Towards Large Scale Summarization

Janara Christensen

A dissertation
submitted in partial fulfillment of the
requirements for the degree of

Doctor of Philosophy

University of Washington

2014

Reading Committee:

Mausam, Chair

Carlos Guestrin

Stephen Soderland

Program Authorized to Offer Degree:
Computer Science and Engineering

University of Washington

Abstract

Towards Large Scale Summarization

Janara Christensen

Chair of the Supervisory Committee:
Professor Mausam
Computer Science and Engineering

As the Internet grows and information is increasingly available, it is more and more difficult to understand what is most important without becoming overwhelmed by details. We need systems which can organize this information and present it in a coherent fashion. These systems should also be flexible, enabling the user to tailor the results to his or her own needs. Current solutions such as summarization are static and lack coherent organization. Even structured solutions such as timelines are inflexible. These problems become increasingly important as the size of the information grows.

I propose a new approach to scaling up summarization called *hierarchical summarization*, which emphasizes organization and flexibility. In a hierarchical summary, the top level gives the most general overview of the information, and each subsequent level gives more detail. Hierarchical summarization allows the user to understand at a high level the most important information, and then explore what is most interesting to him or her without being overwhelmed by information.

In this work, I formalize the characteristics necessary for good hierarchical summaries and provide algorithms to generate them. I perform user studies which demonstrate the value of hierarchical summaries over competing methods on datasets much larger than used for traditional summarization.

TABLE OF CONTENTS

Page

LIST OF FIGURES

Figure Number Page

LIST OF TABLES

ACKNOWLEDGMENTS

I am very grateful to Mausam for advising me throughout graduate school. I especially appreciated Mausam's encouragement to pursue problems that were most interesting to me. I'd also like to thank Stephen Soderland and Hannaneh Hajishirzi for their collaboration and ideas.

Oren Etzioni has given me excellent feedback throughout graduate school, and from him I learned the importance of identifying the best problems and the importance of ambition in research. Both Carlos Guestrin and Lucy Vanderwende were on my thesis committee and provided excellent feedback through their extensive knowledge of the background material. I also would like to thank Luke Zettlemoyer for always having time to help me, and for creating such an engaging and exciting natural language processing research environment at the University of Washington.

I was very lucky to have two amazing internships with two amazing mentors. Sumit Basu is one of the most energetic and thoughtful people I have ever worked with. Marius Pasca taught me so much about thinking through problems and working towards solutions. Thank you both for inspiring me and believing in me.

To David Musicant and David Liben-Nowell, thank you so much for the joy and beauty you brought to Computer Science, and for first inspiring and encouraging me.

I also want to thank my fellow grad students at the University of Washington, who are really what makes the University of Washington such a great place to study. To Yoav Artzi, Alan Ritter, Stefan Schoenmackers, Thomas Lin, Eunsol Choi, Chloe Kiddon, Xiao Ling, Nicholas FitzGerald, Tom Kwiatkowski, Mark Yatskar, Michael Schmitz, and Kayur Patel, thank you for your support, suggestions, ideas, and friendship. Morgan, Tony, and Abe, I am so lucky to have such funny and supportive officemates. Thank you for the coffee runs, the lunches, and the jokes that made my last year of grad school the best year of

grad school. Thanks especially to my many friends outside of CSE for your encouragement and happiness. Leianna Dixon is one of the kindest and most grounded people I have met, and one of my closest friends. To Adrienne Stilp, I have many great memories of crumpets, cooking, and West Wing. Geoff Findlay and my sister, Charlotte, were my first Seattle friends, and I very much missed the politics, dinners, and jokes after they graduated. To Ron Hause, thank you for your humor and support. Lastly, thanks to Greg Valiant for the happiness and fun he brought to my last year.

Most importantly, I owe everything to my family without whom nothing is possible. My mother, Alice, my father, Gordon, and my sister, Charlotte, have made me the person I am today and always see the best in me.

DEDICATION

for my parents, Alice and Gordon

Chapter 1
INTRODUCTION

After decades of research, why is it still so difficult to find the information one needs? Consider a user interested in a topic that encompasses a wide range of information like the 1998 embassy bombings or semantic parsing. These basic information needs are beyond the capabilities of even the most sophisticated systems.

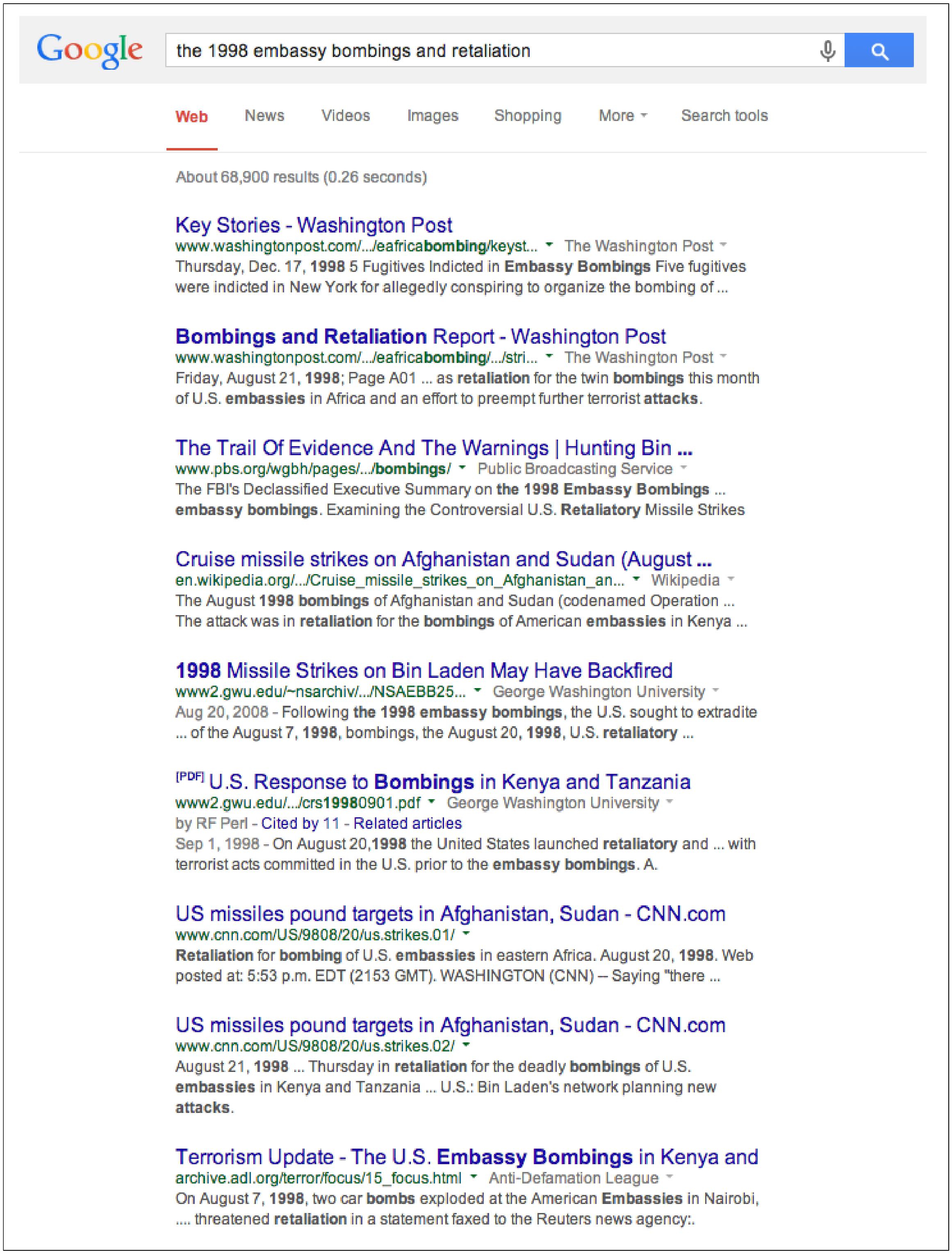
The answer for many information needs lies fragmented across pages on the Internet. The difficulty occurs in identifying the information and coalescing it into a whole, particularly when the question encompasses a sprawling, complex topic.

Search engines like Google handle these questions by returning a set of webpages (see Figure 1.1). To their credit, the answer is often contained in these pages, but the burden is on the user to read the pages, find the relevant information, and put it together. We would argue that this is just the first step of the solution – identifying the sources that contain the information. The second part of the solution is to identify the most relevant information and put it together into a coherent whole. We will focus on this second task in this work.

Other solutions to this second task do exist. Wikipedia is a manually generated encyclopedia that provides excellent overviews to some topics. However for many topics, especially those that are less common or more complicated, there are no satisfactory entries. For semantic parsing, there are no entries at all, and for multi-document summarization, the entry includes nothing on the main research techniques.

Users need an automatic solution that will gather the relevant information and collate it into a coherent and manageable summary. I refer to this problem as *large-scale summarization* because the input is potentially much larger than the traditional multi-document summarization task of 10 news articles[1], and the desired output length may be much longer

[1] DUC 2003 and DUC 2004 both used 10 documents as the input size for multi-document summarization tasks.



the 1998 embassy bombings and retaliation

About 68,900 results (0.26 seconds)

Key Stories - Washington Post
www.washingtonpost.com/.../eafricabombing/keyst... The Washington Post
Thursday, Dec. 17, **1998** 5 Fugitives Indicted in **Embassy Bombings** Five fugitives were indicted in New York for allegedly conspiring to organize the bombing of ...

Bombings and Retaliation Report - Washington Post
www.washingtonpost.com/.../eafricabombing/.../stri... The Washington Post
Friday, August 21, **1998**; Page A01 ... as **retaliation** for the twin **bombings** this month of U.S. **embassies** in Africa and an effort to preempt further terrorist **attacks**.

The Trail Of Evidence And The Warnings | Hunting Bin ...
www.pbs.org/wgbh/pages/.../bombings/ Public Broadcasting Service
The FBI's Declassified Executive Summary on **the 1998 Embassy Bombings** ... **embassy bombings**. Examining the Controversial U.S. **Retaliatory** Missile Strikes

Cruise missile strikes on Afghanistan and Sudan (August ...
en.wikipedia.org/.../Cruise_missile_strikes_on_Afghanistan_an... Wikipedia
The August **1998 bombings** of Afghanistan and Sudan (codenamed Operation ... The attack was in **retaliation** for the **bombings** of American **embassies** in Kenya ...

1998 Missile Strikes on Bin Laden May Have Backfired
www2.gwu.edu/~nsarchiv/.../NSAEBB25... George Washington University
Aug 20, 2008 - Following **the 1998 embassy bombings**, the U.S. sought to extradite ... of the August 7, **1998**, bombings, the August 20, **1998**, U.S. **retaliatory** ...

[PDF] U.S. Response to **Bombings** in Kenya and Tanzania
www2.gwu.edu/.../crs19980901.pdf George Washington University
by RF Perl - Cited by 11 - Related articles
Sep 1, 1998 - On August 20,**1998** the United States launched **retaliatory** and ... with terrorist acts committed in the U.S. prior to the **embassy bombings**. A.

US missiles pound targets in Afghanistan, Sudan - CNN.com
www.cnn.com/US/9808/20/us.strikes.01/
Retaliation for **bombing** of U.S. **embassies** in eastern Africa. August 20, **1998**. Web posted at: 5:53 p.m. EDT (2153 GMT). WASHINGTON (CNN) -- Saying "there ...

