Summary Generation through Intelligent Cutting and Pasting of the Input Document

Ph.D. Thesis Proposal

Hongyan Jing
hjing@cs.columbia.edu

Department of Computer Science
Columbia University
New York, NY 10027

January 30, 1999

Contents

List of Figures

List of Tables

Abstract

We present a framework for fast generating concise and coherent summaries in domain-independent, single-document summarization. The proposed generation approach, called *cut-and-paste*, generates summaries through reusing the input document. Rather than using the extracted document sentences directly for producing summaries, the cut-and-paste approach edits the sentences in some way so that they are more concise, coherent, and appropriate for summaries.

We specially investigate two effective techniques, sentence reduction and sentence combination, for transforming extracted sentences into appropriate summary sentences. We also constructed a reusable, large-scale lexicon to support the generation process. This study is based on a corpus of human-written summaries; we have developed an automatic program to analyze the composition of human-written summaries. The system is designed to be a general generation tool portable to any domain-independent, single-document summarizer in need of a generation component.

Chapter 1

The Problem

1.1 Problem description

Automatic text summarization draws substantial interest since it provides a promising solution to the acute information overload problem people face in this Digital Era. One of the pressing research issues in summarization is summary generation. Almost all automatic summarizers, from the first system built at IBM more than 40 years ago [Luhn, 1958] to dozens of commercial and research products nowadays, rely on extracting sentences from the original documents to produce summaries. Since the sentences are extracted without context, at best, they can be incoherent, and at worst, they can convey misleading information or be incomprehensible. It is clearly time to develop better summary generation techniques than naïve extraction.

In this proposal, we address the generation problem in domain-independent, single-document summarization, a topic rarely studied in previous research. We propose a *cut-and-paste* based approach for summary generation. The main idea is that, rather than using extracted sentences directly to produce summaries, we edit them in some way so that they are more concise, coherent, and appropriate for summaries. The editing includes removing extraneous information from a sentence, combining a sentence with related sentences, and syntactic transformations. We call it cut-and-paste since summary generation reuses the input document, cutting words, phrases, or clauses from the document and putting them together in novel ways.

1.2 Why is summary generation important?

Extraction-based summaries (called extracts) have many shortcomings:

(1) **Misleading or incomprehensible information**

Placing unrelated sentences side by side can result in a misleading text. For example, *The department chair went to the meeting.* followed by *He went home late.* will lead people to think that *he* refers to *the department chair*, although *he* might refer to another person in the document and *the chair* is actually a female.

(2) **Incoherent**

An extracted sentence can contain anaphora, logical and rhetorical connectives, and long-range references, which will cause incoherence when the sentence is used in the summaries without context. [Paice, 1990] discussed this problem in more detail.

(3) **Redundant**

Extracted sentences usually contain more detail than necessary. For example, an extracted sentence can be *I wrote a proposal which is 30 pages long, not including the appendix, and I wrote it in the semester of...* , while what we really want to say in the summary is *I wrote a proposal.*

The extraneous information lengthens the summary. Our automatic corpus analysis of 90 summaries of news articles shows that the human-written abstracts are 52% length of their semantic-equivalent extracts.

[Marcu, 1998] shows that the human-written abstracts are 36% length of their semantic-equivalent extracts for 10 documents on computers. The inclusion of extraneous information also affects the clarity of the main points.

(4) **Lack of structure**

A text generally has a structure – for example, research papers usually have sections like *Introduction, Proposed Method, Evaluation,* and *Conclusion.* Extraction, however, selects sentences based on vague importance scores; therefore, the extracts do not have such a structure. [Paice, 1990] discussed this problem in more detail.

(5) **Applicability**

Extraction can not be applied when documents consist of extraordinary long sentences, as in certain patent databases.

Given all these shortcomings, it is imperative to investigate better summary generation techniques than extraction.

1.3 Why is summary generation difficult?

Summary generation is a difficult problem because available generation techniques can not meet the demands of summarization. A traditional generation system requires sophisticated semantic information and extensive domain knowledge, which is impossible to provide by a summarizer which works in a general domain and relies on text analysis tools to extract information. All traditional generation systems work in a specific domain [Swartout, 1983, McKeown, 1985, McCoy, 1986, Meteer et al., 1987, Miller and Rennels, 1988, Meteer, 1989, Elhadad, 1992, Robin, 1994, McKeown et al., 1994]. Since there are usually limited types of messages in a specific domain, generation systems can rely on the deep understanding of the semantic input and comprehensive domain knowledge to decide what to say and how to say it. Such method, however, can not be scaled up to a general domain since the types of messages are unlimited and comprehensive knowledge in a general domain is unavailable. For these reasons, we can not rely on the available generation techniques to tackle the summary generation problem but have to investigate new method.

1.4 Overview of the proposal

The proposed work relates to three research areas: summarization, generation and lexicons. We focus on the generation problem in summarization and build a large-scale lexicon to support generation.

The main problem we tackle in this work is how to edit the extracted sentences and transform them into concise, coherent sentences appropriate for summaries. We propose two main techniques: sentence reduction and sentence combination. Sentence reduction removes extraneous information from a sentence; thus it shortens the summaries, stresses the main points, and delete potential incoherence. Sentence combination merges a sentence with phrases and clauses from other sentences in order to substitute an incoherent component or add closely related information; thus it reduces incoherence and enhance the informativeness of summaries.

This research is corpus-based. We collect a corpus of human-written summaries and use them for training and evaluating sentence reduction and sentence combination program. To automatically analyze human-written summaries, we develop a program to analyze the composition of human-written summary sentences. The program is based on the Hidden Markov Model and it automatically analyzes the origins of sentence components in the document. The output of the composition analysis program is the basis of investigation on sentence reduction and sentence combination. The program has been implemented and evaluated.

The proposed sentence reduction program can trim less important words, phrases, or clauses off an extracted sentence. It relies on three types of information to make reduction decisions: the local context of a sentence, the probabilities from a corpus of human-written sentences, and linguistic knowledge. It has been implemented and evaluated on a small corpus.

We also propose a sentence combination technique to substitute the incoherent phrases in an extracted sentence with related and appropriate phrases from nearby sentences, or to merge two sentences with related content and similar structure. The proposed algorithm is yet to be implemented and tested.

Both sentence reduction and sentence combination rely on linguistic knowledge. We built a reusable lexicon for generation by combining multiple, large-scale linguistic resources. The lexicon has been constructed and is being used in summary generation. It has also been used in traditional generation systems and the word sense disambiguation task.

The ultimate goal of this work is to build a general generation tool which, upon receiving extracted document sentences, can automatically transform them into concise and coherent summaries. It can be plugged into any single-document, generic summarizer which is in need of a generation component.

The next chapter discusses related work in summarization and generation. Then we introduce the cut-and-paste approach by example and present the system architecture. Chapter 4 describes the corpus analysis of human-written summaries and presents an algorithm to analyze the composition of a summary sentence. In chapter 5, we discuss how to remove extraneous material from a sentence so that the remaining sentence is more concise and coherent. In chapter 6, we discuss how to combine a sentence with related sentences so that the resulting sentence is more coherent and informative. Chapter 7 introduces the construction of a reusable, large-scale lexicon for generation and its application in summary generation. The planned work is presented in Chapter 8. In the conclusion, we present contributions of the proposed work.

Chapter 2

Related Work

2.1 Related work in summarization

A summary is defined as a text that retains some essential qualities of the original text but reduces it in complexity and length. The types of summaries are usually defined as following:

- *Indicative vs. informative*: depending on whether a summary contains preliminary information for quick categorization or contains the main content of the document.
- *Generic vs. query-based*: depending on whether a summary is on the central subject matter of the document or on special matter related to a user's query.
- *Single-document vs. multi-document*: depending on whether the input is a single document or multiple documents.

Summarization methods can be roughly grouped into three categories: statistical, knowledge-based, and general linguistic. The statistical approach summarizes without understanding; it relies on the statistical distribution of certain features. Recent representative works include the classification-based method [Kupiec et al., 1995], the IR-based method [Salton et al., 1994], the position-based method [Lin and Hovy, 199 among others. The statistical approach is domain-independent and fast, but has an inherent upper bound of performance.

The knowledge-based approach stands in direct contrast to the statistical approach. This approach interprets the text using extensive domain knowledge as well as natural language techniques and then summarizes it. Representative works include the summarization of multiple news articles in the terrorist domain [Radev, 1998], the analysis of technical papers [Paice and Jones, 1993], and other works like [Jacobs and Rau, 1990, Young and Hayes, 1985]. This approach is usually used in a specific domain. Compared to the statistical approach, it is more expensive but produces better quality summaries for a specific domain.

The general linguistic approach also relies on natural language processing techniques. It differs from the knowledge-based approach in the aspect of dependence on domain knowledge. The general linguistic approach works in a general domain and does not rely on domain knowledge. The understanding of text, however, is still shallow at the current stage. It usually relies on lexical information or discourse markers [Rau et al., 1989, Gladwin et al., 1991, Spark Jones, 1993, Johnson et al., 1993, Ono et al., 1994, Marcu, 1997, Barzilay and Elhadad, 1997, Mani and Bloedorn, 1997, Aone et al., 1997]. To understand the document at a semantic level remains a challenge.

Only a few works addressed the problem of generation for summarization. [Radev, 1998] studied generating summaries over multiple documents in the terrorist domain, using the outputs from an Information Extraction system. Our proposed work is different in that we focus on single-document, domain-independent summary generation, and we do not use the traditional generation architecture as he did.

Other researchers [Hirst et al., 1997] proposed a sentence transformation approach for generating HealthDoc summaries. The idea is to select material from pre-existing text and then post-edit it to recover coherence and cohesion. They posit four repair modules for post-editing: co-reference, rhetorical relations, aggregation, and constituent ordering. The sentence transformer is in the same spirit as our cut-and-paste approach in that both methods aim to reuse the original document for generation. However, the sentence reduction and combination techniques we propose are quite different from the four repair modules in the above work.

2.2 Related work in generation

A generation system generally consists of 3 modules: a content planner, a sentence planner, and a surface realizer. The input to a generation system contains semantic information to be conveyed. The semantic input is passed through each module in turn and augmented with more detailed information at each stage. Our proposed work does not follow this general architecture. We take advantage of the fact that the input to a summarization system is text and reuse the text in generation.

In contrast to summarizing texts, some generation systems, including STREAK [Robin, 1994], PlanDOC [McKeown et al., 1994], FLOWDOC [Passonneau et al., 1996], and ZEDDOC [Passonneau et al., 1997], generate summaries from databases. A representative system, STREAK, adopted a revision-based method for generating basketball reports. The input to the system is a database including historical records of basketball games and players. The summary generated is in the same style as newswire basketball reports. It follows the traditional generation architecture as we have described above. Although our goal is also to generate summaries, our inputs are documents rather than databases; hence, the method to tackle the problem is also different.

We use certain aggregation [Shaw, 1995, Dalianis and Hovy, 1993] techniques in the sentence combination module. Aggregation combines related information to generate concise and fluent sentences. For example, the propositions *Mary likes singing* and *Mary likes dancing* can be realized in a single sentence *Mary likes singing and dancing* through aggregation. We use some of the aggregation operators identified in previous work for combining sentences.

Chapter 3

The Cut-and-Paste Approach

3.1 Introduction

The main idea of the cut-and-paste approach is to generate summaries by reusing the original document, that is, we extract phrases or sentences from the original document and edit them for producing summaries. We choose this approach because this is the fastest, most reliable and a plausible way for generation in domain-independent, single-document summarization. Professional summarizers also use this approach for producing summaries. We show the major operations involved in the cut-and-paste procedure by analyzing a sample human-written summary.

Figure 3.1 shows a sample human-written summary and Figure 3.2 shows the original full article from the New York Times. The sentences have been labeled with numbers for easy reference. As we can see, the sample summary was built almost totally by reusing the original input text. We manually analyzed a set of summaries and found the major operations involved in the cut-and-paste procedure by humans, as shown in the following:

- (1)**sentence reduction**

  This is a frequently used technique. Through this method, humans select a single sentence from the document, remove less important material from the sentence, and then use the reduced sentence in the summary. For example, sentence (4) in Figure 3.1 is based on sentence (5) in the original document, but the to-infinitive adjunct is removed.

  The deleted material can be at any granularity: a word, a phrase, or a clause. Multiple components can be removed, as shown in the following example.

  > Document sentence: *James Rittinger,* **an attorney for the company pointed out that** *several west features such as* **syllabuses and headnotes still can't be** *legally* **copied.**
  > Summary sentence: **An attorney for the company noted that syllabuses and headnotes of West still cannot be copied.**

- (2)**sentence combination**

  Humans also generate a summary sentence by combining material from several sentences in the document. For example, sentence (1) in Figure 3.1 is combined from sentences (1), (2) and (3) in the original document. This is another frequently used technique. It can be used together with sentence reduction, as shown in the following example, which also uses paraphrases.

