F:4/nation(nation) []
 ->F:1/man(men) []
 ->F:1/equal(equal) []
 F:1/life(lives) [10843624 10866987 10867147 10867885
 10868236 10868596 10869347]
 ->F:1/long(long) []
 ->F:1/seven(seven) []
 ->F:1/liberty(liberty) []
 ->F:1/year(years) []
 ->F:1/fourscore(Fourscore) []

Now the fifth sentence is processed.

COMMON: It
 COMMON: is
 LWORD: altogether
 LWORD: fitting
 COMMON: and
 LWORD: proper
 BONUS: that
 COMMON: we
 COMMON: should
 COMMON: do
 BONUS: this

BONUS words are marked and entered into the tree, then the LWORDS are passed for chaining. "altogether" and "proper" form trivial chains. "fitting" is

added to the chain with "gave."

CHAIN STACK DUMP

```
F:1/proper(proper) []
F:1/altogether(altogether) [10371715]
F:1/resting-place(resting-place) []
F:1/fit(fitting) [3856995 3858670]
  ->F:1/give(gave) []
F:1/live(live) []
F:1/come(come) []
F:1/great(great) []
F:1/field(field) [6384327 3638970 6282882 6318813
  6324781 6335561 6346705]
  ->F:1/battlefield(battlefield) []
F:1/meet(met) [5543177 278242 5551445 5551573 5551776
  5551854 5552012]
F:1/war(war) [609840 610138 610268 610417 617941 619159
  621140 621298 643613 643714 803370 852529 852735
  852986 853179 853664 854360 854569 854887 855068
  855316 855623 855981 856551 856729 856891 857620
  857807 858055 858212 858512 858919 859169 859377
  859637 859928 10072161]
  ->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []
F:1/ago(ago) []
F:1/bring(brought) []
F:1/final(final) [5554348 5549662 5550428]
  ->F:1/test(testing) []
  ->F:1/continent(continent) []
F:2/conceive(conceived) []
F:3/dedicate(dedicated) []
F:1/portion(portion) [18916 9945697 4749333 4867079
  9946426 9946775 9947031 9947198 9947291 9947647]
  ->F:1/proposition(proposition) []
F:1/create(created) []
F:1/father(fathers) [7075378 7461763]
  ->F:4/nation(nation) []
  ->F:1/man(men) []
  ->F:1/equal(equal) []
```

```
F:1/life(lives) [10843624 10866987 10867147 10867885
 10868236 10868596 10869347]
->F:1/long(long) []
->F:1/seven(seven) []
->F:1/liberty(liberty) []
->F:1/year(years) []
->F:1/fourscore(Fourscore) []
```

Now the sixth sentence is processed.

```
STIGMA: But
COMMON: in
COMMON: a
LWORD: larger
LWORD: sense
COMMON: we
COMMON: cannot
LWORD: dedicate
COMMON: we
COMMON: cannot
LWORD: consecrate
COMMON: we
COMMON: cannot
LWORD: hallow
BONUS: this
LWORD: ground
```

BONUS and STIGMA words are marked and placed in the tree, and the LWORDS are passed for chaining. "larger," "consecrate," "hallow," "ground," and "dedicate" all form new trivial chains. "sense" is added to the chain with "portion."

CHAIN STACK DUMP

```
F:1/consecrate(consecrate) []
F:1/hallow(hallow) []
F:1/grind(ground) [7367396]
```

F:1/large(larger) [3974960]
F:1/proper(proper) []
F:1/altogether(altogether) [10371715]
F:1/resting-place(resting-place) []
F:1/fit(fitting) [3856995 3858670]
 ->F:1/give(gave) []
F:1/live(live) []
F:1/come(come) []
F:1/great(great) []
F:1/field(field) [6384327 3638970 6282882 6318813
 6324781 6335561 6346705]
 ->F:1/battlefield(battlefield) []
F:1/meet(met) [5543177 278242 5551445 5551573
 5551776 5551854 5552012]
F:1/war(war) [609840 610138 610268 610417 617941 619159
 621140 621298 643613 643714 803370 852529 852735
 852986 853179 853664 854360 854569 854887 855068
 855316 855623 855981 856551 856729 856891 857620
 857807 858055 858212 858512 858919 859169 859377
 859637 859928 10072161]
 ->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []
F:1/ago(ago) []
F:1/bring(brought) []
F:1/final(final) [5554348 5549662 5550428]
 ->F:1/test(testing) []
 ->F:1/continent(continent) []
F:2/conceive(conceived) []
F:4/dedicate(dedicated) []
F:1/sense(sense) [4951584 4952578]
 ->F:1/portion(portion) []
 ->F:1/proposition(proposition) []
F:1/create(created) []
F:1/father(fathers) [7075378 7461763]
 ->F:4/nation(nation) []
 ->F:1/man(men) []
 ->F:1/equal(equal) []
F:1/life(lives) [10843624 10866987 10867147 10867885
 10868236 10868596 10869347]
 ->F:1/long(long) []
 ->F:1/seven(seven) []
 ->F:1/liberty(liberty) []

->F:1/year(years) []
->F:1/fourscore(Fourscore) []

Now the seventh sentence is processed.

COMMON: The
LWORD: brave
LWORD: men
LWORD: living
COMMON: and
LWORD: dead
COMMON: who
LWORD: struggled
COMMON: here
COMMON: have
LWORD: consecrated
COMMON: it
COMMON: far
COMMON: above
BONUS: our
LWORD: poor
LWORD: power
COMMON: to
LWORD: add
COMMON: or
LWORD: detract

BONUS words are marked and LWORDS are sent for chaining. "poor," "add," "detract," "consecrate," and "hallow" all form new trivial chains. "brave" and "dead" are added to the chain with "ground." "men" is added by word repetition (man). "struggled" is added to the chain with "war." "power" is added to the chain with "larger."

CHAIN STACK DUMP

F:1/poor(poor) []
F:1/add(add) []
F:1/detract(detract) []
F:2/consecrate(consecrate) []
F:1/hallow(hallow) []
F:1/dead(dead) [5957883 5960295 5962749 7203851]
 ->F:1/brave(brave) []
 ->F:1/grind(ground) []
F:1/power(power) [3714294 4050476 1754717 1756614
 3930426 3930785 3931015 3931458 4042313 4042432
 4042603 4042993 4043761 4043876 4043948 4045365
 4046299 4046475 4046609 4046763 4046959 4047189]
 ->F:1/large(larger) []
F:1/proper(proper) []
F:1/altogether(altogether) [10371715]
F:1/resting-place(resting-place) []
F:1/fit(fitting) [3856995 3858670]
 ->F:1/give(gave) []
F:2/live(live) []
F:1/come(come) []
F:1/great(great) []
F:1/field(field) [6384327 3638970 6282882 6318813
 6324781 6335561 6346705]
 ->F:1/battlefield(battlefield) []
F:1/meet(met) [5543177 278242 5551445 5551573
 5551776 5551854 5552012]
F:1/struggle(struggled) [503611 506044 506162 5555979]
 ->F:1/war(war) []
 ->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []
F:1/ago(ago) []
F:1/bring(brought) []
F:1/final(final) [5554348 5549662 5550428]
 ->F:1/test(testing) []
 ->F:1/continent(continent) []
F:2/conceive(conceived) []
F:4/dedicate(dedicated) []
F:1/sense(sense) [4951584 4952578]
 ->F:1/portion(portion) []
 ->F:1/proposition(proposition) []
F:1/create(created) []

```

F:1/father(fathers) [7075378 7461763]
  ->F:4/nation(nation) []
  ->F:2/man(men) []
  ->F:1/equal(equal) []
F:1/life(lives) [10843624 10866987 10867147 10867885
10868236 10868596 10869347]
  ->F:1/long(long) []
  ->F:1/seven(seven) []
  ->F:1/liberty(liberty) []
  ->F:1/year(years) []
  ->F:1/fourscore(Fourscore) []

```

Now the eighth sentence is processed.

```

COMMON: The
LWORD: world
COMMON: will
COMMON: little
BONUS: note
COMMON: nor
LWORD: long
LWORD: remember
COMMON: what
COMMON: we
LWORD: say
COMMON: here

```

BONUS words are marked and LWORDS are sent for chaining. "remember" and "say" form new trivial chains. "world" is added to the chain with "dead." "long" is added to the chain with "lives."

CHAIN STACK DUMP

```

F:1/remember(remember) []
F:1/say(say) [10390345]
F:1/poor(poor) []

```

F:1/add(add) []
F:1/detract(detract) []
F:2/consecrate(consecrate) []
F:1/hallow(hallow) []
F:1/world(world) [17954 5957883]
 ->F:1/dead(dead) []
 ->F:1/brave(brave) []
 ->F:1/grind(ground) []
F:1/power(power) [3714294 4050476 1754717 1756614
 3930426 3930785 3931015 3931458 4042313 4042432
 4042603 4042993 4043761 4043876 4043948 4045365
 4046299 4046475 4046609 4046763 4046959 4047189]
 ->F:1/large(larger) []
F:1/proper(proper) []
F:1/altogether(altogether) [10371715]
F:1/resting-place(resting-place) []
F:1/fit(fitting) [3856995 3858670]
 ->F:1/give(gave) []
F:2/live(live) []
F:1/come(come) []
F:1/great(great) []
F:1/field(field) [6384327 3638970 6282882 6318813
 6324781 6335561 6346705]
 ->F:1/battlefield(battlefield) []
F:1/meet(met) [5543177 278242 5551445 5551573
 5551776 5551854 5552012]
F:1/struggle(struggled) [503611 506044 506162 5555979]
 ->F:1/war(war) []
 ->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []
F:1/ago(ago) []
F:1/bring(brought) []
F:1/final(final) [5554348 5549662 5550428]
 ->F:1/test(testing) []
 ->F:1/continent(continent) []
F:2/conceive(conceived) []
F:4/dedicate(dedicated) []
F:1/sense(sense) [4951584 4952578]
 ->F:1/portion(portion) []
 ->F:1/proposition(proposition) []
F:1/create(created) []
F:1/father(fathers) [7075378 7461763]


```

->F:4/nation(nation) []
->F:2/man(men) []
->F:1/equal(equal) []
F:1/life(lives) [10843624 10866987 10867147
10867885 10868236 10868596 10869347]
->F:2/long(long) []
->F:1/seven(seven) []
->F:1/liberty(liberty) []
->F:1/year(years) []
->F:1/fourscore(Fourscore) []

```

Now the ninth sentence is processed.

```

STIGMA: But
COMMON: it
COMMON: can
COMMON: never
LWORD: forget
COMMON: what
COMMON: they
COMMON: did
COMMON: here

```

The one STIGMA word is marked and the one LWORD is sent for chaining.

“forget” forms a new trivial chain.

CHAIN STACK DUMP

```

F:1/forget(forget) []
F:1/remember(remember) []
F:1/say(say) [10390345]
F:1/poor(poor) []
F:1/add(add) []
F:1/detract(detract) []
F:2/consecrate(consecrate) []
F:1/hallow(hallow) []
F:1/world(world) [17954 5957883]

```

->F:1/dead(dead) []
->F:1/brave(brave) []
->F:1/grind(ground) []
F:1/power(power) [3714294 4050476 1754717 1756614
3930426 3930785 3931015 3931458 4042313 4042432
4042603 4042993 4043761 4043876 4043948 4045365
4046299 4046475 4046609 4046763 4046959 4047189]
->F:1/large(larger) []
F:1/proper(proper) []
F:1/altogether(altogether) [10371715]
F:1/resting-place(resting-place) []
F:1/fit(fitting) [3856995 3858670]
->F:1/give(gave) []
F:2/live(live) []
F:1/come(come) []
F:1/great(great) []
F:1/field(field) [6384327 3638970 6282882 6318813
6324781 6335561 6346705]
->F:1/battlefield(battlefield) []
F:1/meet(met) [5543177 278242 5551445 5551573
5551776 5551854 5552012]
F:1/struggle(struggled) [503611 506044 506162 5555979]
->F:1/war(war) []
->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []
F:1/ago(ago) []
F:1/bring(brought) []
F:1/final(final) [5554348 5549662 5550428]
->F:1/test(testing) []
->F:1/continent(continent) []
F:2/conceive(conceived) []
F:4/dedicate(dedicated) []
F:1/sense(sense) [4951584 4952578]
->F:1/portion(portion) []
->F:1/proposition(proposition) []
F:1/create(created) []
F:1/father(fathers) [7075378 7461763]
->F:4/nation(nation) []
->F:2/man(men) []
->F:1/equal(equal) []
F:1/life(lives) [10843624 10866987 10867147
10867885 10868236 10868596 10869347]

->F:2/long(long) []
->F:1/seven(seven) []
->F:1/liberty(liberty) []
->F:1/year(years) []
->F:1/fourscore(Fourscore) []

Now the tenth sentence is processed.

COMMON: It
COMMON: is
BONUS: for
COMMON: us
COMMON: the
LWORD: living
COMMON: rather
COMMON: to
COMMON: be
LWORD: dedicated
COMMON: here
COMMON: to
COMMON: the
LWORD: unfinished
BONUS: work
COMMON: which
COMMON: they
COMMON: who
LWORD: fought
COMMON: here
COMMON: have
BONUS: thus
COMMON: far
BONUS: so
LWORD: nobly
LWORD: advanced

BONUS words are marked and LWORDS are sent for chaining. "unfinished," "nobly," and "dedicated" form new trivial chains. "fought" is added to the chain with "struggled." "advanced" is added to the chain with "sense."

CHAIN STACK DUMP

F:1/unfinished(unfinished) []
F:1/nobly(nobly) []
F:1/forget(forget) []
F:1/remember(remember) []
F:1/say(say) [10390345]
F:1/poor(poor) []
F:1/add(add) []
F:1/detract(detract) []
F:2/consecrate(consecrate) []
F:1/hallow(hallow) []
F:1/world(world) [17954 5957883]
 ->F:1/dead(dead) []
 ->F:1/brave(brave) []
 ->F:1/grind(ground) []
F:1/power(power) [3714294 4050476 1754717 1756614
 3930426 3930785 3931015 3931458 4042313 4042432
 4042603 4042993 4043761 4043876 4043948 4045365
 4046299 4046475 4046609 4046763 4046959 4047189]
 ->F:1/large(larger) []
F:1/proper(proper) []
F:1/altogether(altogether) [10371715]
F:1/resting-place(resting-place) []
F:1/fit(fitting) [3856995 3858670]
 ->F:1/give(gave) []
F:3/live(live) []
F:1/come(come) []
F:1/great(great) []
F:1/field(field) [6384327 3638970 6282882 6318813
 6324781 6335561 6346705]
 ->F:1/battlefield(battlefield) []
F:1/meet(met) [5543177 278242 5551445 5551573
 5551776 5551854 5552012]
F:1/fight(fought) [614914 79453 615322 759289 759418
 759560 759639 759769 760027 760227 760925 761897
 762001 762099 762195 762286 762395]
 ->F:1/struggle(struggled) []
 ->F:1/war(war) []
 ->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []
F:1/ago(ago) []

F:1/bring(brought) []
F:1/final(final) [5554348 5549662 5550428]
 ->F:1/test(testing) []
 ->F:1/continent(continent) []
F:2/conceive(conceived) []
F:5/dedicate(dedicated) []
F:1/advance(advanced) [5355937]
 ->F:1/sense(sense) []
 ->F:1/portion(portion) []
 ->F:1/proposition(proposition) []
F:1/create(created) []
F:1/father(fathers) [7075378 7461763]
 ->F:4/nation(nation) []
 ->F:2/man(men) []
 ->F:1/equal(equal) []
F:1/life(lives) [10843624 10866987 10867147 10867885
10868236 10868596 10869347]
 ->F:2/long(long) []
 ->F:1/seven(seven) []
 ->F:1/liberty(liberty) []
 ->F:1/year(years) []
 ->F:1/fourscore(Fourscore) []

Now the eleventh sentence is processed.

COMMON: It
COMMON: is
COMMON: rather
BONUS: for
COMMON: us
COMMON: to
COMMON: be
COMMON: here
LWORD: dedicated
COMMON: to
COMMON: the
LWORD: great
LWORD: task
LWORD: remaining
COMMON: before

COMMON: us
BONUS: that
COMMON: from
BONUS: these
LWORD: honored
LWORD: dead
COMMON: we
BONUS: take
LWORD: increased
LWORD: devotion
COMMON: to
BONUS: that
BONUS: cause
BONUS: for
COMMON: which
COMMON: they
LWORD: gave
COMMON: the
COMMON: last
LWORD: full
LWORD: measure
COMMON: of
LWORD: devotion
BONUS: that
COMMON: we
COMMON: here
LWORD: highly
LWORD: resolve
BONUS: that
BONUS: these
LWORD: dead
COMMON: shall
COMMON: not
COMMON: have
LWORD: died
COMMON: in
LWORD: vain
BONUS: that
BONUS: this
LWORD: nation
COMMON: under
LWORD: God
COMMON: shall

COMMON: have
COMMON: a
COMMON: new
LWORD: birth
COMMON: of
LWORD: freedom
COMMON: and
BONUS: that
LWORD: government
COMMON: of
COMMON: the
LWORD: people
COMMON: by
COMMON: the
LWORD: people
COMMON: and
BONUS: for
COMMON: the
LWORD: people
COMMON: shall
COMMON: not
LWORD: perish
COMMON: from
COMMON: the
LWORD: earth

BONUS words and the one UWORD are marked and the LWORDS are sent for chaining. Empty chains are created for “dedicated,” “remaining,” “full,” “highly,” “died,” “vain,” and “perish.” “measure” is added to the chain with “resolve.” All the instances of “people” are added to the chain with “measure.” “great” is added by word repetition. “task” is added to the chain with “devotion.” “honored” is added to the chain with “advanced.” All instances of “dead” are added to the chain with “measure.” “increased” is added to the chain with “honored.” All instances of “devotion” are added to the chain with “task.” “gave” is added to the chain with

“fitting.” “resolve” is added to the chain with “measure.” “nation” is added by word repetition. “birth” is added to the chain with “lives.” “freedom” is added to the chain with “say.” “government” is added to the chain with “measure.” “earth” is added to the chain with “measure.”

CHAIN STACK DUMP

```

F:1/remain(remaining) []
F:1/full(full) []
F:1/highly(highly) []
F:1/die(died) [2530714 2691709 2718771 3342242]
F:1/vain(vain) []
F:1/perish(perish) []
F:1/unfinished(unfinished) []
F:1/nobly(nobly) []
F:1/forget(forget) []
F:1/remember(remember) []
F:1/freedom(freedom) [16185 1007914 1011333 10080250
10080443 10081549 10081721 10081855 10082961 10083322]
->F:1/say(say) []
F:1/poor(poor) []
F:1/add(add) []
F:1/detract(detract) []
F:2/consecrate(consecrate) []
F:1/hallow(hallow) []
F:1/measure(measure) [3971772 2185491 2186718 3933851
3972791 3973015 3973089 3973414 3974960 3978095 3996081]
->F:1/resolve(resolve) []
->F:1/god(God) []
->F:1/government(government) []
->F:3/people(people) []
->F:1/earth(earth) []
->F:1/world(world) []
->F:3/dead(dead) []
->F:1/brave(brave) []
->F:1/grind(ground) []
F:1/power(power) [3714294 4050476 1754717 1756614 3930426
3930785 3931015 3931458 4042313 4042432 4042603 4042993

```


4043761 4043876 4043948 4045365 4046299 4046475 4046609
4046763 4046959 4047189]
->F:1/large(larger) []
F:1/proper(proper) []
F:1/altogether(altogether) [10371715]
F:1/resting-place(resting-place) []
F:1/fit(fitting) [3856995 3858670]
->F:2/give(gave) []
F:3/live(live) []
F:1/come(come) []
F:2/great(great) []
F:1/field(field) [6384327 3638970 6282882 6318813
6324781 6335561 6346705]
->F:1/battlefield(battlefield) []
F:1/meet(met) [5543177 278242 5551445 5551573 5551776
5551854 5552012]
F:1/task(task) [377835 408767 502692 503108 509169 509332
509451 509558 509663]
->F:1/devotion(devotion) []
->F:1/devotion(devotion) []
->F:1/fight(fought) []
->F:1/struggle(struggled) []
->F:1/war(war) []
->F:1/civil-war(civil-war) []
F:1/endure(endure) []
F:1/engage(engaged) []
F:1/ago(ago) []
F:1/bring(brought) []
F:1/final(final) [5554348 5549662 5550428]
->F:1/test(testing) []
->F:1/continent(continent) []
F:2/conceive(conceived) []
F:6/dedicate(dedicated) []
F:1/honor(honored) [5013089 5097156 5020842 5028973 5029066
5029189 5029345 5029514 5029668 5032036 5425157 5425347]
->F:1/increase(increased) []
->F:1/advance(advanced) []
->F:1/sense(sense) []
->F:1/portion(portion) []
->F:1/proposition(proposition) []
F:1/create(created) []
F:1/father(fathers) [7075378 7461763]
->F:5/nation(nation) []

```
->F:2/man(men) []
->F:1/equal(equal) []
F:1/birth(birth) [10965545 10865688 10867885 10867398]
->F:1/life(lives) []
->F:2/long(long) []
->F:1/seven(seven) []
->F:1/liberty(liberty) []
->F:1/year(years) []
->F:1/fourscore(Fourscore) []
```

Normally at this point the sentences in the last paragraph would be marked to receive a lastParaScore, but since there is only one paragraph this is irrelevant. This score is zero and will be ignored. Likewise the globalTitle and paraTitleScores are always zero here and will be ignored.