US missiles pound targets in Afghanistan, Sudan - CNN.com
www.cnn.com/US/9808/20/us.strikes.02/
August 21, **1998** ... Thursday in **retaliation** for the deadly **bombings** of U.S. **embassies** in Kenya and Tanzania ... U.S.: Bin Laden's network planning new **attacks**.

Terrorism Update - The U.S. **Embassy Bombings** in Kenya and
archive.adl.org/terror/focus/15_focus.html Anti-Defamation League
On August 7, **1998**, two car **bombs** exploded at the American **Embassies** in Nairobi, ... threatened **retaliation** in a statement faxed to the Reuters news agency:.

Figure 1.1: Results for a Google query for the 1998 embassy bombings and retaliation.

than a single paragraph.

Automatic solutions to large-scale summarization, however, lack coherence and organization. Research on multi-document summarization (MDS) has, for the most part, ignored large-scale summarization, and state-of-the-art MDS systems do not consider coherence in sentence selection, resulting in incoherent summaries. Incoherence becomes especially problematic as the length of the summary grows. Consider reading a three sentence incoherent summary versus a three page incoherent summary – the incoherence becomes more and more frustrating as the length of the summary grows.

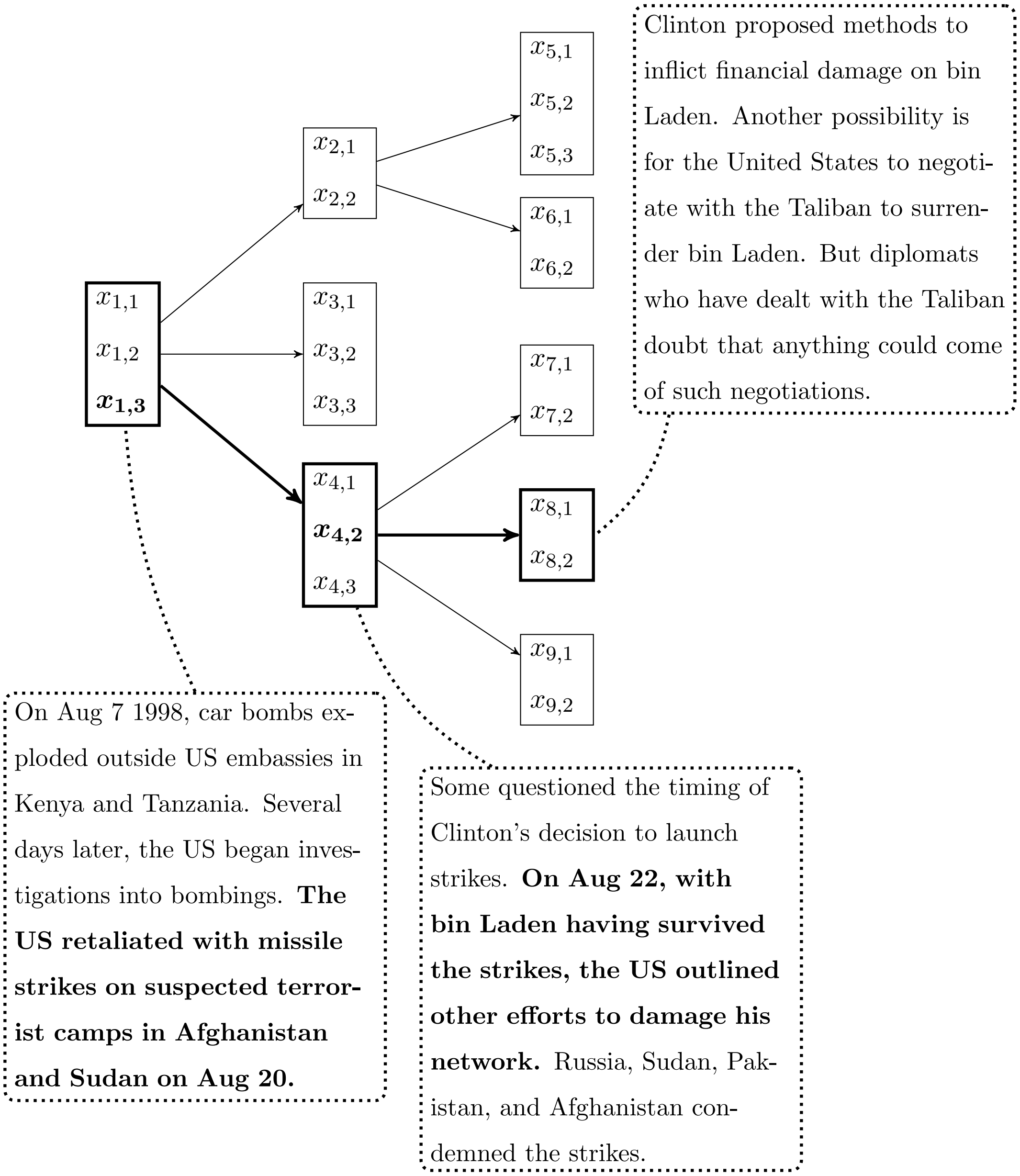
Some existing work (*e.g.* (Sauper and Barzilay, 2009)) attempts to generate structured summaries, but these solutions are incapable of collating information. Other solutions, such as timelines, have structure, but present all users with a single, inflexible summary. Different users will have different information needs, and information should be organized such that users can read a high level overview and then view more details on aspects that interest them.

1.1 Hierarchical Summaries for Large-Scale Summarization

In this thesis, I introduce an approach to large-scale summarization, called *hierarchical summarization*, which enables organized summaries (Figure 1.2). Hierarchical summarization is designed to mimic how someone with a general interest in a topic would learn about it from an expert. First, the expert would give an overview, and then more specific information about areas of interest. In hierarchical summarization, the user is first presented with a short summary of the entire topic. The user can click on sentences, and the system will present another short summary describing the sentence in question.

For example, given the topic, "1998 embassy bombings and retaliation," the overview summary might mention that the US retaliated by striking Afghanistan and Sudan. The user can click on this information to learn more about these attacks. In this way, the system can present large amounts of information without overwhelming the user, and the user can tailor the output to his or her interests.

Automatically constructed hierarchical summaries provide many advantages over the solutions enumerated above.



Clinton proposed methods to inflict financial damage on bin Laden. Another possibility is for the United States to negotiate with the Taliban to surrender bin Laden. But diplomats who have dealt with the Taliban doubt that anything could come of such negotiations.

On Aug 7 1998, car bombs exploded outside US embassies in Kenya and Tanzania. Several days later, the US began investigations into bombings. **The US retaliated with missile strikes on suspected terrorist camps in Afghanistan and Sudan on Aug 20.**

Some questioned the timing of Clinton's decision to launch strikes. **On Aug 22, with bin Laden having survived the strikes, the US outlined other efforts to damage his network.** Russia, Sudan, Pakistan, and Afghanistan condemned the strikes.

Figure 1.2: An example of a hierarchical summary for the 1998 embassy bombings, with one branch of the hierarchy highlighted. Each rectangle represents a summary and each $x_{i,j}$ represents a sentence within a summary. The root summary provides an overview of the events of August 1998. When the last sentence is selected, a more detailed summary of the missile strikes is produced, and when the middle sentence of that summary is selected, a more detailed summary bin Laden's escape is produced.

- **Coherent and collated information** Unlike the results returned by search engines, hierarchical summaries contain only information relevant to the result and the information is organized into a coherent whole.

- **Automated solution** While manually generated overviews like Wikipedia are extremely useful, they require a great deal of time and effort to generate. Automated solutions are much more desirable.

- **Organized output** Unlike traditional flat summaries, hierarchical summaries provide a structure to the information, which is particularly important as the output length grows.

- **Personalization** Hierarchical summaries allow users to explore what is most interesting to them with minimal need to read additional information. Users can personalize the output to their desires.

- **Interaction** Reading hierarchical summaries is an interactive task. Users click on sentences to learn more, and collapse summaries if they find them uninteresting.

1.2 Thesis Statement and Approach

In this thesis, I investigate how organization and coherence can enable large-scale summarization. Specifically our hypothesis is that:

(1) Systems that incorporate coherence generate summaries that humans substantially prefer. (2) Hierarchically structured summaries can effectively organize a large body of knowledge and enable interactive exploration, allowing users to focus on the aspects that interest them.

In this thesis, I introduce systems aimed at two tasks, the second task is hierarchical summarization and the first is *coherent multi-document summarization* which will serve as an intermediate step towards the goal of hierarchical summarization:

- **Task 1: Coherent Multi-Document Summarization**

  Given a set of related documents D as input, our system produces a *coherent* short summary X as output.

- **Task 2: Hierarchical Summarization**

  Given a set of related documents D as input, our system generates a hierarchical summary X of the input documents. The hierarchy arranges the sentences such that the most salient and most general sentences are at the top most levels and parent sentences are directly related to the child summaries they connect to. The summary should cover the most salient information in the documents while maintaining coherence.

In the next sections, I provide an overview of the systems I built for each of these tasks.

1.2.1 Overview of Approach – Task 1 Coherent Multi-document Summarization

Before building hierarchical summaries, I approach an intermediate task – to generate a coherent, short multi-document summary of a set of related documents. Coherence is an important subgoal to hierarchical summarization because the individual flat sumaries that make up a single hierarchical summary are connected through coherence relationships. In other words, the summary that is produced when a sentence is clicked on must be highly related to the sentence in question.

Previous work in extractive[2] multi-document summarization has taken a pipeline approach first performing sentence selection and then sentence ordering. However, if coherence is not considered when selecting sentences, it is likely that no coherent ordering exists for the selection. Instead, I introduce a system called G-Flow, which performs *joint* sentence selection and ordering. By jointly selecting and ordering, G-Flow can select a subset of sentences which are coherent together.

I identify three criteria necessary for summaries: salience, coherence, and lack of redundancy. Salience and redundancy have been studied extensively in the past. I formulate

[2] In this thesis, I examine extractive summarization only; however, our ideas are also applicable to abstractive summarization.

salience as a supervised learning task in which sentences are the instances, and I automatically label each instance with ROUGE scores. G-Flow measures the salience of a summary simply by adding together the salience of the individual sentences. G-Flow identifies redundant sentences with Open Information Extraction tuples. By looking at the tuple level, two sentences can be considered redundant if they contain overlapping information.

While salience and redundancy have been studied before, coherence requires a novel formulation. I approach coherence by identifying pairwise ordering constraints necessary for a coherent summary over the sentences in the input. Any coherent summary must necessarily obey such constraints. I represent these ordering constraints by a graph. Each sentence in the input is a node and an edge indicates that the source sentence may be placed before the destination sentence in a summary and the summary will be coherent between the two sentences. This graph is G-Flow's mechanism for measuring summary coherence.

I combine these three criteria into an objective function which maximizes salience and coherence such that there are no redundant sentences and the summary fits within the given budget. This objective function is NP Hard and is not submodular. G-Flow approximates a solution with gradient descent with random restarts.

In human evaluations, I show that people substantially prefer the summaries produced by G-Flow to those produced by other state-of-the-art systems.