  > Text sentence 1: **But it also raises serious questions about the privacy of such highly personal information** *wafting about the digital world.*
  >
  > Text sentence 2: **The issue thus fits squarely into the broader debate about privacy and security on the internet**, *whether it involves protecting credit card number or keeping children from offensive information.*
  >
  > Summary sentence: **But it also raises the issue of privacy of such personal information** *and* **this issue hits the head on the nail in the broader debate about privacy and security on the internet.**

SENATE PANEL DEBATES CHILDREN'S ONLINE PRIVACY

(1) Arthur B. Sackler, vice president for law and public policy of Time Warner Inc. and a member of the Direct Marketing Association, told the communications subcommittee of the Senate Commerce Committee that legislation to protect children's privacy online could destroy the spontaneous nature that makes the Internet unique. **(2)**"For example, a child's e-mail address is necessary in order to respond to inquiries such as updates on Mark McGuire's and Sammy Sosa's home run figures this year or updates of an online magazine," said Mr. Sackler. "In this context, a standard of reasonable efforts to provide parental notification, rather than obtaining parental consent, is the only standard that will work if the legislation is not to eliminate all spontaneous interactivity for children." **(3)**The subcommittee is considering the Children's Online Privacy Protection Act <http://thomas.loc.gov/cgi-bin/bdquery/z?d105:s.02326:>, which was drafted on the recommendation of the Federal Trade Commission. **(4)**The bill would require the FTC to right privacy rules for commercial web sites.

Figure 3.1: A sample human-written summary

TITLE: SENATE PANEL DEBATES CHILDREN'S ONLINE PRIVACY

WASHINGTON – **(1)**A proposed new law that would require Web publishers to obtain parental consent before collecting personal information from children could destroy the spontaneous nature that makes the Internet unique, a member of the Direct Marketing Association told a Senate panel Thursday.

(2) Arthur B. Sackler, vice president for law and public policy of Time Warner Inc., said the association supported efforts to protect children online, but he urged lawmakers to find some middle ground that also allows for interactivity on the Internet.

(3)"For example, a child's e-mail address is necessary in order to respond to inquiries such as updates on Mark McGuire's and Sammy Sosa's home run figures this year or updates of an online magazine," Sackler said in testimony to the communications subcommittee of the Senate Commerce Committee. "In this context, a standard of reasonable efforts to provide parental notification, rather than obtaining parental consent, is the only standard that will work if the legislation is not to eliminate all spontaneous interactivity for children."

(4) The subcommittee is considering the Children's Online Privacy Protection Act, which was drafted on the recommendation of the Federal Trade Commission. **(5)**The bill would require the FTC to prescribe rules for commercial Web sites to follow when collecting personal information from children.

(6) The rules, to be developed within a year, would require Web publishers to ...

(7)... ...

SOURCE: New York Times (CyberTimes), AUTHOR: Jeri Clausing
http://www.nytimes.com/library/tech/98/09/cyber/articles/24privacy.html

Figure 3.2: The full document for the sample summary

- (3) **syntactic transformation**

  In both sentence reduction and sentence combination, syntactic transformations may be involved. For example, the phrase *Sackler said* in sentence (2) of the document is changed to *said Sackler* in the summary during sentence reduction. A more complicated example is sentence (1) in the summary. The structure of this combined sentence is based on that of the first sentence in the document, but the position of the subject has been moved.

- (4) **lexical paraphrasing**

  Humans may borrow a block of texts from the original document and then replace certain phrases with their paraphrases. For example, they replaced *point out* by *note*, or *fits squarely into* by a more picturesque description *hits the head on the nail* as in previous examples.

- (5) **generalization**

  Similarly, humans may borrow a block of text from the document and then replace certain phrases or clauses with a more generalized high-level description. For example, they replaced *a proposed new law that would require Web publishers to obtain parental consent before collecting personal information from children* with *legislation to protect children's privacy on-line.*

- (6) **reordering**

  The borrowed sentences from the document do not necessarily retain their precedence orders in the summary. For example, a concluding sentence in the document may be placed at the beginning of a summary as an opening sentence.

To generate a single sentence, multiple operations from the above can be involved. For instance, sentence (4) in the sample summary is by sentence reduction and lexical paraphrasing, and sentence (1) in the summary is by sentence combination, syntactic transformation, and generalization.

Our work focuses on sentence reduction and sentence combination since they are the two most important and most frequently used techniques by humans, and they are very effective in producing concise, coherent sentences.

3.2 Cut-and-paste in professional summarizing

The cut-and-paste approach is a standard technique for text production by expert summarizers. Six experts agreed in a psycholinguistic study [Endres-Niggemeyer and Neugebauer, 1995] that summary production is usually by selecting texts from the original document, putting them together, and adapting them to the new surroundings. Another linguistic study [Fries, 1997] manually analyzed 100 summaries on the front page of the London *Times*. It concluded that the words in a headline are usually abbreviated forms of those in the summary and the summaries themselves consist of abbreviated forms in comparison to the full text. Ten types of abbreviations were identified, including the omission of time phrases, the possible deletion of prepositions and adverbs, and the reduction of relative clauses to verb phrases.

3.3 System architecture

The summarization task we consider is domain-independent, single-document, generic summarization compared to domain-dependent, multi-document, or query-based summarization.

The generation component is designed to be a portable module for any single-document, generic summarizer in need of a generation component. After the extraction module determines the most important material in the document, it sends such information to the generation module by giving a list of selected important sentences. The generation module takes the extracted sentences and also the original document as input. It then edits the selected sentences, reducing it or combining it with other sentences. The modified sentences constitute the final summary.

Appendix A.1 shows the system architecture. The generation component consists of a control module, individual operation modules, a paste module, and a surface repair module. The control module decides what operation(s) to take, and each operation module realizes a specific operation, such as sentence reduction. The paste module rebuilds a sentence based on the decisions made by the operation modules. Finally, the surface repair module makes minor surface changes that might be necessary due to sentence reorganization, for example, adding/removing determiners or changing singular/plural of nouns.

The generator relies on a corpus of human-written summaries, the WordNet lexical database, and a reusable lexicon we combined from multiple, large-scale resources [Jing and McKeown, 1998]. It also uses general NLP tools such as syntactic parser, anaphora resolving, and co-reference resolving tool.

Chapter 4

Corpus Analysis

4.1 The need for corpus analysis

No prior research can show us how the cut-and-paste should be carried out in summarization. Are there any rules in this process? If so, what are they? When and how should they be applied? Are there only a limited number of frequently used rules, or are there many rules depending on various conditions? The only way to find the answers is to analyze human-written summaries.

Our experience indicates that manual analysis of a small number of documents is not sufficient to answer the above questions; the discovery of the answers relies on large-scale, automatic analysis. After manually examining 15 human-written summaries by analyzing the construction of each sentence, we concluded that humans make different decisions under different circumstances. The diversity of both the decisions and the circumstances under which the decisions are made makes it impossible to generalize any rules by a small-scale manual analysis. Thus we propose a corpus-based method for investigating the cut-and-paste technique.

4.2 Corpus collection and cleaning

The corpus we need should consist of human-written summaries as well as their original documents. We also require the documents to cover broad topics and not to possess a particular writing style, given that the goal of this research is to develop domain-independent techniques.

The corpus was collected from *Communications-related Headlines*, a free daily online news service provided by the Benton Foundation (http://www.benton.org). The articles are communication related, but the topics involved are very broad, including law, company mergers, new technologies, labor issues and so on. The articles do not possess a particular writing style since they come from various sections of various newspapers.

The corpus collection and cleaning include four major steps. First, we process the daily email sent by the service to extract human-written summaries. We also separate single-document based summaries from multiple-document summaries. Second, we retrieve the original documents from the Internet. Then we clean the retrieved documents, removing irrelevant material such as indexes and advertisements. Finally, we check the correspondence between a summary and the original document. The cleaning and verification are done manually.

4.3 The Decomposition of summary sentences

4.3.1 The problem

To learn the cut-and-paste rules from human-written summaries, we must understand the composition of each summary sentence. Is a summary sentence constructed by cut-and-paste? If so, what sentence components come from the document? Where exactly in the document do they come from? We call this *the summary sentence decomposition problem*. Only after this problem is solved can we analyze large corpora of human-written summaries to learn how to cut-and-paste.

The decomposition is a difficult problem. The sentence components coming from the original document can be at any granularity. Therefore, identifying the boundary of a component is a complex issue. Determining the origin of a component is also a hard problem since the component may occur multiple times in the document, and it may also appear in a slightly changed form.

4.3.2 The proposed solution

Given this difficult problem, our solution hinges on the novel approach of reducing it to a problem of finding for each word in a summary sentence a document position that it most likely comes from. Based on the human practice of cutting and pasting, a set of general heuristic rules are produced. We use these heuristic rules to create a Hidden Markov Model (HMM) [Ross, 1988, Allen, 1995]. The Viterbi algorithm [Viterbi, 1967, Allen, 1995] is used to efficiently find the most likely document position for each word in a summary sentence.

4.3.2.1 Formulating the problem

We first mathematically formulate the summary sentence decomposition problem. An input summary sentence can be represented by a word sequence: $(W_1, ..., W_N)$ where W_1 is the first word of the sentence, ..., and W_N is the last.

The position of a word in a document can be uniquely identified by the sentence position and the word position within the sentence: $(SNUM, WNUM)$. For example, $(4, 8)$ uniquely refers to the 8th word in the 4th sentence.

Multiple occurrences of a word in the document can be represented by a list of word positions: $\{(SNUM_1, WNUM_1), ...(SNUM_m, WNUM_m)\}$.

Using the above notations, we formulate the decomposition problem as follows: *Given a word sequence $(W_1, ..., W_N)$ and for each word in the sequence, its position(s) in the original document: $\{(SNUM_1, WNUM_1), ..., (SN$ determine for each word in the sequence, its most likely document position.*

By formulating the problem as above, we reduce the difficult tasks of identifying component boundaries and determining component origins into a single, unified problem of finding a most likely document position for each word. As shown in Figure 4.1, when each word in the summary sequence chooses a position, we get a sequence of positions. For example, $\{(0, 21), (2, 40), (2, 41), (0, 31)\}$ is the position sequence we get when every summary word chooses its first occurrence of the same word in the document. $\{(0, 21), (2, 40), (2, 41), (0, 31)\}$ is another position sequence. Every time a summary word chooses a different position, we get a different position sequence. For this 4-word sequence, there are a total of 1,936 $(44 \times 1 \times 2 \times 22)$ possible position sequences.

To find a most likely document position for each word is to find the most likely position sequence among all possible position sequences. For the example in Figure 4.1, as we can see, the most likely position sequence should be $\{(2, 39), (2, 40), (2, 41), (2, 42)\}$ (that is, the fragment comes from document sentence 2 and its position within the sentence is word number 39-41). However, how can we automatically find this most likely sequence among 1,936 possible sequences?

4.3.2.2 The Hidden Markov Model (HMM)

Exactly what document position a word comes from depends on the positions of the words surrounding it. Using the bigram model, we assume that the probability a word comes from a certain position in the document only depends on the word directly before it in the sequence. Suppose W_1 and W_2 are two adjacent words in a summary sentence and W_1 is before W_2, we use $PROB(W_2 = (S_2, W_2) | W_1 = (S_1, W_1))$ to represent the probability that W_2 comes from sentence number S_2 and word number W_2 of the document when W_1 comes from sentence number S_1 and word number W_1.

To decompose a summary sentence, we must consider how humans are likely to generate it; we draw here on the operations we noted in Chapter 3. There are two general heuristic rules we can safely assume: first, humans are more likely to cut phrases for a summary than single, isolated words; second, humans are

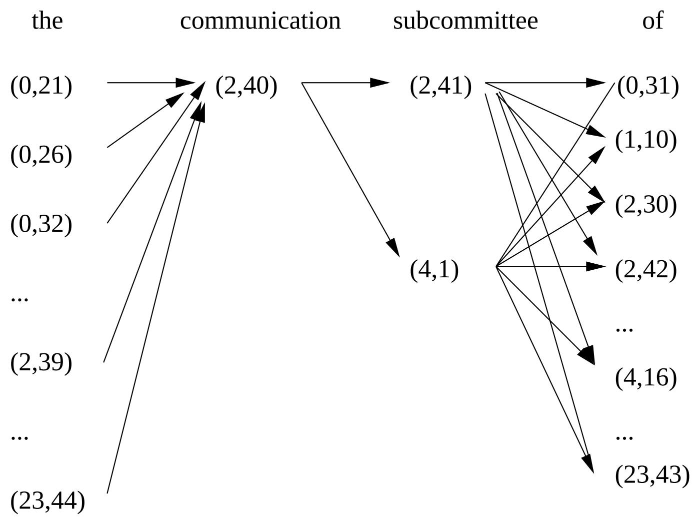

Figure 4.1: Representing all possible sequences of positions for a summary fragment

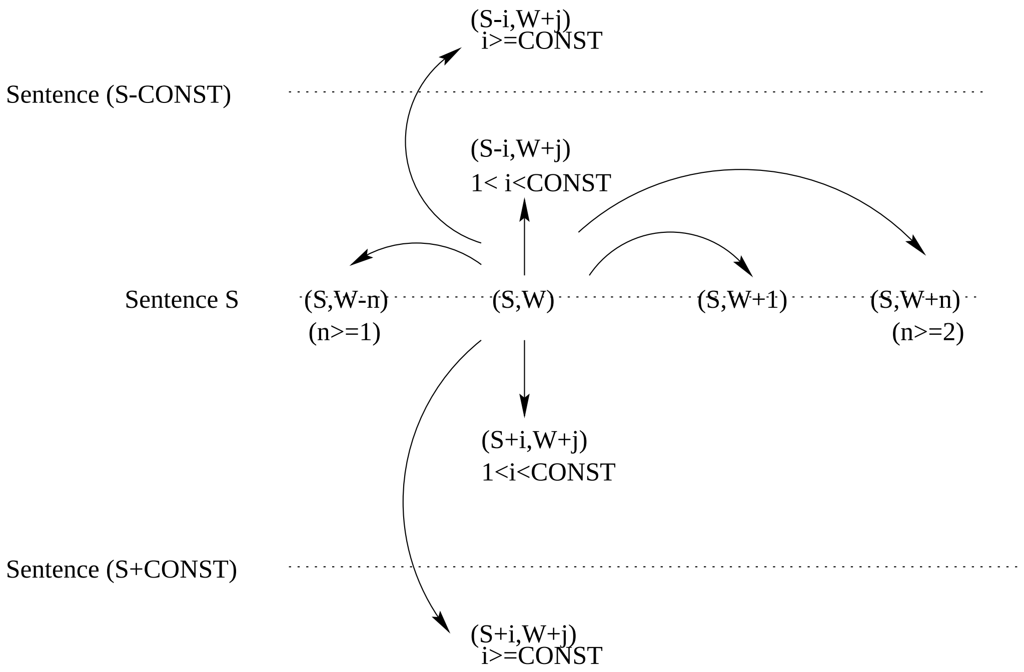

Figure 4.2: The Hidden Markov Model for sentence decomposition

more likely to combine nearby sentences into a single sentence than sentences far apart. These two rules are our guidance in the decomposition process.