Now that all the marking and chaining are done, the sentence scoring procedure is called to total the raw parameter scores first.

Every word has 11 separate scores mentioned at the beginning of the example, but we are ignoring lastPara, globalTitle, and paraTitle, so there are 8 separate scores. If we dump these remaining 8 integers for each word in the first sentence we would get the following totals. Remember COMMON words always have zero in every score.

Only the first node in the sentence carries the positional scores (e.g. first-SentScore).

Fourscore	MSRchainScore: 0
XSRchainScore: 0	firstSentScore: 0
SRchainScore: 0	lastSentScore: 0
MSRchainScore: 78	bonusScore: 0
firstSentScore: 1	uwordScore: 0
lastSentScore: 0	stigmaScore: 0
bonusScore: 0	
uwordScore: 1	our
stigmaScore: 0	XSRchainScore: 0
	SRchainScore: 0
and	MSRchainScore: 0
XSRchainScore: 0	firstSentScore: 0
SRchainScore: 0	lastSentScore: 0
MSRchainScore: 0	bonusScore: 1
firstSentScore: 0	uwordScore: 0
lastSentScore: 0	stigmaScore: 0
bonusScore: 0	
uwordScore: 0	fathers
stigmaScore: 0	XSRchainScore: 0
	SRchainScore: 0
seven	MSRchainScore: 71
XSRchainScore: 0	firstSentScore: 0
SRchainScore: 0	lastSentScore: 0
MSRchainScore: 78	bonusScore: 0
firstSentScore: 0	uwordScore: 0
lastSentScore: 0	stigmaScore: 0
bonusScore: 0	
uwordScore: 0	brought
stigmaScore: 0	XSRchainScore: 0
	SRchainScore: 0
years	MSRchainScore: 0
XSRchainScore: 0	firstSentScore: 0
SRchainScore: 0	lastSentScore: 0
MSRchainScore: 75	bonusScore: 0
firstSentScore: 0	uwordScore: 0
lastSentScore: 0	stigmaScore: 0
bonusScore: 0	
uwordScore: 0	forth
stigmaScore: 0	XSRchainScore: 0
	SRchainScore: 0
ago	MSRchainScore: 0
XSRchainScore: 0	firstSentScore: 0
SRchainScore: 0	lastSentScore: 0

bonusScore:	0		
uwordScore:	0	new	
stigmaScore:	0	XSRchainScore:	0
		SRchainScore:	0
upon		MSRchainScore:	0
XSRchainScore:	0	firstSentScore:	0
SRchainScore:	0	lastSentScore:	0
MSRchainScore:	0	bonusScore:	0
firstSentScore:	0	uwordScore:	0
lastSentScore:	0	stigmaScore:	0
bonusScore:	0		
uwordScore:	0	nation	
stigmaScore:	0	XSRchainScore:	500
		SRchainScore:	0
this		MSRchainScore:	0
XSRchainScore:	0	firstSentScore:	0
SRchainScore:	0	lastSentScore:	0
MSRchainScore:	0	bonusScore:	0
firstSentScore:	0	uwordScore:	0
lastSentScore:	0	stigmaScore:	0
bonusScore:	1		
uwordScore:	0	conceived	
stigmaScore:	0	XSRchainScore:	200
		SRchainScore:	0
continent		MSRchainScore:	0
XSRchainScore:	0	firstSentScore:	0
SRchainScore:	0	lastSentScore:	0
MSRchainScore:	71	bonusScore:	0
firstSentScore:	0	uwordScore:	0
lastSentScore:	0	stigmaScore:	0
bonusScore:	0		
uwordScore:	0	in	
stigmaScore:	0	XSRchainScore:	0
		SRchainScore:	0
a		MSRchainScore:	0
XSRchainScore:	0	firstSentScore:	0
SRchainScore:	0	lastSentScore:	0
MSRchainScore:	0	bonusScore:	0
firstSentScore:	0	uwordScore:	0
lastSentScore:	0	stigmaScore:	0
bonusScore:	0		
uwordScore:	0	liberty	
stigmaScore:	0	XSRchainScore:	0

	SRchainScore: 0	lastSentScore: 0
	MSRchainScore: 77	bonusScore: 0
	firstSentScore: 0	uwordScore: 0
	lastSentScore: 0	stigmaScore: 0
	bonusScore: 0	
	uwordScore: 0	proposition
	stigmaScore: 0	XSRchainScore: 0
		SRchainScore: 0
and		MSRchainScore: 78
	XSRchainScore: 0	firstSentScore: 0
	SRchainScore: 0	lastSentScore: 0
	MSRchainScore: 0	bonusScore: 0
	firstSentScore: 0	uwordScore: 0
	lastSentScore: 0	stigmaScore: 0
	bonusScore: 0	
	uwordScore: 0	that
	stigmaScore: 0	XSRchainScore: 0
		SRchainScore: 0
dedicated		MSRchainScore: 0
	XSRchainScore: 600	firstSentScore: 0
	SRchainScore: 0	lastSentScore: 0
	MSRchainScore: 0	bonusScore: 1
	firstSentScore: 0	uwordScore: 0
	lastSentScore: 0	stigmaScore: 0
	bonusScore: 0	
	uwordScore: 0	all
	stigmaScore: 0	XSRchainScore: 0
		SRchainScore: 0
to		MSRchainScore: 0
	XSRchainScore: 0	firstSentScore: 0
	SRchainScore: 0	lastSentScore: 0
	MSRchainScore: 0	bonusScore: 0
	firstSentScore: 0	uwordScore: 0
	lastSentScore: 0	stigmaScore: 0
	bonusScore: 0	
	uwordScore: 0	men
	stigmaScore: 0	XSRchainScore: 200
		SRchainScore: 0
the		MSRchainScore: 0
	XSRchainScore: 0	firstSentScore: 0
	SRchainScore: 0	lastSentScore: 0
	MSRchainScore: 0	bonusScore: 0
	firstSentScore: 0	uwordScore: 0

stigmaScore: 0

are

XSRchainScore: 0

SRchainScore: 0

MSRchainScore: 0

firstSentScore: 0

lastSentScore: 0

bonusScore: 0

uwordScore: 0

stigmaScore: 0

created

XSRchainScore: 0

SRchainScore: 0

MSRchainScore: 0

firstSentScore: 0

lastSentScore: 0

bonusScore: 0

uwordScore: 0

stigmaScore: 0

equal

XSRchainScore: 0

SRchainScore: 0

MSRchainScore: 79

firstSentScore: 0

lastSentScore: 0

bonusScore: 0

uwordScore: 0

stigmaScore: 0

When we add up all the individual word totals, we get the following totals for the first sentence.

```
XSRchainScore: 1500.000000
SRchainScore:  0.000000
MSRchainScore: 607.000000
firstSentScore: 1.000000
lastSentScore: 0.000000
bonusScore:    3.000000
uwordScore:    1.000000
stigmaScore:   0.000000
```

firstSentScore is only kept in the first node for the entire sentence. The totals for all of the sentences are shown below.

Fourscore and seven years ago our fathers brought forth upon this continent a new nation conceived in liberty and dedicated to the proposition that all men are created equal.

```
XSRchainScore: 1500.000000
SRchainScore:  0.000000
MSRchainScore: 607.000000
firstSentScore: 1.000000
lastSentScore:  0.000000
bonusScore:    3.000000
uwordScore:    1.000000
stigmaScore:   0.000000
```

Now we are engaged in a great civil war testing whether that nation or any nation so conceived and so dedicated can long endure.

```
XSRchainScore: 600.000000
SRchainScore:  180.000000
MSRchainScore: 0.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
```

bonusScore: 3.000000
uwordScore: 0.000000
stigmaScore: 0.000000

We are met on a great battlefield of that war.

XSRchainScore: 200.000000
SRchainScore: 180.000000
MSRchainScore: 0.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 1.000000
uwordScore: 0.000000
stigmaScore: 0.000000

We have come to dedicate a portion of that field as a final resting-place for those who here gave their lives that that nation might live.

XSRchainScore: 700.000000
SRchainScore: 270.000000
MSRchainScore: 76.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 5.000000
uwordScore: 0.000000
stigmaScore: 0.000000

It is altogether fitting and proper that we should do this.

XSRchainScore: 0.000000
SRchainScore: 0.000000
MSRchainScore: 77.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 2.000000
uwordScore: 0.000000
stigmaScore: 0.000000

But in a larger sense we cannot dedicate we cannot consecrate we cannot hallow this ground.

XSRchainScore: 300.000000

SRchainScore: 0.000000
MSRchainScore: 231.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 1.000000
uwordScore: 0.000000
stigmaScore: 1.000000

The brave men living and dead who struggled here have
consecrated it far above our poor power₂ to add or detract.

XSRchainScore: 600.000000
SRchainScore: 90.000000
MSRchainScore: 153.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 1.000000
uwordScore: 0.000000
stigmaScore: 0.000000

The world will little note nor long remember what we say here.

XSRchainScore: 100.000000
SRchainScore: 90.000000
MSRchainScore: 77.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 1.000000
uwordScore: 0.000000
stigmaScore: 0.000000

But it can never forget what they did here.

XSRchainScore: 0.000000
SRchainScore: 0.000000
MSRchainScore: 0.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.000000
uwordScore: 0.000000
stigmaScore: 1.000000

It is for us the living rather to be dedicated here to the

unfinished work which they who fought here have thus far so nobly advanced.

XSRchainScore: 200.000000
SRchainScore: 180.000000
MSRchainScore: 0.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 4.000000
uwordScore: 0.000000
stigmaScore: 0.000000

It is rather for us to be here dedicated to the great task remaining before us that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion that we here highly resolve that these dead shall not have died in vain that this nation under God shall have a new birth of freedom and that government of the people by the people and for the people shall not perish from the earth.

XSRchainScore: 600.000000
SRchainScore: 270.000000
MSRchainScore: 917.000000
firstSentScore: 0.000000
lastSentScore: 1.000000
bonusScore: 14.000000
uwordScore: 1.000000
stigmaScore: 0.000000

Once all the individual sentence parameter scores are totaled, the max values of each are used to normalize scores so that they range from 0.0 to 1.0. It is these normalized scores that are multiplied by their appropriate scaling factors to get the final score used for sentence selection. The sentences with their normalized scores and totals are list below.

Fourscore and seven years ago our fathers brought forth upon

this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal.

total score: 832.864202

XSRchainScore: 0.900000
SRchainScore: 0.000000
MSRchainScore: 1.000000
firstSentScore: 1.000000
lastSentScore: 0.000000
bonusScore: 0.214286
uwordScore: 1.000000
stigmaScore: 0.000000

Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure.

total score: 629.237381

XSRchainScore: 0.900000
SRchainScore: 0.333333
MSRchainScore: 0.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.214286
uwordScore: 0.000000
stigmaScore: 0.000000

We are met on a great battlefield of that war.

total score: 119.620326

XSRchainScore: 0.100000
SRchainScore: 0.666667
MSRchainScore: 0.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.071429
uwordScore: 0.000000
stigmaScore: 0.000000

We have come to dedicate a portion of that field as a final resting-place for those who here gave their lives that that nation might live.

total score: 537.773925

XSRchainScore: 0.600000
SRchainScore: 1.000000
MSRchainScore: 0.122779
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.357143
uwordScore: 0.000000
stigmaScore: 0.000000

It is altogether fitting and proper that we should do this.

total score: 55.685679

XSRchainScore: 0.000000
SRchainScore: 0.000000
MSRchainScore: 0.124394
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.142857
uwordScore: 0.000000
stigmaScore: 0.000000

But in a larger sense we cannot dedicate, we cannot consecrate, we cannot hallow this ground.

total score: 76.926040

XSRchainScore: 0.000000
SRchainScore: 0.000000
MSRchainScore: 0.373183
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.071429

uwordScore: 0.000000
stigmaScore: 1.000000

The brave men, living and dead, who struggled here, have
consecrated it far above our poor power to add or detract.

total score: 171.051492

XSRchainScore: 0.100000
SRchainScore: 0.333333
MSRchainScore: 0.252019
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.071429
uwordScore: 0.000000
stigmaScore: 0.000000

The world will little note, nor long remember, what we say here.

total score: 87.170794

XSRchainScore: 0.100000
SRchainScore: 0.333333
MSRchainScore: 0.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.000000
uwordScore: 0.000000
stigmaScore: 0.000000

But it can never forget what they did here.

total score: -42.723503

XSRchainScore: 0.000000
SRchainScore: 0.000000
MSRchainScore: 0.000000
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.000000

uwordScore: 0.000000
stigmaScore: 1.000000

It is for us, the living, rather to be dedicated here to the unfinished work which they who fought here have thus far so nobly advanced.

total score: 288.317322

XSRchainScore: 0.300000
SRchainScore: 0.333333
MSRchainScore: 0.117932
firstSentScore: 0.000000
lastSentScore: 0.000000
bonusScore: 0.285714
uwordScore: 0.000000
stigmaScore: 0.000000

It is rather for us to be here dedicated to the great task remaining before us, that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion; that we here highly resolve that these dead shall not have died in vain; that this nation, under God, shall have a new birth of freedom, and that government of the people, by the people, and for the people, shall not perish from the earth.

total score: 1018.196877

XSRchainScore: 1.000000
SRchainScore: 1.000000
MSRchainScore: 0.980614
firstSentScore: 0.000000
lastSentScore: 1.000000
bonusScore: 1.000000
uwordScore: 0.000000
stigmaScore: 0.000000

We now sort the sentences based on total score with the highest score at the top.

It is rather for us to be here dedicated to the great task remaining before us, that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion; that we here highly resolve that these dead shall not have died in vain; that this nation, under God, shall have a new birth of freedom, and that government of the people, by the people, and for the people, shall not perish from the earth.

total score: 1018.196877

Fourscore and seven years ago our fathers brought forth upon this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal.

total score: 832.864202

Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure.

total score: 629.237381

We have come to dedicate a portion of that field as a final resting-place for those who here gave their lives that that nation might live.

total score: 537.773925

It is for us, the living, rather to be dedicated here to the unfinished work which they who fought here have thus far so nobly advanced.

total score: 288.317322

The brave men, living and dead, who struggled here, have consecrated it far above our poor power to add or detract.

total score: 171.051492

We are met on a great battlefield of that war.

total score: 119.620326

The world will little note, nor long remember, what we say here.

total score: 87.170794

But in a larger sense we cannot dedicate, we cannot consecrate, we cannot hallow this ground.

total score: 76.926040

It is altogether fitting and proper that we should do this.

total score: 55.685679

But it can never forget what they did here.

total score: -42.723503

For the testing against Microsoft Word97 AutoSummarize and Sinope I used a summary of three sentences. To get that here, I would take the three top scoring sentences, sort them back into their order in the original document, then print the output. This is the summary shown in Appendix A on page 232.)

Fourscore and seven years ago our fathers brought forth upon this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal.

total score: 832.864202

Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure.

total score: 629.237381

It is rather for us to be here dedicated to the great task remaining before us, that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion; that we here highly resolve that these dead shall not have died in vain; that this nation, under God, shall have a new birth of freedom, and that government of the people, by the people, and for the people, shall not perish from the earth.

total score: 1018.196877

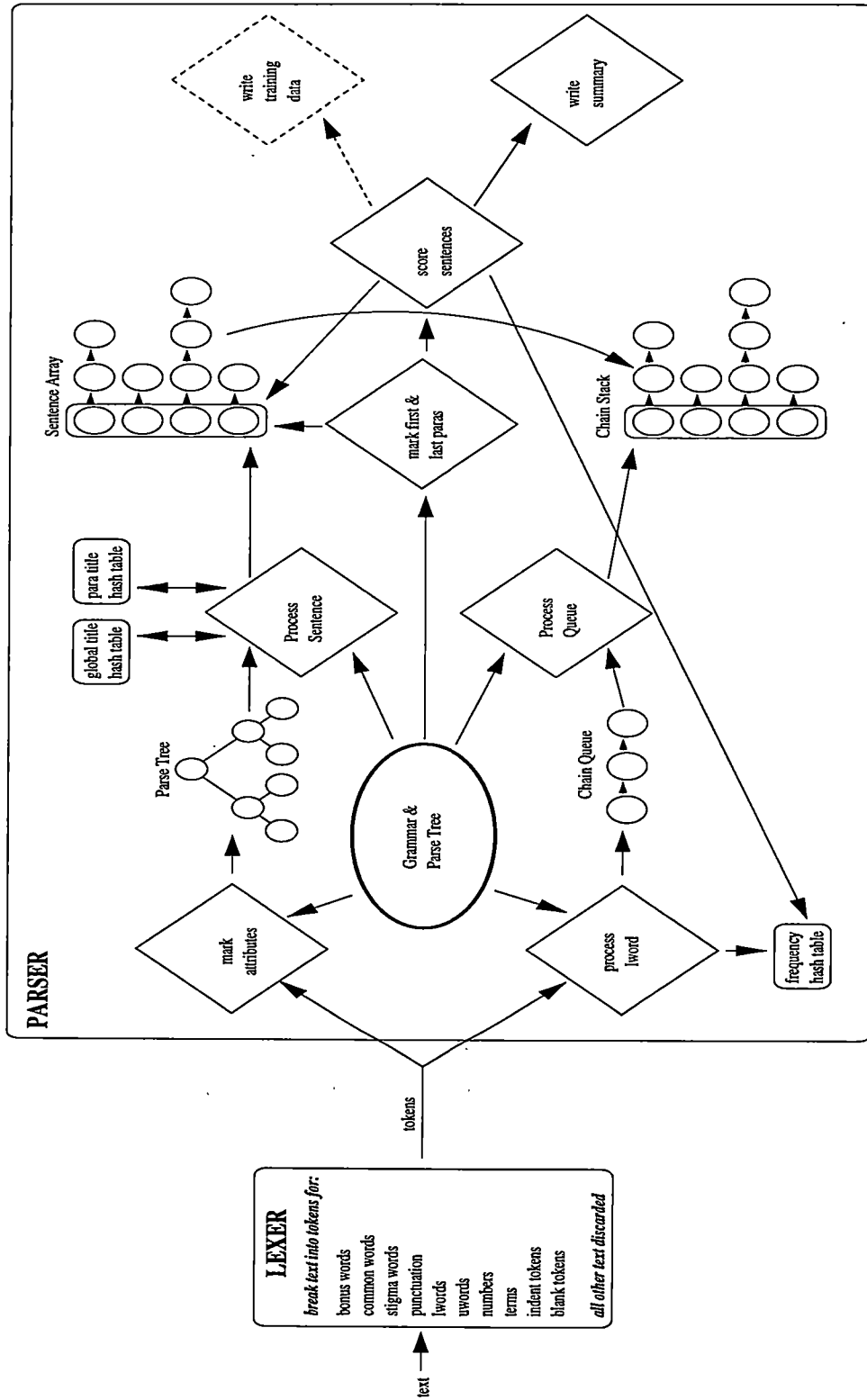


Figure 7.1: Overall Organization of Summarizer.

```
text:
    paragraphList
    | paragraphList blankTokenList
    ;

paragraphList:
    paragraph
    | paragraphList paragraph
    ;

paragraph:
    blankTokenList sentenceList
    ;

sentenceList:
    sentence
    | sentenceList sentence
    ;

sentence:
    wordList period
    | wordList exclamation
    | wordList question
    | wordList
    ;

blankTokenList:
    blankToken
    | indentToken
    | blankTokenList blankToken
    | blankTokenList indentToken
    ;

wordList:
    word
    | wordList word
    ;

word:
    LWORD
    | UWORD
    | BONUS
    | COMMON
    | STIGMA
    | TERM
    | NUMBER
    ;
```

Figure 7.2: Simple Bison grammar used to create parse tree.