1.2.2 Overview of Approach – Task 2 Coherent Multi-document Summarization

After generating coherent flat summaries, I am able to look at the more difficult problem of generating hierarchical summaries. I start by formally defining hierarchical summaries.

Definition A *hierarchical summary* H of a document collection D is a set of summaries X organized into a hierarchy. The top of the hierarchy is a summary X_1 representing all of D, and each summary X_i consists of summary units $x_{i,j}$ (*e.g.* the jth sentence of summary i) that point to a child summary, except at the leaf nodes of the hierarchy.

The summary should be organized hierarchically such that more important and more general information is at the top or root summary and less important and more specific information is at the leaves. Furthermore, the individual cluster summaries $X_i \dots X_{i+M}$ for a given level should be logically distinct. There should be a guiding organizational principle that is obvious to the reader.

To organize the information into a hierarchy, I begin by hierarchically clustering the input sentences. By clustering the sentences into topically related groups, I can then summarize over the hierarchy. Thus I split the summarization process into two steps, first hierarchically clustering, and second hierarchically summarizing over the clustering.

I create two systems to perform hierarchical summarization. The first, SUMMA, operates over news articles, and the second, SCISUMMA, operates over scientific papers. SUMMA clusters the sentences temporally, because SUMMA is focused on identifying events, which are often indicated by the burstiness of the news. SCISUMMA clusters using information from the documents as well as the citation graph.

These systems summarize over the hierarchy by once again considering salience, redundancy, and coherence. Salience is measured via a classifier trained on ROUGE scores. SUMMA uses the classifier trained for G-FLOW. For SCISUMMA, I automatically generate new training data for scientific documents by identifying related work sections that contain large numbers of citations. The sentences from cited papers are used as the input instances and the related work section is used to generate the labels (ROUGE scores).

Both SUMMA and SCISUMMA measure redundancy with a trained classifier. Because the summaries are much larger, the chance of producing two redundant sentences is higher, so a higher recall redundancy metric is necessary. The classifier uses features like word overlap, and Open Information Extraction tuple overlap.

Finally, the systems measure coherence for the hierarchical setting. They split coherence into two subtypes: parent-to-child coherence and intra-cluster coherence. Parent-to-child coherence is the coherence between a parent sentence and the child summary it leads to. SUMMA measures parent-to-child coherence with the discourse graph built for G-FLOW. And for SCISUMMA, I identify how to build a discourse graph for scientific documents. Intra-cluster coherence is a measure of the amount of confusion a user would experience when reading a sentence in the summary. I measure this coherence through missing references that have not been fulfilled before reaching the given sentence.

In human evaluations, Summa is substantially preferred over other state-of-the-art methods for large-scale summarization, and users learn at least as much. With SciSumma, users learn much more and prefer SciSumma to other state-of-the-art methods.

1.3 Contributions

My primary contributions are as follows:

- Multi-Document Discourse
  - Automatically constructing a domain-independent graph of ordering constraints over sentences in a document collection, without any manual annotation of the collection, based on syntactic cues and redundancy across documents (Chapters 3 and 6).
- Multi-document summarization
  - Formalizing a notion of coherence by using the discourse graph (Chapter 3).
  - Identifying a method for joint sentence selection and sentence ordering for traditional multi-document summarization (Chapter 3).
  - Performing human evaluations that show the value of these methods over other state-of-the-art methods for traditional multi-document summarization (Chapter 3).
- Hierarchical Summarization
  - Introducing and defining the task of hierarchical summarization (Chapter 4).
  - Adapting hierarchical summarization to the news domain (Chapter 5), and the scientific document domain (Chapter 6).
  - Formalizing the notion of coherence for hierarchical summarization (Chapters 5 and 6).
  - Providing efficient algorithms for hierarchical summarization (Chapters 5 and 6).

- Presenting a user study which demonstrates the value of hierarchical summarization for news articles over timelines and flat multi-document summaries in learning about a complex topic (Chapter 5).
- Performing a user study which shows the value of hierarchical summarization over state-of-the-art methods in learning about a new scientific research area (Chapter 6).

1.4 Outline

In the next section, I discuss background work for my dissertation. Chapter 3 discusses my work on coherent multi-document summarization. In Chapter 4, I introduce the new task of hierarchical summarization. In Chapters 5 and 6, I discuss the first systems for hierarchical summarization for the news and scientific document domains. Finally, I end with conclusions and future work in this area.

Chapter 2
BACKGROUND

Researchers have studied automatic summarization extensively, with earliest works dating back half a century ago (Luhn, 1958). This section is not intended to provide a comprehensive survey of the field. Rather, I focus on generic, extractive multi-document summarization and large-scale summarization. I reserve related works targeted at specific contributions for those chapters. For comprehensive surveys, see the following:

- Inderjeet Mani's book provides a history of the field up to 2001 (Mani, 2001).
- Dipanjan Das and André Martins' technical report on summarization is a survey of different methodologies up to 2007 (Das and Martins, 2007).
- Ani Nenkova and Kathleen McKeown's journal article surveys the field up to 2011 and describes open challenges (Nenkova and McKeown, 2011).

In the following section, I describe the scope of this dissertation and other areas of summarization research which are not directly related. This section provides context for how this work fits into the larger picture of summarization research. In Section 2.2, I discuss work in generic, extractive summarization and, in Section 2.3, work in large-scale summarization.

2.1 Problem Scope

Summarization in its broadest form takes as input a set of related articles and produces as output a summary which conveys the most important information. Summarization encompasses a wide range of tasks that vary along many dimensions including input size, output size, single document versus multi-document, extractive versus abstractive, and generic versus update or query-relevant summarization. In this dissertation, I focus on generic, extractive multi-document summarization. I consider two sizes of input and two sizes of output. The input to the problem is either small (10 documents) or larger (30-300 documents). The output size is correspondingly large: short summaries of three to five sentences or longer summaries of 30 to 50 sentences.

In the following subsections, I briefly discuss problem formulations outside the scope of this dissertation.

2.1.1 Single-Document Summarization

Early work in summarization focused on single-document summarization – summarization in which the input consists of just a single document. Single document summarization has lost popularity in recent years as researchers have moved to multi-document summarization. The task was discontinued at DUC when no automatic system could with statistical significance outperform the baseline of simply returning the beginning of the article (Nenkova, 2005).

Research in single document summarization can be roughly categorized into two approaches: machine learning methods and methods that rely on deep natural language analysis (Das and Martins, 2007). Many machine learning techniques have been employed, including naive-bayes methods (Kupiec et al., 1995), decision trees (Lin, 1999), hidden markov models (Conroy and O'leary, 2001), log-linear models (Osborne, 2002), and neural networks (Svore et al., 2007). Svore et al. (2007)'s system NetSum, based on neural nets, was the first system to outperform the baseline of returning the beginning of the article with statistical significance. NetSum used a two-layer neural net trained on sentences matched by the CNN highlights. The features include features based on word frequency, news search query logs and Wikipedia entities. Woodsend and Lapata (2010) later presented an abstractive approach which also relied on the CNN highlights.

Approaches based on natural language analysis generally rely on notions of cohension, coherence, and discourse. Ono et al. (1994) and Marcu (1998) summarized the document by first identifying the rhetorical structure of the document. Barzilay and Elhadad (1997) and Silber and McCoy (2000) used lexical chains (a chain of related words in a document) to identify summary sentences.

2.1.2 Abstractive Summarization

Abstractive summarization is an alternative to sentence extraction. In abstractive summarization, the output may be composed of entirely new sentences or of revised sentences drawn from the input documents. Because of the challenges of abstractive summarization, this area has received far less attention than extractive summarization.

A few papers have investigated extracting the most important information from sentences and forming new sentences from that information (Radev and McKeown, 1998; Genest and Lapalme, 2011, 2012). These papers often use information extraction techniques to identify specific information targeted at the category to which the document collection belongs. For example, for a 'bombing' event, the information extraction component would identify what was bombed, and who the perpetrator was. After extracting this information, a natural language generation component is used to generate the summary. While this approach often performs well for a small number of predefined examples, it is difficult to apply to arbitrary information.

Alternatively, summaries may be composed of sentences from the input that are compressed, fused, or revised. Sentence compression techniques aim to remove unnecessary information. Rule-based approaches for compression use syntactic and discourse knowledge to compress sentences (Jing, 2000; Zajic et al., 2007; Cohn and Lapata, 2008; Siddharthan et al., 2004). Alternatively, statistical approaches learn which syntactic constituents can be deleted, and therefore do not use linguistic rules (Knight and Marcu, 2002; Galley and McKeown, 2007; Turner and Charniak, 2005; Clarke and Lapata, 2010). In sentence fusion, techniques first identify themes in the source documents, select which are important for the summary, and then generate a sentence for each theme by combining the sentences in the theme (Barzilay and McKeown, 2005; Marsi and Krahmer, 2005; Filippova and Strube, 2008). Sentence revision algorithms have been developed to merge phrases, to revise references, and to correct errors (Radev and McKeown, 1997; Nenkova and McKeown, 2003; Tanaka et al., 2009). Other groups have researched extracting snippets from text (Yan et al., 2011b) or generating headlines (Banko et al., 2000; Witbrock and Mittal, 1999).

2.1.3 Query-Based Summarization

In query-based summarization, the goal is to summarize only the information that is relevant to a specific query. Researchers have roughly taken two approaches to this task: (1) adapting existing generic summarization methods and (2) formulating new techniques that are specifically adapted to the query type. The first set of approaches is based on the idea that the importance of a sentence should be judged by how important it is to the user's query and how important it is to the original set of documents. Various groups have adapted the use of topic signature words (Conroy et al., 2005), graph-based approaches (Erkan and Radev, 2004; Otterbacher et al., 2005), and submodular approaches (Lin and Bilmes, 2011). Others have proposed approaches which are specifically designed for biographical queries (Schiffman et al., 2001; Duboue et al., 2003; Zhou et al., 2004; Feng and Hovy, 2005; Biadsy et al., 2008) and definitional questions (Blair-Goldensohn et al., 2003). These papers leverage the nature of the expected output.

2.1.4 Update Summarization

Recently, the task of update summarization has gained attention. In update summarization, the summary should convey the development of an event beyond what is already known. Systems that address this problem must be able to identify novelty in addition to salience. (Toutanova et al., 2007; Dang and Owczarzak, 2008; Steinberger and Jezek, 2009). Delort and Alfonseca (2012) approached this task by modeling novelty with a variation of Latent Dirichlet Allocation. Wang and Li (2010) used incremental hierarchical clustering to model new information.

2.1.5 Intrinsic Evaluations of Automatic Summarization Methods

All automatic methods of summary evaluation require some gold standard data. If the gold standard summaries are extractive, the systems can be evaluated with precision and recall. However, this methodology does not account for sentence ordering and the gold standard summaries are unlikely to be an exhaustive list of the acceptable sentences. Furthermore, sentences are unlikely to be of equal value to the final summary. Another automatic evluation method that seeks to address some of these issues is relative utility (Radev and Tam, 2003). Relative utility requires multiple judges to score each sentence in the input according to relative value. Judges also mark which sentences should not belong in the same summary. This methodology requires substantial manual effort on the part of the judges. Neither precision and recall nor relative utility can evaluate the summaries as a whole – they operate only on the sentence level.