We translate the heuristic rules to the above bigram probability $PROB(W_2 = (S_2, W_2)|W_1 = (S_1, W_1))$ (abbreviated as $PROB(W_2|W_1)$ in the following discussion). The probability values are assigned in the following manner:

- IF $((S_1 = S_2)and(W_1 = W_2 - 1))$ (i.e., the words are in two adjacent positions in the document), THEN $PROB(W_2|W_1)$ is assigned the maximal value P1. For example, $PROB(subcommittee = (2, 41)|communications = (2, 40))$ in the example of Figure 4.1 will be assigned the maximal value.(Rule: Two adjacent words in the summary are most likely to come from two adjacent words in the document.)
- IF $((S_1 = S_2)and(W_1 < W_2 - 1))$, THEN $PROB(W_2|W_1)$ is assigned the second highest value P2.

For example, $PROB(of = (4,16)|subcommittee = (4,1))$ will be assigned a high probability. (Rule: Adjacent words in the summary are highly likely to come from the same sentence in the document, retaining their relative precedent relation, as in sentence reduction. This rule can be further refined by adding restrictions on distance between words.)

- IF $((S_1 = S_2) and (W_1 > W_2))$, THEN $PROB(W_2|W_1)$ is assigned the third highest value P3. For example, $PROB(of = (2,30)|subcommittee = (2,41))$. (Rule: Adjacent words in the summary are likely to come from the same sentence in the document but reverse their relative orders, such as in the case of sentence reduction with syntactic transformations)
- IF $(S_2 - CONST < S_1 < S_2)$, THEN $PROB(W_2|W_1)$ is assigned the fourth highest value P4. For example, $PROB(of = (3,5)|subcommittee = (2,41))$. (Rule: Adjacent words in the summary can come from nearby sentences in the document and retain their relative order, such as in sentence combination. CONST is a small constant such as 3 or 5.)
- IF $(S_2 < S_1 < S2 + CONST)$, THEN $PROB(W_2|W_1)$ is assigned the fifth highest value P5. For example, $PROB(of = (1,10)|subcommittee = (2,41))$. (Rule: Adjacent words in the summary can come from nearby sentences in the document but reverse their relative orders.)
- IF $(|S_2 - S_1| >= CONST)$ THEN $PROB(W_2|W_1)$ is assigned a small value P6. For example, $PROB(of = (23,43)|subcommitte = (2,41))$. (Rule: Adjacent words in the summary are not very likely to come from sentences far apart.)

Based on the above principles, we create a Hidden Markov Model as shown in Figure 4.2. The nodes in the figure represent possible positions in the document, and the edges output the probability of going from one node to another. This HMM is used in finding the most likely position sequence in the next step. Assigning values to P1-P6 is experimental. The maximal value is assigned 1 and others are assigned evenly decreasing values 0.9, 0.8 and so on. We determined the orders of the above rules based on the observations on a set of summaries. These values, however, can be adjusted or even trained for different corpora.

4.3.2.3 The Viterbi Algorithm

To find the most likely sequence is to find a sequence of positions which maximizes the probability $PROB(W_1, ..., W_N)$. Using the bigram model, it can be approximated as:

$PROB(W_1, ..., W_N) = \prod_{i=1}^{N} PROB(W_i|W_{i-1})$

$PROB(W_i|W_{i-1})$ has been assigned in HMM. Therefore we have all the information needed to solve the problem. We use the Viterbi Algorithm to find the most likely sequence. For a N-word sequence, supposing each word has a document frequency of M, the Viterbi Algorithm is guaranteed to find the most likely sequence using $k \times N \times M^2$ steps, for some constant k, compared to M^N for the brute force search algorithm.

The Viterbi Algorithm [Viterbi, 1967, Allen, 1995] finds the most likely sequence incrementally. It first finds the most likely sequence for $(W_1 W_2)$, for each possible position of W_2. This information is then used to compute the most likely sequence for $(W_1 W_2 W_3)$, for each possible position of W_3. The process repeats until all the words in the sequence have been considered.

The pseudo-algorithm is shown in Appendix B.1. We slightly revised the original algorithm for our application. In the initialization step, equal chance is assumed for each possible document position for the first word in the sequence. In the iteration step, we take special measures to handle the case when a summary word does not appear in the document (thus has an empty position list). The record for previous entry is copied to the current entry so that the computation continues. In the sequence identification step, we add a revision step at the end to handle the words which do not exist in the original document.

4.3.2.4 An example

Figure 4.3 shows a sample output. The output at each step is shown in Appendix C. The components in the summary are tagged as *(FNUM:SNUM actual-text)*, where *FNUM* is the sequential number of the component and *SNUM* is the number of the document sentence where the component comes from. SNUM = -1 means that the component does not come from the original document. The borrowed components are tagged as *(FNUM actual-text)* in the document sentences.

As we can see from this example, the program correctly identified the four document sentences which the summary sentence was combined from; it correctly divided the summary sentence into components and pinpointed the exact document origin of each component. In this example, the components that were borrowed from the document range from a single word to a long verb phrase. Certain borrowed phrases have also been syntactically transformed, such as *Children's online privacy* → *Children's privacy online.* Despite these, the program successfully decomposed the sentence.

Summary sentence:
(F0:S1 **arthur b sackler vice president for law and public policy of time warner inc**) (F1:S-1 *and*) (F2:S0 **a member of the direct marketing association told**) (F3:S2 **the communications subcommittee of the senate commerce committee**) (F4:S-1 *that legislation*) (F5:S1**to protect**) (F6:S4 **children' s**) (F7:S4 **privacy**) (F8:S4 **online**) (F9:S0 **could destroy the spontaneous nature that makes the internet unique**)

Source document sentences:
Sentence 0: a proposed new law that would require web publishers to obtain parental consent before collecting personal information from children **(F9 could destroy the spontaneous nature that makes the internet unique) (F2 a member of the direct marketing association told)** a senate panel thursday
Sentence 1: **(F0 arthur b sackler vice president for law and public policy of time warner inc)** said the association supported efforts **(F5 to protect)** children online but he urged lawmakers to find some middle ground that also allows for interactivity on the internet
Sentence 2: for example a child's e-mail address is necessary in order to respond to inquiries such as updates on mark mcguire's and sammy sosa's home run figures this year or updates of an online magazine sackler said in testimony to **(F3 the communications subcommittee of the senate commerce committee)**
Sentence 4: the subcommittee is considering the **(F6 children's) (F8 online) (F7 privacy)** protection act which was drafted on the recommendation of the federal trade commission

Figure 4.3: A sample output of corpus analysis program

4.3.3 Evaluation

We evaluated the program using the data collected by [Marcu, 1998]. 10 documents from the Ziff-Dowis corpus were selected and presented to 14 human judges, together with their human-written summaries. The human judges were instructed to extract sentences from the original document which are semantically equivalent to the summary sentences. Sentences selected by majority of human judges were collected to build an *extract* of the document.

Our program can provide a set of relevant document sentences for each summary sentence as shown in Figure 4.3. Taking the union of the selected sentences, we also build an *extract* for the document. We compared this extract with the gold standard extract based on human judgments. We achieved an average 81.5% precision,78.5% recall, and 79.1% F-measure for 10 documents. The average performance of 14 human judges are 88.8% precision, 84.4% recall, and 85.7% F-measure. [Marcu, 1998] used an IR similarity based method coupled with discourse model and reported a comparable result (78.1% F-measure) [1]. The result for the individual document is shown in Table 4.1.

Further analysis indicates that there are two types of errors by the program. The first type is that the program missed finding semantically equivalent sentences which have very different wordings. For

[1] Note that Marcu's system is very different from ours. While Marcu's system is specially developed for finding semantically equivalent extracts, our system is designed for sentence decomposition. Marcu's system processes at sentence or clause level; we process at sentence component level (a sentence component can either be a word, a phrase, or a clause; the boundaries of sentence components are automatically identified by the system)

Docno	1	2	3	4	5	6	7	8	9	10	*Average*
Prec	0.67	0.75	1	0.86	1	0.56	0.79	0.67	0.86	1	0.815
Recall	0.67	1	1	0.55	1	0.64	0.79	0.67	0.55	1	0.785
F-measure	0.67	0.86	1	0.67	1	0.6	0.79	0.67	0.67	1	0.791

Table 4.1: The evaluation result of the decomposition program

example, it failed to find the correspondence between the summary sentence *Running Higgins is much easier than installing it* and the document sentence *The program is very easy to use, although the installation procedure is somewhat complex.* This is not really an "error" of the program since it is not designed to find such paraphrases. For the sentence decomposition problem, the program only needs to indicate that the summary sentence is not produced by cutting and pasting text from the original document. The program correctly indicated it by returning no matching sentence.

The second problem is that the program may identify a irrelevant document sentence as relevant if it contains some amount of common words with the summary sentence. This typically occurs when a summary sentence is not constructed by cutting and pasting text from the document but does share some amount of common words with certain document sentences. Our post-editing steps are designed to cancel such false matchings although it does not work perfectly.

The program also demonstrates some advantages. One of them is that it can capture the duplications in the extract. The gold standard extract based on multiple human judgments is not perfect. If two paraphrases from the document are chosen by an equal number of human subjects then both will be included in the extract. This is exactly what happened in the extract of document ZF109-601-903. The program, in contrast, picked up only one of the paraphrases. However, this correct decision is penalized in the evaluation due to the mistake in the gold standard.

The program won perfect scores for 3 out of 10 documents. We checked the 3 summaries and found that their texts were largely produced by cut-and-paste, compared to other summaries which might have new sentences written by humans. This indicates that when only the decomposition task is considered, the algorithm performs very well and finishes the task successfully.

This evaluation is not a direct assessment of the decomposition program. An evaluation at sentence component level, however, requires tremendous human effort and complex evaluation models. Thus, we did not attempt it at this stage.

4.4 Corpus study

Using the decomposition program, we analyzed a corpus of 100 summaries. 100 summaries contain 543 sentences in total, ranging from 2 sentences per summary to 16 sentences per summary. The results show that 76 (14%) sentences do not have matching sentences in the document, 353 (65%) sentences match to a single sentence in the document, 108 (20%) sentences match to 2 or 3 sentences in the document, and only 6 (1%) sentences match to more than 3 sentences in the document. These results show that a significant portion(85%) of summary sentences produced by human are based on cut-and-paste. We will use sentence decomposition results for sentence reduction and sentence combination study.

4.5 Conclusion

We for the first time defined the problem of decomposing a human-written summary sentence and proposed a novel Hidden Markov Model based solution for the problem. The decomposition program can automatically determine whether a summary sentence is constructed by cutting and pasting text from the original document; it can accurately recognize components in a sentence despite the wide variety of granularities that occur; it can also pinpoint the exact document origin of a component. The algorithm is fast and straightforward. It does not need other tools such as a tagger or parser as preprocessor. It does not have complex processing steps. The evaluation shows that the program performs very well for the decomposition task.

Chapter 5

Sentence Reduction

5.1 Introduction

In this chapter, we propose a method for automatic sentence reduction to produce concise and coherent summaries. For the extracted sentence *" When it arrives sometime next year in new TV sets,* **the V-chip will give parents a** *new and potentially revolutionary* **device to block out programs they don't want their children to see.** *"*, the program automatically reduces it to *" The V-chip will give parents a device to block out programs they don't want their children to see."*. Sentence reduction is a very effective technique for producing concise and coherent summaries. By removing extraneous information, it shortens the length and stresses the main points; the deletion of extraneous material also decreases the chance of incoherence.