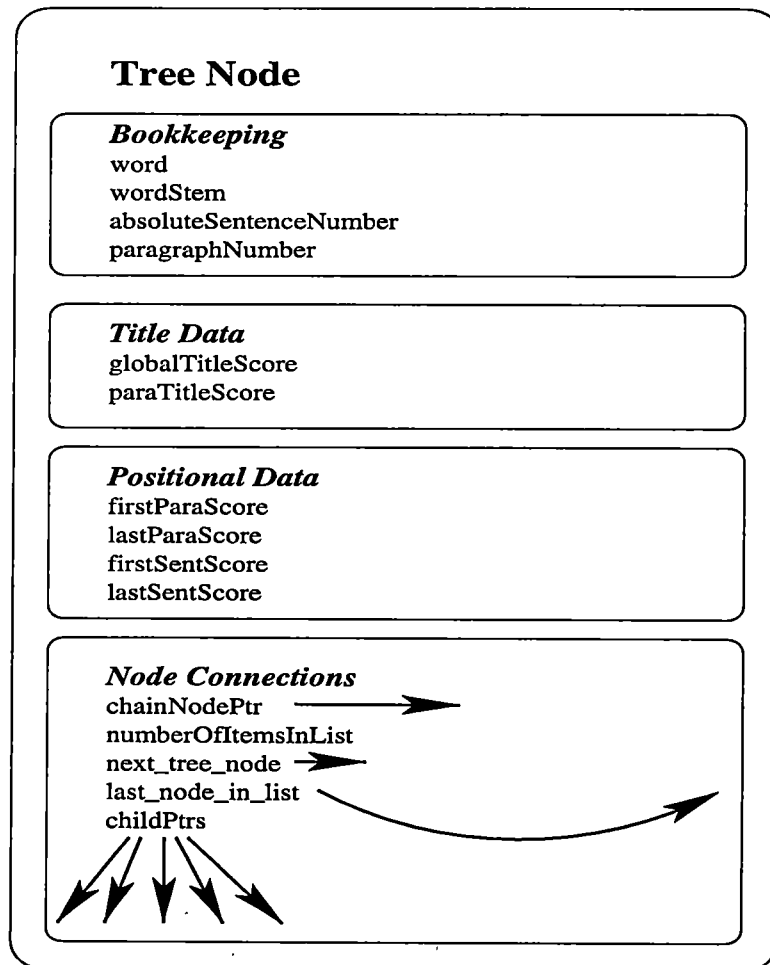


Figure 7.3: Internal Organization of TreeNode.

PROCESS LWORD, UWORD, OR TERM

```
/* store treenode */
store_treenode_in_tree(word);

/* WordNet assumes lower case for words */
word = tolowercase(word);

/* try first to find noun stem */
stem = find_morphological_noun_stem(word);

if(!stem) {

    /* if no noun stem try to find verb stem */
    stem = find_morphological_verb_stem(word);

    if(!stem) {

        /* if no verb stem try to find adjective stem */
        stem = find_morphological_adjective_stem(word);

        if(!stem) {

            /* if no adjective stem try to find adverb stem */
            stem = find_morphological_adverb_stem(word);
        }
    }
}

/* try to form noun compound with previous word in sentence */
if(previous_word) {
    new_stem = find_morphological_noun_stem(word-compound);

    if(new_stem) {
        stem = new_stem;
    }
}

store_chainnode_in_queue(stem);
```

Figure 7.4: Processing LWORD, UWORD, or TERM.

PROCESS CHAIN QUEUE

Restart:

```
tempQueuePtr = queueBase;
while(tempQueuePtr) {

    /* check entire chain structure for extra-strong match */
    found = find_extra_strong_relation(tempQueuePtr);
    if(found) {
        remove_word_from_queue(tempQueuePtr);
        goto Restart;
    }

    /* check plus or minus 7 sentences for strong match */
    found = find_strong_relation(tempQueuePtr);
    if(found) {
        remove_word_from_queue(tempQueuePtr);
        goto Restart;
    }

    /* check plus or minus 4 sentences for medium match */
    found = find_medium_relation(tempQueuePtr);
    if(found) {
        remove_word_from_queue(tempQueuePtr);
        goto Restart;
    }

    /* Finally, if no match can be found a new chain is created */
    create_new_chain(tempQueuePtr);
    remove_word_from_queue(tempQueuePtr);
    goto Restart;
}
```

Figure 7.5: Queue Processing.

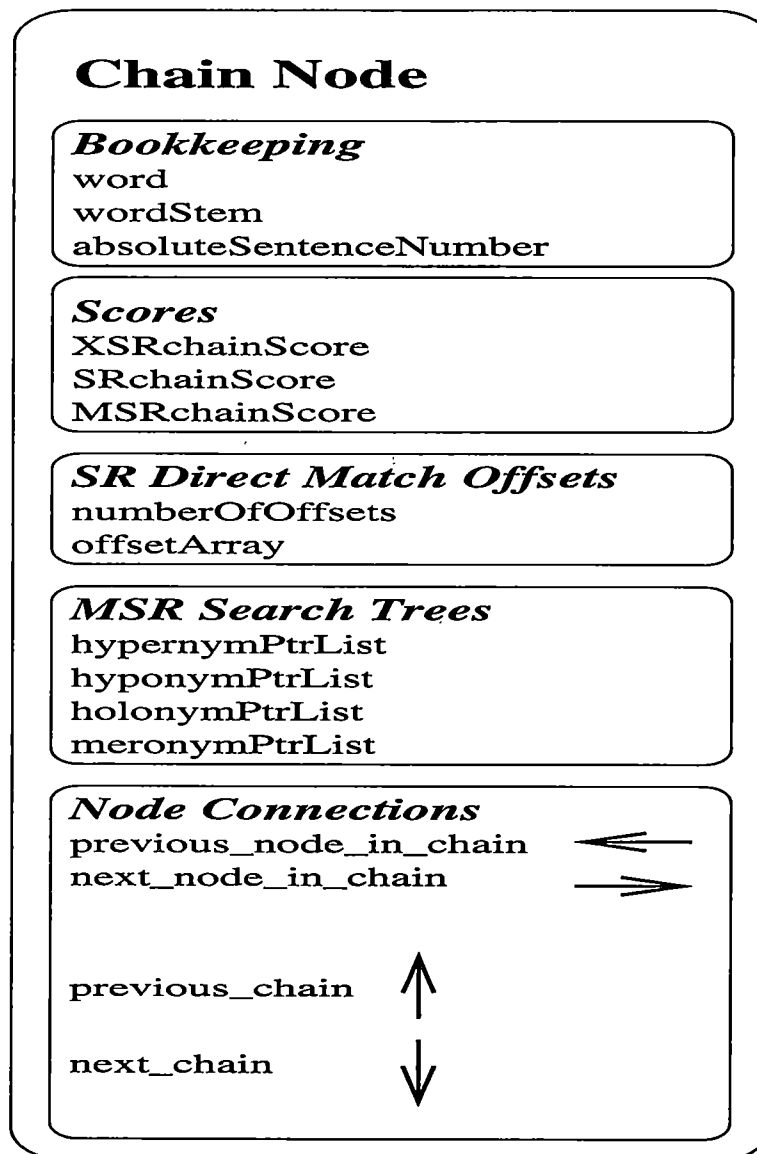


Figure 7.6: Internal Organization of ChainNode.

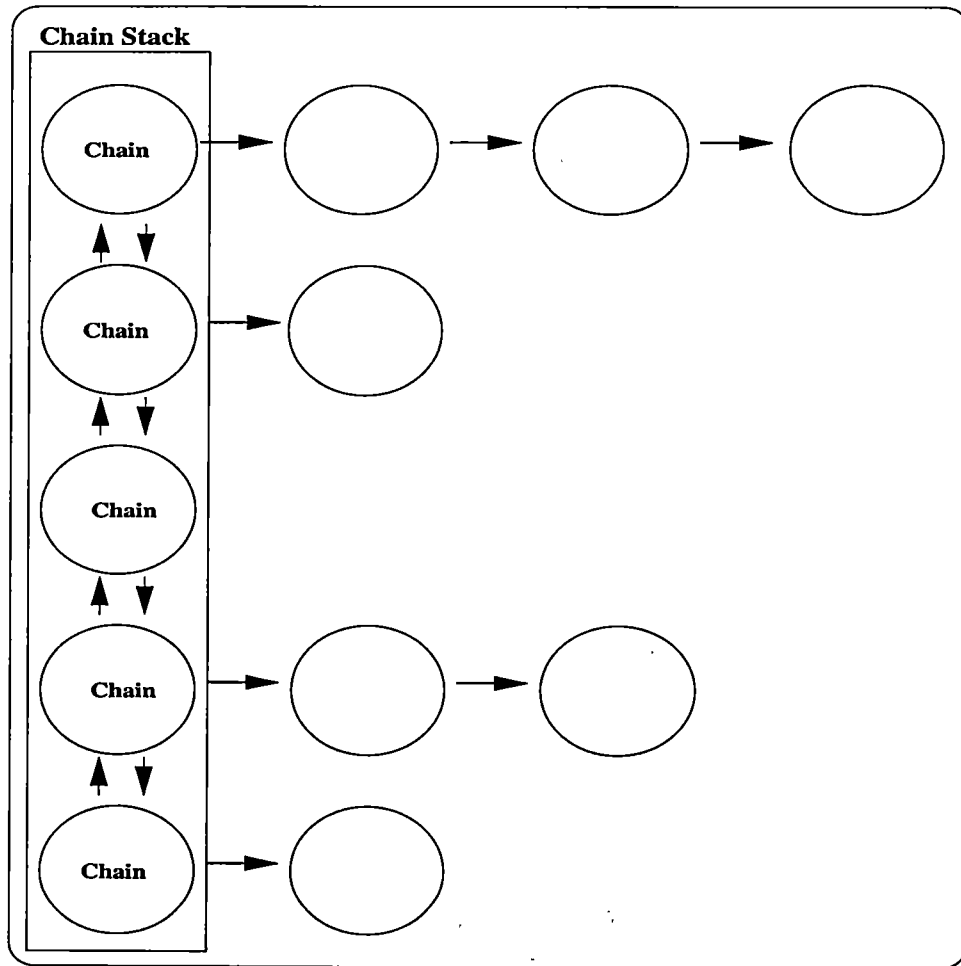


Figure 7.7: Lexical chain stack organization.

Find Strong Relation

```
SRrestart:
/* start at base of queue */
tempQueuePtr = queueBase;
while(tempQueuePtr) {

    /* start at base of chain stack */
    stackPtr = stackBase;
    while(stackPtr) {

        /* only check for words in sentences + or - 7 distance */
        if( abs(tempQueuePtr->absSentNum - stackPtr->absSentNum) < 8 ){

            if(find_immediate_synonym_match())
                insert_node_in_chain();
            purge_first_node();
            set_chain_strength();
            goto SRrestart;
        }

        if(find_immediate_antonym_match())
            insert_node_in_chain();
            purge_first_node();
            set_chain_strength();
            goto SRrestart;
        }

        if(find_immediate_meronym_match())
            insert_node_in_chain();
            purge_first_node();
            set_chain_strength();
            goto SRrestart;
        }

        if(find_immediate_hyponym_match())
            insert_node_in_chain();
            purge_first_node();
            set_chain_strength();
            goto SRrestart;
        }

        if(find_immediate_holonym_match())
            insert_node_in_chain();
            purge_first_node();
            set_chain_strength();
            goto SRrestart;
        }

        if(find_immediate_hyponym_match())
            insert_node_in_chain();
            purge_first_node();
            set_chain_strength();
            goto SRrestart;
        }
    }
}
}
```

Figure 7.8: Find Strong Relation.

Find Medium-Strength Relation

```

MSR_RESTART:

/* start at base of queue */
tempQueuePtr = queueBase;
while(tempQueuePtr) {

    /* start at base of chain stack */
    stackPtr = stackBase;
    while(stackPtr) {

        /* only check for words in sentences + or - 4 distance */
        if( abs(tempQueuePtr->absSentNum - stackPtr->absSentNum) < 5 ){

            /* explore hypernym tree */
            ↑
            if(find_extended_hypernym_match())
                insert_node_in_chain();
                purge_first_node();
                set_chain_strength();
                goto SRrestart;
            }

            /* explore hyponym tree */
            ↓
            if(find_extended_hyponym_match())
                insert_node_in_chain();
                purge_first_node();
                set_chain_strength();
                goto SRrestart;
            }

            /* explore holonym tree */
            ↑
            if(find_extended_holonym_match())
                insert_node_in_chain();
                purge_first_node();
                set_chain_strength();
                goto SRrestart;
            }

            /* explore meronym tree */
            ↓
            if(find_extended_meronym_match())
                insert_node_in_chain();
                purge_first_node();
                set_chain_strength();
                goto SRrestart;
            }
        }
    }
}

```

Figure 7.9: Find Medium Strength Relation.

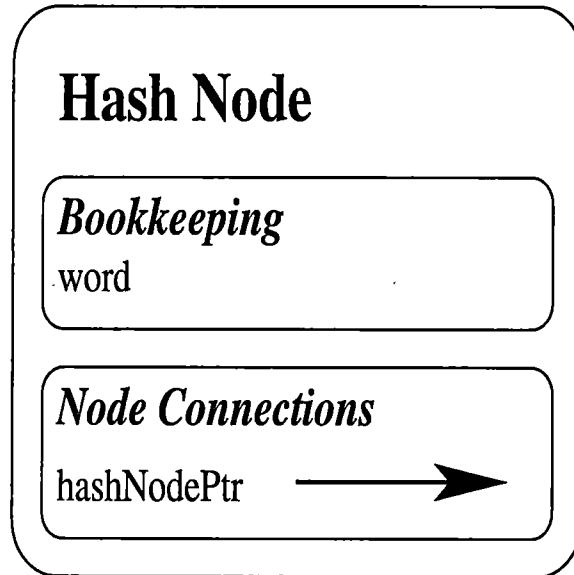


Figure 7.10: Internal Organization of HashNode.



Figure 7.11: Internal Organization of SortNode.

```
0.130435,0.500000,0.358788,0.000000,0.500000,0.000000,1.000000,0.000000,0.000000,0.100000,0.000000,0.0
0.173913,0.000000,0.450909,0.000000,0.000000,0.000000,1.000000,0.000000,0.500000,0.000000,0.000000,0.0
0.391304,0.500000,0.169697,0.000000,0.000000,0.000000,1.000000,0.000000,0.250000,0.500000,0.000000,0.0
0.043478,0.500000,0.444848,0.000000,0.000000,0.000000,0.000000,1.000000,0.000000,0.200000,0.000000,0.0
0.130435,0.500000,0.555152,0.000000,0.000000,0.000000,1.000000,0.000000,0.250000,0.100000,1.000000,0.0
0.086957,0.000000,0.630303,0.200000,0.000000,1.000000,1.000000,0.000000,0.250000,0.600000,0.000000,0.0
1.000000,0.000000,0.168514,0.000000,0.000000,0.000000,1.000000,0.000000,0.000000,0.333333,0.000000,1000.0
0.552632,0.500000,1.000000,0.125000,0.000000,0.000000,1.000000,0.000000,0.750000,0.266667,0.000000,1000.0
0.263158,0.500000,0.343681,0.625000,0.333333,0.000000,1.000000,0.000000,1.000000,0.133333,0.000000,1000.0
0.157895,1.000000,0.674058,0.000000,0.666667,0.000000,0.000000,1.000000,0.500000,0.000000,0.000000,1000.0
0.078947,1.000000,0.172949,0.125000,0.333333,0.000000,1.000000,0.000000,0.250000,0.066667,0.000000,1000.0
0.105263,0.500000,0.501109,0.125000,0.000000,0.000000,1.000000,0.000000,0.000000,0.200000,0.000000,1000.0
0.210526,0.000000,0.849224,0.250000,0.000000,0.000000,1.000000,0.000000,0.000000,0.000000,0.000000,0.0
0.421053,0.500000,0.702882,1.000000,0.000000,0.000000,1.000000,0.000000,0.000000,1.000000,0.000000,0.0
0.052632,0.000000,0.341463,0.250000,0.000000,0.000000,1.000000,0.000000,0.500000,0.000000,0.000000,0.0
0.289474,0.500000,0.332594,0.625000,0.666667,0.000000,1.000000,0.000000,0.000000,0.400000,0.000000,0.0
0.026316,1.000000,0.000000,0.000000,0.000000,0.000000,0.000000,1.000000,0.000000,0.000000,1.000000,0.0
0.000000,0.000000,0.170732,0.000000,0.000000,0.000000,1.000000,0.000000,0.000000,0.000000,0.000000,0.0
```

Figure 7.12: Partial Comma-Delimited Training Data.

```

double interp(double x[])
/*-----*
DataFit version 7.1.44
Date Created: May 24 2001
Time Created: 12:33:26 PM

This function returns a predicted f(x) value
for the function a*x1+b*x2+c*x3+d*x4+e*x5+f*x6+g*x7+h*x8+i*x9+j*x10+k*x11.
The independent variables x1, x2, .. xn are passed in the
array x[], where x[1] = x1, et.
for the function a*x1+b*x2+c*x3+d*x4+e*x5+f*x6+g*x7+h*x8+i*x9+j*x10+k*x11.

Regression Statistics:
R2 = 0.382230081622972
Adjusted R2 = 0.301262762971781
Standard Error = 50.872269732267
*-----*/
{
double a = 642.027611975991;
double b = 68.9040992351196;
double c = 295.212206160026;
double d = -17.1983268493788;
double e = 111.445331265914;
double f = -80.1695970821782;
double g = 74.3741705353783;
double h = -114.96501062744;
double i = 132.740985706689;
double j = -142.991521939385;
double k = -42.7235031520912;
double result;

result = ((((((((((a * x[1]) + (b * x[2])) + (c * x[3])) + (d * x[4]))
+ (e * x[5])) + (f * x[6])) + (g * x[7])) + (h * x[8])) + (i * x[9]))
+ (j * x[10])) + (k * x[11]));
return result;
}

```

Figure 7.13: DataFit Least Squares Solution.

Chapter 8

Conclusions

8.1 Evaluation

There are three kinds of lies:
lies, damn lies, and statistics.

Benjamin Disraeli

(attributed by Mark Twain)

In this chapter we detail how the AutoExtract summarizer was evaluated, what our results were, lessons learned, our contributions, and our recommendations for future research.

Summaries can be evaluated extrinsically, in which case the quality of a summary is judged by how it affects the completion of some other task, or intrinsically, in which case humans judge the quality of the summary directly based on fluency, coverage, or how well it matches a manually-constructed ideal summary. This latter method is the one that most researchers use, and the one that we will use.

Zechner (1995) extracted six test texts from the Daily Telegraph Corpus (British), then used human volunteers to create *ideal* extracts of five to seven sentences each. He then combined the extracts from the volunteers by using the top ranked mean scores of sentences to produce the best overall extracts. These were then used to test his automatic text abstracting system. We shall use these same six texts to test our system. (See Appendix B on page 235 for the six texts.)

We must bear in mind though that there really is no such thing as an *ideal* summary. Marcu (1997c) found that overall, human abstractors only agree on which sentences should be in a summary about 71% of the time. Rath et al. (1961) showed in a study that extracts selected by four different human judges had only a 25% overlap, and for a given judge over time only 55%. Thus, not only don't human summarizers agree with each other, they agree with themselves only about half the time.

For each of the test texts I set my AutoExtract summarizer to produce the same number of output sentences as in the ideal summary, then simply measured the overlap of my summaries with the ideal human ones. Since the number of sentences retrieved equals the number of sentences in the ideal summary, precision and recall are identical and equal my overlap measurement. Complete results are shown in Appendix C on page 250.

I ran the same tests using the commercial text summarizer, Sinope (See Appendix D on page 257.) and Microsoft Word97 AutoSummarize (See Appendix E on page 263.). Unfortunately, I couldn't compare directly with Zechner (1995) be-

cause he didn't align his automatically created summaries with the ideal human ones. That is, recall and precision for each of his automatically created summaries were not identical. For example, the ideal human summary might be six sentences while Zechner's automatic summary might be eight sentences. The mean precision of his program was 0.47 and the mean recall was 0.54, both lower than the 0.60 mean overlap (precision, recall) of my program.

Zechner, however, did provide some interesting statistics on randomly generated summaries:

The random chance of getting (at least) one three-sentence passage when forming an abstract of about 6 or 7 sentences from about 19 sentences – which are the values for the six test articles – is about 38%; getting such passages for at least 5 out of 6 articles (as it is the case in the experiment's results) has a random chance of only 3%.

Comparisons of the results all three programs are shown in Table 8.1 on page 212.

8.2 Discussion of Overlap Test

Some problems are just too complicated for rational, logical solutions.
They admit of insights, not answers.

Jerome Wiesner

The test data (six texts) is, of course, too small to render any definitive judgments. Moreover, all the test texts were drawn from one corpus (using British English). Ideally, we would have a training corpus of at least a thousand randomly

selected texts and a test corpus of at least one hundred randomly selected texts. As the old saying goes, if wishes were horses, beggars would ride. In other words, I had to make do with the corpus that I could gather.

Nevertheless, my AutoExtract program must be doing something right to clearly beat both Microsoft's Word97 AutoSummarize and the Sinope commercial summarizer using the ideal overlap measurement. To be fair, my program doesn't appear to be as fast as the two commercial entries. My unoptimized program took about 30 seconds per text, while the commercial entries *appeared* to take about 5 seconds. I say *appeared* because the Microsoft entry clearly did some processing before the AutoSummarize option was selected. I say this because the program was already displaying statistical information on number of words and number of sentences before the type of summarization was selected. It would be logical to assume that word frequencies had already been computed. The Sinope summarizer emailed me the results fairly rapidly, but I can't be sure how long the processing took. The work of some researchers, who were not trying to achieve fast results, sometimes took hours (Barzilay & Elhadad 1997).

I originally tried to compile data on 12 *independent* variables (or parameters) for my summarizer. That they are independent is a convenient fiction that allows us to create a first-order approximation to a linear salience function. The variable first-ParaScore (meaning that the sentence is in the first paragraph) had to be dropped, because there was no way to reliably and automatically determine the first paragraph. Yes, a human can very easily find the first paragraph of a text, but I wanted

my summarizer to do this automatically without human intervention. One problem is that paragraphs are sometimes only one sentence. Another problem is that titles sometimes end with periods. What do you do if there are multiple titles or bylines? Finally, the length cutoff feature that many researchers use for discarding titles is just not effective for all types of text, as it misses some long titles and discards some important short sentences.