The Pyramid method attempts to identify the most important units of human generated summaries and measures the inclusion of those in the automatic summaries (Nenkova et al., 2007). Specifically, human summaries are analyzed to identify summary content units (SCU). Then each SCU is weighted based on the number of human summaries in which it was included. The score of an automatically generated summary is the ratio of the sum of the weights of its SCUs to the weight of an optimal summary that includes the same number of SCUs. The Pyramid method has the downside of being labor intensive and, like the other automatic methods discussed, blind to linguistic quality.

The most popular automatic metric for summary evaluation is ROUGE (Recall-Oriented Understudy for Gisting Evaluation) (Lin, 2004). ROUGE measures n-gram overlap between an automatically generated summary and one or more manually written summaries. ROUGE has parameters such as word-stemming, stop word removal, and n-gram size which can be varied according to the task at hand. ROUGE is fast and inexpensive, and is thus the method preferred by most summarization papers. However, ROUGE has no way of accounting for coherence, redundancy, or even informativeness, for which word overlap is only a rough measure.

None of the methods described thus far has any way of measuring linguistic quality. In this dissertation, I avoid reliance on automatic metrics and perform human evaluations.

2.2 Extractive Multi-Document Summarization

Researchers have investigated a wide variety of approaches for generic, extractive MDS. In this section, I survey some of the more popular approaches. A common thread through these techniques is the emphasis of salience (typically measured through coverage or centrality) and lack of redundancy.

Some of the earlier work in MDS focused on modeling content through simple word frequency-based methods (Luhn, 1958; Nenkova and Vanderwende, 2005). Nenkova and Vanderwende (2005) introduced a method which uses word frequency exclusively to generate summaries. This method originated from the observation that word frequency is a very good indicator of sentence importance. Sentences were selected according to the average probability of the words in the sentence, and the probabilities were updated after each selection, naturally controlling for redundancy. This simple method outperformed many of the summarization systems at DUC 2004.

Radev et al. (2004) introduced the use of cluster centroids for document summarization. This method first performs agglomerative clustering to group together news articles describing the same event. The centroids of these clusters are then used to identify the sentences most central to the topic of the cluster. The algorithm selects sentences by approximating their cluster-based relative utility (CBRU) and cross-sentence informational subsumption (CSIS), which are metrics for sentence relevance and redundancy respectively. These two measures resemble the two parts of MMR (Carbonell and Goldstein, 1998), but are not query-dependent.

Graph-based approaches to MDS are also popular (Radev, 2004; Wan and Yang, 2006; Qazvinian and Radev, 2008). Radev (2004) introduced a method called LexRank, which measures sentence importance by eigenvector centrality in a graph representation of sentences. The graph is composed of nodes which represent sentences and edges which represent the similarity between sentences. Another method, C-LexRank, builds upon LexRank for summarization of scientific articles (Qazvinian and Radev, 2008). C-LexRank clusters the input documents and then uses LexRank within each cluster to identify the most salient sentences.

Others have investigated the use of probabilistic topic models for representing document content (Barzilay and Lee, 2004; Daumé and Marcu, 2006; Haghighi and Vanderwende, 2009; Celikyilmaz and Hakkani-Tur, 2010; Celikyilmaz and Hakkani-Tür, 2011). Haghighi and Vanderwende (2009) described several different generative probabilistic models for multi-document summarization and reported results showing structured topic models use produces higher quality summaries.

Recently, Lin and Bilmes (2011) introduced a submodular approach to MDS which emphasizes coverage and diversity. Lin and Bilmes (2011) also observed that the objective functions of many existing extractive methods for MDS are submodular. These methods are able to achieve submodularity because they score summaries at the sentence or sub-sentence level. The methods we introduce in this thesis will not be submodular, because we score summaries as a whole. In other words, the addition of a single sentence does not satisfy the property of diminishing returns. If a sentence serves to provide context to another sentence or connects two seemingly dissimilar sentences, the value of that sentence changes radically depending on what set of sentences it is added to.

2.3 Large-Scale Summarization

Very little research has targeted large-scale summarization specifically. In this section, I discuss the most relevant related work: generating structured summaries (Section 2.3.1), generating timelines (Section 2.3.2) and identifying threads of related documents (Section 2.3.3).

2.3.1 Structured Summaries

Some research has explored generating structured summaries, primarily for Wikipedia and biographies. Approaches to generating Wikipedia articles focus on identifying the major aspects of the topic. This goal is accomplished either via a training corpus of articles in the same domain (Sauper and Barzilay, 2009), by topic modeling (Wang et al., 2009), by an entity-aspect LDA model (Li et al., 2010), or by Wikipedia templates of related topics (Yao et al., 2011). (Biadsy et al., 2008) and (Liu et al., 2010) researched biography generation. (Biadsy et al., 2008) trained a biographical sentence classifier on data generated from biography Wikipedia articles and the TDT4 corpus, and generated biographies using biographical sentences in online resourses. (Liu et al., 2010) proposed a framework called BioSnowball which used Markov Logic Networks as the underlying statistical model.

A few papers have examined the relationship between summarization and hierarchies. Buyukkokten et al. (2001) and Otterbacher et al. (2006) investigated creating a hierarchical summary of a single document. Other work has created a hierarchy of words or phrases to hierarchically organize a set of documents (Lawrie et al., 2001; Lawrie, 2003; Takahashi et al., 2007; Haghighi and Vanderwende, 2009). Lastly, there is a related thread of research on identifying the hierarchical structure of the input documents and generating a summary which prioritizes the more general information according to the hierarchical structure (Ouyang et al., 2009; Celikyilmaz and Hakkani-Tur, 2010), or spreads the summary out across the hierarchy (Yang and Wang, 2003; Wang et al., 2006).

2.3.2 Timeline Generation

Much of the research on timelines has focused on extracting a short sentence or summary for dates in a news story or collection of news story.

Work that focuses on a single article has emphasized complex temporal expressions. (Kolomiyets et al., 2012) proposed an approach where all events in a narrative are linked by partial ordering relations. They annotated children's stories with temporal dependency trees, and compared different parsing models for temporal dependency structures. (Do et al., 2012) proposed a temporal representation based on time intervals. They formalized a joint inference model that can be solved efficiently and investigated the use of event coreference for timeline construction.

Some work on generating timelines from multiple articles emphasized the summarization aspect. (Yan et al., 2011c) formalized the task of generating a timeline as an optimization problem that balanced coherence and diversity and local and global summary quality. In (Yan et al., 2011a), the authors presented a summarization-based approach to automatically generate timelines using inter-date and intra-date sentence dependencies.

Others have researched identifying the most important dates. Most work in this area has relied on the bursts of news which surrounds important dates (Swan and Allen, 2000; Chieu and Lee, 2004; Akcora et al., 2010; Hu et al., 2011; Kessler et al., 2012).

2.3.3 Document Threads

A related track of research focuses on discovering threads of related documents. While the work described in this dissertation aims to summarize collections of information, this track seeks to display relationships between documents.

Some work in this area targeted finding documents within the same storyline. Nallapati et al. (2004) examined the problem of identifying threads of events and their dependencies in a news topic through event models. They used a supervised approach and features such as temporal locality of stories for event recognition and time-ordering for capturing dependencies. Ahmed et al. (2011) proposed a hybrid clustering and topic modeling approach to group news articles into storylines. Tang and Yang (2012) presented a statistical model to detect trends (a series of events or a storyline) and topics (clusters of co-occurring words) from document streams.

Others have investigated identifying *coherent* threads of documents. Shahaf and Guestrin (2010) formalized the characteristics of a good chain of articles and proposed an efficient algorithm to connect two specified articles. Gillenwater et al. (2012) proposed a probabilistic technique for extracting a diverse set of threads from a given collection. Shahaf et al. (2012b) extended work on coherent threads to coherent maps of documents, where a map is set of intersecting threads which are meant to represent how different threads of documents interact and related to one another. Shahaf et al. (2012a) applied the idea of metro maps to mapping research in scientific areas.

Chapter 3

COHERENT MULTI-DOCUMENT SUMMARIZATION

Building hierarchical summaries is a complex task. Before approaching this task, we focus on an intermediate goal – producing *coherent* short summaries. This goal will be a first test of our theory that providing organization to summaries improves summary quality, and we will heavily leverage our coherence metric in the following chapters on hierarchical summaries.

The goal of multi-document summarization (MDS) is to produce high quality summaries of collections of related documents. Most previous work in extractive MDS has studied the problems of sentence selection (*e.g.*, (Radev, 2004; Haghighi and Vanderwende, 2009)) and sentence ordering (*e.g.*, (Lapata, 2003; Barzilay and Lapata, 2008)) separately, but we believe that a joint model is necessary to produce coherent summaries. The intuition is simple: if the sentences in a summary are first selected—without regard to coherence—then a satisfactory ordering of the selected sentences may not exist.

An extractive summary is a subset of the sentences in the input documents, ordered in some way.[1] Of course, most *possible* summaries are incoherent. Now, consider a directed graph where the nodes are sentences in the collection, and each edge represents a pairwise ordering constraint necessary for a coherent summary (see Figure 3.1 for a sample graph). By definition, any *coherent* summary must obey the constraints in this graph.

Previous work has constructed similar graphs automatically for single document summarization and manually for MDS (see Section 3.1). In this chapter, we introduce a novel system, G-Flow, which extends this research in two important ways. First, it tackles automatic graph construction for MDS, which requires novel methods for identifying inter-document edges (Section 3.2). It uses this graph to estimate coherence of a candidate summary. Second, G-Flow introduces a novel methodology for joint sentence selection and or-

[1] We focus exclusively on extractive summaries, so we drop the word "extractive" henceforth.

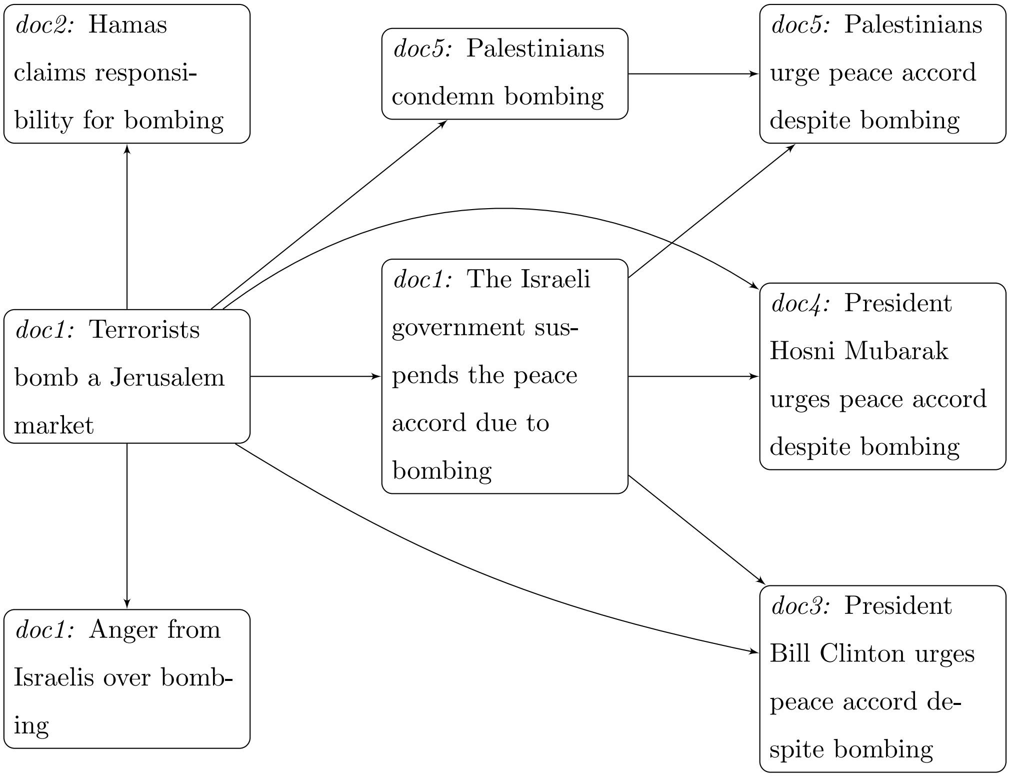

doc2: Hamas claims responsibility for bombing
doc5: Palestinians condemn bombing
doc5: Palestinians urge peace accord despite bombing
doc1: Terrorists bomb a Jerusalem market
doc1: The Israeli government suspends the peace accord due to bombing
doc4: President Hosni Mubarak urges peace accord despite bombing
doc1: Anger from Israelis over bombing
doc3: President Bill Clinton urges peace accord despite bombing

Figure 3.1: An example of a discourse graph covering a bombing and its aftermath. Each node represents a sentence from the original documents and is labeled with the source document id. A coherent summary should begin with the bombing and then describe the reactions. Sentences are abbreviated for compactness.