To emphasize the reliability and robustness of the reduction, the proposed algorithm relies on different types of knowledge, including the context of a sentence, probabilities from corpus, and linguistic characteristics. We have implemented and evaluated the system. The results show that the system works well for the reduction task – it very rarely makes fatal errors such as reducing a component that is indispensable; and the output conforms well with outputs produced by human experts. The average precision and recall are 85% and 91% respectively for an evaluation on a corpus of 103 human-written, reduction-based sentences. A larger-scale evaluation is planned in the near future.

5.2 The Proposed Algorithm

Whether a sentence component can be removed depends on many factors, including but not limited to, the syntactic role of a component, the specific context of the sentence, collocations, and the reduction rate. Moreover, the reduction is not an absolute operation; different subjects may shorten a sentence in different ways. It is a challenge to design a reliable reduction model which can take into account different factors and can also learn from human subjects is a challenge.

We propose a novel reduction algorithm, which considers the main factors in making reduction decisions and can also be trained using examples of human reduced sentences. The algorithm consists of four steps; the first three steps consider respectively the context, corpus probabilities, and linguistic constraints, and the last step combines them. The reduction program first parses extracted sentences and all subsequent steps are based on parse trees. When we mention *sentence component* in the following discussions, it means any node in a parse tree; therefore, a component can be at any granularity: a word, a phrase, or a clause, which are the units we consider for possible deletion. In the remainder of this section, we describe each step in more detail.

Step 1: Measuring context importance of components based on lexical links

To decide whether a sentence component could be removed, we must consider the specific context of the sentence. For example, we can reduce the sentence *I went swimming with Maria* to *I went swimming* if the focus is my activity, while the sentence should not be reduced if the focus is *Maria.* Therefore, we need to find a method to distinguish which components are closely related to the focus of the local context and which are not. Since the components are made up words, the problem reduces to deciding whether a word is related to the focus of the local context.

The relatedness of a word to the focus of the context is to some degree reflected by its connections with other words. Words closely related to the main topic are likely to be connected with many words by some relations, while words not so closely related to the main topic tend to be less linked. For example, if the local context is on a murder case, the word *killing* is related to the main topic, which can be indicated by its connections with many related words that may occur in the local context, for example, *shot, victim, death, die* and so on.

Based on this observation, we use a method based on lexical links to detect whether a word [1] is related to the focus of the discussion. After each word has been assigned a context importance score, we then compute a score for each component by summing up the scores of all the words in the component. We now present how the context importance score for each word is computed.

First, we link each word with its related words in local context, according to 9 types of lexical relations: repetition, inflectional variants, derivational variants, synonyms, hypernyms, antonyms, part-of, entailment (for example: *kill* $\rightarrow$ *die*), and causative (for example: *eat* $\rightarrow$ *chew*). Inflectional and derivational relations are derived from the CELEX database [CELEX, 1995], and other types of relations are extracted from WordNet. The local context we consider are the 5 sentences before and the 5 sentences after the current sentence. Figure 5.1 shows the lexical links in a sample document.

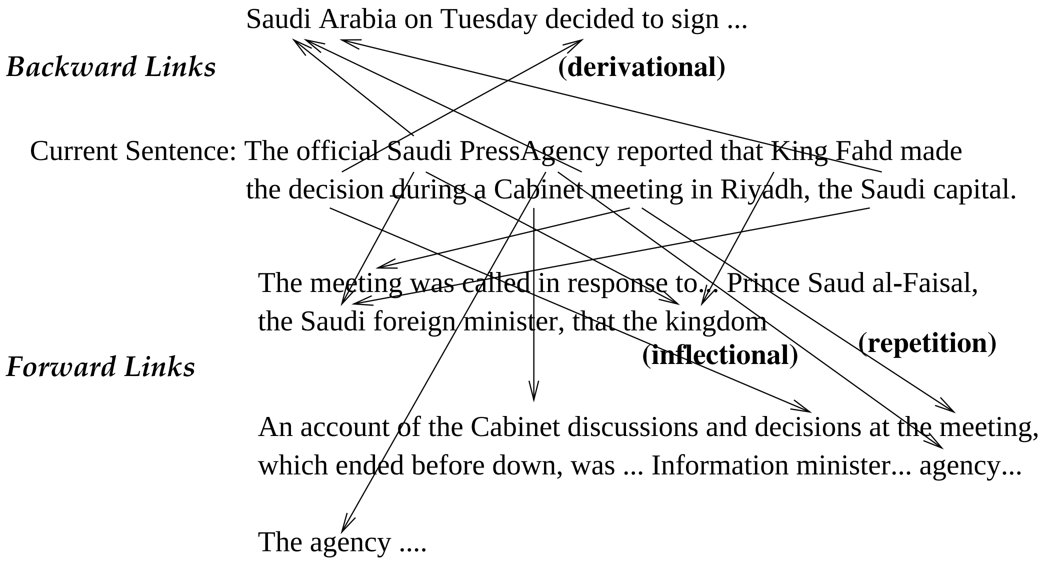

Figure 5.1: Sample lexical links

When all lexical links have been found, we compute an importance score for each word. We call all links from the current sentence to subsequent sentences *forward links* and those from the current sentence to preceeding sentences *backward links*. The score is computed in the following way:

$$SCORE(w) = Loc(w) \times (ForWeight(w) + BacWeight(w))$$

$$ForWeight(w) = \sum_{i=1}^{9}(W_i \times Fnum_i(w))$$

[1] only non-stop words are considered

$$BacWeight(w) = \sum_{i=1}^{9}(W_i \times Bnum_i(w))$$

$$Loc(w) = \frac{\max(ForWeight(w), BacWeight(w))}{(ForWeight(w) + BacWeight(w))}$$

$ForWeight(w)$ computes the weight of forward links and $BacWeight(w)$ computes the weight of backward links. To compute the weight of links, each type of link is assigned a weight, according to its importance. The 9 types of relations we listed earlier are considered in descending order of importance and are assigned decreasing weights, based on a linguistic study [Hoey, 1991]. We also take into consideration of the location of the word. When a word introduces a topic or ends a topic, it is considered more important and has a higher location value.

The reason we use various types of lexical relations is to capture the relatedness of a word to the main topic. Using only simple relations like repetition and synonymy will miss the deeper, more subtle connections between words. As a result, a component which is actually closely related to the main topic can be considered irrelevant and result in its removal. Therefore, it is important to consider many different types of lexical links. We consider only 5 sentences before and after the current sentence because this range of text is sufficient to identify the focus of the local context and also avoids a high cost in computing lexical links.

When the first step finishes, each node in the parse tree has been assigned an importance score, indicating how related the component is to the focus of the discussion.

Step 2: Introducing corpus probabilities

Although context analysis in step 1 distinguishes the components in terms of context importance, it does not give any information on interdependency of components, which is acquired in step 2 by analyzing a corpus of human reduced sentences. Sentence components are interdependent – for example, the inclusion of a verb in the reduced sentences implies that the subject of the action, and the objects, if there are any, should also be included. While this example is more related to grammar, there are other correlations which are more related to the semantics of a word. For example, if the main verb is *begin, start, or end*, the time phrase or clause probably should not be reduced. There is not, however, such strong correlation between verbs like *jump, skate, fish* and time phrases. This kind of correlation is subtle and hard to derive from existing knowledge bases; however, probabilities from a corpus of human-written summaries can indicate the degree of correlations between two components, such as a given verb and its subclause. Therefore, we use corpus probabilities to capture the interdependency of components.

The other important reason we use corpus probabilities is to learn the way humans write. As noted, reduction is not an absolute operation; while some components are essential and must be retained, some components are border-line and can be either retained or removed. To deal with such situations, we analyze examples written by experts and learn what humans usually do.

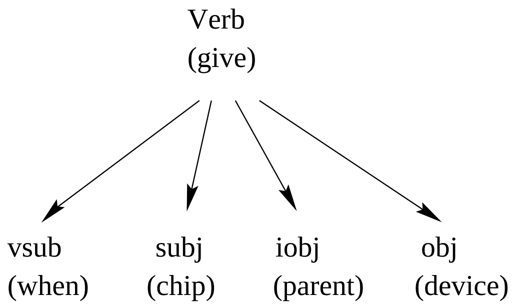

Figure 5.2: A fragment of a sample parse tree

Our probability computation is based on parse trees. We use probabilities to indicate the degree of correlation between a parent node and its child node in the parse tree. We elaborate by an example. Figure 5.2 shows a fragment of a parse tree. The main verb *give* is the parent node, and it has 4 children -

they are subclause conjunct, subject, indirect object and object respectively. To measure the interdependency between the verb *give* and its subclause, we compute the probability that the subclause is reduced when the verb is *give*, represented by $PROB(vsub[reduced]|verb = give)$. This conditional probability is transformed into the following using Bayes's rule:

$$PROB(vsub[reduced]|verb = give) = \frac{PROB(verb = give|vsub[reduced])PROB(vsub[reduced])}{PROB(verb = give)}$$

Similarly, we compute the probability for other child nodes: $PROB(subj[reduced]|verb = give)$, and so on. Afterwards, we go down one level in the parse tree and consider the previous 4 child nodes as top nodes and repeat this process, until every parent-child node pair has been considered.

Note that this corpus analysis is actually performed in advance, using a corpus of human-written, reduction-based summary sentences. The probabilities are stored and are looked up when a reduction program runs.

This corpus based step is the key module responsible for learning from the examples by human summarizers. This step enables the system to tune to a specific domain if necessary.

Step 3: Judgments based on linguistic knowledge

Context importance scores and corpus probabilities rank the relative importance of sentence components, but they do not indicate which component(s) must be present in the reduced sentence to keep it grammatically correct. Step 3 is to make such decisions using linguistic knowledge.

We use the syntactic knowledge from a rich, reusable lexicon, which we have constructed by combining multiple, large-scale resources [Jing and McKeown, 1998], for this process. In the lexicon, each verb is marked with a list of subcategorizations and alternate syntactic patterns; we use this information to decide for a node in a parse tree which of its children are essential to maintain the grammatical correctness of the component. For example, in Figure 5.2, the nodes *subj, iobj, obj* will be marked essential for the verb *give* since the lexicon indicates that *give* needs a subject and two objects; while the subclause will be marked as reducible. The lexicon also includes collocations such as *consist of* or *replace ... with ...*, which prevents us from removing indispensable components.

We represent the linguistic judgments by assigning a tag to each node in the parse tree. The value of a tag is either *essential* or *reducible*, indicating whether or not a node is indispensable to its parent node.

Step 4: Combining context, corpus, and linguistic information

In the last step, we combine the information from previous analysis and make final decisions. The reduction process can be considered to be a series of decision making steps along the edges of a parse tree. We start from the root node of the parse tree and first decide which child nodes of the root can be removed. A child node is removed if three conditions are satisfied: (1) the component is not a local focus. We compute the ratio of context importance score of the child node to that of the root node, and the child node is considered unimportant if the ratio is smaller than a threshold; (2) the corpus probability that the special syntactic component of the root is removed is higher than a threshold, for example, if the root is the verb *put* and the child is a prepositional phrase, we require that $PROB(pp[reduced]|verb = put)$ have a high value; (3) the linguistic analysis in step 3 marks the child node as *reducible*.

If a child node is decided to be removed, the processing on that branch terminates. For all the child nodes which are retained, we go down one level in the parse tree and repeat this process. The program halts when there are no more nodes to consider.

We have presented the major steps of the proposed reduction algorithm. Each of the first three steps considers a different type of information which may indicate the salience of a sentence component. These steps are independent from each other and can be used separately. All types of information are synthesized at the last stage to make final decisions.

The algorithm is robust because it combines many types of information and reduces a component only when all resources confirm that the component is reducible, and also because we use corpus probabilities for guidance. The algorithm is very flexible because it can reduce extraneous information at any granularity, such as a word or a clause; i.e., any component which corresponds to a node in the parse tree. In the next section, we show the evaluations and sample outputs of the program.

5.3 Evaluation

We use 103 human-written, reduction-based sentences in the corpus to to train and evaluate the reduction program. Sample sentences, system outputs, and human produced outputs are shown in Appendix D.1. We also run the program on documents not in the corpus, and a sample output is shown in Appendix D.2.

We evaluate the system using precision and recall measure. A human reduced sentence is considered as the gold standard. The system output is compared with this gold standard, and precision, recall, f-measure are computed using the following formulae:

$$\text{Precision} = \frac{\text{the number of words in system output and also in human output}}{\text{the total number of words in system output}}$$

$$\text{Recall} = \frac{\text{the number of words in system output and also in human output}}{\text{the total number of words in human output}}$$

$$\text{F-measure} = \frac{2 \times \text{Recall} \times \text{Precision}}{\text{Recall} + \text{Precision}}$$

The experiment includes two steps: training and testing. In the training stage, we use 50% of sentences in the corpus to compute the probabilities and also to adjust the threshold values so that the average f-measure of the training set is optimal. We then test on the other 50% of sentences and compute average precision and recall. To avoid bias in the selection of training sets, we did cross-validation. We switched 20% of the training set with 20% of the test set each time; this was repeated 5 times until the training set and the test set were completely switched. The results are shown in Table 5.1.