After discarding firstParaScore, I was left with the eleven independent variables listed below.

I compiled all the variable values for all the sentences for all the training examples into one comma-delimited ASCII data file, then fed the data into the DataFit program from Oakdale Engineering. DataFit found three least squares error solutions, one non-linear using exponentials, one linear with an added constant, and one linear that only involved constant multipliers for the parameters. I wanted to focus on the parameters, I didn't want the overhead of computing exponentials, and the error for all three was about the same. I, therefore, chose the linear function that only involved constant multipliers for the parameters. That function is shown in Figure 7.13 on page 199. The constants that DataFit generated are shown below, next to the parameter names. (Other researchers have also found that a simple linear summation gives the best results (Zechner 1995)).¹

¹R², the coefficient of multiple determination, measures the proportion of variation in the data points that is explained by the regression model. An R² of 0.80 would mean that 80% of the variation in the output is explained by the regression model. An R² of 1.0 would mean that the curve passes through every data point. Conversely, an R² of 0.0 would mean that the regression model doesn't describe the data any better than a horizontal line passing through the average of the data points.

The DataFit program from Oakdale Engineering is designed to find the function that gives the best fit for a data set, where best fit is defined as minimizing least squares error. It has both linear and non-linear functions in its database and you can define your own function if you wish. You can direct it to try out all the possible solutions in its repertoire, then give you the ones that work. That is what I did.

Based on the training data, the DataFit program returned the following constants for a linear least squares error solution.

XSRchainScore	642.027611975991
MSRchainScore	295.212206160026
uwordScore	-142.991521939385
bonusScore	132.740985706689
lastSentScore	-114.96501062744
paraTitleScore	111.445331265914
lastParaScore	-80.1695970821782
firstSentScore	74.3741705353783
SRchainScore	68.9040992351196
stigmaScore	-42.7235031520912
globalTitleScore	-17.1983268493788

R_{a2} , the adjusted coefficient of multiple determination, is used to balance the cost of using a model with more parameters against the increase in R^2 .

Standard error, or standard deviation, is a measure of the dispersal of the probability distribution about the mean; the smaller the better.

An examination of the constants generated by the DataFit program (As shown above and in Figure 7.13 on page 199.) shows some very interesting results, with the size of the constants clearly indicating the importance of the parameters.

8.2.1 XSRchainScore

The largest constant was assigned to XSRchainScore (extra-strong chain score). As noted earlier, the XSRchainScore is really just another name for Luhn's thematic term frequency, first invented in 1958. That is, this score reflects the word frequency of uncommon words. The numbers show that this parameter is far more important than any other parameter in automatic text summarization. This would seem to validate Microsoft's choice of this parameter as the primary component in its Word97 AutoSummarize tool.

8.2.2 MSRchainScore

Surprisingly, the parameter second in importance was not the SRchainScore (strong chain score). Most researchers using lexical chains give the second highest weight in chain score computation to this parameter. This score reflects direct synonym, antonym, and meronym connections between words.

Our results, however, show that the MSRchainScore (medium strength chain score), which reflects longer chains of hypernyms-hyponyms and holonyms-meronyms, is the second most important factor in selecting sentences, and far more important than the SRchainScore.

8.2.3 uwordScore

In the past, researchers have used the presence of upper-case words as a weight for selecting sentences for extraction. In our solution, however, the uwordScore seemed to be very important for not selecting sentences. This could easily be due to the type of material in our training corpus. Nevertheless, the use of this parameter needs to be examined more. We might want to discard uwordScore altogether. Kupiec et al. (1995) did find a very low correlation between upper case word frequency and sentences selected for extraction.

8.2.4 bonusScore

The size of the bonusScore (bonus words, cue words) multiplier reinforces the work of earlier researchers stressing the importance of this parameter. Examining all of the constants, however, indicates that the XSRchainScore (thematic word score) is far more important for selecting sentences. (At least with the cue-word set and training corpus that we used.)

8.2.5 lastSentScore

The positional information encoded in lastSentScore (bonus for being the last sentence in a paragraph) did not function the way we had hoped either. Like the uwordScore, it actually functioned as a negative selector for extraction. As noted by earlier researchers, using positional information is a gamble, where you may win big or lose big.

8.2.6 paraTitleScore

The paraTitleScore (bonus for having a word in the sentence that is also in the paragraph or header title) worked quite well as a sentence selector parameter. Curiously, the globalTitleScore, like lastSentScore and uwordScore, functioned as a negative selector.

8.2.7 lastParaScore

The lastParaScore parameter didn't function any better than the lastSentScore parameter. This is more reinforcement for being wary of positional information.

8.2.8 firstSentScore

This is the one positional parameter that did perform as expected, serving as a very positive parameter for sentence selection. Curiously, as we noted earlier, Microsoft appears to rely on this parameter (in addition to word frequency) in its Word97 AutoSummarize tool.

8.2.9 SRchainScore

The SRchainScore (strong chain score - immediate synonyms, antonyms etc.) served as a good positive selector, but was not nearly as important as we were led to believe by other research on lexical chains. Instead, as noted above, the MSRchainScore was more than four times more important as a selector.

8.2.10 stigmaScore

The stigmaScore parameter performed as expected as a negative sentence selector. The difference in the size of the bonusScore and stigmaScore multipliers confirms that we were right to evaluate these parameters separately. The natural extension of this idea would be to have a pseudo-continuum of bonus/stigma values. That is we could have a hundred different bonus/stigma parameters/groups each with its own constant. This idea is explored more in the section on future research.

8.2.11 globalTitleScore

The globalTitleScore parameter didn't perform as theory and past research would have indicated. In fact, it served as a negative selector. This could be due to the type of training data that we used. In any case, the use of this parameter needs further study.

8.3 Extract Comparison Using Human Volunteers

When all is said and done, when we have finished with our overlap, precision, and recall metrics, only one thing really matters. The summaries produced must be usable by humans, and, therefore, must be validated by humans. Human judgement, of course, has its own biases, defects, needs, and preferences. This is the reason that I used as large a group of humans as possible for my testing and validation, to arrive at an *average* human judgement.

Moreover, we are seeking the *best* subset of sentences for our summary, and it is possible, because of properties of the subset as a whole, that the best subset might not be composed of the *best* individual sentences. I, therefore, decided to do something that no other researcher to my knowledge has tried. I used thirteen human volunteers to read and judge each computer-generated summary as a complete document.

For this test, I created a test composed of Zechner's six articles, with each followed by four different summaries in random order. The three machine extracts created above plus Zechner's ideal extracts. Since all of Zechner's texts are from a British Corpus, I decided to include something uniquely American as a seventh test, Lincoln's *Gettysburg Address*. I also chose this document for the seventh test, because this is the document that Microsoft chose to show off its AutoSummarize tool (Gore 1997). The Gettysburg Address and four different extracts are in Appendix A on page 232. The original texts and ideal human summaries are contained in Appendix B on page 235. All of the AutoExtract summaries of Zechner's articles are in Appendix C on page 250. All of the Sinope summaries are in Appendix D on page 257. All of the Microsoft summaries are in Appendix E on page 263.

All the extracts were formatted the same, and all identifying marks and notations were removed. No one had any idea where any of the extracts came from. I deliberately did not tell the volunteers what to look for. I told them to use their own judgement. The instructions at the beginning of the test were:

There are seven reading selections (about 20 sentences each) to read. Following each selection are four different extracts in no particular order.

I'm interested in your opinion (based on whatever criteria you may choose to use) as to how good each extract is as a summary.

- Please read each of the selections.
- Read each of the four summaries that follow each selection.
- Rank the summaries from 1 to 4. That is, mark the best summary with a 1, the second best with a 2, the third best with a 3, and finally the worst summary with a 4.

The results of the test are shown in Table 8.2 on the following page and in Figure 8.1 on the next page.

The results of the Gettysburg test are shown in Table 8.3 on page 213 and in Figure 8.2 on page 213.

Some volunteers said they looked for inclusion of topic sentences in the summaries. Others just considered general overall impressions of the summaries. The idea was that with enough volunteers we would get a feel for what the average reasonable human would expect of a summary.

Several volunteers remarked on the problem of anaphoric resolution. That is, all the programs produced output where pronouns appeared to refer to the wrong person. This is one of the harder problems to resolve in automatic text summarization, and no program really does it well.

The results of the human extract comparison test clearly reinforce the results of the ideal extract overlap test used above. The order of the test results was the same for the automatic summarization programs. What was surprising, was that

Table 8.1: Comparison of text summarization programs.

Text	1	2	3	4	5	6	average
AutoExtract	50%	83%	71%	50%	57%	50%	60%
Microsoft AutoSummarize	50%	33%	43%	33%	57%	67%	47%
Sinope	50%	33%	57%	0%	43%	50%	39%

Table 8.2: Human Extract Comparison For Zechner Texts.

	average	standard deviation
AutoExtract	1.962	0.932
Human Ideal	1.987	0.781
Microsoft AutoSummarize	2.487	1.041
Sinope	3.551	0.907

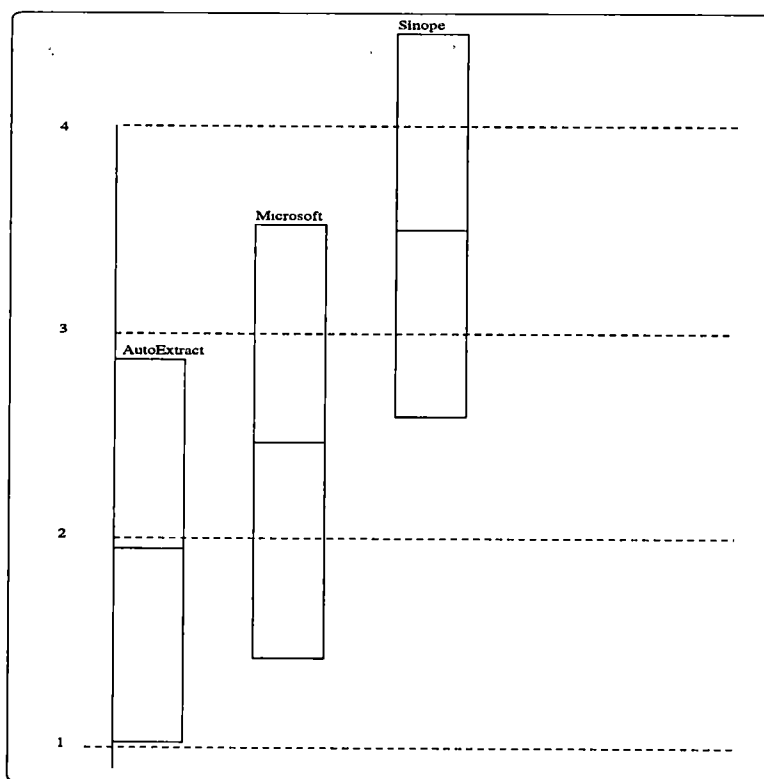


Figure 8.1: Std Dev Graph of Human Extract Comparison For Zechner Texts.

Table 8.3: Human Extract Comparison For Gettysburg Address.

	average	standard deviation
AutoExtract	1.308	0.630
Sinope 32%	2.615	0.870
Sinope 3 sentences	2.615	1.044
Microsoft AutoSummarize	3.462	0.776

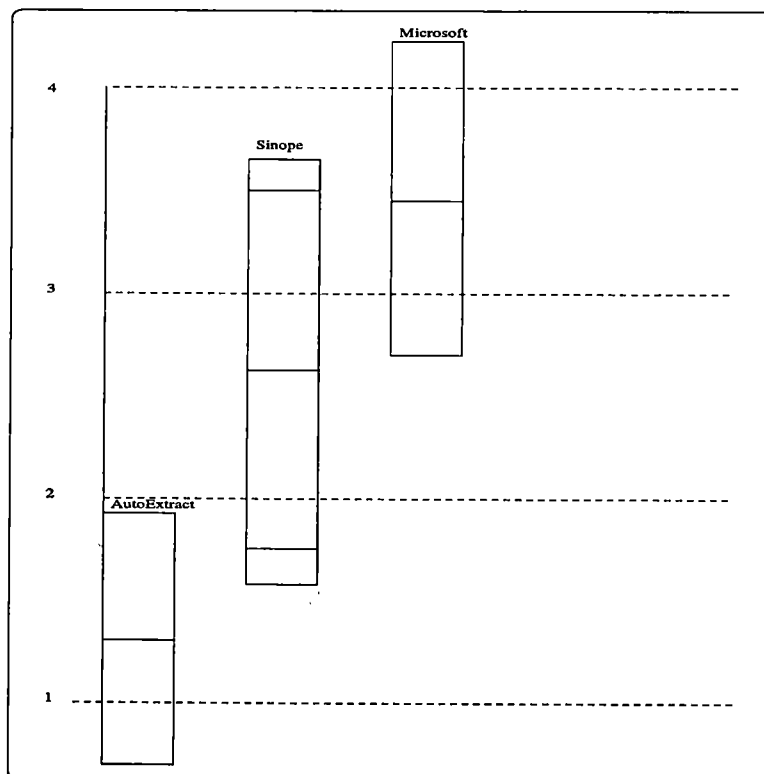


Figure 8.2: Std Dev Graph of Human Extract Comparison For Gettysburg Address.

the output of my AutoExtract program was judged slightly better than the ideal human extracts. Perhaps this was because I trained on an American corpus and the human testers were American, while the ideal summaries were selected by British volunteers from a British corpus. I was disappointed that the Sinope summarizer didn't score higher. However, this was a demo version of the summarizer available on the Internet; the final commercial version might do better. Also the Sinope summarizer was the only summarizer that attempted to combine sentences to create a more readable summary. That attempt may have backfired. Since word frequency is the main component in both my AutoExtract program and the Microsoft program, it's not surprising that the Microsoft entry did as well as it did.

The Gettysburg Address test was scored separately from the other six, since it was American and the other six were British. Moreover, for Zechner's six articles I had the ideal human extracts, but there was no ideal extract for *The Gettysburg Address*. For *The Gettysburg Address* test I forced each extract to be three sentences, except for Sinope. I included two different summaries from the Sinope program, since Sinope produces different summaries for the percentage and number of sentences options. (Also, I wanted all texts to have four summaries for testing purposes.) My AutoExtract program calculates output percentage based on numbers of sentences. Microsoft allows you to specify number of sentences or percentage, but the percentage option and number of sentences option give you the same output. Curiously, the Microsoft program scored dead last here. This is ironic since this is the output that was used to demonstrate the AutoSummarize tool in an article in *Slate Magazine*

(Gore 1997), which is published by Microsoft.

8.4 Summary of Contributions

This work clearly reinforces the overall importance of thematic word frequency (XS-RchainScore) for automatic text summarization, and demonstrates that the other lexical chain information components (MSRchainScore and SRchainScore) are at least as important as bonus words, stigma words, and first sentence position information. Moreover, combining all these factors into one linear salience function yields a very powerful and fast method of automatic text summarization. The evidence supports the conclusion that this work shows that the long topic-defining hypernym-hyponym and holonym-meronym chains of the MSRchainScore procedure are more important than the short synonym-antonym chains of the SRchainScore procedure.

This work is unique in that it is the only work to my knowledge that combines all the input parameters that it does with the use of lexical chains to create an automatic text summarizer. (See Table 8.4 on page 218 for a comparison.)

This work is unique by virtue of the nature and speed of the algorithms constructed. My method of lexical chaining is much faster than most other lexical chaining algorithms and compares favorably with commercial entries in the text summarization field such as Microsoft's Word97 AutoSummarize in terms of speed.

This work is unique in performing significantly better than both the Sinope commercial summarizer and Microsoft's Word97 AutoSummarize on the overlap

measurement and more importantly on the human volunteer testing.

In testing, I've done something that no other researcher has done. I used human volunteers to read and judge the final summaries, rather than just rely on recall, precision, or overlaps measurements. I didn't find any evidence that anyone else has used humans to validate the final results.

This work is unique in the way that lexical chain scores are kept as separate inputs to an overall salience function. The overall method that I use for the creation of lexical chains is quite similar to the approach of Barzilay & Elhadad (1997). However, I do not sum the chain metrics to create an overall chain score that is used to select chains or the sentences that they correspond to. Instead, in my approach the three chain scores that I compute are kept separate (extra-strong chain relation (word repetition), strong chain relation (e.g. synonymy), and medium-strength chain relation (e.g. link in an extended hypernym chain)), then used as three of the eleven input parameters to my salience function. To the best of my knowledge, I am the only one who does it this way, and the evidence supports the conclusion that the results that I achieved are due to this.

I wanted to keep the three types of chain scores separate because I wanted to test the rank ordering that past researchers have used. That is, it has always been assumed that the extra-strong relation was most important (and weighted the highest), and the medium-strength relation was the least important and weighted the least. My decision to keep the chain scores separate was validated by the results of the overlap test and the human volunteer test, which show that the so-called

medium-strength relation may actually be more important than the strong relation.

The training setup that I created, which allowed the medium-strength relation to receive the proper weighting, is unique in lexical chain construction. The evidence supports the conclusion that the assumption of past researchers about the relative importance of the three lexical chain scores may be wrong, and that in fact the medium-strength relationship based on longer hypernym-hyponym and holonym-meronym relationships may be more important than the short-length synonym type relations of the strong relation.

This work is also unique in being the only scheme using lexical chains that uses all the WordNet parts of speech (nouns, verbs, adverbs, adjectives) in lexical chain creation. Most researchers just rely on chains created from nouns. The performance of my summarizer would seem to validate this choice.

This work is significant because it creates a summarization scheme that performs better than both Microsoft Word97 AutoSummarize and the Sinope commercial summarizer. This work is important because it demonstrates that the longer chains of the medium-strength chaining relation are more important than the short length strong chain relation. This work is also important because of the speed of the chaining algorithms created.

Table 8.4: Saliency Function Input Parameters.

	Structure	Stop Words	Stigma Words	Word Case	Word Freq	Position/ Location	Cue Phrase	Length	Title Words	Word Stem
Byler 2001	lexical chains	✓	✓	✓	✓	✓	✓		✓	✓
Microsoft Word97 2001		✓			✓	?				
Sinope 2001	semantic structure	✓				✓	✓			
Dalianis 2000					✓	✓			✓	✓
Aone 1997							✓			
Barzilay 1997	lexical chains				✓					✓
Hovy 1997						✓	✓		✓	
Myaeng 1997							✓	✓	✓	
Teufel 1997					✓	✓	✓	✓	✓	
Kupiec 1995		✓		✓	✓	✓	✓	✓	✓	
Zechner 1995					✓	✓	✓	✓	✓	
Paice 1990					✓	✓	✓			
Pollock 1975			✓				✓			
Mathis 1972							✓			
Edmundson 61-68		✓			✓	✓	✓		✓	
Luhn 1958		✓			✓					?

8.5 Future Research

8.5.1 Corpus Creation

Clearly, the greatest need in automatic text summarization by extraction is the creation of a large corpus (hundreds or thousands) of texts with matching human validated extracts. This corpus, however, should not simply have a human validated extract for each text.

My idea is that each and every sentence, whether going into the extract or not, must be assigned a ranking (from lowest to highest) of how important the sentence would be to a summary. These scores could then be normalized over some range, say from 0 to 1000. This is the data that is critical for calculating parameter constants and validating the importance of different parameters.

8.5.2 Speed-up

My summarizer (which took about 30 seconds to summarize each of the test documents) could be speeded up even more by doing away with the extra-strength chaining procedure. As words are placed into the parse tree we just need to insert a pointer into the frequency hash table inside each of the tree nodes. My summarizer also could be speeded up a great deal by consolidating the memory allocation for each node (using malloc) into larger block allocations of memory.

8.5.3 Type of Document

Perhaps summarization should be adaptable to the type of documents to be summarized. This might be accomplished by having a learning mechanism for adjusting the weights of the summarization function parameters. Of course, you would need some sort of feedback mechanism.

8.5.4 Type of Sentence

Perhaps we should weight based on the type of sentence that we are scanning, with different weights for opinions, questions, statements, conjectures, facts, and quotes.

8.5.5 Continuous Range for Bonus Words

Luhn showed us that the importance of words in a summary follows a bell curve shaped distribution, with bonus words in the middle of the bell curve and stigma words at the edges. (See Figure 6.1 on page 118.) From this picture, it is obvious that bonus and stigma words are really just opposite ends of the same continuous spectrum. As such we should have a continuous range of weights from some very low positive number for the stigma end to some high positive number for the high bonus end of the spectrum. This is an extension of an earlier idea about cue phrases that divided them into five different classes. (Teufel & Moens 1997)

To implement this idea, we must first have or construct the corpus mentioned above. Then the score of each sentence in each text must be normalized for that text (e.g. the best sentence in each text could receive a score of 1000.0 and the

worst 0.0). That sentence's normalized score would then be assigned to each word in that sentence. Then all the words in all the sentences would be sorted based on score. The resulting output would then be divided into 100 separate groups. Any word that appeared in more than one group would be considered *noise* and removed. What we would be left with then would be the unique words that were typical of sentences of that scoring range. Each of these groups would be assigned its own parameter. The final multipliers for each of these parameters could be assigned based on the type of data fitting that I did with the DataFit program (or using a neural network or the Simplex Method from linear programming). I feel strongly that this would result in a much better salience function.