State-of-the-art MDS system	G-Flow
• The attack took place Tuesday near Cailaco in East Timor, a former Portuguese colony, according to a statement issued by the pro-independence Christian Democratic Union of East Timor. • The United Nations does not recognize Indonesian claims to East Timor.	• In a decision welcomed as a landmark by Portugal, European Union leaders Saturday backed calls for a referendum to decide the fate of East Timor, the former Portuguese colony occupied by Indonesia since 1975. • Indonesia invaded East Timor in 1975 and annexed it the following year.
• Bhichai Rattakul, deputy prime minister and president of the Bangkok Asian Games Organizing Committee, asked the Foreign Ministry to urge the Saudi government to reconsider withdrawing its 105-strong team. • The games will be a success.	• Thailand won host rights for the quadrennial games in 1995, but setbacks in preparations led officials of the Olympic Council of Asia late last year to threaten to move the games to another country. • Thailand showed its nearly complete facilities for the Asian Games to a tough jury Thursday - the heads of the organizing committees from the 43 nations competing in the December event.
• Jose Saramago became the first writer in Portuguese to win the Nobel Prize for Literature on Thursday. • They were given the prize for their discoveries concerning nitric oxide as a signaling molecule in the cardiovascular system, according to the citiation from the Karolinska lnstitute.	• Jose Saramago, a 75-year-old Portuguese writer who took up literature relatively late in life, was awarded this year's Nobel Prize in Literature Thursday by the Swedish Academy in Stockholm. • The literature prize is one of five established by Alfred Nobel, the Swedish industrialist and inventor of dynamite.

Table 3.1: Pairs of sentences produced by a pipeline of a state-of-the-art sentence extractor (Lin and Bilmes, 2011) and sentence orderer (Li et al., 2011a), and by G-Flow.

dering (Section 3.3). It casts MDS as a constraint optimization problem where salience and coherence are soft constraints, and redundancy and summary length are hard constraints. Because this optimization problem is NP-hard, G-Flow uses local search to approximate it.

We report on a Mechanical Turk evaluation that directly compares G-Flow to state-of-the-art MDS systems. Using DUC'04 as our test set, we compare G-Flow against a combination of an extractive summarization system with state-of-the-art ROUGE scores (Lin and Bilmes, 2011) followed by a state-of-the-art sentence reordering scheme (Li et al., 2011a). We also compare G-Flow to a combination of an extractive system with state-of-the-art coherence scores (Nobata and Sekine, 2004) followed by the reordering system. In both cases participants substantially preferred G-Flow. Participants chose G-Flow 54% of the time when compared to Lin, and chose Lin's system 22% of the time. When compared to Nobata, participants chose G-Flow 60% of the time, and chose Nobata only 20% of the time. The remainder of the cases were judged equivalent.

A further analysis shows that G-Flow's summaries are judged superior along several dimensions suggested in the DUC'04 evaluation (including coherence, repetitive text, and referents). A comparison against manually written, gold standard summaries, reveals that while the gold standard summaries are preferred in direct comparisons, G-Flow has nearly equivalent scores on almost all dimensions suggested in the DUC'04 evaluation.

In this chapter, we make the following contributions:

- We present G-Flow, a novel MDS system that jointly solves the sentence selection and ordering problems to produce coherent summaries.

- G-Flow automatically constructs a domain-independent graph of ordering constraints over sentences in a document collection, without any manual annotation of the collection, based on syntactic cues and redundancy across documents. This graph is the backbone for estimating the coherence of a summary.

- We perform human evaluations on blind test sets and find that G-Flow dramatically outperforms state-of-the-art MDS systems.

- We release the code for G-Flow to the research community at `http://knowitall.cs.washington.edu/gflow/`.

3.1 Related Work

Automatic text summarization is a long-studied problem with early papers dating back over fifty years (Luhn, 1958; Rath et al., 1961). In recent times, multi-document summarization (MDS) has been the mainstay of summarization research.

Most existing research in multi-document summarization (MDS) focuses on sentence selection for increasing coverage and does not consider coherence of the summary (Section 3.1.1). Although coherence has been used in ordering of summary sentences (Section 3.1.2), this work is limited by the quality of summary sentences given as input. In contrast, G-Flow incorporates coherence in both selection and ordering of summary sentences.

G-Flow can be seen as an instance of discourse-driven summarization (Section 3.1.3). There is prior work in this area, but primarily for summarization of single documents. There is some preliminary work on the use of manually-created discourse models in MDS. Our approach is fully automated.

3.1.1 Subset Selection in MDS

Most extractive summarization research aims to increase the coverage of concepts and entities while reducing redundancy. Approaches include the use of maximum marginal relevance (Carbonell and Goldstein, 1998), centroid-based summarization (Saggion and Gaizauskas, 2004; Radev et al., 2004), covering weighted scores of concepts (Takamura and Okumura, 2009; Qazvinian et al., 2010), formulation as minimum dominating set problem (Shen and Li, 2010), and use of submodularity in sentence selection (Lin and Bilmes, 2011). Graph centrality has also been used to estimate the salience of a sentence (Erkan and Radev, 2004). Approaches to content analysis include generative topic models (Haghighi and Vanderwende, 2009; Celikyilmaz and Hakkani-Tur, 2010; Li et al., 2011b), and discriminative models (Aker et al., 2010).

The approaches listed above do not consider coherence as one of the desiderata in sentence selection. Moreover, they do not attempt to organize the selected sentences into an intelligible summary. They are often evaluted by ROUGE (Lin, 2004), which is coherence-insensitive. In practice, these approaches often result in incoherent summaries.

3.1.2 Sentence Reordering

A parallel thread of research has investigated taking a set of summary sentences as input and reordering them to make the summary fluent. Various algorithms use some combination of topic-relatedness, chronology, precedence, succession, and entity coherence for reordering sentences (Barzilay et al., 2001; Okazaki et al., 2004; Barzilay and Lapata, 2008; Bollegala et al., 2010). Recent work has also used event-based models (Zhang et al., 2010) and context analysis (Li et al., 2011a).

The hypothesis in this research is that a pipelined combination of subset selection and reordering will produce high-quality summaries. Unfortunately, this is not true in practice, because sentences are selected primarily for coverage without regard to coherence. This methodology often leads to an inadvertent selection of a set of disconnected sentences, which cannot be put together in a coherent summary, irrespective of how the succeeding algorithm reorders them. In our evaluation, reordering had limited impact on the quality of the summaries.

3.1.3 Coherence Models and Summarization

Research on discourse analysis of documents provides a basis for modeling coherence in a document. Several theories have been developed for modeling discourse, *e.g.*, Centering Theory, Rhetorical Structure Theory (RST), Penn Discourse TreeBank (Grosz and Sidner, 1986; Mann and Thompson, 1988; Wolf and Gibson, 2005; Prasad et al., 2008). Numerous discourse-guided summarization algorithms have been developed (Marcu, 1997; Mani, 2001; Taboada and Mann, 2006; Barzilay and Elhadad, 1997; Louis et al., 2010). However, these approaches have been applied to single document summarization and not to MDS.

Discourse models have seen some application to summary generation in MDS, for example, using a detailed semantic representation of the source texts (McKeown and Radev, 1995; Radev and McKeown, 1998). A multi-document extension of RST is Cross-document

Structure Theory (CST), which has been applied to MDS (Zhang et al., 2002; Jorge and Pardo, 2010). However, these systems require a stronger input, such as a manual CST-annotation of the set of documents. Our work can be seen as an instance of summarization based on lightweight CST. However, a key difference is that our proposed algorithm is completely automated and does not require any additional human annotation. Additionally, while incorporating coherence into selection, this work does not attempt to order the sentences coherently, while our approach performs joint selection and ordering.

Discourse models have also been used for evaluating summary quality (Barzilay and Lapata, 2008; Louis and Nenkova, 2009; Pitler et al., 2010). There is also some work on generating coherent summaries in specific domains, such as scientific articles (Saggion and Lapalme, 2002; Abu-Jbara and Radev, 2011) using domain-specific cues like citations. In contrast, our work generates summaries without any domain-specific knowledge. Finally, Celikyilmaz and Hakkani-Tür (2011) introduced an unsupervised probabilistic approach to generate coherent and non-redundant summaries for query-focused summarization. Unlike our approach, they did not consider sentence ordering, but only focused on sentence selection. Other research has investigated identifying coherent threads of *documents* rather than sentences (Shahaf and Guestrin, 2010).

3.2 Discourse Graph

As described above, our goal is to identify pairwise ordering constraints over a set of input sentences. These constraints specify a multi-document discourse graph, which is used by G-Flow to evaluate the coherence of a candidate summary.

In this graph G, each vertex is a sentence and an edge from s_i to s_j indicates that s_j can be placed right after s_i in a coherent summary. In other words, the two share a discourse relationship. In the following three sentences (from possibly different documents) there should be an edge from s_1 to s_2, but not between s_3 and the other sentences:

s_1 *Militants attacked a market in Jerusalem.*

s_2 *Arafat condemned the bombing.*

s_3 *The Wye River Accord was signed in October.*

Discourse theories have proposed a variety of relationships between sentences such as background and interpretation. RST has 17 such relations (Mann and Thompson, 1988) and PDTB has 16 (Prasad et al., 2008). While we seek to identify pairs of sentences that have a relationship, we do not attempt to label the edges with the exact relation. In that sense, our graph is a lightweight discourse graph.

We use textual cues from the discourse literature in combination with the redundancy inherent in related documents to generate edges. Because this methodology is noisy, the graph used by G-Flow is an approximation, which we refer to as an approximate discourse graph (ADG). We first describe the construction of this graph, and then discuss the use of the graph for summary generation (Section 3.3). Our indicators are as follows.