RunNo	1	2	3	4	5	6	*Average*
Precision	0.86	0.80	0.90	0.84	0.82	0.88	0.85
Recall	0.91	0.89	0.92	0.88	0.90	0.92	0.91
F-measure	0.88	0.84	0.91	0.86	0.86	0.90	0.88

Table 5.1: The evaluation result of the sentence reduction program

We also manually checked system output to examine whether the system makes fatal errors such as removing an indispensable component which results in an incomprehensible reduced sentence. 2 out of 103 reduced sentences have such errors: one due to a collocation missing in the lexicon and one due to lack of lexical links between context words.

These two evaluations show that the system perform well for the reduction task. We are collecting more summaries to do a larger scale evaluation and to train the system so that it is more robust.

5.4 Related work and corpus analysis

[Grefenstette, 1998] proposed a technique to produce telegraphic text reduction for blind people. The work is different from ours in that it makes reduction decisions only based on syntactic roles. For example, all subclauses are to be reduced or all adjective modifiers to be removed. Our system, in contrast, considers the context of the sentence and also considers how humans reduce sentences.

We have illustrated a 4-step sentence reduction algorithm, which considers local context, corpus statistics, and linguistic knowledge. The reason we rely on various types of information from multiple resources is because a single type of information is far from sufficient for the reduction task, according to our experiments. Using a corpus of human-written, reduction-based summary sentences, we tested whether reduction can be based on term frequency (TF), term frequency × inverse document frequency (TF×IDF), or syntactic structures alone. The results indicate that there are no correlations between those indicators and whether a component should be reduced.

Chapter 6

Sentence Combination

6.1 Corpus study

Sentence combination is another frequently used operation by expert summarizers to produce summaries. Through this method, phrases from a few sentences are appropriately combined to generate a single concise and coherent sentence. Using the automatic decomposition program on 543 sentences from 100 summaries, we found that 108 (20%) sentences were constructed by sentence combination. Of the 108 sentences, 71% sentences were combined from 2 document sentences, 24% were combined from 3 sentences, and only 5% were combined from more than 3 sentences. Based on the decomposition results, we have the following observations.

Observation 1: Existence of base-sentence(s)

We observed that when a few document sentences are combined, very often, one of the sentences will serve as a base-sentence for generating the new sentence. A base-sentence provides most of the content and structure of the new sentence, while other sentences only provide some phrases to substitute or augment a component in the base-sentence. A good example is a sentence we have used earlier in Chapter 4 (see Figure 4.3). Although the summary sentence was combined from 4 document sentences, document sentence 0 was the important base-sentence while document sentences 1, 2, and 4 contribute a phrase respectively.

Occasionally, there might be two parallel base-sentences rather than a single base-sentence, as shown in the following example, where two base-sentences are reduced and combined as conjuncts in a single sentence.

> Text sentence 1: **But it also raises serious questions about the privacy of such highly personal information** *wafting about the digital world.*
>
> Text sentence 2: **The issue thus fits squarely into the broader debate about privacy and security on the internet**, *whether it involves protecting credit card number or keeping children from offensive information.*
>
> Summary sentence: **But it also raises the issue of privacy of such personal information** *and* **this issue hits the head on the nail in the broader debate about privacy and security on the internet.**

Observation 2: Various combination methods

We found examples of various techniques that human experts use to combine sentences. We categorize them as:

(1) substitution

This is one of the most frequently used techniques in the corpus example. By substitution, a phrase in the base-sentence is replaced by a more complete/general/specific phrase cut from other sentences. For instance, replace *a Senate panel* with a more specific phrase *the Communications Subcommittee of the Senate Commerce Committee* as in the example in Figure 4.3.

The resolving of anaphora and dangling references are also accomplished by substitution. For example, replace *they* with *government lawyers*, or replace *in the country* with *in the U.S.*.

(2) augmentation

The augmentation operation cuts phrases from other sentences and adds them to an existing phrase in the base-sentence. For example, the general description *a member of the direct marketing association* was augmented and generated as *Arthur B. Sackler, vice president for law and public policy of Time Warner Inc. and a member of the direct marketing association* in the example in Figure 4.3.

(3) parallel sentences

When there are two parallel base-sentences, various techniques have been used to combine them. We find the examples of linking two sentences by adding conjunctions, adding punctuations, changing one sentence to a subclause, extracting the common subject, and so on.

Observation 3: The range of combined sentences

Most sentences which are combined are close to each other in the document, usually right next to each other or 2 or 3 sentences apart. However, sentences more than 10 sentences apart are also occasionally combined. A description more than 40 sentences away from the base sentence can also be combined.

Observation 4: The effect on conciseness and coherence

Sentence combination is very effective in improving conciseness and coherence of generated sentences. First of all, the generated sentences are significantly shortened compared to the document sentences they are combined from. Secondly, it greatly reduces incoherence by substituting and augmenting phrases or adding smooth transitions. We find sentence reduction and sentence combination are two major techniques human experts use to reduce incoherence of extracted sentences.

6.2 The proposed method

6.2.1 The algorithm

Based on the observations from the corpus, we propose an algorithm for sentence combination, which is yet to be implemented and tested. The algorithm uses corpus information, linguistic knowledge, and also some general NLP tools. Similar to sentence reduction, the inputs of the sentence combination module are the sentences that have been selected by the prior extraction module. The original document is also forwarded to and used by the sentence combination module. The outputs are combined sentences. The algorithm is shown in Appendix E.1.

Given an extracted sentence from the document, the sentence reduction program will first process the sentence and mark the components which could possibly be removed before the sentence is sent to the combination module. Upon receiving an input sentence, the combination module will assume it as a base-sentence for generating the output. The first step in the combination is to decide whether the input sentence should be the sole base-sentence for generating the sentence, or another sentence should also be introduced thus generating the output based on two parallel base-sentences. Determining single or parallel base-sentences could be based on features which we shall learn from corpus examples, the similarity of sentence structures and verb arguments, or discourse markers.

Once we have decided whether it is the single base-sentence situation or the parallel base-sentences situation, we will invoke separate modules which have been designed for each task. For the single base-sentence scenario, we will first identify all the components in the base-sentence that might cause incoherence, for example, pronouns, proper names, definite noun phrases, or any other kind of expression we may discover by training on the corpus. If the incoherent component has been marked as reducible by the reduction module, we will ignore the incoherence and remove the component. If the component is essential, however, we need to find appropriate phrases that can be used for substituting or augmenting the component. To do this, we plan to first test the general tools including a co-reference resolver, anaphora resolver, and description identifiers to see to what degree they can help in the task and what their biggest problems are. There have been many efforts in resolving co-references, especially in MUC systems. We will import a co-reference module if possible or implement a published algorithm which has reported the best result.

We will also see whether the lexical cohesion links which we have used in sentence reduction can be used to find related components. In the above process, our main concern will be accuracy, since substituting a component with a wrong reference could be very misleading and is worse than not substituting at all. After we have found the appropriate phrase, we will decide which action to take, i.e, to substitute the phrase in the original sentence or to augment it, based on the relation between the new phrase and the original phrase. This operation is performed on every possible incoherent component. The final output should be a combined sentence, which is based on the input base-sentence but with incoherent components properly substituted or augmented.

For the second scenario, where parallel base-sentences are involved, the procedure is similar to the single base-sentence scenario except that it has two additional steps. The first additional step is that after the program decides that there should be another base-sentence, it must find the other sentence, either from the extract given by the extraction or more likely from the nearby document sentences. The procedure to find the other base-sentence is quite related to the procedure to decide whether there should be another base-sentence. So they might be able to use the same knowledge, such as learning corpus examples and sentence structures. After both base-sentences are available, analysis similar to the single base-sentence scenario will be performed separately for each sentence. At the end, we will have a second additional step to decide how the two sentences should be combined, for example, whether to change one sentence to a subclause to extract common verb arguments. This step can use data from the corpus or perhaps the previous study on aggregation can also help.

Both sentence reduction and sentence combination start from the parse trees provided by the parser. They will make decisions as to which nodes in the tree should be deleted or should be moved to another tree. A paste module will perform these operations on the parse trees and generate a combined parse tree for the new sentence. A sentence repair module will fix the minor surface generation problems that might arise. Finally, the new sentence will be generated, which is based on the input sentence and possibly also some closely related sentences in the document, with the unimportant components trimmed and the incoherent components replaced.

6.2.2 An example

We show by example how the proposed algorithm works. Suppose we have the following extracted sentence from a document:

> **Input (Extracted Document Sentence):**
> The new version also replaces the vague "indecency" standard, to which the court objected, with the better-defined one of material ruled "harmful to minors."

The sentence reduction program will first mark reducible components and produce the following result:

> **After sentence reduction** *(extracted components are shown in bold and omitted components are shown in italics)*
> **The new version also replaces the vague "indecency" standard,** *to which the court objected,* **with the** *better-defined* **one of material ruled "harmful to minors."**

The program will then decide whether there is another base-sentence. The word *also* is an indicator that the sentence might be a part of continuing arguments or facts. We examine the nearby sentences in the input document and find that the two sentences right before it are quite related to it, according to co-references, lexical links or other standards. The two sentences are shown in the following:

> **Nearby sentences in the document:**
> The new measure is an echo of the original bad idea, blurred just enough to cloud prospects both for enforcement and for court review.
>
> Unlike the 1996 act, this one applies only to commercial Web sites – thus sidestepping 1996 objections to the burden such regulations would pose for museums, libraries and freewheeling conversation deemed "indecent" by somebody somewhere.

Since the second sentence has a very similar structure to the input sentence and also the same subject as the input sentence, we will choose it as the other parallel sentence. This sentence then will be sent to the reduction module, which produces the following result:

> **Reduced second base-sentence:**
> *Unlike the 1996 act,* **This one applies only to commercial Web sites** – *thus sidestepping 1996 objections to the burden such regulations would pose for museums, libraries and freewheeling conversation deemed "indecent" by somebody somewhere.*

Now the combination module has two reduced sentences as the following:

> **Reduced base-sentences**:
> base-sentence 1: this one applies only to commercial Web sites.
> base-sentence 2: The new version also replaces the vague "indecency" standard with the one of material ruled "harmful to minors."

The program will then find all the incoherent components: *this one*, *the new version*, *the vague "indecency" standard*. Suppose it can correctly identify that *this one* and *the new version* actually refers to the same entity (that is, the new law that has been proposed) and suppose it can correctly identify what *indecency standard* refers to and that it has been refered in the summary before. Thus, there is no incoherence problem if it is used as it is. Then the program would generate the following output:

> **Output of the program:**
> This new law applies only to commercial Web sites and also replaces the vague "indecency" standard with the one of material ruled "harmful to minors."

The generated sentence includes more information in less words and is also more coherent. For the reference, the following is the actual summary sentence produced by a human summarizer. It combined three sentences and added the contrast conjunctions, which is very difficult for a computer program to accomplish.

> **The sentence produced by human:**
> While the new law replaces the "indecency" standard with "harmful to minors" and now only applies to commercial Web sites, the "new measure is an echo of the original bad idea."

6.3 Evaluation

We plan to evaluate our sentence combination module by two means: corpus-based evaluation and human judgments. In the first evaluation, we will quantitatively measure the system's ability to detect incoherence, the ability to find the correct replacement phrases, the ability to determine the base-sentence(s), and the ability to find the most appropriate structure for the combined sentence, using the corpus of human-written summaries. The second evaluation is to evaluate the overall quality of the output. We will ask a number of human subjects to score the produced summaries, in terms of accuracy, fluency, conciseness, coherence, and readability. As the research goes on, we will also perform any evaluations that are necessary when new situations arise.

Chapter 7

Building Lexical Resources for Generation

7.1 The need for a rich, reusable generation lexicon

Both sentence reduction and sentence combination need syntactic and semantic knowledge to make decisions. Existing linguistic resources are not suitable to use directly since each resource provides a different type of knowledge needed in the process, but they can not be used simultaneously due to different representations and organizations. Moreover, most large-scale linguistic resources so far provide information at the surface word level, while generation usually starts from semantic concepts so an ideal generation lexicon should be at word sense level. In this chapter, we introduce our work of building a rich, reusable generation lexicon by combining multiple, large-scale resources and demonstrate its applications in sentence reduction, sentence combination, and other generation tasks.