8.5.6 Better Chaining

The evidence supports the conclusion that the assumption of past researchers about the relative importance of the three lexical chain scores may be wrong, and that in fact the medium-strength relationship based on longer hypernym-hyponym and holonym-meronym relationships is more important than the short-length synonym type relations of the strong relation. When we think about it, this makes sense since the long hypernym-type chains tend to group items into topics, and these topics are the nuclei about which a summary should form. The extra-strong lexical chain relation is really just thematic term frequency, and, although it is very important for computing similarity between sentences, it not really at the heart of lexical chaining. I believe the heart of lexical chaining lies with the topic-defining chains of

the so-called medium strength relationship.

8.5.7 Better Saliency Function

Since summarization parameters are not really independent variables, more work needs to be done on developing a better, non-linear saliency function for combining the parameters.

Since we don't know what our function will look like in advance, an artificial neural network (ANN) would be ideal here. An ANN with an input node for each parameter (now eleven) would allow us to find an optimum (we might find a local rather than global minimum) fit for our training data. Once the training was done, we could always disassemble the network to get our function, or leave the net in place and use it to compute outputs. Leaving the network in place wouldn't place too great a burden on the system, since after training, the network would be feed-forward only. Moreover, leaving the network in place would allow you to adjust the weights of the network in the future for a different training corpus.

8.5.8 Compound Words

I accepted the recommendations of other researchers to use compound words wherever possible, but have begun to have doubts. We need to test whether using compounds instead of simple words composed of single morphemes actually improves chaining.

8.5.9 Include Bonus Words in Chaining

I treated bonus words completely separately from lexical chain construction. It would be worthwhile to try including bonus words in lexical chaining to see if this would improve chaining. I'm also beginning to think that words selected for any kind of relationship (e.g. extra-strong chain relation) should still be available for forming other kinds of relations (e.g. medium-strength chain relation).

BIBLIOGRAPHY

Bibliography

- Aho, A. V., Sethi, R. & Ullman, J. D. (1988), *Compilers, Principles, Techniques, and Tools*, Addison-Wesley Publishing Company, Reading, Massachusetts.
- Andersen, T. H. (2000), 'Text categorization using lexical chains'.
- Aone, C., Gorlinsky, J., Larsen, B. & Okurowski, M. E. (1997), *A Trainable Summarizer with Knowledge Acquired from Robust NLP Techniques*, The MIT Press, Cambridge, MA, pp. 71–80.
- Aretoulaki, M. (1994), 'Towards a hybrid abstract generation system', *In Proceedings of the International Conference on New Methods in Language Processing* pp. 220–227.
- Barzilay, R. & Elhadad, M. (1997), 'Using lexical chains for text summarization', *In Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS'97), ACL*.
- Baxendale, P. B. (1958), 'Machine-Made Index for Technical Literature-An Experiment', *IBM Journal* pp. 354–361.
- Boguraev, B. & Kennedy, C. (1997), 'Salience-based content characterization of text documents', in *Proc. of a workshop sponsored by the Association for Computational Linguistics* (Mani & Maybury 1997), pp. 2–9.
- Borko, H. & Bernier, C. (1975), *Abstracting Concepts and Methods*, Academic Press, New York, NY.
- Borko, H., ed. (1968), *Automated language processing*, Wiley, New York, NY.
- Brandow, R., Mitze, K. & Rau, L. F. (1995), 'Automatic condensation of electronic publications by sentence selection', *Information Processing and Management* 31:5, 675–685.
- Climenson, W. D., Hardwick, H. H. & Jacobson, S. N. (1961), 'Automatic Syntax Analysis in Machine Indexing and Abstracting', *American Documentation* 12(3), 178–183.

- Cremmins, E. T. (1996), *The Art of Abstracting*, Information Resources Press.
- Dalianis, H. (1999), 'ASTROGEN - Aggregated deep Surface naTuRal language GENEration', <http://www.dsv.su.se/hercules/ASTROGEN/ASTROGEN.html>.
- Dalianis, H. (2000), 'SweSum - A Text Summarizer for Swedish', *Royal Institute of Technology*.
- Dalianis, H. & Hovy, E. (1996), 'Aggregation in Natural Language Generation', *Trends in Natural Language Generation: an Artificial Intelligence Perspective, EWNLG'93, Fourth European Workshop, Lecture Notes in Artificial Intelligence* **1036**, 88-105.
- DeJong, G. (1978), 'Fast Skimming of News Stories: The FRUMP System', *Ph.D. diss. Yale University*.
- DeJong, G. (1982), 'An overview of the FRUMP system', pp. 149-175.
- Donlan, D. (1980), 'Locating Main Ideas in History Textbooks', *Journal of Reading* pp. 135-140.
- Edmundson, H. P. (1961), 'Automatic abstracting and indexing: Survey and recommendations', *Comm. ACM* **5:5**, 226-235.
- Edmundson, H. P. (1963), 'Automatic abstracting'.
- Edmundson, H. P. (1968), 'New methods in automatic extracting', *Journal of the ACM* **16:2**, 264-285.
- Endres-Niggemeyer, B. (1998), *Summarizing Information*, Springer-Verlag, New York, NY.
- Endres-Niggemeyer, B., Maier, E. & Sigel, A. (1995), 'How to implement a naturalist model of abstracting: four core working steps of an expert abstractor', *Information Processing and Management* **31(5)**, 631-674.
- Fum, D., Guida, G. & Tasso, C. (1982), 'Forward and backward reasoning in automatic abstracting', *Proc. 9th International Conference on Computational Linguistics* pp. 83-88.
- Fum, D., Guida, G. & Tasso, C. (1984), *A propositional language for text representation*, Amsterdam, Holland, pp. 121-150.
- Fum, D., Guida, G. & Tasso, C. (1985a), 'A rule-based approach to evaluating importance in descriptive texts', *Second conference of the European Chapter of the Association for Computational Linguistics Proc. EAACL85*, 244-250.

- Fum, D., Guida, G. & Tasso, C. (1985b), 'Evaluating importance: A step towards text summarization', *IJCAI-85: Proc. 9th International Joint Conference on Artificial Intelligence* pp. 840-844.
- Georgantopoulos, B. (1996), Automatic summarizing based on sentence extraction: A statistical approach, Master's thesis, University of Edinburgh.
- Gibb, F. (1993), 'Knowledge-based indexing in SIMPR: Integration of natural language processing and principles of subject analysis in an automated indexing system', *Journal of Document and Text Management* 1:2, 131-153.
- Gibb, F. & Smart, G. (1990), 'Structured information management using new techniques for processing text', *Online Review* 14:3, 159-171.
- Gore, K. (1997), 'Cogito Auto Sum', *Slate Magazine*. Posted online Saturday, Feb. 8, 1997, at 4:30 p.m. PT.
- Green, S. J. (1996), 'Building hypertext links in newspaper articles using semantic similarity', *University of Toronto*.
- Guha, R. V. & Lenat, D. B. (1990), 'CYC: A Mid-Term Report', *AI Magazine* pp. 32-59.
- Halliday, M. A. K. & Hasan, R. (1976), *Cohesion in English*, Longman, London, England.
- Hearst, M. (1994), 'Multi-paragraph segmentation of expository text', *Proc. 32nd Annual Meeting of the Association for Computational Linguistics* pp. 9-16.
- Hearst, M. A. & Plaunt, C. (1993), 'Subtopic structuring for full-length document access', *Proc. ACM-SIGIR'93* pp. 59-68.
- Hirst, G. & St-Onge, D. (1995), 'Lexical chains as representations of context for the detection and correction of malapropisms', *WordNet*.
- Hovy, E. (1988), *Generating natural language under pragmatic constraints*, Erlbaum, Hillsdale, NJ.
- Hovy, E. & Lin, C.-Y. (1997), 'Automated Text Summarization in SUMMARIST', *In Proceedings of the Workshop of Intelligent Scalable Text Summarization* pp. 18-24.
- Jacobs, P. S. & Rau, L. F. (1990), 'SCISOR: Extracting information from online news', *Comm. ACM* 33:11, 88-97.
- Jones, K. S. (1993), *What might be in a summary?*, Universitätsverlag, Konstanz, pp. 9-26.

- Jones, K. S. (1998), 'Automatic summarizing: factors and directions', *Computation and Language*.
- Karetnyk, D., Karlsson, F. & Smart, G. (1991), 'Knowledge-based indexing of morphosyntactically analyzed language', *Expert Systems for Information Management* 4, 1-29.
- Kintsch, W. (1974), *The representation of meaning in memory*, Erlbaum, Hillsdale, NJ.
- Kintsch, W. & van Dijk, T. A. (1978), 'Toward a model of text comprehension', *Psychological Review* 85, 363-394.
- Kupiec, J., Pedersen, J. & Chen, F. (1995), 'A trainable document summarizer', *SIGIR '95* pp. 68-73.
- Lehnert, W. & Ringle, M., eds (1982), *Strategies for natural languages processing*, Erlbaum, Hillsdale, NJ.
- Lehnert, W. G. (1981), 'Plot Units: A Narrative Summarization Strategy', *Strategies for natural languages processing* pp. 223-244.
- Levine, J. R., Mason, T. & Brown, D. (1992), *lex & yacc*, O'Reilly & Associates, Inc., 103 Morris Street, Suite A, Sebastopol, California 95472.
- Liddy, E. D. (1991), 'Discourse-level structure of empirical abstracts: an exploratory study', *Information Processing and Management* 27(1), 55-81.
- Lie, D. H. (1998), 'Sumatra: A System for Automatic Summary Generation', *Carp Technologies*.
- Lie, D. H., Hulstijn, J., op den Akker, R. & Nijholt, A. (1997), 'A Transformational Approach to NL Understanding in Dialogue Systems'.
- Luhn, H. P. (1958), 'The automatic creation of literature abstracts', *IBM Journal Research and Development* 2:2, 159-165.
- Mani, I. & Maybury, M., eds (1997), *Intelligent scalable text summarization*, Madrid.
- Mann, W. C. & Thompson, S. A. (1988), 'Rhetorical Structure Theory: Toward a functional theory of text organization', *Text* 8:3, 243-281.
- Marcu, D. (1997a), 'From Discourse Structures to Text Summaries', *The Proceedings of the ACL'97/EACL'97 Workshop on Intelligent Scalable Text Summarization* pp. 82-88.
- Marcu, D. (1997b), 'The rhetorical parsing of natural language texts', *Proc. 35th Annual Meeting of the Association for Computational Linguistics* pp. 96-103.

- Marcu, D. (1997c), *The Rhetorical Parsing, Summarization, and Generation of Natural Language Texts*, Ph.D. dissertation, University of Toronto, Toronto, Canada.
- Mathis, B. A. (1972), *Techniques for the evaluation and improvement of computer-produced abstracts*, Technical Report OSU-CISRC-TR-72-15, Ohio State University, Columbus OH.
- McKeown, K. & Radev, D. R. (1995), 'Generating summaries of multiple news articles', *SIG195* pp. 74-82.
- McKeown, K., Robin, J. & Kukich, K. (1995), 'Generating concise natural language summaries', *Information Processing and Management* 31:5, 703-733.
- Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D. & Miller, K. J. (1990), 'Introduction to WordNet: An on-line lexical database', *International Journal of Lexicography (special issue)* 3(4), 235-312.
- Mitra, M., Singhal, A. & Buckley, C. (1997), *Automatic text summarization by paragraph extraction*, in Mani & Maybury (1997), pp. 39-46.
- Morris, J. & Hirst, G. (1991), 'Lexical cohesion computed by thesaural relations as an indicator of the structure of text', *Computational Linguistics* 17(1), 21-48.
- Myaeng, S. H. & Jang, D.-H. (1997), *Development and Evaluation of a Statistically-Based Document Summarization System*, The MIT Press, Cambridge, MA, pp. 61-70.
- Ono, K., Sumita, K. & Miike, S. (1994), 'Abstract generation based on rhetorical structure extraction', *Proc. 15th International Conference on Computational Linguistics* 1, 344-348.
- Paice, C. D. (1981), *The automatic generation of literature abstracts*, Butterworths, London, pp. 172-191.
- Paice, C. D. (1990), 'Constructing literature abstracts by computer: Techniques and prospects', *Information Processing and Management* 26:1, 171-186.
- Peat, F. D. (1988), *Artificial Intelligence: How Machines Think*, Baen Books, Simon & Schuster, New York, New York.
- Pollock, J. J. & Zamora, A. (1975), 'Automatic Abstracting Research at Chemical Abstracts Service'.
- Rath, G. J., Resnick, A. & Savage, T. R. (1961), 'The formation of abstracts by the selection of sentences. Part 1. Sentence selection by men and machines', *American Documentation* 12:2, 139-141.

- Rau, L., Jacobs, P. & Zernik, U. (1989), 'Information extraction and text summarization using linguistic knowledge acquisition', *Information Processing and Management* 25:4, 419-428.
- Reimer, U. & Hahn, U. (1988), *Knowledge-Based Text Summarization: Salience and Generalization Operators for Knowledge Base Abstraction*, The MIT Press, Cambridge, MA, pp. 215-232.
- Resnick, A. (1961), 'The formation of abstracts by the selection of sentences. Part 2. The reliability of people in selecting sentences', *American Documentation* 12:2, 141-143.
- Salton, G., Singhal, A., Mitra, M. & Buckley, C. (1997), 'Automatic Text Structuring and Summarization', *Information Processing and Management* 33:2, 193-208.
- Skorohodko, E. (1971), 'Adaptive method of automatic abstracting and indexing', *Proc. IFIP Conference 1971* pp. 133-137.
- Skorohodko, E. (1981), *Semantische Relationen in der Lexik und in Texten [Semantic relations in the lexicon and in texts]*, Brockmeyer, Bochum.
- Stairmand, M. A. (1996), *A Computational Analysis of Lexical Cohesion with Applications in Information Retrieval*, Ph.D. dissertation, Center for Computational Linguistics, UMIST, Manchester, England.
- Teufel, S. & Moens, M. (1997), *Sentence extraction as a classification task*, in Mani & Maybury (1997), pp. 58-65.
- vanDijk, T. A. (1979), *Recalling and summarizing complex discourse*, Walter de Gruyter, Berlin, Germany.
- Weizenbaum, J. (1976), *Computer Power and Human Reason*, W. H. Freeman and Co., New York, New York.
- Witbrock, M. J. & Mittal, V. O. (1998), 'Ultra-Summarization: A Statistical Approach to Generating Highly Condensed Non-Extractive Summaries'.
- Wyllys, R. E. (1968), *Extracting and abstracting by computer*, in Borko (1968), pp. 127-179.
- Zechner, K. (1995), *Automatic Text Abstracting by Selecting Relevant Passages*, Master's thesis, Centre for Cognitive Science, University of Edinburgh, Edinburgh, Scotland.

APPENDICES

Appendix A

The Gettysburg Address

A.1 Original Text

Fourscore and seven years ago our fathers brought forth upon this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal. Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure. We are met on a great battlefield of that war. We have come to dedicate a portion of that field as a final resting-place for those who here gave their lives that that nation might live. It is altogether fitting and proper that we should do this. But in a larger sense we cannot dedicate, we cannot consecrate, we cannot hallow this ground. The brave men, living and dead, who struggled here, have consecrated it far above our poor power to add or detract. The world will little note, nor long remember, what we say here. But it can never forget what they did here. It is for us, the living, rather to be dedicated here to the unfinished work which they who fought here have thus far so nobly advanced. It is rather for us to be here dedicated to the great task remaining before us, that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion; that we here highly resolve that these dead shall not have died in vain; that this nation, under God, shall have a new birth of freedom, and that government of the people, by the people, and for the people, shall not perish from the earth.

A.2 Summary Produced by my AutoExtract Summarizer

My AutoExtract summarizer computes percentages for the extract based on whole sentences.

A.2.1 3 Sentences

Fourscore and seven years ago our fathers brought forth upon this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created

equal . Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure . It is rather for us to be here dedicated to the great task remaining before us, that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion; that we here highly resolve that these dead shall not have died in vain; that this nation, under God, shall have a new birth of freedom, and that government of the people, by the people, and for the people, shall not perish from the earth .

A.3 Summaries Produced by the Sinope Summarizer

The Sinope summarizer has separate choices for percentage or number of sentences for the summary. Selections from 25% to 32% gave back the same two sentences.

A.3.1 3 Sentences

But in a larger sense we cannot dedicate, we cannot consecrate, we cannot hallow this ground. It is for us, the living, rather to be dedicated here to the unfinished work which they who fought here have thus far so nobly advanced. It is rather for us to be here dedicated to the great task remaining before us, that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion; that we here highly resolve that these dead shall not have died in vain; that this nation, under God, shall have a new birth of freedom, and that government of the people, by the people, and for the people, shall not perish from the earth.

A.3.2 25 to 32 Percent Summary - 2 sentences

It is for us, the living, rather to be dedicated here to the unfinished work which they who fought here have thus far so nobly advanced. It is rather for us to be here dedicated to the great task remaining before us, that from these honored dead we take increased devotion to that cause for which they gave the last full measure of devotion; that we here highly resolve that these dead shall not have died in vain; that this nation, under God, shall have a new birth of freedom, and that government of the people, by the people, and for the people, shall not perish from the earth.

A.4 Summary Produced by Microsoft Word97

The Microsoft Word97 AutoSummarize tool allows you to choose a percentage for summary. You can choose number of sentences, but this is converted into a percentage. So they are not really different settings. To generate a summary of three sentences, I started at 40% and reduced the percentage until at 32% the summary

dropped from 4 sentences to 3 sentences. Thus, 32% is the maximum overall percentage that will generate 3 sentences. As it turned out, this produced the exact same summary as 25% to 32%, 3 sentences.

A.4.1 25 to 32 Percent Summary - 3 sentences

Fourscore and seven years ago our fathers brought forth upon this continent a new nation, conceived in liberty, and dedicated to the proposition that all men are created equal. Now we are engaged in a great civil war, testing whether that nation, or any nation so conceived and so dedicated, can long endure. The brave men, living and dead, who struggled here, have consecrated it far above our poor power to add or detract.

Appendix B

Newspaper Texts Used for Final Testing

Zechner (1995) extracted the following six texts from the Daily Telegraph Corpus, then used human volunteers to create “ideal” extracts of five to seven sentences each. He then combined these extracts by using the top ranked mean scores of sentences to produce the best overall extracts. These were then used to test his automatic text abstracting system.

B.1 Countryside Alliance Fights Trespass Law

B.1.1 Original Text

Countryside Alliance Fights Trespass Law.
By Charles Clover, Environment Editor.

AN unprecedented alliance of 115 countryside, wildlife and sporting organisations has written to Mr Ian Lang, the Scottish Secretary, asking him to scrap proposals making trespass a criminal offence in Scotland [1].

They express concern that parts of the Criminal Justice and Public Order Bill, which gives the police new powers to control the disturbance of hunting and grouse shooting, could be used by landowners to “intimidate” people using the outdoors for recreation [2].

The letter says the new offence of aggravated trespass “potentially affects every man, woman or child who wants to enjoy Scotland’s countryside” [3].

It is signed by Mr Bill Murray, chairman of the Scottish Countryside Activities Council, Mr Fred Nelson, chairman of the Scottish Sports Association, and Mr

Seaton Baxter, chairman of Scottish Wildlife and Countryside Link [4].

The groups say intimidation could take the form of restrictive notices, verbal confrontations and the threat of the involvement of police officers to investigate alleged criminal offences [5].

Mr Lang has already been urged by Scotland's police constables and by Magnus Magnusson, chairman of Scottish Natural Heritage, the country recreation watchdog, to scrap the proposed new offence in Scotland [6].

The new alliance of outdoor groups, which includes ramblers, mountaineers, cyclists, skiers and wildlife enthusiasts, says the Government has taken no account of the different tradition and legal basis for public access in Scotland and has made no attempt to consult with any of their organisations [7]. "There appears to be no recognition of the fact that virtually everyone going for a walk or run in Scotland's countryside is, technically speaking, exposed to accusations of being a trespasser," say the chairmen [8].

They say the public right to cross land in Scotland is often wrongly assumed to be a general right [9]. It is instead almost entirely based on assertion by the public of an accustomed right [10].

This, they say, is "a traditional right which many of the members we represent cherish deeply" [11].

The number of footpaths, tracks and wild areas where the public has secured a legal right to roam are "minute" when compared to the area of land to which the public presently enjoys access in Scotland - or with the official rights of way network in England and Wales [12].

The introduction of a crime of aggravated trespass would, they argue, lead to a "confrontational" situation with landowners [13].

They note that the provisions in the Criminal Justice Bill which seek to control "rares" - or mass parties - apply only to England and Wales and suggest the same should apply to aggravated trespass [14].

They say the need for measures to control the disruption of hunting and grouse shooting in Scotland has yet to be demonstrated [15]. Even if disruption were to take place it would be relatively easy for the Government to bring in an amendment to a future Criminal Justice Bill [16].

They add that the present Bill makes a nonsense of a review of access to the

outdoors being conducted by Scottish Natural Heritage [17].