3.2.1 Deverbal Noun Reference

Often, the main description of an event is mentioned in a verbal phrase and subsequent references use deverbal nouns (nominalization of verbs) (*e.g.*, 'attacked' and 'the attack'). In this example, the noun is derivationally related to the verb, but that is not always the case. For example, 'bombing' in s_2 above refers to 'attacked' in s_1.

We identify verb-noun pairs with this relationship as follows. First, we locate a set of candidate pairs from WordNet (Miller, 1995): for each verb v, we determine potential noun references n using a path length of up to two in WordNet (moving from verb to noun is possible via WordNet's 'derivationally related' links).

This set captures verb-noun pairs such as ('to attack', 'bombing'), but also includes generic pairs such as ('to act', 'attack'). To filter such errors we score the candidate references. Our goal is to emphasize common pairs and to deemphasize pairs with common verbs or verbs that map to many nouns. To this end, we give each pair a score:

$$(c/p) * (c/q) \tag{3.1}$$

where c is the number of times the pair (v, n) appears in adjacent sentences, p is the number of times the verb appears, and q is the number of times that v appears with a different noun. We generate these statistics over a background corpus of 60,000 articles from the New York Times and Reuters, and filter out candidate pairs scoring below a threshold identified over a

verb	noun
to indict	indictment
to storm	attack
to finalize	end
to collide	crash
to detonate	explosion
to compensate	settlement
to elect	election
to overthrow	revolution
to covene	meeting
to relocate	move
to bombard	attack
to postpone	delay
to rip	attack
to poll	vote
to free	release
to gun	shooting

verb	noun
to mold	time
to market	market
to shop	market
to spring	movement
to classify	number
to derail	move
to shout	attack

Table 3.2: Examples of correct deverbal noun pairs identified by the system (above) and incorrect pairs identified by the system (below). In general, G-Flow benefits substantially from the fact that all documents are related, and so, even though incorrect pairs could be recognized by the systems, in practice, they rarely occur together.

small training set. In Table 3.2, we show correct and incorrect deverbal noun pairs identified by the system.

We construct edges in the ADG between pairs of sentences containing these verb to noun mappings. To our knowledge, we are the first to use deverbal nouns for summarization.

We performed a simple experiment to evaluate the usefulness of the deverbal noun pairs. In this experiment, we randomly chose 200 verb to noun pairs identified in the DUC'04 dataset and marked each pair correct or incorrect. Note that in this experiment we evaluate the pairs rather than the sentences. Thus a pair is correct if the noun could represent a deverbal noun reference, whether or not that noun actually does refer to that verb (*e.g.* ('argue','debate') would be marked correct even if that specific 'debate' does not refer to that specific 'argue' instance. In this experiment, we found that 74% of pairs were correct.

3.2.2 Event/Entity Continuation

Our second indicator is related to lexical chains (Barzilay and Lapata, 2008). We add an edge in the ADG from a sentence s_i to s_j if they contain the same event or entity and the timestamp of s_i is less than or equal to the timestamp of s_j (timestamps generated with (Chang and Manning, 2012)).

s_4 *The bombing was the second since the summit in Wye River.*

s_5 *The summit led to an agreement between Israel and the Palestinian Authority .*

3.2.3 Discourse Markers

We use 36 explicit discourse markers (*e.g.*, 'but', 'however', 'moreover') to identify edges between two adjacent sentences of a document (Marcu and Echihabi, 2002). This indicator lets us learn an edge from s_6 to s_7 below:

s_6 *Arafat condemned the bombing.*

s_7 ***However,*** *Netanyahu suspended peace talks.*

3.2.4 Inferred Edges

We exploit the redundancy of information in MDS documents to infer edges to related sentences. An edge (s, s'') can be inferred if there is an existing edge (s, s') and s' and s'' express similar information. As an example, the edge (s_8, s_9) can be inferred based on edge (s_6, s_7):

s_8 *Arafat condemned the attack.*

s_9 *Netanyahu has suspended the talks.*

To infer edges we need an algorithm to identify sentences expressing similar information. To identify these pairs, we extract Open Information Extraction (Banko et al., 2007) relational tuples for each sentence, and we mark any pair of sentences with an equivalent relational tuple as redundant (see Section 3.3.3). The inferred edges allow us to propagate within-document discourse information to sentences from other documents.

3.2.5 Co-referent Mentions

A sentence s_j will not be clearly understood in isolation and may need another sentence s_i in its context, if s_j has a general reference (*e.g.*, 'the president') pointing to a specific entity or event in s_i (*e.g.*, 'President Bill Clinton'). We construct edges based on coreference mentions, as predicted by Stanford's coreference system (Lee et al., 2011). We are able to identify syntactic edge (s_{10}, s_{11}):

s_{10} *Pres. Clinton expressed sympathy for Israel.*

s_{11} ***He** said the attack should not derail the deal.*

and (s_{12}, s_{13}):

s_{12} *Israel suspended peace talks after the attack.*

s_{13} *Untoward incidents had actually decreased since the start of **the talks**.*

3.2.6 Edge Weights and Negative Edges

We weight each edge in the ADG by adding the number of distinct indicators used to construct that edge – if sentences s and s' have an edge because of a discourse marker and a deverbal reference, the edge weight $w_G(s, s')$ will be two. We also include negative edges in the ADG. $w_G(s, s')$ is negative if s' contains a deverbal noun reference, a discourse marker, or a co-reference mention that is not fulfilled by s. For example, if s' contains a discourse marker, and s is not the sentence directly preceding s' and there is no inferred discourse link between s and s', then we will add a negative edge $w_G(s, s')$. If we know that 'He' refers to Clinton, we can add a negative edge (s_{14}, s_{15}):

s_{14} *Netanyahu suspended peace talks.*

s_{15} ***He** said the attack should not derail the deal.*

3.2.7 Preliminary Graph Evaluation

We evaluated the quality of the ADG used by G-Flow, which is important not only for its use in MDS, but also because the ADG may be used for other applications like topic tracking and decomposing an event into sub-events. One author randomly chose 750 edges and labeled an edge correct if the pair of sentences did have a discourse relationship between them and incorrect otherwise. 62% of the edges accurately reflected a discourse relationship. We evaluate yield rather than recall because of the difficulty of manually identifying every edge in the documents. Our ADG has on average 31 edges per sentence for a dataset in which each document cluster has on average 253 sentences. This evaluation includes only the positive edges.

3.3 Summary Generation

We denote a candidate summary X to be a sequence of sentences $\langle x_1, x_2, \ldots, x_{|X|} \rangle$. G-Flow's summarization algorithm searches through the space of ordered summaries and scores each candidate summary along the dimensions of coherence (Section 3.3.1), salience (Section 3.3.2) and redundancy (Section 3.3.3). G-Flow returns the summary that maximizes a joint objective function (Section 3.3.4). In this function, salience and coherence are soft constraints and redundancy and length are hard. Coherence is by necessity a soft constraint as the algorithm for graph construction is approximate.

weight	feature
-0.037	position in document
0.033	from first three sentences
-0.035	number of people mentions
0.111	contains money
0.038	sentence length > 20
0.137	length of sentence
0.109	number of sentences verbs appear in (any form)
0.349	number of sentences common nouns appear in
0.355	number of sentences proper nouns appear in

Table 3.3: Linear regression features and learned weights for salience.

3.3.1 Coherence

G-Flow estimates coherence of a candidate summary via the ADG. We define coherence as the sum of edge weights between successive summary sentences. For disconnected sentence pairs, the edge weight is zero.

$$Coh(X) = \sum_{i=1..|X|-1} w_{G+}(x_i, x_{i+1}) + \lambda w_{G-}(x_i, x_{i+1})$$

w_{G+} represents positive edges and w_{G-} represents negative edge weights. λ is a tradeoff coefficient for positive and negative weights, which is tuned using the methodology described in Section 3.3.4.

Because coherence is defined as the sum of the coherence between each pair of adjuacent sentences, our summaries may not necessarily exhibit topic coherence as human generated summaries would. In the future, we hope to investigate this problem more.

3.3.2 Salience

Summaries should not just be coherent, they should also convey the most important information in the documents. Salience is the inherent value of each sentence to the documents.

We compute salience of a summary ($Sal(X)$) as the sum of the saliences of individual sentences:

$$Sal(X) = \sum_i Sal(x_i) \tag{3.2}$$

To estimate salience of a sentence, G-FLOW uses a linear regression classifier trained on ROUGE scores over the DUC'03 dataset. ROUGE scores are good indicators of salience, since they measure word (or n-gram) overlap with gold standard summaries. The classifier uses surface features designed to identify sentences that cover important concepts. The complete list of features and learned weights is in Table 3.3. The classifier finds a sentence more salient if it mentions nouns or verbs that are present in more sentences across the documents. The highest ranked features are the last three – number of other sentences that mention a noun or a verb in the given sentence. We use the same procedure as in deverbal nouns for detecting verb mentions that appear as nouns in other sentences (Section 3.2.1).

3.3.3 Redundancy

We also wish to avoid redundancy in our summaries. G-FLOW first processes each sentence with a state-of-the-art Open Information extractor OLLIE (Mausam et al., 2012), which converts a sentence into its component relational tuples of the form (arg1, relational phrase, arg2).[2] For example, it finds (Militants, bombed, a marketplace) as a tuple from sentence s_{16}.

Two sentences will express redundant information if they both contain the same or synonymous component fact(s). Unfortunately, detecting synonymy even at the relational tuple level is very hard. G-FLOW approximates this synonymy by considering two relational tuples synonymous if the relation phrases contain verbs that are synonyms of each other, have at least one synonymous argument, and are timestamped within a day of each other. Because the input documents cover related events, these relatively weak rules provide good performance. The same algorithm is used for inferring edges for the ADG (Section 3.2.4). This algorithm can detect that the following sentences express redundant information:

s_{16} *Militants bombed a marketplace in Jerusalem.*

[2] Available from http://ollie.cs.washington.edu

s_{17} *He alerted Arafat after assailants attacked the busy streets of Mahane Yehuda.*

3.3.4 Objective Function

The objective function needs to balance coherence, salience and redundancy and also honor the given budget, *i.e.*, maximum summary length B. G-FLOW treats redundancy and budget as hard constraints and coherence and salience as soft. Coherence is necessarily soft as the graph is approximate. While previous MDS systems specifically maximized coverage, in preliminary experiments on a development set, we found that adding a coverage term did not improve G-FLOW's performance. We optimize:

$$\begin{aligned} \textbf{maximize:} \quad & F(x) \triangleq Sal(X) + \alpha Coh(X) - \beta |X| \\ s.t. \quad & \textstyle\sum_{i=1..|X|} len(x_i) < B \\ & \forall x_i, x_j \in X : \text{redundant}(x_i, x_j) = 0 \end{aligned}$$

Here *len* refers to the sentence length. We add $|X|$ term (the number of sentences in the summary) to avoid picking many short sentences, which may increase coherence and salience scores at the cost of overall summary quality. Every time another sentence is added to the summary, there is an opportunity for increasing the coherence score, which will push the summary to many short sentences rather than several medium or long sentences.