7.2 Constructing a generation lexicon by merging linguistic resources

We combined four large-scale resources semi-automatically to build a rich, reusable lexicon containing syntactic, semantic, and lexical knowledge. The four resources are listed in the following, each of which provides a particular type of knowledge necessary for generation:

1. The WordNet on-line lexical database [Miller et al., 1990]. WordNet is the largest lexical base to date, consisting more than 120,000 words and more than 170,000 word senses. It also provides rich lexical relations between word senses, such as synonymy, hyponymy, and meronymy.
2. English Verb Classes and Alternations (EVCA) [Levin, 1993]. EVCA studies 80 alternations for over 3,000 verbs. An alternation is a variation in the realization of verb arguments. For example, the alternation "there-insertion" transforms *A ship appeared on the horizon* to *There appeared a ship on the horizon*.
3. The COMLEX syntax dictionary [Grishman et al., 1994]. COMLEX provides syntactic information for 38,000 English words, using 94 pre-defined subcategorization patterns.
4. The Brown Corpus tagged with WordNet senses [Miller et al., 1993]. We use the original Brown corpus [Kučera and Francis, 1967] to provide frequency measurements and exacting selectional constraints on verb arguments.

The detailed description of the combining process is in [Jing and McKeown, 1998]. We omit the details and summarize the work here due to space limitations. The combination includes 3 steps. First, EVCA is encoded using COMLEX style representations and combined with COMLEX; therefore, each verb is associated with a list of syntactic subcategorizations and alternations. In the second step, the above result is merged with WordNet so the syntactic information is refined to word sense level and lexical information

Word: *appear*
sense 1 give an impression
((PP-TO-INF-RS :PVAL ("to") :SO ((sb, −)) :FRE (...))
(TO-INF-RS :SO ((sb, −)) :FRE (...))
(NP-PRED-RS :SO ((sb, −)) :FRE (...))
(ADJP-PRED-RS :SO ((sb, −) (sth, −)) :FRE (...))))

sense 2 become visible
((PP-TO-INF-RS :PVAL ("to") :SO ((sb, −) (sth, −)) :FRE (...))
(INTRANS THERE-V-SUBJ :ALT there-insertion :SO ((sb, −) (sth, −)) :FRE (...)))
...

sense 8 have an outward expression
((NP-PRED-RS :SO ((sth, −)) :FRE (...))
(ADJP-PRED-RS :SO ((sb, −) (sth, −)) :FRE (...)))

(Expressions like *PP-To-INF-RS* are encoded subcategorization patterns; *:SO ((sb, −))* represents the constraints on verb arguments – it means the subject of the verb should be *somebody*; expressions like *INTRANS THERE-V-SUBJ :ALT there-insertion* represents alternate patterns of an alternation; *FRE (...)* represents the frequency of the subcategorization or the alternation in the Brown Corpus)

Figure 7.1: A sample entry in the combined lexicon

from WordNet is added. Finally, we enrich each entry with corpus frequency and constraints. A sample entry in the final lexicon is shown in Figure 7.1.

As we can see from the sample entry, the final lexicon consists of: (1) A complete list of syntactic subcategorizations for each sense of a verb; (2) A large variety of transitivity alternations for each sense of a verb; (3) Selectional constraints and frequency information derived from the corpus; (4) Rich lexical relations between lexical concepts as provided by WordNet. The combined lexicon consists of 5,676 verbs. On average, each verb has 2.5 senses and each sense is associated with 3.1 verb subcategorizations and 0.2 alternations. We show in the next section how the lexicon is used in sentence reduction and sentence combination.

7.3 Applications of the lexicon

We have used the combined lexicon in sentence reduction to provide verb subcategorization information so that we can decide which components are essential to keep the sentence grammatical and must not be removed. The alternate patterns in the lexicon are helpful in avoiding removing essential components.

The fact that the lexicon links syntactic knowledge with semantic knowledge also makes it possible to do certain inferencing. For example, suppose in the training set, we have computed the probability $Pr(pp[reduced]|v = say)$, the probability that the prepositional phrase is reduced when the main verb is *say*. Now in the test set, we find the main verb is *state* but it has not appeared in the training set before. Because *state* and *say* are synonyms in WordNet and our lexicon shows that when they are synonyms they have the same subcategorizations, we can use $Pr(pp[reduced]|v = say)$ to estimate $Pr(pp[reduced]|v = state)$ to avoid the sparse data problem. When we evaluate the sentence reduction problem on a larger corpus, we plan to test whether the above inferencing can improve performance.

The lexicon can also be used in sentence combination. For example, suppose there are two sentences: *This course demands strong programming skills* and *it also requires basic knowledge in compilers*. From the lexicon, we can find that *demand* and *require* are synonyms; therefore, we can combine them together as *This course demands strong programming skills and basic knowledge in compilers*.

We have also applied the lexicon in a practical generation system, PlanDOC, developed for Bellcore [Jing and McKeown, 1998]. [Jing et al., 1997] discusses the usage of the lexicon in word sense disambiguation. [Jing, 1998] discusses how to adapt such a general lexicon to a specific application domain.

Chapter 8

System Status, Planned Work, and Proposed Evaluation

8.1 System status

Currently, the following modules of the system have been fully implemented:

- The summary sentence decomposition module for corpus analysis, fully implemented, evaluated, and now in use.
- The sentence reduction module, fully implemented, evaluated on the corpus, but planned to be further improved.
- The generation lexicon constructed by combining resources. The lexicon is finished and has been used in the sentence reduction task and another generation system, PlanDOC.
- A lexical cohesion based summarization system. The system extracts sentences from the original document and is used as the front-end to provide input for the generation module.
- Corpus analysis. We have collected hundreds of summaries and the corpus collection is continuing. We used 100 summaries for prior experiments.

8.2 Planned work and proposed evaluation

8.2.1 Further evaluation of the sentence reduction program on a larger corpus

We plan to evaluate our sentence reduction module on a larger corpus, analyze failure cases, explore better ways to use corpus information, and more importantly, analyze what fatal errors the program makes and how to avoid them, and how to make the program robust to the errors of the parser. In the end, we hope to have a robust, fast, reliable sentence reduction module which is trainable on a corpus.

8.2.2 Developing sentence combination module

The combination module is the major remaining work. As we have proposed in Chapter 6, we will investigate how to decide single versus parallel base-sentences, how to find the other base-sentence, how to find incoherent components that need to be modified, how to find phrases that can replace or augment the incoherent components, how to combine parallel sentences, and so on. We will investigate how corpus information can be used in training or evaluating the combination module. We will also investigate how to design a robust and reliable system when the analysis result is not always reliable. There are a lot of topics involved in this research; we will prioritize the tasks based on their effects on accuracy and robustness of the system and aim to provide good solutions to the key problems.

8.2.3 The paste and surface repair module

Both sentence reduction and sentence combination are performed on parse trees; they mark which node(s) in the tree to be removed or to be moved to another tree. To regenerate a sentence from the marked parse trees, we develop a paste module which does tree reduction and merging. We plan to formulate the tree operations using the Tree Adjoining Grammar (TAG) [Joshi, 1987b, Joshi, 1987a, Joshi, 1994] formalism for the paste module. After sentence components are reduced or combined, minor changes might be needed occasionally to keep the sentence grammatical, for example, changing singular/plural of nouns or adding conjunctions like *and*. This is the work of the sentence repair module.

8.2.4 Testing in real summarization systems

After each module has been separately developed and tested, we will integrate the individual modules and test them in real summarization systems. We have developed a summarizer which extracts sentences based on lexical cohesion. We plan to link the generation component with our summarizer first and solve the problems that may arise in the integration. For example, we need to develop a control panel for the generation component to decide when to call which module. We will test the generation component in a full-fledged summarization system. The input is a document to be summarized, then the extraction module selects most important sentences, then the generator does its work, then we will examine the outputs and fix any problems that may exist in the generator. The generation component should also be able to hook up with other summarizers.

8.2.5 Testing on different corpora

We will test the system on two other corpora to evaluate its portability. One of the two corpora should be similar to the corpus we are using now; that is, a general corpus consisting of news articles on broad topics. Since our corpus is communication-related, we will use a second corpus which is also general but not communication-related to test whether the results will be significantly different. We also plan to test the system on a corpus in a specific domain, for example, a medical domain or a computer domain to investigate whether the system can be successfully used in a specific domain and what changes are needed to make it successful.

8.2.6 Extending to query-based or multi-document summarization

All the discussions in this proposal have been based on single-document, generic summarization. In the end, we will also investigate whether the methods we have proposed can be modified for query-based summarization or multi-document summarization. We forsee that the techniques can be easily modified for query-based summarization; the only modification is that we also need to consider the relatedness of a sentence component to the query while deciding which component to drop in sentence reduction and which components to merge in sentence combination.

Multi-document summarization is a more complex issue. Suppose a summarizer extracts a number of sentences from different documents to form a summary, we do not yet know whether it is sufficient to perform sentence reduction and combination on individual sentences from each document in a way similar to the single-document based approach and then place the modified sentences together, or whether a new type of incoherence will occur and we need some new methods to deal with the new situation. We will investigate whether our system can be used within a multi-document summarization system with minor revisions.

8.2.7 Packaging and delivering the system

The generation component we build is designed to be a portable module which can be easily plugged into summarization systems. Upon the completion of the work, we will package the system and share it with people in the research community. We will design easy-to-use interface functions, write utilities, and also make the system modular so that it can be easily modified if necessary. We will also write short technical documents for the system. The large-scale lexicon we have combined will also be available to other researchers.

8.2.8 Proposed evaluation

We have evaluated or proposed evaluation methods for each individual module of the system. We also plan to evaluate the overall performance of the generation component according to human subjects perception or statistical counts. The first system evaluation is to compare the qualities of summaries provided by our generation component and the summaries based solely on extraction. We will select a set of documents, run an extraction based summarizer, and get an extract for each document. We will then send those extracts to the generation component so that they will be modified through sentence reduction and sentence combination. Afterwards, both the extracts and the outputs of the generation module will be presented to human judges. The judges will read the texts and score them in terms of clarity of main points, existence of incoherence, readability and so on. Then we will compare the average scores of the generated outputs with the average scores of extracts to conclude whether the generation component has successfully made a significant improvement over the original extracts.

The second evaluation of the system will be quantitative measures based on statistical counts. We will compare the average length of the summaries produced by the generation component with the average length of the extracts to measure the reduction rate. We will also count the number of incoherent phrases in the summaries and the number of incoherent phrases which have been successfully solved or mistakenly replaced by the generation module to compute accuracy and error rate.

8.3 Schedule for completion

This dissertation should be finished by January 2000. A detailed timetable is shown in Figure 8.1.

Year	*Month*	*Implementation*	Writing/Defense
1999	Feb.10		**Propose**
	Feb.	further evaluation and improvement of sentence reduction module	
	Mar - May	sentence combination	write a paper on combination submit a journal article
	June	develop the paste and surface repair module test on real summarization systems	
	July	test on different corpora evaluation of the system	
	August	extend to query-based	
	September	package and deliver the system distribute the combined lexicon	
	October - December		write thesis
2000	January		submit to conferences **Defend**

Table 8.1: Timetable for completion

Chapter 9

Conclusion and Contributions

We have described a cut-and-paste technique to generate concise, coherent summaries fast and reliably in automatic text summarization. Specially, we study two effective techniques: sentence reduction and sentence combination. Sentence reduction removes less important material from a sentence and sentence combination merges a few sentences into a single sentence. Jointly, they generate a concise, coherent summary sentence based on extracted sentences. We have also constructed a rich, large-scale lexicon to provide syntactic, semantic, and lexical knowledge needed in reduction and combination. We have collected a large corpus for the study and developed a summary sentence decomposition program. The system is designed to be a portable generation component which can be used with any extraction based system to produce better quality summaries.

This dissertation has 4 major contributions to the state of art. First, we propose a framework for fast generating reliable summaries in a general domain by reusing input texts. We specially studied two techniques which are very effective in producing concise, coherent summaries. Second, we construct a large-scale lexicon to support the cut-and-paste generation and other generation tasks. Third, we show how a corpus can be used to study generation techniques and how they can be used to train automatic domain-independent generators. Finally, we build a general tool which can be plugged into any summarization system, serve as the generation module, and provide better quality summaries.