B.1.2 Optimal Extract - 35.29% of original sentences

AN unprecedented alliance of 115 countryside, wildlife and sporting organisations has written to Mr Ian Lang, the Scottish Secretary, asking him to scrap proposals making trespass a criminal offence in Scotland [1].

They express concern that parts of the Criminal Justice and Public Order Bill, which gives the police new powers to control the disturbance of hunting and grouse shooting, could be used by landowners to “intimidate” people using the outdoors for recreation [2].

The new alliance of outdoor groups, which includes ramblers, mountaineers, cyclists, skiers and wildlife enthusiasts, says the Government has taken no account of the different tradition and legal basis for public access in Scotland and has made no attempt to consult with any of their organisations [7]. “There appears to be no recognition of the fact that virtually everyone going for a walk or run in Scotland’s countryside is, technically speaking, exposed to accusations of being a trespasser,” say the chairmen [8].

They say the public right to cross land in Scotland is often wrongly assumed to be a general right [9].

The number of footpaths, tracks and wild areas where the public has secured a legal right to roam are “minute” when compared to the area of land to which the public presently enjoys access in Scotland - or with the official rights of way network in England and Wales [12].

B.2 Lyell “Did Not Report Heseltine Concern”

B.2.1 Original Text

Lyell “Did Not Report Heseltine Concern”.

By Sean O’Neill.

THE ROLE of Sir Nicholas Lyell, the Attorney General, in the arms-to-Iraq affair was called into question again yesterday, shortly before he appears at the Scott inquiry [1].

Mr Alan Moses, QC, leading prosecution counsel in the Matrix Churchill trial, told the inquiry that Sir Nicholas did not inform him that Mr Michael Heseltine, President of the Board of Trade, had grave reservations about signing a public interest immunity certificate [2].

The trial of three former executives of Matrix Churchill, charged with illegally exporting arms-making machinery to Iraq, collapsed in November 1992, after certificates were overturned by the judge, releasing to the defence Government papers showing ministerial knowledge of the trade [3].

Mr Moses told Lord Justice Scott that the use of such certificates in the Matrix Churchill case, which has been defended by Sir Nicholas, had undermined public confidence in justice [4]. Such evidence from the Government's lawyer has further undermined the Attorney General, who is to give evidence at the inquiry tomorrow [5].

He is certain to face a gruelling interrogation about the quality of his advice to ministers [6].

On his second day before the inquiry, Mr Moses said he would have "wanted to know" about Mr Heseltine's views, but because he had not been told of them he felt he had not represented the minister's position properly at the trial [7].

Last month the inquiry was told that Mr Heseltine had refused to sign a gagging order to withhold Whitehall papers from the defence in the case because he feared he would be taking part in a "cover-up" [8].

Sir Nicholas told Mr Heseltine that despite his concerns it was his ministerial "duty" to sign [9]. Mr Heseltine agreed to sign a much amended certificate after Sir Nicholas assured him that the "unusual wording" and "limited scope" of the order would be drawn to the attention of the judge by counsel, yet three days later, on Sept 10 1992, Sir Nicholas met Mr Moses but did not mention the issue [10]. Mr Moses had further contacts with the Attorney's office and with the Treasury Solicitor's Department, but no one instructed him that Mr Heseltine wanted his dissenting view "flagged up" to the court [11].

On Sept 30, Mr Moses went to court and argued to uphold the certificate [12]. He told Judge Smedley, the trial judge, not that Mr Heseltine had been a reluctant signatory, but that the four ministers who had signed gagging orders were of one mind - that disclosure of documents would damage the public interest [13].

Mr Moses admitted: "My submission was not a sufficient explanation of the President's views as I now understand them [14]."

Mr Moses's evidence to the inquiry raised doubts about a statement issued by Sir Nicholas after Mr Heseltine's appearance at the inquiry [15].

Sir Nicholas had said that Mr Heseltine's certificate had been "expressly amended after consultation with the Attorney General and leading counsel for the prosecution" [16]. Mr Moses told the inquiry that he had received a copy of Mr Heseltine's amended certificate but he was not told what changes Mr Heseltine had made, or their significance [17].

"What I did not take on board and did not appreciate was that the President had expressed positive views as a matter of independent judgment about the individual documents," said Mr Moses [18].

He added: "I can well see a minister would be very ill-advised to sign a certificate unless he was quite satisfied as to the reasoning of the prosecution for fear, as I think Mr Heseltine feared, that he would be accused of withholding documents ... which a judge has ordered to be disclosed [19]."

B.2.2 Optimal Extract - 31.58% of original sentences

THE ROLE of Sir Nicholas Lyell, the Attorney General, in the arms-to-Iraq affair was called into question again yesterday, shortly before he appears at the Scott inquiry [1].

Mr Alan Moses, QC, leading prosecution counsel in the Matrix Churchill trial, told the inquiry that Sir Nicholas did not inform him that Mr Michael Heseltine, President of the Board of Trade, had grave reservations about signing a public interest immunity certificate [2].

The trial of three former executives of Matrix Churchill, charged with illegally exporting arms-making machinery to Iraq, collapsed in November 1992, after certificates were overturned by the judge, releasing to the defence Government papers showing ministerial knowledge of the trade [3].

Last month the inquiry was told that Mr Heseltine had refused to sign a gagging order to withhold Whitehall papers from the defence in the case because he feared he would be taking part in a "cover-up" [8].

Sir Nicholas told Mr Heseltine that despite his concerns it was his ministerial "duty" to sign [9]. Mr Heseltine agreed to sign a much amended certificate after Sir Nicholas assured him that the "unusual wording" and "limited scope" of the order

would be drawn to the attention of the judge by counsel, yet three days later, on Sept 10 1992, Sir Nicholas met Mr Moses but did not mention the issue [10].

B.3 Major Is Ready For Long Fight With EC

B.3.1 Original Text

Major Is Ready For Long Fight With EC.

By George Jones and Christopher Lockwood.

Britain's rift with the rest of Europe deepened last night after EC foreign ministers failed to break the deadlock over voting rights and Mr John Major, adopting a strident anti-Brussels stance, signalled that the Government was digging in for a long battle [1].

The Prime Minister will now seek to use the issue to the Conservatives' advantage in the forthcoming elections for the European parliament [2].

However his confrontational tone heightened fears of a new Tory split over Europe [3]. Pro-European Tory MPs accused the Prime Minister of bowing to the views of the party's Euro-sceptics and claimed that Mr Douglas Hurd, the Foreign Secretary, was unhappy with Mr Major's tough stand [4].

Yesterday's talks in Brussels broke up when it became clear there was no compromise acceptable to Britain and Spain - the only other country opposing an increase in the number of votes needed to block EC legislation [5].

Another effort will be made to broker a settlement when EC foreign ministers meet in Greece at the weekend, but positions on both sides appear to be hardening [6].

Britain's European partners said that failure to solve the voting issue within the next few days could derail plans to enlarge the Community from 12 members to 16 with the inclusion of Austria, Sweden, Norway and Finland next year [7].

While Mr Hurd emerged from yesterday's talks emphasising that Britain was "not inflexible", Mr Major was in uncompromising mood in the House of Commons [8].

Euro-sceptics cheered when he vowed to "fight Britain's corner hard" [9]. He said "phoney threats" to delay enlargement of the EC would not sway him [10].

He also took an electioneering swipe at the Labour Party [11]. Claiming Labour would sign away "competitiveness and money", he described Mr John Smith, the Labour leader, as: "The man who likes to say 'yes' in Europe - Monsieur Oui, the poodle of Brussels [12]."

Delighted Tory Euro-sceptics claimed that Mr Major, faced with the prospect of a Conservative "civil war" or a clash with Europe, had chosen to battle with the EC [13].

But there was dismay among the party's European wing, who fear enlargement of the EC is threatened and that Mr Major is jeopardising compromise within the party over the issue [14].

The voting row centres on whether the number of votes in the EC's ruling Council of Ministers needed to block legislation should be increased from 23 to 27 when the four new countries join [15].

There remains strong opposition within the Cabinet to giving way on 27 [16]. Mr Kenneth Clarke, the Chancellor, said during a visit to Sweden that London felt the voting issue should be uncoupled from the enlargement process and deferred until 1996, when EC leaders are to hold an inter-governmental conference [17].

Emerging from the Brussels talks, Mr Hurd said he had rejected a compromise proposal, which would allow for a delay of one or two months in cases where there were 23 votes against a proposal, but not the full 27 [18].

"In our view, this does not give adequate protection to our interests," he said [19]. The compromise would have included a non-binding declaration that the EC would try to avoid imposing decisions blocked by 23 votes [20].

Britain is demanding a protocol to the enlargement treaty which would commit the EC not to override a vote of 23 within a still unspecified time limit [21].

B.3.2 Optimal Extract - 33.33% of original sentences

Britain's rift with the rest of Europe deepened last night after EC foreign ministers failed to break the deadlock over voting rights and Mr John Major, adopting a strident anti-Brussels stance, signalled that the Government was digging in for a long battle [1].

Yesterday's talks in Brussels broke up when it became clear there was no compromise acceptable to Britain and Spain - the only other country opposing an increase

in the number of votes needed to block EC legislation [5].

Britain's European partners said that failure to solve the voting issue within the next few days could derail plans to enlarge the Community from 12 members to 16 with the inclusion of Austria, Sweden, Norway and Finland next year [7].

The voting row centres on whether the number of votes in the EC's ruling Council of Ministers needed to block legislation should be increased from 23 to 27 when the four new countries join [15].

There remains strong opposition within the Cabinet to giving way on 27 [16].

Emerging from the Brussels talks, Mr Hurd said he had rejected a compromise proposal, which would allow for a delay of one or two months in cases where there were 23 votes against a proposal, but not the full 27 [18].

Britain is demanding a protocol to the enlargement treaty which would commit the EC not to override a vote of 23 within a still unspecified time limit [21].

B.4 International - Senate To Stage Public Inquiry On Whitewater

B.4.1 Original Text

International - Senate To Stage Public Inquiry On Whitewater.
By Maurice Weaver in Washington.

WASHINGTON began preparing yesterday for one of the biggest Congressional spectacles since the Iran-Contra inquiry following a Senate decision to hold public hearings on President Clinton's Whitewater property dealings [1].

Leaders of the minority Republican Party, having bulldozed Democratic objections aside, hope the "show" will start in the spring [2].

But first they must agree a timetable with the Democrats who, despite having caved in, remain deeply uneasy at the prospect of seeing the President's pre-election record of private financial manoeuvres subjected to public scrutiny [3].

The television and press coverage will be unwelcome and possibly damaging at a crucial stage in Mr Clinton's legislative programme [4].

The President, faced with a *fait accompli*, has said diplomatically that Congress must "do whatever it is they think is the right thing to do" [5].

But one of his closest political advisers, Mr James Carville, gave a more candid insight into White House feelings yesterday, when he grumbled that the affair was turning into "a zoo" [6].

Another White House insider, Presidential Counsellor David Gergen, expressed dismay at the "overheated atmosphere" in Washington and the vehemence of Mr Clinton's critics, which he attributed largely to his ambitious reform programme [7]. The risk, he said, is that the Whitewater controversy will paralyse the Presidency [8].

Even as the Clinton team rallied to the banner, however, new information was leaking out about the Clintons' family finances which, while not suggesting any improprieties, does indicate how the couple benefited from the "cronyism" in their native state of Arkansas [9].

Mrs Hillary Clinton, the New York Times disclosed, made nearly \$660,000 by trading in cattle futures in the late 1970s on the advice of a lawyer friend who worked for Tyson Foods, one of Arkansas's biggest companies [10].

The revelation is considered significant because it shows how a couple from relatively humble beginnings were able, by virtue of knowing the right people, to attain the financial security that would help them scale the social and political heights [11].

The Clintons arrived in Washington as the ultimate meritocrats, the baby-boomers who made it to the very pinnacle on their own abilities [12]. How much damage will be done to that reputation by the disclosures that will come spilling out at public Congressional hearings is the key question [13].

Yesterday, the leader of the Senate's Democratic majority, Senator George Mitchell, and his Republican opposite number, Senator Robert Dole, were discussing with their inner circles what an "appropriate" timetable for the hearings will be [14].

The Democrats initially argued that a Congressional inquiry would hinder the investigation being carried out by Mr Robert Fiske, the special counsel appointed by the Justice Department [15].

But with the collapse of unanimity within the party as more members concluded that full public disclosure of the Whitewater facts was preferable to perceived secretiveness, the leadership was forced to give way to Republican demands [16].

Senator Dole is now calling for the hearings to begin as soon as possible, and his

representatives speak unofficially of a spring start [17].

It would, theoretically, be possible for Congress to call Mr Clinton to testify [18]. But constitutionalists consider that highly unlikely [19].

B.4.2 Optimal Extract - 31.58% of original sentences

WASHINGTON began preparing yesterday for one of the biggest Congressional spectacles since the Iran-Contra inquiry following a Senate decision to hold public hearings on President Clinton's Whitewater property dealings [1].

But first they must agree a timetable with the Democrats who, despite having caved in, remain deeply uneasy at the prospect of seeing the President's pre-election record of private financial manoeuvres subjected to public scrutiny [3].

The risk, he said, is that the Whitewater controversy will paralyse the Presidency [8]. Even as the Clinton team rallied to the banner, however, new information was leaking out about the Clintons' family finances which, while not suggesting any improprieties, does indicate how the couple benefited from the "cronyism" in their native state of Arkansas [9].

Mrs Hillary Clinton, the New York Times disclosed, made nearly \$660,000 by trading in cattle futures in the late 1970s on the advice of a lawyer friend who worked for Tyson Foods, one of Arkansas's biggest companies [10].

But with the collapse of unanimity within the party as more members concluded that full public disclosure of the Whitewater facts was preferable to perceived secretiveness, the leadership was forced to give way to Republican demands [16].

B.5 International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision

B.5.1 Original Text

International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision.

British forces will help UN monitor delicate ceasefire.
By Peter Almond, Defence Correspondent, and Robert Fox in Zagreb.

THE British troops who began arriving in Split last night will help the United Nations monitor a delicate Croat-Muslim ceasefire in Bosnia [1]. They will initially be deployed there for four months [2].

The first VC-10 load of troops, an advance party from the 1st Bn The Duke of Wellington's Regiment, left RAF Brize Norton, Oxon, within an hour of Mr Malcolm Rifkind, the Defence Secretary, announcing to the Commons that the Government had decided to send the extra forces to support the UN's request for 10,650 [3].

At the same time, six C-130 Hercules carrying Land Rovers left RAF Lyneham for Split, and three Navy auxiliary ships began loading 109 Saxon armoured troop carriers and other vehicles at Marchwood, Southampton [4].

They are expected to sail this morning and arrive in Split on March 20 [5]. By then, 792 soldiers, including engineers and logistics troops, along with 113 men with 15 Scimitar tracked reconnaissance vehicles from D Squadron of The Light Dragoons at Hohne, Germany, will have been deployed to Bosnia to join 2,300 already there [6].

Explaining the Government's change of mind about extra troops, Mr Rifkind told the Commons that the ceasefires had brought new opportunities and responsibilities that were greatly stretching the ability of the Coldstream Guards battalion group to fulfil its UN mission [7].

"The Coldstream Guards, whose mission hitherto had been to support humanitarian aid convoys, have found themselves with a major peacekeeping task on their doorstep," he said [8].

"It has become clear that the effort involved, while tolerable for a time, is unsustainable beyond the short term with their current manpower [9]. Although the UK contribution to the region is already a large one, a further UK contribution at this stage as part of a coordinated international effort would help to make the difference between success and failure for the ceasefires [10]."

The Duke of Wellington's will provide Lt-Gen Sir Michael Rose, the UN Commander in Bosnia, with his most powerful new body to help cement the Croat-Muslim ceasefire [11].

The troops will work to the complex ceasefire map drawn up by a joint commission of Bosnian army and Croat HVO militia commanders under the chairmanship of Brig John Reith, Commander of the new UN Sector SouthWest in Bosnia [12].

British officials say the map shows a ceasefire to be policed along 125 miles of

front lines between the two sides, some of which cross high mountain ridges [13].

The soldiers are needed to patrol no man's land, help take weapons to collection sites and man checkpoints and observation posts [14].

Next week, the UN soldiers will begin overseeing the disengagement of forces [15]. Surprisingly, only a few miles of the front line are currently under dispute - part of the perimeter of Mostar airfield and a heavily booby-trapped municipal office in the middle of Vitez [16].

But UN commanders fear that any part of the front line could flare into conflict at any time, as the war is a hotchpotch of village rivalries and local feuds [17].

Jim Muir in Sarajevo writes: Gen Rose welcomed the announcement from London and other indications that he is likely to get the extra troops he needs [18].

B.5.2 Optimal Extract - 38.89% of original sentences

THE British troops who began arriving in Split last night will help the United Nations monitor a delicate Croat-Muslim ceasefire in Bosnia [1].

Explaining the Government's change of mind about extra troops, Mr Rifkind told the Commons that the ceasefires had brought new opportunities and responsibilities that were greatly stretching the ability of the Coldstream Guards battalion group to fulfil its UN mission [7].

"Although the UK contribution to the region is already a large one, a further UK contribution at this stage as part of a coordinated international effort would help to make the difference between success and failure for the ceasefires [10]."

The troops will work to the complex ceasefire map drawn up by a joint commission of Bosnian army and Croat HVO militia commanders under the chairmanship of Brig John Reith, Commander of the new UN Sector SouthWest in Bosnia [12].

British officials say the map shows a ceasefire to be policed along 125 miles of front lines between the two sides, some of which cross high mountain ridges [13].

The soldiers are needed to patrol no man's land, help take weapons to collection sites and man checkpoints and observation posts [14].

Next week, the UN soldiers will begin overseeing the disengagement of forces [15].

B.6 "Thatcher Circle" In Pergau Row

B.6.1 Original Text

"Thatcher Circle" In Pergau Row.
By Robert Shrimmsley and Anthony Looch.

The "close circle" around Lady Thatcher was criticised yesterday by Sir David Steel, the Liberal Democrats' foreign affairs spokesman, for involvement in the Pergau Dam project [1].

Speaking in an acrimonious Commons debate on the £234 million of British aid for the dam, Sir David said: "In the course of battling for Britain, which our former Prime Minister did extremely well, her close circle often seems to have been involved [2].

"In this case Sir Tim Bell, well known as her PR adviser, is also adviser to the Malaysian Prime Minister and to Tam Sri Armugam, who controls GEC Malaysia, heavily involved in several of the contracts under the deal [3].

"One other person who helped broker parts of the deal is Mr Steve Tipping, a business associate of Mr Mark Thatcher and indeed best man at his wedding [4].

"Tim Bell told the Sunday Times when asked to clarify Tipping's exact role: 'What he does for a living is introduce people to each other [5]."

Opening the debate, Sir David said the only possible explanation for the funding of the project was the Government's "eagerness to lubricate the trade channels with Malaysia, especially the arms trade" [6].

He said Malaysia, which last year received £20.4 million in aid - the 12th largest donation from Britain - was far from poor [7]. Seeking to establish a link between aid and arms sales, he pointed out that Oman, whose GNP per head is higher than Portugal's had had its aid from Britain doubled since 1979 "and it currently ranks third largest in the list of purchasers of arms from Britain [8]."

Referring to an "unhealthy dominance" of Tory Party benefactors among firms in the Pergau project, Sir David said: "Companies linked by such donations have been the main beneficiaries to the tune of 42 per cent of the Aid and Trade Provision [9]." He cited Trafalgar House, Balfour Beatty, GEC and Biwater [10].

Mr Alistair Goodlad, a Foreign Office minister, ridiculed Sir David's figures, adding: "It is ridiculous to suggest that a country where there is a defence sales

relationship should, because of that relationship, be ineligible for aid for trade provision [11]."

He said: "If we had not carried through our commitment our credibility as a trading and investment partner would have been seriously damaged and with it a wide range of British interests and prospects for the future [12]."

The Lib-Dem motion attacking the Government generally over its conduct of overseas aid and on Pergau, was defeated in a 305-159 division (Government majority 146 [13].)

Speaking from New York, Mr Major said last night that he had "no regrets" over the deal [14]. He was adopting a "wait and see" policy towards Malaysia's reimposition of a "buy British last" policy [15].

He refused to go into the question of retaliation, saying: "It is a matter I hope we will be able to sort out [16]."

This afternoon Mr Hurd, the Foreign Secretary, will give evidence to the Commons Foreign Affairs Select Committee, which is conducting an inquiry into the affair [17].

He will attempt to smooth ruffled Malaysian feathers while also explaining why he overruled advice from the top civil servant at the Overseas Development Administration, who described the dam as "a very bad buy" [18].

B.6.2 Optimal Extract - 33.33% of original sentences

The "close circle" around Lady Thatcher was criticised yesterday by Sir David Steel, the Liberal Democrats' foreign affairs spokesman, for involvement in the Pergau Dam project [1].

Speaking in an acrimonious Commons debate on the £234 million of British aid for the dam, Sir David said: "In the course of battling for Britain, which our former Prime Minister did extremely well, her close circle often seems to have been involved [2].

"In this case Sir Tim Bell, well known as her PR adviser, is also adviser to the Malaysian Prime Minister and to Tam Sri Armugam, who controls GEC Malaysia, heavily involved in several of the contracts under the deal [3].