The parameters α, β and λ (see Section 3.3.1) are tuned automatically using a grid search over a development set as follows. We manually generate *extractive* summaries for each document cluster in our development set (DUC'03) and choose the parameter setting that minimizes $|F(X_{\text{G-FLOW}}) - F(X^*)|$ summed over all document clusters. F is the objective function, $X_{\text{G-FLOW}}$ is the summary produced by G-FLOW and X^* is the manual summary.

This constraint optimization problem is NP hard, which can be shown by a simple reduction of the longest path problem as follows.

Proof that Solving for the Objective Function is NP Hard. Suppose there exists a graph R for which we wish to find a simple path of maximum length (longest path problem). There are two parts to this problem: (1) choosing a sequence of nodes with maximum edge weight and (2) choosing a sequence that forms a path in R.

To reduce the longest path problem to solving for the objective function, R will be the ADG, so each node represents a “sentence” and each edge represents the “coherence” between sentences.

We begin by simplifying the objective function to maximize only coherence, as it has several terms which are unnecessary for the longest path problem. Remove the budget constraint by setting $B = \infty$, remove the salience term by setting $Sal(x_i) = 0$ for all x_i, and remove the number of sentences term by setting $\beta = 0$. Now, the objective function is simply to maximize $Coh(X)$ with no redundant sentences.

We must now enforce the path constraint of the longest path problem. Begin by adding “redundant” pairs (x_i, x_i) for all nodes $x_i \in R$. The objective function can now only choose a node once. Next, add edges (x_i, x_j) with weight $-\infty$ for each pair of previously unconnected nodes x_i and x_j. Suppose a sequence x_i, x_j, x_k, x_l is considered with $(x_j, x_k) = -\infty$. The score of the total sequence will be improved by removing either x_i and x_j or x_k and x_l. Thus, the sequence that maximizes the objective function must be a path in the original graph R.

□

3.3.5 Approximation of Objective Function

G-Flow uses local search to reach an approximation of the optimum.

G-Flow employs stochastic hill climbing with random restarts as the base search algorithm. At each step, the search algorithm either adds a sentence, removes a sentence, replaces a sentence with another, or reorders a pair of sentences. We also allow for removing one sentence and replacing a second sentence or replacing one sentence and adding a second sentence. These two steps are useful because the budget is most often defined in bytes rather than sentences. Thus we are able to remove two short sentences and add a long sentence. Or remove one long sentence and add two short sentences. Likewise, we allow for insertion in any place in the current summary and removal of any sentence in the current summary. While this search procedure works well for traditional multi-document summarization tasks, the branching factor becomes quite large when the size of the budget is very large and the size of the input documents is very large.

The initial summary for random restarts is constructed as follows. We first pick the highest salience sentence with no incoming negative edges as the first sentence. The following sentences are probabilistically added one at a time based on the summary score up to that sentence. The initial summary is complete when there are no possible sentences left to fit within the budget. Intuitively, this heuristic chooses a good starting point by selecting a first sentence that does not rely on context and subsequent sentences that build a high scoring summary. One could also modify the objective function to place a higher weight on the first sentence's negative coherence.

3.3.6 Optimizations for Efficiency

As related in the previous section, G-Flow uses stochastic hill climbing with random restarts to find a solution. Hill climbing is a good solution, particularly for traditional MDS problems where in the input is 10 documents and the output is a 665 byte summary. However, to enable efficient processing, we make a number of basic optimizations.

First, as related above, we are careful to choose a good start state. The bottleneck to processing is primarily in the summary scores which must be calculated for all branches. Therefore, we cache as many of the calculations as possible (*e.g.* salience calculations for each sentence are stored) and check that we are not recomputing summaries scores. Additionally, when adding or replacing sentences, we process the sentences shortest to longest and when a sentence that does not fit in the summary is identified, we do not bother to process any longer sentences. We also incrementally compute the summary score. For example, if we are considering inserting a sentence at the beginning of the summary, the score for the rest of the summary is calculated and then each potential sentence's incremental contribution is calculated.

3.4 Experiments

Because summaries are intended for human consumption we focused on human evaluations. We hired workers on Amazon Mechanical Turk (AMT) to evaluate the summaries. Our evaluation addresses the following questions:

- How do G-Flow summaries compare against summaries produced by state-of-the-art MDS systems (Section 3.4.2)?
- What is G-Flow's performance along important summarization dimensions such as coherence and redundancy (Section 3.4.3)?
- How does G-Flow perform on coverage as measured by ROUGE (Section 3.4.3)?
- How much do the components of G-Flow's objective function contribute to performance (Section 3.4.4)?
- How do G-Flow's summaries compare to human summaries (Section 3.4.2)?

3.4.1 Data and Systems

We evaluated the systems on the Task 2 DUC'04 multi-document summarization dataset. This dataset consists of 50 clusters of related documents, each of which contains 10 documents. Each cluster of documents also includes four gold standard summaries used for evaluation. As in the DUC'04 competition, we allowed 665 bytes for each summary including spaces and punctuation. We used DUC'03 as our development set, which contains 30 document clusters, again with approximately 10 documents each.

We compared G-Flow against four systems. The first is a recent MDS extractive summarizer, which we chose for its state-of-the-art ROUGE scores (Lin and Bilmes, 2011).[3] Lin and Bilmes (2011) present an extractive MDS algorithm that performs sentence selection by maximizing coverage over all sentences in the set of documents and diversity of the summary set. We refer to this system as Lin.

Because our algorithm performs both sentence selection and sentence ordering, we also compare against a pipeline of Lin's system followed by a reimplementation of a state-of-the-art sentence reordering system (Li et al., 2011a). This reordering algorithm determines whether two sentences should be placed adjacent to eachother by comparing the similarity

[3] We thank Lin and Bilmes for providing us with their code. Unfortunately, we were unable to obtain other recent MDS systems from their authors.

Gold Standard Summary	Summary Produced by G-Flow
President Boris Yeltsin's health has become a matter of great concern to the Russian leadership. The concern began in 1996 when he had a heart attack followed by bypass surgery. Illness has often sidelined him during his seven years in power. He recently cut short a trip to Central Asia because of a respiratory infection and he later canceled two out-of-country summits. This revived questions about his ability to lead Russia through any crisis. Yeltsin refuses to admit he is seriously ill and his condition is kept secret, even the cause for burns on his hands. Russia's leaders are calling for his resignation and question his legal right to seek reelection.	Russian President Boris Yeltsin cut short a trip to Central Asia on Monday due to a respiratory infection that revived questions about his overall health and ability to lead Russia through a sustained economic crisis. Doctors insisted Monday that Yeltsin fly home from Central Asia a day ahead of schedule because he was suffering from an upper respiratory infection and had a mild fever of 37.4 Celsius. The president and his doctors say Yeltsin has no serious health problems and will serve out the final two years of his term. Yeltsin is still planning to go to Vienna for an Oct. 27-28 summit of European nations, the president's office said.
At least 60 teenagers were killed and another 160 were injured in a dance hall fire in Goteborg, Sweden, Sweden's second largest city. The fire was the worst in Sweden's modern history. At least 400 teenagers, attending a Halloween dance, were crammed into a facility meant to hold 150. The dance attendees were mostly immigrant children from representing 19 nationalities, including Somalia, Ethiopia, Iraq, Iran and former Yugoslavia. The cause of the fire, which quickly engulfed the two-story brick building is unknown as investigators continue to probe the ruins. Emergency help was delayed by about three minutes because of language difficulties.	A fire turned a dance hall jammed with teen-age Halloween revelers into a deathtrap, killing at least 60 people and injuring about 180 in Sweden's second-largest city. The worst previous fire disaster in modern Sweden was in 1978 in Boraas, when 20 people died in a hotel fire. Forensic experts examining heavily burned bodies were able Saturday to identify more of the 60 young people who died in a dance hall fire, but the catastrophe's most tormenting question was still unanswered. On Saturday, hundreds of people stood quietly outside the gutted building amid flowers and candles as they attempted to come to grips with catastrophe.

Table 3.4: Examples of pairs of gold standard summaries and summaries produced by G-Flow.

of each sentence to the context surrounding the other. We refer to this system as Lin-Li. This second baseline allows us to quantify the advantage of using coherence as a factor in both sentence extraction and ordering.

We also compare against the system that had the highest coherence ratings at DUC'04 (Nobata and Sekine, 2004), which we refer to as Nobata. As this system did not preform sentence ordering on its output, we also compare against a pipeline of Nobata's system and the sentence reordering system. We refer to this system as Nobata-Li. Nobata and Sekine (2004) perform sentence selection by scoring sentences by their sentence position, length, tf*idf, and similarity with the headlines.

Lastly, to evaluate how well the system performs against human generated summaries, we compare against the gold standard summaries provided by DUC.

3.4.2 Overall Summary Quality

To measure overall summary quality, we performed human evaluations in which Amazon Mechanical Turk (AMT) workers compared two candidate system summaries. The workers first read a gold standard summary, followed by the two system summaries, and were then asked to choose the better summary from the pair. The system summaries were shown in a random order to remove any bias.

To ensure that workers provided high quality data we added two quality checks. First, we restricted to workers who have an overall approval rating of over 95% on AMT. Second, we asked the workers to briefly describe the main events of the summary. We manually filtered out work where this description was incorrect. For example, if the summary described an airplane crash, and the worker wrote that the summary described an election, we would remove the worker's input.

Six workers compared each pair of summaries. We recorded the scores for each cluster, and report three numbers: the percentages of clusters where a system is more often preferred over the other and the percentage where the two systems are tied. G-Flow is preferred almost three times as often as Lin:

G-Flow	Indifferent	Lin
56%	24%	20%

Next, we compared G-Flow and Lin-Li. Sentence reordering improves performance slightly, but G-Flow is still overwhelmingly preferred:

G-Flow	Indifferent	Lin-Li
54%	24%	22%

These results suggest that incorporating coherence in sentence extraction adds significant value to a summarization system.

In these experiments, Lin and Lin-Li are preferred in some cases. We analyzed those summaries more carefully, and found that occasionally, G-Flow will sacrifice a small amount of coverage for coherence, resulting in lower performance in those cases (see Section 3.4.3). Additionally, G-Flow summaries tend to be more focused than Lin and Lin-Li. If the wrong topic is chosen by G-Flow, the entire summary may be off-topic, whereas Lin and Lin-Li are more likely to cover multiple topics.

We also compared Lin and Lin-Li, and found that reordering does not improve performance by much.

Lin-Li	Indifferent	Lin
32%	38%	30%

While the scores presented above represent comparisons between G-Flow and a summarization system with state-of-the-art ROUGE scores, we also compared against a summarization system with state-of-the-art coherence scores – the system with the highest coherence scores from DUC'04, (Nobata and Sekine, 2004). We found that G-Flow was again preferred:

G-Flow	Indifferent	Nobata
68%	10%	22%

Adding in sentence ordering again improved the scores for the comparison system somewhat:

G-Flow	Indifferent	Nobata-Li
60%	20%	20%

While these scores show a significant improvement over previous sytems, they do not convey how well G-Flow compares to the gold standard – manually generated summaries. As a final experiment, we compared G-Flow and a second, manually generated summary:

G-Flow	Indifferent	Gold
14%	18%	68%

While we were pleased that in 32% of the cases, Turkers either preferred G-Flow or were indifferent, there is clearly a lot of room for improvement despite the gains reported over previous sytems.