Appendix A

System Architecture

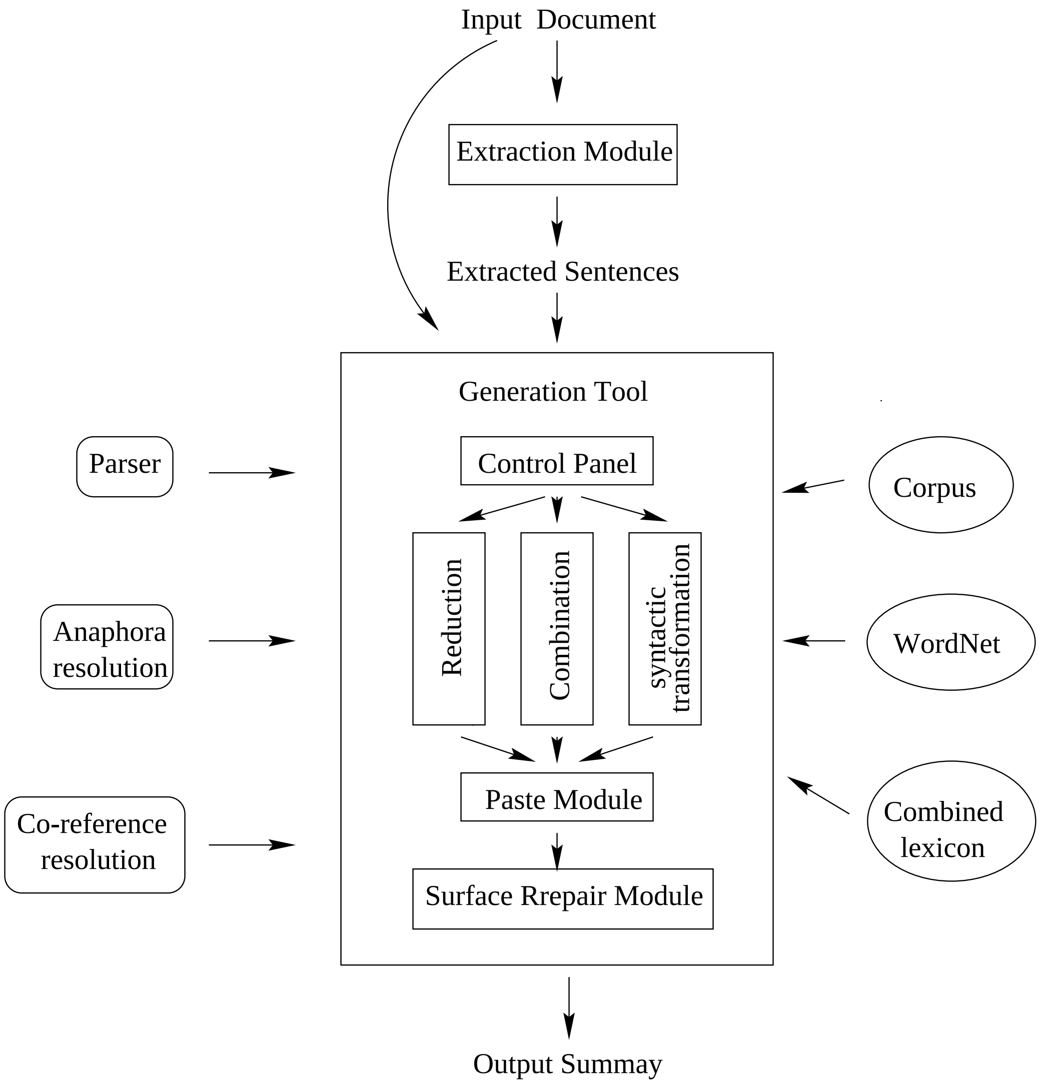

Figure A.1: The system architecture

Appendix B

Viterbi Algorithm

Problem Statement: Given word sequence $(W_1, ..., W_N)$, a list of possible document positions for each word $Wi : \{(SNUM_{i_1}, WNUM_{i_1}), ..., (SNUM_{i_{m_i}}, WNUM_{i_{m_i}})\}$, and bigram probabilities $PROB(W_i = (SNUM_i, WNUM_i)|W_{i-1} = (SNUM_{i-1}, WNUM_{i-1}))$, find the most likely sequence of positions.

Data Structures: The array SEQSCORE tracks the most likely sequence for each possible position of each word. The array BACKPTR is a backward pointer, recording the position of previous word when the most likely sequence is found.

Algorithm:

Initialization Step:
  FOR i= 1 to m_1 do
    SEQSCORE (i, 1) = 1;
    BACKPTR (i, 1) = 0;
Iteration Step:
  FOR j = 2 to N
    IF W_j occurs in the document
      FOR i= 1 to M_j
        $SEQSCORE(i, j) = \max_{k=1}^{M_{j-1}}(SEQSCORE(k, j-1) \times$
  $PROB(W_j = (SNUM_{j_i}, WNUM_{j_i})|W_{j-1} = (SNUM_{j-1_k}, WNUM_{j-1_k}))$
        BACKPTR (i, j) = index of k that gave the max
    ELSE
      FOR i=1 to Mj-1
        SEQSCORE (i, j) = SEQSCORE (i, j-1)
        BACKPTR (j, j) = j-1;
Sequence Identification Step:
  P(N) = i that maximizes SEQSCORE(i,n)
  FOR i = n-1 to 1 do
    P(i) = BACKPTR(P(i+1), i+1)
  FOR i=1 to n
    if W_i does not occur in the document
      P(i) = Φ;

Figure B.1: The Viterbi Algorithm for summary sentence decomposition

Appendix C

An Example of Sentence Decomposition

To demonstrate the program, we show an example from the beginning to the end. The sample summary sentence is as follows:

The input summary sentence (also shown in Figure C.1):
Arthur B. Sackler, vice president for law and public policy of Time Warner Inc. and a member of the Direct Marketing Association, told the communications subcommittee of the Senate Commerce Committee that legislation to protect children's privacy online could destroy the spontaneous nature that makes the Internet unique.

We first index the document, listing for each word its possible positions in the document. Stemming can be performed before indexing but is optional. Augmenting each word with its possible document positions, we therefore have the input for the Viterbi program as shown in the following:

Input to the Viterbi Program (words and its possible document positions):

arthur	:	1,0			
b	:	1,1			
sackler	:	1,2	2,34	...	15,6
...					
the	:	0,21	0,26	...	23,44
internet	:	0,27	1,39	...	18,16
unique	:	0,28			

For this 48-word sentence, there are a total of 5.08×10^{27} possible position sequences. Using the HMM in Figure 4.2, we run the Viterbi Program to find the most likely position sequence. The intermediate output of the Viterbi program is shown as follows:

Intermediate output of the Viterbi program:*(each position of each word is attached with two parameters in the bracket. The first parameter is the highest score of the sequence which ends with that position. The second parameter indicates which position the previous word should take for the sequence to achieve the highest score. For example, 0,27[0.0027,2] for the word* internet *means that for all position sequences which end with the word* internet *which takes the document position (0,27), the highest sequence score we can achieve is 0.0027; to achieve this score, the previous word* the *should come from its second possible document position, i.e., (0, 26), which is adjacent to the position of* internet.*)*

arthur	:	1,0[1,0]			
b	:	1,1[1,1]			
sackler	:	1,2[1,1]	2,34[0.6,1]	...	12,2[0.5,1]
...	:				
the	:	0,21[0.0019,1]	0,26[0.0027,1]	...	23,44[0.0014,1]
internet	:	0,27[0.0027,2]	1,39[0.0016,2]	...	18,16[0.0014,2]
unique	:	0,28[0.0027,1]			

Choosing the sequence with the highest score, we find the most likely position sequence so every word is assigned a most likely document position. After that, we differentiate the components in the sentence by combining words coming from adjacent document positions.

After the sentence components have been identified, we did some simple post-editing. It includes two steps. The first step is to cancel certain asserted matchings. The Viterbi program assigns each word in the input sequence a position in the document, once the word ever appears in the document. In this step, we examine all the identified source document sentences. If any sentence contributes only stop words for the summary, we cancel such matchings since the stop words are more likely to be inserted by humans rather than coming from the original document. Similarly, we remove document sentences providing only a single non-stop word.

The second post-editing operation is to adjust the overlapped matching. If a component in the summary is a subset of a nearby component, for example, *arthur b sackler vice president for law and public policy of time warner inc* and *and*, the short component might be matched to overlapped document positions with the longer component. In this example, *and* is matched to the *and* in the document phrase *arthur b ... law and policy ...*, which is also matched to the other component. For cases like this, we cancel the matching of the shorter component. Therefore, the matching for the word *and* will be canceled. It is actually a word inserted by a human in cut-and-paste.

Figure C.1 shows the final result. The components in the summary are tagged as *(FNUM:SNUM actual-text)*, where *FNUM* is the sequential number of the component and *SNUM* is the number of the document sentence where the component comes from. SNUM = -1 means that the component does not come from the original document. The borrowed components are tagged as *(FNUM actual-text)* in the document sentences.

As we can see from this example, the program correctly identified the four document sentences which the summary sentence was combined from; it correctly divided the summary sentence into components and pinpointed the exact document origin of each component. In this example, the components that were borrowed from the document range from a single word to a long verb phrase. Certain borrowed phrases have also been syntactically transformed, such as *Children's online privacy* → *Children's privacy online*. Despite these, the program successfully decomposed the sentence.

Summary sentence:
(F0:S1 **arthur b sackler vice president for law and public policy of time warner inc**) (F1:S-1 *and*) (F2:S0 **a member of the direct marketing association told**) (F3:S2 **the communications subcommittee of the senate commerce committee**) (F4:S-1 *that legislation*) (F5:S1**to protect**) (F6:S4 **children' s**) (F7:S4 **privacy**) (F8:S4 **online**) (F9:S0 **could destroy the spontaneous nature that makes the internet unique**)

Source document sentences:
Sentence 0: a proposed new law that would require web publishers to obtain parental consent before collecting personal information from children **(F9 could destroy the spontaneous nature that makes the internet unique) (F2 a member of the direct marketing association told)** a senate panel thursday
Sentence 1: **(F0 arthur b sackler vice president for law and public policy of time warner inc)** said the association supported efforts **(F5 to protect)** children online but he urged lawmakers to find some middle ground that also allows for interactivity on the internet
Sentence 2: for example a child's e-mail address is necessary in order to respond to inquiries such as updates on mark mcguire's and sammy sosa's home run figures this year or updates of an online magazine sackler said in testimony to **(F3 the communications subcommittee of the senate commerce committee)**
Sentence 4: the subcommittee is considering the **(F6 children's) (F8 online) (F7 privacy)** protection act which was drafted on the recommendation of the federal trade commission

Figure C.1: A sample output of corpus analysis program

Appendix D

Sample Outputs of the Sentence Reduction Program

Sentence: The company's goal is to bring radio to underdeveloped areas of the world, many of which are beyond the reach of conventional ground-based radio broadcasting.
Reduced(program): The company's goal is to bring radio to underdeveloped areas of the world.
Reduced(human): The company's goal is to bring radio to underdeveloped areas of the world.

Sentence: Consumer groups are worried that a pending decision by federal regulators eventually could lead to people paying per-minute rates for using the Internet, just as they do now with long-distance phone calls.
Reduced(program): Consumer groups are worried that a decision by federal regulators could lead to people paying per-minute rates for using the Internet.
Reduced(human): Consumer groups are worried that a pending decision could lead to people paying per-minute rates for using the Internet.

Figure D.1: Sample sentences and their reduced versions by the program and human experts

Document:
Saudi Arabia on Tuesday decided to sign the nuclear weapons non-proliferation treaty, a strong indication it will not seek nuclear warheads for intermediate-range missiles it recently acquired from China.
The official Saudi Press Agency reported that King Fahd made the decision during a Cabinet meeting in Riyadh, the Saudi capital.
The meeting was called in response to a recommendation by Prince Saud al-Faisal, the Saudi foreign minister, that the kingdom sign the international treaty against the spread of nuclear arms.
An account of the Cabinet discussions and decisions at the meeting, which ended before dawn, was issued by Information Minister Ali al-Shaer and distributed by the agency. The agency, monitored in Bahrain, did not elaborate. ...

Reduced:
Saudi Arabia decided to sign the nuclear weapons non-proliferation treaty, an indication it will not seek nuclear warheads for intermediate-range missiles.
The official Saudi Press Agency reported that King Fahd made the decision during a Cabinet meeting in the Saudi capital.
The meeting was called in response to a recommendation by Prince Saud al-Faisal, the Saudi foreign minister.
An account of the Cabinet discussions and decisions at the meeting was distributed by the agency. The agency did not elaborate. ...