Opening the debate, Sir David said the only possible explanation for the funding of the project was the Government's "eagerness to lubricate the trade channels with Malaysia, especially the arms trade" [6].

Referring to an "unhealthy dominance" of Tory Party benefactors among firms in the Pergau project, Sir David said: "Companies linked by such donations have been the main beneficiaries to the tune of 42 per cent of the Aid and Trade Provision [9]."

The Lib-Dem motion attacking the Government generally over its conduct of overseas aid and on Pergau, was defeated in a 305-159 division (Government majority 146 [13].)

Appendix C

AutoExtract Summaries

The following summaries were produced by my AutoExtract summarization program. The extraction percentage was set to produce the same number of sentences as Zechner's manual "ideal" extract summaries in the previous appendix. This allowed me to measure effectiveness by simply measuring the overlap with the "ideal" summaries, since the recall, precision, and overlap measurements are all the same. The sentences that match the ideal extract are emphasized.

C.1 Countryside Alliance Fights Trespass Law

Overlap with ideal extract: 50%.

sentNum: 1, score: 718.485453

AN unprecedented alliance of 115 countryside, wildlife and sporting organisations has written to Mr Ian Lang, the Scottish Secretary, asking him to scrap proposals making trespass a criminal offence in Scotland [1].

sentNum: 2, score: 818.118035

They express concern that parts of the Criminal Justice and Public Order Bill, which gives the police new powers to control the disturbance of hunting and grouse shooting, could be used by landowners to "intimidate" people using the outdoors for recreation [2].

sentNum: 3, score: 455.939571

The letter says the new offence of aggravated trespass "potentially affects every man, woman or child who wants to enjoy Scotland's countryside" [3].

sentNum: 4, score: 426.463276

It is signed by Mr Bill Murray, chairman of the Scottish Countryside Activities Council, Mr Fred Nelson, chairman of the Scottish Sports Association, and Mr Seaton Baxter, chairman of Scottish Wildlife and Countryside Link [4].

sentNum: 6, score: 399.213521

Mr Lang has already been urged by Scotland's police constables and by Magnus Magnusson, chairman of Scottish Natural Heritage, the country recreation watchdog, to scrap the proposed new offence in Scotland [6].

sentNum: 7, score: 519.314534

The new alliance of outdoor groups, which includes ramblers, mountaineers, cyclists, skiers and wildlife enthusiasts, says the Government has taken no account of the different tradition and legal basis for public access in Scotland and has made no attempt to consult with any of their organisations [7].

C.2 Lyell "Did Not Report Heseltine Concern"

Overlap with ideal extract: 83%.

sentNum: 1, score: 419.177363

THE ROLE of Sir Nicholas Lyell, the Attorney General, in the arms-to-Iraq affair was called into question again yesterday, shortly before he appears at the Scott inquiry [1].

sentNum: 2, score: 762.623876

Mr Alan Moses, QC, leading prosecution counsel in the Matrix Churchill trial, told the inquiry that Sir Nicholas did not inform him that Mr Michael Heseltine, President of the Board of Trade, had grave reservations about signing a public interest immunity certificate [2].

sentNum: 3, score: 584.454884

The trial of three former executives of Matrix Churchill, charged with illegally exporting arms-making machinery to Iraq, collapsed in November 1992, after certificates were overturned by the judge, releasing to the defence Government papers showing ministerial knowledge of the trade [3].

sentNum: 4, score: 367.746079

Mr Moses told Lord Justice Scott that the use of such certificates in the Matrix Churchill case, which has been defended by Sir Nicholas, had undermined public confidence in justice [4].

sentNum: 8, score: 499.108454

Last month the inquiry was told that Mr Heseltine had refused to sign a gagging order to withhold Whitehall papers from the defence in the case because he feared he would be taking part in a "cover-up" [8].

sentNum: 10, score: 343.614382

Mr Heseltine agreed to sign a much amended certificate after Sir Nicholas assured him that the "unusual wording" and "limited scope" of the order would be drawn to the attention of the judge by counsel, yet three days later, on Sept 10 1992, Sir Nicholas met Mr Moses but did not mention the issue [10].

C.3 Major Is Ready For Long Fight With EC

Overlap with ideal extract: 71%.

sentNum: 1, score: 939.098441

Britain's rift with the rest of Europe deepened last night after EC foreign ministers failed to break the deadlock over voting rights and Mr John Major, adopting a strident anti-Brussels stance, signalled that the Government was digging in for a long battle [1].

sentNum: 5, score: 472.918317

Yesterday's talks in Brussels broke up when it became clear there was no compromise acceptable to Britain and Spain - the only other country opposing an increase in the number of votes needed to block EC legislation [5].

sentNum: 7, score: 551.799689

Britain's European partners said that failure to solve the voting issue within the next few days could derail plans to enlarge the Community from 12 members to 16 with the inclusion of Austria, Sweden, Norway and Finland next year [7].

sentNum: 13, score: 392.736233

Delighted Tory Euro-sceptics claimed that Mr Major, faced with the prospect of a Conservative "civil war" or a clash with Europe, had chosen to battle with the EC [13].

sentNum: 14, score: 480.222090

But there was dismay among the party's European wing, who fear enlargement of the EC is threatened and that Mr Major is jeopardising compromise within the party over the issue [14].

sentNum: 15, score: 406.822147

The voting row centres on whether the number of votes in the EC's ruling Council of Ministers needed to block legislation should be increased from 23 to 27 when the four new countries join [15].

sentNum: 18, score: 443.223740

Emerging from the Brussels talks, Mr Hurd said he had rejected a compromise proposal, which would allow for a delay of one or two months in cases where there were 23 votes against a proposal, but not the full 27 [18].

C.4 International - Senate To Stage Public Inquiry On Whitewater

Overlap with ideal extract: 50%.

sentNum: 1, score: 943.676482

WASHINGTON began preparing yesterday for one of the biggest Congressional spectacles since the Iran-Contra inquiry following a Senate decision to hold public hearings on President Clinton's Whitewater property dealings [1].

sentNum: 2, score: 539.717943

Leaders of the minority Republican Party, having bulldozed Democratic objections aside, hope the "show" will start in the spring [2].

sentNum: 3, score: 572.454575

But first they must agree a timetable with the Democrats who, despite having caved in, remain deeply uneasy at the prospect of seeing the President's pre-election record of private financial manoeuvres subjected to public scrutiny [3].

sentNum: 9, score: 529.365285

Even as the Clinton team rallied to the banner, however, new information was leaking out about the Clintons' family finances which, while not suggesting any improprieties, does indicate how the couple benefited from the "cronyism" in their native state of Arkansas [9].

sentNum: 11, score: 546.550978

The revelation is considered significant because it shows how a couple from relatively humble beginnings were able, by virtue of knowing the right people, to attain the financial security that would help them scale the social and political heights [11].

sentNum: 14, score: 557.960115

Yesterday, the leader of the Senate's Democratic majority, Senator George Mitchell, and his Republican opposite number, Senator Robert Dole, were discussing with their inner circles what an "appropriate" timetable for the hearings will be [14].

C.5 International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision

Overlap with ideal extract: 57%.

sentNum: 1, score: 397.541107

THE British troops who began arriving in Split last night will help the United Nations monitor a delicate Croat-Muslim ceasefire in Bosnia [1].

sentNum: 3, score: 759.618610

The first VC-10 load of troops, an advance party from the 1st Bn The Duke of Wellington's Regiment, left RAF Brize Norton, Oxon, within an hour of Mr Malcolm Rifkind, the Defence Secretary, announcing to the Commons that the Government had decided to send the extra forces to support the UN's request for 10,650 [3].

sentNum: 7, score: 587.813210

Explaining the Government's change of mind about extra troops, Mr Rifkind told the Commons that the ceasefires had brought new opportunities and responsibilities that were greatly stretching the ability of the Coldstream Guards battalion group to fulfil its UN mission [7].

sentNum: 9, score: 385.515507

"It has become clear that the effort involved, while tolerable for a time, is unsustainable beyond the short term with their current manpower [9].

sentNum: 10, score: 504.264163

Although the UK contribution to the region is already a large one, a further UK contribution at this stage as part of a coordinated international effort would help to make the difference between success and failure for the ceasefires [10]."

sentNum: 11, score: 371.840518

The Duke of Wellington's will provide Lt-Gen Sir Michael Rose, the UN Commander in Bosnia, with his most powerful new body to help cement the Croat-Muslim ceasefire [11].

sentNum: 12, score: 598.588667

The troops will work to the complex ceasefire map drawn up by a joint commission of Bosnian army and Croat HVO militia commanders under the chairmanship of Brig John Reith, Commander of the new UN Sector SouthWest in Bosnia [12].

C.6 "Thatcher Circle" In Pergau Row

Overlap with ideal extract: 50%.

sentNum: 1, score: 749.406379

The "close circle" around Lady Thatcher was criticised yesterday by Sir David Steel, the Liberal Democrats' foreign affairs spokesman, for involvement in the Pergau Dam project [1].

sentNum: 3, score: 507.581144

"In this case Sir Tim Bell, well known as her PR adviser, is also adviser to the Malaysian Prime Minister and to Tam Sri Armugam, who controls GEC Malaysia, heavily involved in several of the contracts under the deal [3]."

sentNum: 4, score: 518.886394

"One other person who helped broker parts of the deal is Mr Steve Tipping, a business associate of Mr Mark Thatcher and indeed best man at his wedding [4]."

sentNum: 9, score: 541.233762

Referring to an "unhealthy dominance" of Tory Party benefactors among firms in the Pergau project, Sir David said: "Companies linked by such donations have been the main beneficiaries to the tune of 42 per cent of the Aid and Trade Provision [9]."

sentNum: 11, score: 524.938756

Mr Alistair Goodlad, a Foreign Office minister, ridiculed Sir David's figures, adding: "It is ridiculous to suggest that a country where there is a defence sales relationship should, because of that relationship, be ineligible for aid for trade provision [11]."

sentNum: 12, score: 560.934118

He said: "If we had not carried through our commitment our credibility as a trading and investment partner would have been seriously damaged and with it a wide range of British interests and prospects for the future [12]."

Appendix D

Sinope Summaries

The following summaries were produced by the Sinope summarization program, available at www.carp-technologies.nl/sinope_demo_e.html or www.carp-technologies.nl/sinope_demo_java_e.html. The demo versions of the program allow you to paste text into a box and receive the summary by email. The program was set to produce the same number of sentences as the Zechner ideal summaries. I measured effectiveness by measuring overlap with those “ideal” summaries. The sentences that match the ideal extract are emphasized.

D.1 Countryside Alliance Fights Trespass Law

Overlap with ideal extract: 50%.

From: <sinope-info@carp-technologies.nl>
To: <byler@cs.utk.edu>
Subject: Sinope generated summary
Date: Monday, June 25, 2001 9:51 PM

Sinope generated summary (approximately 6 sentences)

It is signed by Mr Bill Murray, chairman of the Scottish Countryside Activities Council, Mr Fred Nelson, chairman of the Scottish Sports Association, and Mr Seaton Baxter, chairman of Scottish Wildlife and Countryside Link [4].

Mr Lang has already been urged by Scotland's police constables and by Magnus Magnusson, chairman of Scottish Natural Heritage, the country recreation watchdog, to scrap the proposed new offence in Scotland [6].

The new alliance of outdoor groups, which includes ramblers, mountaineers, cyclists, skiers and wildlife enthusiasts, says the Government has taken no account of the different tradition and legal basis for public access in Scotland and has made

no attempt to consult with any of their organisations [7]. "There appears to be no recognition of the fact that virtually everyone going for a walk or run in Scotland's countryside is, technically speaking, exposed to accusations of being a trespasser," say the chairmen [8].

The number of footpaths, tracks and wild areas where the public has secured a legal right to roam are "minute" when compared to the area of land to which the public presently enjoys access in Scotland - or with the official rights of way network in England and Wales [12].

The introduction of a crime of aggravated trespass would, they argue, lead to a "confrontational" situation with landowners [13].

They note that the provisions in the Criminal Justice Bill which seek to control "rares" - or mass parties - apply only to England and Wales and suggest the same should apply to aggravated trespass [14].

D.2 Lyell "Did Not Report Heseltine Concern"

Overlap with ideal extract: 33%.

From: <sinope-info@carp-technologies.nl>
To: <byler@cs.utk.edu>
Subject: Sinope generated summary
Date: Monday, June 25, 2001 9:45 PM

Sinope generated summary (approximately 6 sentences)

THE ROLE of Sir Nicholas Lyell, the Attorney General, in the arms-to-Iraq affair was called into question again yesterday, shortly before he appears at the Scott inquiry [1].

Mr Alan Moses, QC, leading prosecution counsel in the Matrix Churchill trial, told the inquiry that Sir Nicholas did not inform him that Mr Michael Heseltine, President of the Board of Trade, had grave reservations about signing a public interest immunity certificate [2].

He is certain to face a gruelling interrogation about the quality of his advice to ministers [6].

On his second day before the inquiry, Mr Moses said he would have "wanted to know" about Mr Heseltine's views, but because he had not been told of them he

felt he had not represented the minister's position properly at the trial [7].

Mr Moses's evidence to the inquiry raised doubts about a statement issued by Sir Nicholas after Mr Heseltine's appearance at the inquiry [15].

D.3 Major Is Ready For Long Fight With EC

Overlap with ideal extract: 57%.

From: <sinope-info@carp-technologies.nl>

To: <byler@cs.utk.edu>

Subject: Sinope generated summary

Date: Monday, June 25, 2001 9:47 PM

Sinope generated summary (approximately 7 sentences)

Britain's rift with the rest of Europe deepened last night after EC foreign ministers failed to break the deadlock over voting rights and Mr John Major, adopting a strident anti-Brussels stance, signalled that the Government was digging in for a long battle [1].

Yesterday's talks in Brussels broke up when it became clear there was no compromise acceptable to Britain and Spain - the only other country opposing an increase in the number of votes needed to block EC legislation [5].

Delighted Tory Euro-sceptics claimed that Mr Major, faced with the prospect of a Conservative "civil war" or a clash with Europe, had chosen to battle with the EC [13].

But there was dismay among the party's European wing, who fear enlargement of the EC is threatened and that Mr Major is jeopardising compromise within the party over the issue [14].

There remains strong opposition within the Cabinet to giving way on 27 [16]. Mr Kenneth Clarke, the Chancellor, said during a visit to Sweden that London felt the voting issue should be uncoupled from the enlargement process and deferred until 1996, when EC leaders are to hold an inter-governmental conference [17].

Emerging from the Brussels talks, Mr Hurd said he had rejected a compromise proposal, which would allow for a delay of one or two months in cases where there were 23 votes against a proposal, but not the full 27 [18].

D.4 International - Senate To Stage Public Inquiry On White-water

Overlap with ideal extract: 0%.

From: <sinope-info@carp-technologies.nl>
To: <byler@cs.utk.edu>
Subject: Sinope generated summary
Date: Monday, June 25, 2001 9:48 PM

Sinope generated summary (approximately 6 sentences)

The President, faced with a fait accompli, has said diplomatically that Congress must "do whatever it is they think is the right thing to do" [5].

But one of his closest political advisers, Mr James Carville, gave a more candid insight into White House feelings yesterday, when he grumbled that the affair was turning into "a zoo" [6].

Another White House insider, Presidential Counsellor David Gergen, expressed dismay at the "overheated atmosphere" in Washington and the vehemence of Mr Clinton's critics, which he attributed largely to his ambitious reform programme [7].

How much damage will be done to that reputation by the disclosures that will come spilling out at public Congressional hearings is the key question [13].

Yesterday, the leader of the Senate's Democratic majority, Senator George Mitchell, and his Republican opposite number, Senator Robert Dole, were discussing with their inner circles what an "appropriate" timetable for the hearings will be [14].

D.5 International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision

Overlap with ideal extract: 43%.

From: <sinope-info@carp-technologies.nl>
To: <byler@cs.utk.edu>
Subject: Sinope generated summary
Date: Monday, June 25, 2001 9:55 PM

Sinope generated summary (approximately 7 sentences)

At the same time, six C-130 Hercules carrying Land Rovers left RAF Lyneham for Split, and three Navy auxiliary ships began loading 109 Saxon armoured troop carriers and other vehicles at Marchwood, Southampton [4].

Although the UK contribution to the region is already a large one, a further UK contribution at this stage as part of a coordinated international effort would help to make the difference between success and failure for the ceasefires [10].

The troops will work to the complex ceasefire map drawn up by a joint commission of Bosnian army and Croat HVO militia commanders under the chairmanship of Brig John Reith, Commander of the new UN Sector South West in Bosnia [12].

The soldiers are needed to patrol no man's land, help take weapons to collection sites and man checkpoints and observation posts [14].

Surprisingly, only a few miles of the front line are currently under dispute - part of the perimeter of Mostar airfield and a heavily booby-trapped municipal office in the middle of Vitez [16].

But UN commanders fear that any part of the front line could flare into conflict at any time, as the war is a hotchpotch of village rivalries and local feuds [17].

Jim Muir in Sarajevo writes: Gen Rose welcomed the announcement from London and other indications that he is likely to get the extra troops he needs [18].

D.6 "Thatcher Circle" In Pergau Row

Overlap with ideal extract: 50%.

From: <sinope-info@carp-technologies.nl>
To: <byler@cs.utk.edu>
Subject: Sinope generated summary
Date: Monday, June 25, 2001 9:50 PM

Sinope generated summary (approximately 6 sentences)
(Sinope actually returned 7 sentences, so 50% is the best case overlap.)

Speaking in an acrimonious Commons debate on the £234 million of British aid for the dam, Sir David said: "In the course of battling for Britain, which our former Prime Minister did extremely well, her close circle often seems to have been involved

[2].

"In this case Sir Tim Bell, well known as her PR adviser, is also adviser to the Malaysian Prime Minister and to Tam Sri Armugam, who controls GEC Malaysia, heavily involved in several of the contracts under the deal [3].

He said Malaysia, which last year received £20.4 million in aid - the 12th largest donation from Britain - was far from poor [7]. Seeking to establish a link between aid and arms sales, he pointed out that Oman, whose GNP per head is higher than Portugal's had had its aid from Britain doubled since 1979 "and it currently ranks third largest in the list of purchasers of arms from Britain [8].

Referring to an "unhealthy dominance" of Tory Party benefactors among firms in the Pergau project, Sir David said: "Companies linked by such donations have been the main beneficiaries to the tune of 42 per cent of the Aid and Trade Provision [9]." He cited Trafalgar House, Balfour Beatty, GEC and Biwater [10].

He said: "If we had not carried through our commitment our credibility as a trading and investment partner would have been seriously damaged and with it a wide range of British interests and prospects for the future [12].

Appendix E

Microsoft Word97 Summaries

The following summaries were produced using the Microsoft Word97 AutoSummarize tool. Microsoft calculates the compression percentage based on words rather than sentences, but still extracts whole sentences. The program was set to produce the same number of sentences as in the Zechner ideal summaries. I measured effectiveness by measuring overlap with the “ideal” summaries. The sentences that match the ideal extract are emphasized.

E.1 Countryside Alliance Fights Trespass Law

Overlap with ideal extract: 50%.

AN unprecedented alliance of 115 countryside, wildlife and sporting organisations has written to Mr Ian Lang, the Scottish Secretary, asking him to scrap proposals making trespass a criminal offence in Scotland [1].

They express concern that parts of the Criminal Justice and Public Order Bill, which gives the police new powers to control the disturbance of hunting and grouse shooting, could be used by landowners to “intimidate” people using the outdoors for recreation [2].

The letter says the new offence of aggravated trespass “potentially affects every man, woman or child who wants to enjoy Scotland’s countryside” [3].

It is signed by Mr Bill Murray, chairman of the Scottish Countryside Activities Council, Mr Fred Nelson, chairman of the Scottish Sports Association, and Mr Seaton Baxter, chairman of Scottish Wildlife and Countryside Link [4].

Mr Lang has already been urged by Scotland’s police constables and by Magnus Magnusson, chairman of Scottish Natural Heritage, the country recreation watchdog, to scrap the proposed new offence in Scotland [6].

The number of footpaths, tracks and wild areas where the public has secured a legal right to roam are "minute" when compared to the area of land to which the public presently enjoys access in Scotland - or with the official rights of way network in England and Wales [12].

E.2 Lyell "Did Not Report Heseltine Concern"

Overlap with ideal extract: 33%.

Mr Alan Moses, QC, leading prosecution counsel in the Matrix Churchill trial, told the inquiry that Sir Nicholas did not inform him that Mr Michael Heseltine, President of the Board of Trade, had grave reservations about signing a public interest immunity certificate [2].

Mr Moses told Lord Justice Scott that the use of such certificates in the Matrix Churchill case, which has been defended by Sir Nicholas, had undermined public confidence in justice [4]. *Sir Nicholas told Mr Heseltine that despite his concerns it was his ministerial "duty" to sign [9].* On Sept 30, Mr Moses went to court and argued to uphold the certificate [12]. Mr Moses's evidence to the inquiry raised doubts about a statement issued by Sir Nicholas after Mr Heseltine's appearance at the inquiry [15].

Sir Nicholas had said that Mr Heseltine's certificate had been "expressly amended after consultation with the Attorney General and leading counsel for the prosecution" [16].

E.3 Major Is Ready For Long Fight With EC

Overlap with ideal extract: 43%.

Britain's rift with the rest of Europe deepened last night after EC foreign ministers failed to break the deadlock over voting rights and Mr John Major, adopting a strident anti-Brussels stance, signalled that the Government was digging in for a long battle [1].

Pro-European Tory MPs accused the Prime Minister of bowing to the views of the party's Euro-sceptics and claimed that Mr Douglas Hurd, the Foreign Secretary, was unhappy with Mr Major's tough stand [4].

Euro-sceptics cheered when he vowed to "fight Britain's corner hard" [9]. He said "phoney threats" to delay enlargement of the EC would not sway him [10].

The voting row centres on whether the number of votes in the EC's ruling Council of Ministers needed to block legislation should be increased from 23 to 27 when the four new countries join [15].

The compromise would have included a non-binding declaration that the EC would try to avoid imposing decisions blocked by 23 votes [20].

Britain is demanding a protocol to the enlargement treaty which would commit the EC not to override a vote of 23 within a still unspecified time limit [21].

E.4 International - Senate To Stage Public Inquiry On Whitewater

Overlap with ideal extract: 33%.

WASHINGTON began preparing yesterday for one of the biggest Congressional spectacles since the Iran-Contra inquiry following a Senate decision to hold public hearings on President Clinton's Whitewater property dealings [1].

Leaders of the minority Republican Party, having bulldozed Democratic objections aside, hope the "show" will start in the spring [2].

But first they must agree a timetable with the Democrats who, despite having caved in, remain deeply uneasy at the prospect of seeing the President's pre-election record of private financial manoeuvres subjected to public scrutiny [3].

The television and press coverage will be unwelcome and possibly damaging at a crucial stage in Mr Clinton's legislative programme [4].

Yesterday, the leader of the Senate's Democratic majority, Senator George Mitchell, and his Republican opposite number, Senator Robert Dole, were discussing with their inner circles what an "appropriate" timetable for the hearings will be [14].

It would, theoretically, be possible for Congress to call Mr Clinton to testify [18].

E.5 International - Troops Leave For Bosnia Within Hour Of Defence Secretary Announcing Government Decision

Overlap with ideal extract: 57%.

THE British troops who began arriving in Split last night will help the United Nations monitor a delicate Croat-Muslim ceasefire in Bosnia [1]. At the same time, six C-130 Hercules carrying Land Rovers left RAF Lyneham for Split, and three Navy auxiliary ships began loading 109 Saxon armoured troop carriers and other vehicles at Marchwood, Southampton [4].

Explaining the Government's change of mind about extra troops, Mr Rifkind told the Commons that the ceasefires had brought new opportunities and responsibilities that were greatly stretching the ability of the Coldstream Guards battalion group to fulfil its UN mission [7].

The Duke of Wellington's will provide Lt-Gen Sir Michael Rose, the UN Commander in Bosnia, with his most powerful new body to help cement the Croat-Muslim ceasefire [11].

The troops will work to the complex ceasefire map drawn up by a joint commission of Bosnian army and Croat HVO militia commanders under the chairmanship of Brig John Reith, Commander of the new UN Sector South West in Bosnia [12].

British officials say the map shows a ceasefire to be policed along 125 miles of front lines between the two sides, some of which cross high mountain ridges [13].

The soldiers are needed to patrol no man's land, help take weapons to collection sites and man checkpoints and observation posts [14].

Next week, the UN soldiers will begin overseeing the disengagement of forces [15].

E.6 "Thatcher Circle" In Pergau Row

Overlap with ideal extract: 67%.

The "close circle" around Lady Thatcher was criticised yesterday by Sir David Steel, the Liberal Democrats' foreign affairs spokesman, for involvement in the Pergau Dam project [1].

Speaking in an acrimonious Commons debate on the £234 million of British aid for the dam, Sir David said: "In the course of battling for Britain, which our former

Prime Minister did extremely well, her close circle often seems to have been involved [2].

Opening the debate, Sir David said the only possible explanation for the funding of the project was the Government's "eagerness to lubricate the trade channels with Malaysia, especially the arms trade" [6].

He said Malaysia, which last year received £20.4 million in aid - the 12th largest donation from Britain - was far from poor [7]. Mr Alistair Goodlad, a Foreign Office minister, ridiculed Sir David's figures, adding: "It is ridiculous to suggest that a country where there is a defence sales relationship should, because of that relationship, be ineligible for aid for trade provision [11]."

The Lib-Dem motion attacking the Government generally over its conduct of overseas aid and on Pergau, was defeated in a 305-159 division (Government majority 146 [13].)

Appendix F

Bonus, Stigma, and Common Words

In the following sections are the 140 bonus words (cue words), 16 stigma words, and 386 common words (stop words) used by the AutoExtract summarizer. These lists were compiled by finding the intersection of all the other lists of other researchers that I could find. These words (in both upper and lower case form) are compiled into the lexer portion of the AutoExtract summarizer.

F.1 Bonus Words

accordingly, Accordingly, account, Account, again, Again, also, Also although, Although, argue, Argue, article, Article, as we said, As we said, at once, At once, attempt, Attempt, at this point, At this point, because, Because, begin, Begin, best, Best, better, Better, case, Case, cause, Cause, causes, Causes, chapter, Chapter, circumstance, Circumstance, circumstances, Circumstances, clearly, Clearly, concerning, Concerning, conclude, Conclude, concluding, Concluding, conclusion, Conclusion, connection, Connection, consequently, Consequently, consider, Consider, develop, Develop, efficient, Efficient, eg, e.g., Eg, E.g., eight, Eight, elegant, Elegant, etc, etc., Etc, Etc., example, Example, fifth, Fifth, finally, Finally, first, First, follows, Follows, for, For, fourth, Fourth, further, Further, furthermore, Furthermore, getting back to, Getting back to, good, Good, greatest, Greatest, hence, Hence, however, However, i.e., I.e., ignoring this, Ignoring this, illustrate, Illustrate, important, Important, incidentally, Incidentally, innovative, Innovative, in order to, In order to, in short, In short, insightful, Insightful, investigate, Investigate, investigation, Investigation, method, Method, more, More, moreover, Moreover, most, Most, mostly, Mostly, much, Much, must, Must, namely, Namely, nevertheless, Nevertheless, next, Next, note, Note, often, Often, one, One, ones, Ones, otherwise, Otherwise, our, Our, outperform, Outperform, outperforms, Outperforms, paper, Paper, point, Point, previous, Previous, probably, Probably, propose, Pro-

pose, prove, Prove, provides, Provides, purpose, Purpose, reason, Reason, respect, Respect, result, Result, results, Results, second, Second, secondly, Secondly, seven, Seven, seventh, Seventh, show, Show, significant, Significant, significantly, Significantly, similar, Similar, similarly, Similarly, six, Six, sixth, Sixth, so, So, sum, Sum, summary, Summary, take, Take, taken, Taken, than, Than, that, That, that being so, That being so, that is to say, That is to say, that's, that's, That's, That's, then, Then, thence, Thence, there, There, thereafter, Thereafter, thereby, Thereby, therefore, Therefore, therein, Therein, these, These, third, Third, this, This, those, Those, three, Three, thus, Thus, to put it another way, To put it another way, to recapitulate, To recapitulate, to resume, To resume, to return, To return, to sum up, To sum up, to this end, To this end, to wit, To wit, use, Use, using, Using, viz, Viz, we must now turn to, We must now turn to, with this in mind, With this in mind, work, Work

F.2 Stigma

actually, Actually, anyhow, Anyhow, at all events, At all events, at the same time, At the same time, believe, Believe, but, But, either way, Either way, hardly, Hardly, impossible, Impossible, inadequate, Inadequate, inconclusive, Inconclusive, instead, Instead, insufficient, Insufficient, obvious, Obvious, on the contrary, On the contrary, on the other hand, On the other hand

F.3 Common Words - Stop Words - Noise

's, A, a, about, About, above, Above, Accordingly, accordingly, Across, across, After, after, afterwards, Afterwards, Again, again, Against, against, all, All, allows, Allows, Almost, almost, alone, Alone, along, Along, Already, already, Also, also, Always, always, am, Am, Among, among, amongst, Amongst, an, An, and, And, another, Another, Any, any, Anybody, anybody, anyone, Anyone, Anything, anything, anyway, Anyway, Anyways, anyways, Anywhere, anywhere, apart, Apart, Appear, appear, Appropriate, appropriate, are, Are, around, Around, As, as, Aside, aside, associated, Associated, at, At, Available, available, Away, away, awfully, Awfully, b, B, be, Be, became, Became, Become, become, Becomes, becomes, becoming, Becoming, Been, been, Before, before, beforehand, Beforehand, Behind, behind, being, Being, below, Below, Beside, beside, Besides, besides, between, Between, beyond, Beyond, Both, both, brief, Brief, By, by, C, c, came, Came, Can, can, Can't, can't, Cannot, cannot, certain, Certain, certainly, Certainly, change, Change, Changes, changes, co, Co, Contain, contain, Containing, containing, Contains, contains, Corresponding, corresponding, could, Could, course, Course, Currently, currently, d, D, described, Described, did, Did, Different, different, Do, do, does, Does, Doing, doing, done, Done, down, Down, Downwards, downwards, during, During, e, E, each, Each, Either, either, else, Else, Elsewhere, elsewhere, enough, Enough, Especially,

especially, Et, et, Even, even, Ever, ever, every, Every, Everybody, everybody, everyone, Everyone, everything, Everything, Everywhere, everywhere, Ex, ex, except, Except, f, F, Far, far, Few, few, Five, five, Followed, followed, following, Following, former, Former, Formerly, formerly, Forth, forth, Four, four, from, From, G, g, Get, get, Gets, gets, given, Given, gives, Gives, Go, go, goes, Goes, Going, going, gone, Gone, got, Got, Gotten, gotten, H, h, had, Had, happens, Happens, Has, has, have, Have, having, Having, he, He, Her, her, Here, here, Hereafter, hereafter, hereby, Hereby, Herein, herein, hereupon, Hereupon, Hers, hers, herself, Herself, Him, him, Himself, himself, His, his, Hither, hither, how, How, Howbeit, howbeit, i, I, i'd, I'd, I'll, i'll, I'm, i'm, I've, i've, If, if, ignored, Ignored, Immediate, immediate, in, In, Inasmuch, inasmuch, Inc, inc, Indeed, indeed, Indicate, indicate, Indicated, indicated, Indicates, indicates, inner, Inner, insofar, Insofar, Into, into, Inward, inward, Is, is, It, it, It's, it's, its, Its, Itself, itself, J, j, Just, just, K, k, keep, Keep, kept, Kept, Know, know, known, Known, knows, Knows, L, l, last, Last, Latter, latter, latterly, Latterly, least, Least, less, Less, Lest, lest, let, Let, like, Like, little, Little, looks, Looks, ltd, Ltd, M, m, many, Many, may, May, Me, me, Meanwhile, meanwhile, might, Might, My, my, Myself, myself, n, N, name, Name, near, Near, necessary, Necessary, Neither, neither, Never, never, New, new, Nine, nine, No, no, Nobody, nobody, none, None, noone, Noone, nor, Nor, normally, Normally, Not, not, Nothing, nothing, Novel, novel, Now, now, nowhere, Nowhere, o, O, Of, of, off, Off, oh, Oh, Old, old, on, On, Once, once, Only, only, Onto, onto, Or, or, Other, other, others, Others, ought, Ought, Ours, ours, ourselves, Ourselves, out, Out, outside, Outside, Over, over, overall, Overall, Own, own, P, p, particular, Particular, particularly, Particularly, per, Per, Perhaps, perhaps, Placed, placed, Please, please, plus, Plus, Possible, possible, q, Q, que, Que, Quite, quite, R, r, Rather, rather, re, Re, really, Really, relatively, Relatively, respectively, Respectively, Right, right, S, s, Said, said, Same, same, says, Says, See, see, seem, Seem, seemed, Seemed, seeming, Seeming, seems, Seems, seen, Seen, self, Self, Selves, selves, Sensible, sensible, sent, Sent, Serious, serious, Several, several, Shall, shall, she, She, should, Should, Since, since, Some, some, Somebody, somebody, somehow, Somehow, someone, Someone, Something, something, sometime, Sometime, Sometimes, sometimes, Somewhat, somewhat, Somewhere, somewhere, Specified, specified, Specify, specify, Specifying, specifying, Still, still, Sub, sub, Such, such, Sup, sup, t, T, The, the, their, Their, theirs, Theirs, Them, them, Themselves, themselves, there's, There's, theres, Theres, Thereupon, thereupon, these, These, they, They, thorough, Thorough, Thoroughly, thoroughly, though, Though, through, Through, Throughout, throughout, thru, Thru, To, to, Together, together, Too, too, Toward, toward, towards, Towards, Twice, twice, two, Two, U, u, Un, un, under, Under, unless, Unless, until, Until, unto, Unto, Up, up, Upon, upon, Us, us, Used, used, useful, Useful, Uses, uses, Usually, usually, V, v, value, Value, various, Various, Very, very, via, Via, Vs, vs, w, W, was, Was, Way, way, we, We, Well, well, Went, went, were, Were, what, What, whatever, Whatever, when, When, Whence, whence, whenever, Whenever, where, Where, whereafter,

Whereafter, whereas, Whereas, Whereby, whereby, wherein, Wherein, Whereupon, whereupon, Wherever, wherever, Whether, whether, which, Which, while, While, whither, Whither, Who, who, Whoever, whoever, whole, Whole, Whom, whom, whose, Whose, why, Why, Will, will, with, With, Within, within, without, Without, Would, would, wouldn't, Wouldn't, x, X, y, Y, yes, Yes, Yet, yet, you, You, You're, you're, Your, your, yours, Yours, Yourself, yourself, Yourselves, yourselves, Z, z, Zero, zero

Appendix G

Glossary

Many terms unique to WordNet and linguistics are used in this work. To avoid confusion, the more significant terms are listed here.

abstract

An abstract is both a summarization and reformulation of the original text.

anaphora

Anaphora is a reference to something previously mentioned in the text. Consider:

John is a boy.

He is fast.

The pronoun “he” is anaphoric since it refers to John. The problem of “anaphoric resolution” crops up in text summarization by extraction when we extract a sentence where it isn’t clear what all the antecedents are. If we extracted the second sentence, we would need some way of resolving that “he” refers to “John.”

base form

The base form of a word is the form to which inflections are added.

cohesion

Cohesion is what makes a text a text instead of an assortment of unrelated syntactic structures.

collocation

In WordNet, a collocation is a string of two or more words, connected by spaces or hyphens. In the database itself, spaces are represented as underscore characters.

compression ratio

A measure of how much shorter the summary is than the original.

coordinate

Words with the same hypernym are coordinate terms.

cousin

Words or senses whose hyponyms have some sort of relation to each other.

DARPA

Defense Advanced Research Projects Agency.

discourse markers

Term used by professional text summarizers to refer to words that indicate how one sentence relates to another.

discourse structure

Discourse structure refers to text type and structural organization, structural entities.

entailment (verbs)

A entails B, if A cannot be done unless B is done (or has been done).

extract

An extract is simply a subset of the original set of sentences in the text.

extrinsic evaluations

An extrinsic evaluation uses some task to measure a system's performance.

gloss

A definition or example sentence inside a WordNet synset.

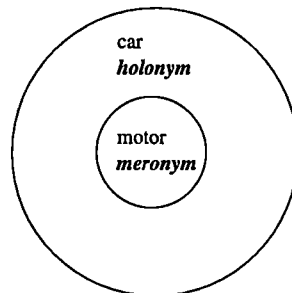
gold standard

The standard for success. Teufel uses "gold standard sentence" to mean a source sentence that matches a summary sentence based on semantic or syntactic similarity.

holonym - opposite of meronym - container

The whole of which the meronym is a part. B is a holonym of A if A is a part of B.

Car is holonym of motor.

**hypernymy**

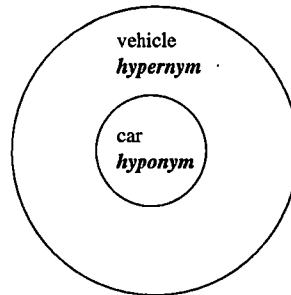
is-a relationship.

hypernym - superordinate - opposite of hyponym- container

B is a hypernym of A if A is a (kind of) B.

Vehicle is a hypernym of car.

Flower is a hypernym of pansy.



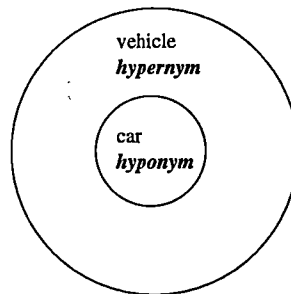
hyponym - subordinate - opposite of hypernym - member

A word with a narrower meaning whose meaning is included in the meaning of a more general term.

Pansy is a hyponym of flower.

Rose is a hyponym of flower.

Car is a hyponym of vehicle.



IE

Information Extraction.

intrinsic evaluations

Intrinsic methods directly measure some "intrinsic" system quality.

IR

Information Retrieval.

lemma

The lower case ASCII form of a word as found in WordNet.

lemmatization

The finding of root forms.

lexical chain

As defined by Morris in 1991, a lexical chain is a sequence of words relating to the same concept.

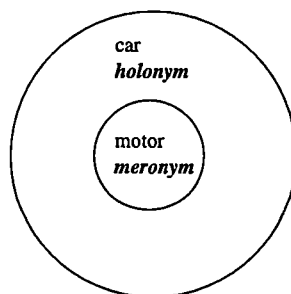
lexical pointer

A lexical pointer indicates a relationship between words in synsets.

meronym - opposite of holonym - member

Part of, substance of, or member of something. A is a meronym of B if A is a part of B.

Motor is a meronym of car.

**morphology**

Literally, a study and description of word formulation in a language, but usually used to refer to the construction of words from more basic components.

MUC

Message Understanding Conference - part of TIPSTER.

NIST

National Institute of Standards and Technology.

part of speech (pos)

WordNet limits "part of speech" to either noun, verb, adjective, or adverb.

pertainym

A relational adjective. Adjectives that are pertainyms are usually defined by such phrases as "of or pertaining to" and do not have antonyms. A pertainym can point to a noun or to another pertainym.

polysemous

The property of having more than one sense in a syntactic category.

polysemy count

In WordNet, the number of senses of a word in a syntactic category.

polysemy

Having many concepts for the same word.

precision

Used to measure IR performance. Precision is defined as the number of relevant documents retrieved divided by the total number of documents retrieved.

If there are 80 documents on widgets in the collection, and the system returns 60 documents, with 40 about widgets, then the precision is $40/60 = 67\%$.

recall

Used to measure IR performance. Recall is defined as the number of relevant documents retrieved divided by the total number of relevant documents in the collection.

For example, suppose there are 80 documents on widgets in the collection, and the system returns 60 documents, with 40 about widgets then the recall is $40/80 = 50\%$.

retention ratio

The percentage of the original information you retain.

semantic concordance

A text corpus (e.g. Brown Corpus) and a lexicon (e.g. WordNet) combined so that every substantive word in the text is linked to its sense in the lexicon by means of a semantic tag.

semantic pointer

A semantic pointer indicates a relation between synsets.

semantic tag

A pointer from a text file to a specific sense of a word in the WordNet database. A semantic tag in a semantic concordance is represented by a sense key.

sense

A meaning of a word in WordNet. Each sense of a word is in a different synset.

sense key

The information necessary to find a sense in WordNet.

sister

Two strings that are both the immediate hyponyms of the same superordinate.

subordinate

Same as hyponym.

SUMMAC

SUMMARization Conference - part of TIPSTER.

superordinate

Same as hypernym.

synonymy

Equivalence of meaning. The state or quality of being synonymous. This term is also used to refer to the problem of many words for the same concept.

synset

A synonym set in WordNet.

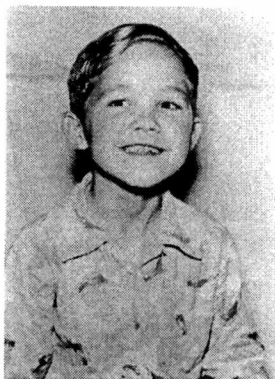
TIPSTER

A research effort examining Document Detection (TREC), Information Extraction (MUC) and Summarization (SUMMAC).

TREC

Text REtrieval Conference - part of TIPSTER.

Vita



Charles Ray Byler was born on 24 November 1949 in the sleepy backwater of Searcy, Arkansas. He graduated with honors from Porterville Union High School in June 1968, then enrolled at California Institute of Technology (CalTech). He dropped out of CalTech in 1970 and was immediately given an all-expense-paid vacation to sunny Southeast Asia by the U. S. Army. After placing second in the Southeast Asian War Games from 1970 to 1974, he returned to CalTech and graduated with Honors with a B.S. in Chemistry/History in 1976. He then re-entered the Army and retired in 1993. Upon retirement he enrolled in the graduate program in computer science at the University of Tennessee, Knoxville. He received an M.S. in Computer Science in December 1996 and was awarded the Doctor of Philosophy degree in Computer Science from the University of Tennessee in August 2001.