Figure D.2: A sample document and its reduced version

Appendix E

The Algorithm for Sentence Combination

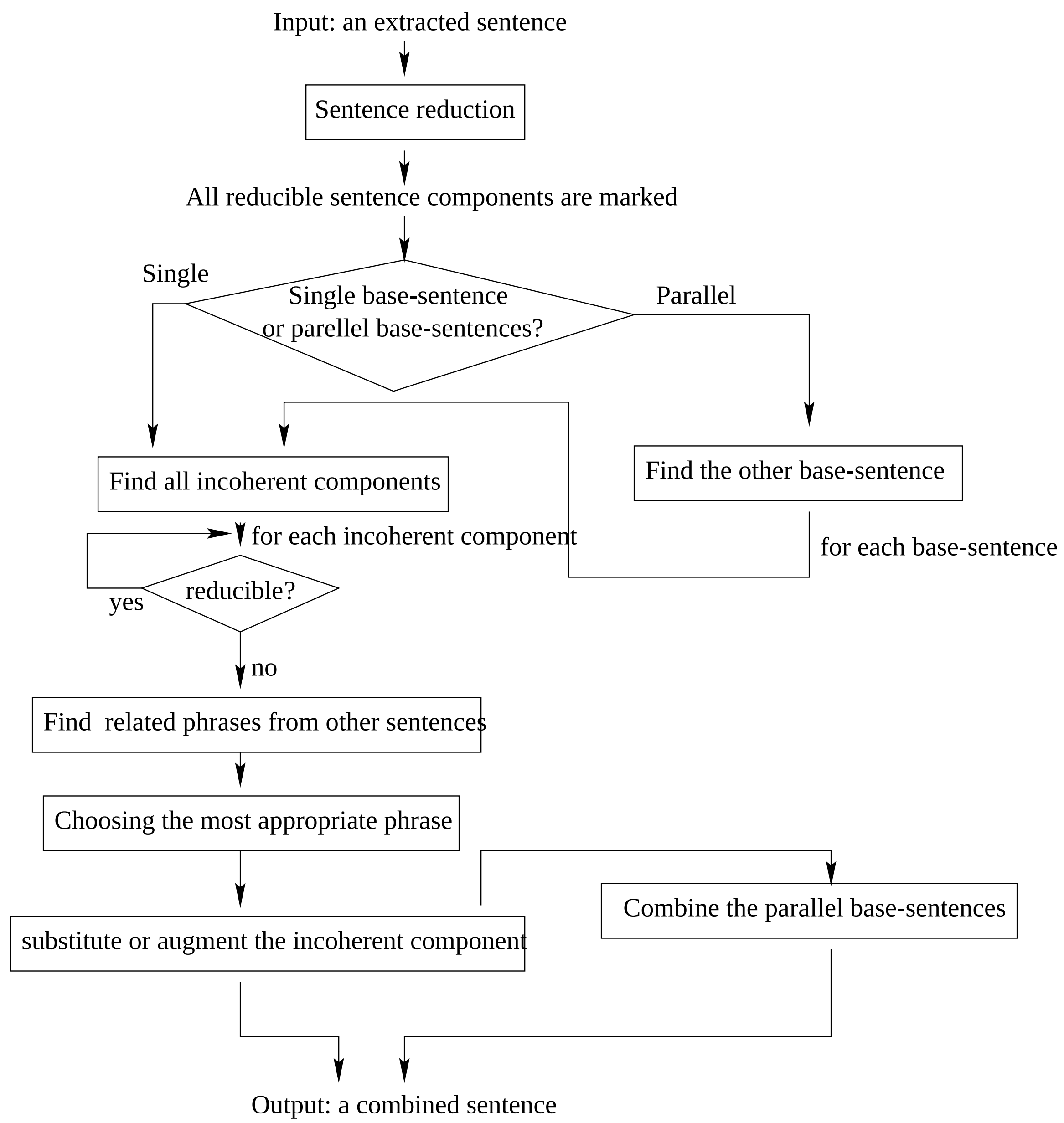

Figure E.1: The proposed algorithm for sentence combination

Bibliography

[Allen, 1995] Allen, J. (1995). *Natural Language Processing.* The Benjamin/Cummings Publishing Company, Inc.

[Aone et al., 1997] Aone, C., Okurowski, M., Gorlinsky, J., and Larsen, B. (1997). A scable summarization system using robust nlp. In *Proceedings of ACL/EACL'97 workshop on summarization*, Madrid, Spain.

[Barzilay and Elhadad, 1997] Barzilay, R. and Elhadad, M. (1997). Using lexical chains for text summarization. In *ACL/EACL-97 summarization workshop*, Madrid, Spain.

[CELEX, 1995] CELEX (1995). The CELEX lexical database—Dutch, English, German. CD-ROM. Centre for Lexical Information, Max Planck Institute for Psycholinguistics, Nijmegen.

[Dalianis and Hovy, 1993] Dalianis, H. and Hovy, E. (1993). Aggregation in natural language generation. *Proceedings of the 4th European Workshop on Natural Language Generation.*

[Elhadad, 1992] Elhadad, M. (1992). *Using Argumentation to Control Lexical Choice: A Functional Unification-Based Approach.* PhD thesis, Department of Computer Science, Columbia University.

[Endres-Niggemeyer and Neugebauer, 1995] Endres-Niggemeyer, B. and Neugebauer, E. (1995). Professional summarising: No cognitive simulation without observation. In *Proceedings of the International Conference in Cognitive Science*, San Sebastian.

[Fries, 1997] Fries, U. (1997). Summaries in newspapers: A textlinguistic investigation. In Fries, U., editor, *The Structure of Texts*, volume 3. Gunter Narr Verlag Tübingen.

[Gladwin et al., 1991] Gladwin, P., Pulman, S., and Spark Jones, K. (1991). Shallow processing and automatic summarising: a first study. Technical Report Technical Report No. 223, University of Cambridge, Computer Laboratory.

[Grefenstette, 1998] Grefenstette, G. (1998). Producing intelligent telegraphic text reduction to provide an audio scanning service for the blind. In *Working Notes of AAAI 1998 Spring Symposium on Intelligent Text Summarization*, Stanford University, Standford, California.

[Grishman et al., 1994] Grishman, R., Macleod, C., and Meyers, A. (1994). COMLEX syntax: Building a computational lexicon. In *Proceedings of COLING'94*, Kyoto, Japan.

[Hirst et al., 1997] Hirst, G., DiMarco, C., Hovy, E., and Parsons, K. (1997). Authoring and generating health-education documents that are tailored to the needs of the individual patient. In Jameson, A., Paris, C., and Tasso, C., editors, *Proceedings of the Sixth International Conference on User Modeling (UM97)*, pages 107–118, Sardinia, Italy. Springer: Vienna and New York.

[Hoey, 1991] Hoey, M. (1991). *Patterns of Lexis in Text.* Oxford University Press.

[Jacobs and Rau, 1990] Jacobs, P. and Rau, L. (1990). Scisor:extracting information from on-line news source. *Communications of the ACM*, 33(11):88–97.

[Jing, 1998] Jing, H. (1998). Applying wordnet to natural language generation. In *To appear in the Proceedings of COLING-ACL'98 workshop on the Usage of WordNet in Natural Language Processing Systems*, University of Montreal, Montreal, Canada.

[Jing et al., 1997] Jing, H., Hatzivassiloglou, V., Passonneau, R., and McKeown, K. (1997). Investigating complementary methods for verb sense pruning. In *Proceedings of ANLP'97 Lexical Semantics Workshop*, pages 58–65, Washington, D.C.

[Jing and McKeown, 1998] Jing, H. and McKeown, K. (1998). Combining multiple, large-scale resources in a reusable lexicon for natural language generation. In *Proceedings of the 36th Annual Meeting of the Association for Computational Linguistics and the 17th International Conference on Computational Linguistics*, volume *1*, pages 607–613, Université de Montréal, Quebec, Canada.

[Johnson et al., 1993] Johnson, F., Paice, C., Black, W., and Neal, A. (1993). The application of linguistic processing to automatic abstract generation. *Journal of Document Text Management*, 1(3):215–242.

[Joshi, 1987a] Joshi, A. (1987a). Introduction to tree-adjoining grammars. In Manaster-Ramis, A., editor, *Mathematics of Language*. John Benjamins, Amsterdam.

[Joshi, 1987b] Joshi, A. (1987b). The relevance of tree-adjoining grammar to generation. In Kempen, G., editor, *Natural Language Generation: New Results in Artificial Intellligence, Psychology and Linguistics*. Martinus Ninjhoff Publishers.

[Joshi, 1994] Joshi, A. (1994). Introduction to tree-adjoining grammars – preface to the special issue on tree adjoining grammars. *Computational Linguistics*.

[Kupiec et al., 1995] Kupiec, J., Pedersen, J., and Chen, F. (1995). A trainable document summarizer. In *ACM SIGIR*, pages 68–73.

[Kučera and Francis, 1967] Kučera, H. and Francis, W. N. (1967). *Computational Analysis of Present-day American English*. Brown University Press, Providence, RI.

[Levin, 1993] Levin, B. (1993). *English Verb Classes and Alternations: A Preliminary Investigation*. University of Chicago Press, Chicago, Illinois.

[Lin and Hovy, 1997] Lin, C.-Y. and Hovy, E. (1997). Identifying topics by position. In *Proceedings of the Fifth Conference on Applied Natural Language Processing*, pages 283–290, Washington, DC, USA.

[Luhn, 1958] Luhn, H. (1958). The automatic creation of literature abstracts. *IBM Journal of research and development*, 2(2).

[Mani and Bloedorn, 1997] Mani, I. and Bloedorn, E. (1997). Multi-document summarization by graph search and matching. In *In proceedings of AAAI-97*, Providence Rhode Island.

[Marcu, 1997] Marcu, D. (1997). From discourse structures to text summaries. In *Proceedings of ACL/EACL'97 Workshop on Intelligent Summarization*, pages 82–88, Madrid, Spain.

[Marcu, 1998] Marcu, D. (1998). The automatic construction of large-scale corpora for summarization research.

[McCoy, 1986] McCoy, K. F. (1986). The romper system: Responding to object-related misconceptions using perspective. In *Proceedings of the 24th Annual Meeting of the ACL*, New York City, New York. Association of Computational Linguistics.

[McKeown, 1985] McKeown, K. (1985). *Text Generation: Using Discourse Strategies and Focus Constraints to Generate Natural Language Text*. Cambridge University Press, Cambridge, England.

[McKeown et al., 1994] McKeown, K., Kukich, K., and Shaw, J. (1994). Practical issues in automatic documentation generation. In *Proceedings of the 1994 Applied Natural Language Processing Conference*, pages 7–14, Stuttgart, Germany.

[Meteer et al., 1987] Meteer, M., McDonald, D., Anderson, S., Forster, D., Gay, L., Huettner, A., and Sibun, P. (1987). Mumble-86: Design and implmentation. Technical Report COINS 87-87, University of Massachussets at Amherst, Ahmerst, Ma.

[Meteer, 1989] Meteer, M. W. (1989). The spokesman natural language generation system. Technical Report Report No. 7090, BBN Systems and Technologies Corporation, 10 Moulton Street, Cambridge, MA 02138.

[Miller et al., 1990] Miller, G. A., Beckwith, R., Fellbaum, C., Gross, D., and Miller, K. J. (1990). Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography (special issue)*, **3**(4):235–312.

[Miller et al., 1993] Miller, G. A., Leacock, C., Tengi, R., and Bunker, R. T. (1993). A semantic concordance. Cognitive Science Laboratory, Princeton University.

[Miller and Rennels, 1988] Miller, P. and Rennels, G. (1988). Prose generation from expert systems. *AI Magazine*, 9(3):37–44.

[Ono et al., 1994] Ono, K., Sumita, K., and Miike, S. (1994). Abstract generation based on rhetorical structure extraction. In *Proceedings of the 15th International Conference on Computational Linguistics (COLING-94)*, volume 1, pages 344–348, Kyoto, Japan.

[Paice and Jones, 1993] Paice, C. and Jones, A. P. (1993). The identification of important concepts in highly structured technical papers. In *Proceedings of the sixteenth annual international ACM SIGIR conference on research and development in IR*.

[Paice, 1990] Paice, C. D. (1990). Constructing literature abstracts by computer: techniques and prospects. *Information Processing and Management*, 26(1):171–186.

[Passonneau et al., 1997] Passonneau, R., Kukich, K., McKeown, K., Radev, D., and Jing, H. (1997). Summarizing web traffic: A portability exercise. Technical Report CUCS-009-97, Department of Computer Science, Columbia University.

[Passonneau et al., 1996] Passonneau, R., Kukich, K., Robin, J., Hatzivassiloglou, V., Lefkowitz, L., and Jing, H. (1996). Generating summaries of work flow diagrams. In *Proceedings of the International Conference on Natural Language Processing and Industrial Applications*, pages 204–210, New Brunswick, Canada. University of Moncton.

[Radev, 1998] Radev, D. R. (1998). *Language Reuse and Regeneration: Generating Natural Language Summaries from Multiple On-Line Sources*. PhD thesis, Department of Computer Science, Columbia University, New York.

[Rau et al., 1989] Rau, L. F., Jacobs, P., and Zernick, U. (1989). Information extraction and text summarization using linguistic knowledge acquisition. *Information Processing and Management*, 25(4):419–428.

[Robin, 1994] Robin, J. (1994). *Revision-Based Generation of Natural Language Summaries Providing Historical Background: Corpus-Based Analysis, Design, Implementation, and Evaluation*. PhD thesis, Department of Computer Science, Columbia University. Also Technical Report CU-CS-034-94.

[Ross, 1988] Ross, S. (1988). *A First Course in Probability*. Macmillan, New York, 3rd edition.

[Salton et al., 1994] Salton, G., Allan, J., Buckley, C., and Singhal, A. (1994). Automatic analysis, theme generation, and summarization of machine readable texts. *Science*, 264(5164):1421–1426.

[Shaw, 1995] Shaw, J. (1995). Conciseness through aggregation in text generation. In *Proceedings of the 33rd Annual Meeting of the Association for Computational Linguistics (Student Session)*, pages 329–331.

[Spark Jones, 1993] Spark Jones, K. (1993). Discourse modelling for automatic summarising. Technical Report Technical Report No. 290, University of Cambridge, Computer Laboratory.

[Swartout, 1983] Swartout, W. (1983). Xplain: a system for creating and explaining expert consulting systems. *Artificial Intelligence*, 21(3):285–325.

[Viterbi, 1967] Viterbi, A. (1967). Error bounds for convolution codes and an asymptotically optimal decoding algorithm. *IEEE Transactions on Information Theory*, 13:260–269.

[Young and Hayes, 1985] Young, S. R. and Hayes, P. (1985). Automatic classification and summarization of banking telexes source. In *Proceedings of the 2nd conference on Artifical Intelligence Applications[CAIA]*, pages 402–408, Miami beach, FL.