We analyzed those cases where G-Flow is preferred over the gold standard summaries. Those summaries were likely preferred because G-Flow is extractive and the gold standard summaries are abstractive. While the sentences that G-Flow chooses are written by professional reporters, the sentences in the gold standard summaries were often simple factual statements, so the quality of the writing did not appear to be as good.

See Table 3.4 for some examples of summaries produced by G-Flow.

3.4.3 Comparison along Summary Dimensions

A high quality summary needs to be good along several dimensions. We asked AMT workers to rate summaries using the quality questions enumerated in DUC'04 evaluation scheme.[4] This setup is similar to that used at DUC and by other authors of recent summarization papers (*e.g.* (Haghighi and Vanderwende, 2009; Celikyilmaz and Hakkani-Tur, 2010)). These questions concern:

- coherence
- useless, confusing, or repetitive text
- redundancy
- nouns, pronouns, and personal names that are not well-specified

[4] http://duc.nist.gov/duc2004/quality.questions.txt

- entities rementioned in an overly explicit way
- ungrammatical sentences
- formatting errors

We evaluated G-Flow, Lin-Li, and Nobata-Li against the gold standard summaries, using the same AMT scheme as in the previous section. To assess automated performance with respect to the standards set by human summaries, we also evaluated a (different) gold standard summary for each document cluster, using the same Mechanical Turk scheme as in the previous section. The 50 summaries produced by each system were evaluated by four workers. The results are shown in Figure 3.2.

G-Flow was rated significantly better than Lin-Li in all categories except 'Redundancy' and significantly better than Nobata-Li on 'Coherence' and 'Referents'. The ratings for 'Coherence', 'Referents', and 'OverlyExplicit' are not surprising given G-Flow's focus on coherence. The results for 'UselessText' may also be due to G-Flow's focus on coherence which ideally prevents it from getting off topic. Lastly, G-Flow may perform better on 'Grammatical' and 'Formatting' because it tends to choose longer sentences than other systems, which are less likely to be sentence segmentation errors. There may also be some bleeding from one dimension to the other – if a worker likes one summary she may score it highly for many dimensions.

Finally, somewhat surprisingly, we find G-Flow's performance to be nearly that of human summaries. G-Flow is rated statistically significantly lower than the gold summaries on only 'Redundancy'. Given the results from the previous section, G-Flow is likely performing worse on categories not conveyed in these scores, such as Coverage, which we examine next.

Coverage Evaluation using ROUGE

Most recent research has focused on the ROUGE evaluation, and thus implicitly on coverage of information in a summary. To estimate the coverage of G-Flow, we compared the systems on ROUGE (Lin, 2004). ROUGE measures the degree of word (or n-gram) overlap

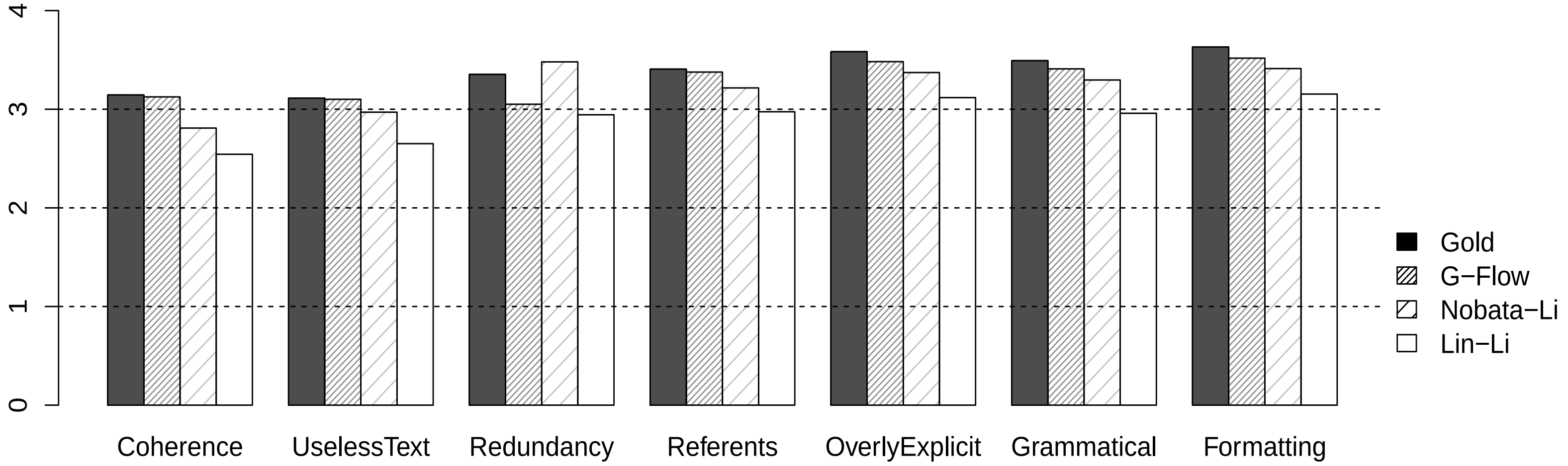


Figure 3.2: Ratings for the systems. 0 is the lowest possible score and 4 is the highest possible score. G-Flow is rated significantly higher than Lin-Li on all categories, except for ‘Redundancy’, and significantly higher than Nobata-Li on ‘Coherence’ and ‘Referents’. G-Flow is only significantly lower than the gold standard on ‘Redundancy’.

System	R	F
NOBATA	30.44	34.36
Best system in DUC-04	38.28	37.94
Takamura and Okumura (Takamura and Okumura, 2009)	38.50	-
LIN	39.35	38.90
G-FLOW	37.33	37.43
Gold Standard Summaries	**40.03**	**40.03**

Table 3.5: ROUGE-1 recall and F-measure results (%) on the DUC 2004 dataset. Some values are missing because not all systems reported both F-measure and recall.

between an automatically generated summary and the gold standard summary. We calculated ROUGE-1 scores for G-FLOW, LIN, and NOBATA.[5] As sentence ordering does not matter for ROUGE, we do not include LIN-LI or NOBATA-LI in this evaluation. Because our algorithm does not explicitly maximize coverage while LIN does, we expected G-FLOW to perform slightly worse than LIN.

The ROUGE-1 scores for G-FLOW, LIN, NOBATA and other recent MDS systems are listed in Table 3.5. We also include the ROUGE-1 scores for the gold summaries (compared to the other gold summaries). G-FLOW has slightly lower scores than LIN and the gold standard summaries, but much higher scores than NOBATA. G-FLOW only scores significantly lower than LIN and the gold standard summaries. G-FLOW likely sacrifices some coverage for overall coherence.

We can conclude that good summaries have both the characteristics listed in the quality dimensions, and good coverage. The gold standard summaries outperform G-FLOW on both ROUGE scores and the quality dimension scores, and therefore, outperform G-FLOW on overall comparison. However, G-FLOW is preferred to LIN-LI in addition to NOBATA-LI indicating that its quality scores outweigh its ROUGE scores in that comparison. An improvement to G-FLOW may focus on increasing coverage while retaining strengths such

[5] ROUGE version 1.5.5 with options: -a -c 95 -b 665 -m -n 4 -w 1.2

as coherence.

3.4.4 Ablation Experiments

In this ablation study, we evaluated the contribution of the main components of G-Flow – coherence and salience. The details of the experiments are the same as in the experiment described in Section 3.4.2.

We first measured the importance of coherence in summary generation. This system G-Flow-sal is identical to the full system except that it does not include the coherence term in the objective function (see Section 3.3.4). The results show that coherence is very important to G-Flow's performance:

G-Flow	Indifferent	G-Flow-sal
54%	26%	20%

Similarly, we evaluated the contribution of salience. This system G-Flow-coh does not include the salience term in the objective function:

G-Flow	Indifferent	G-Flow-coh
60%	20%	20%

Without salience, the system produces readable, but highly irrelevant summaries. Both components of the objective function are crucial to the overall performance of the system.

3.4.5 Agreement of Expert & Amazon Mechanical Turk Workers

Because summary evaluation is a relatively complex task, we compared AMT workers' annotations with expert annotations from DUC'04. We randomly selected ten summaries from each of the seven DUC'04 annotators, and asked four AMT workers to annotate them on the DUC'04 quality questions. For each DUC'04 annotator, we selected all pairs of summaries where one summary was judged more than one point better than the other summary. We compared whether the workers (voting as in Section 3.4.2) likewise judged that summary better than the second summary. We found that the annotations agreed in 75% of cases. When we looked only at pairs more than two points different, the agreement was 80%. Thus, given the subjective nature of the task, we feel reasonably confident that the AMT annotations are informative, and that the dramatic preference of G-Flow over the baseline systems is due to a substantial improvement in its summaries.

3.4.6 Distribution of Sentence Selection Over Documents

As a final experiment, we were interested in how often the selected sentences come from the same document and how often they are from different documents. It may be that summaries appear more coherent because they are drawn from the same documents, or G-Flow may be capable of creating coherent summaries with sentences drawn from different documents. To investigate this question, we counted the number of different documents represented in each summary and found that, on average, 79% of the sentences are drawn from different documents. So in a summary with five sentences, on average, four documents will be represented.

3.5 Discussion

While G-Flow represents the first coherent multi-document summarization system, there are many areas for improvement.

3.5.1 Scalability

Unlike work that attempts to build a summary by first selecting the sentences and then rearranging them (e.g. (Lin and Bilmes, 2011)), G-Flow jointly selects the sentences and ordering. Consequently, G-Flow is much slower than systems like (Lin and Bilmes, 2011), particularly for larger scale problems. As the objective function is NP Hard, G-Flow uses stochastic hill climbing to approximate the solution, adding, removing, replacing and reordering sentences from the current summary at each branch. Unfortunately, the branching factor becomes prohibitively large in cases where there are many options for removal or addition. In particular, when the budget given to the system is very large, and, to a lesser extent, when the number of potential sentences is much larger.

While we have made many small optimizations to the code to improve efficiency and scalability, future research could attempt to discover an algorithm for an efficient and scalable approximation of the objective function.

3.5.2 Building the Discourse Graph

G-Flow uses the approximate discourse graph to measure coherence, and the performance of the system would likely be greatly enhanced by higher precision and higher recall in the graph. There are a number of ways of improving the graph construction. Most simply, the weights on the types of edges could be learned, such that edges with more evidence of coherence could have higher weights than edges with less evidence. Learning the weights will likely require substantial training data and an interesting area of research would be to find an efficient way of collecting this training data. The best training data would be pairs of sentences from potentially different documents with a scalar value indicating the coherence between the two sentences. Unfortunately, extractive summaries are relatively uncommon. More data might exist for topic coherence. For example, Wikipedia articles link citations to sentences, and the source sentence that best matches the Wikipedia sentence could be identified. This methodology would generate pairs of sentences from different documents that theoretically exhibit topical coherence. Much filtering would likely be necessary to generate a clean dataset, but, since Wikipedia is so large, filtering would be less problematic.

While learning edge weights would likely be quite useful, our methods for identifying edges may still miss many edges. Another area of research is additional methods of identifying edges between sentences. We have used a number of heuristics to identify edges and have shied away from machine learning methods, primarily because of the lack of training data. Training data would consist of pairs of sentences potentially from different documents that contain a coherence relation. Such data is difficult to come by, but researchers could potentially learn coherence relations through automatically labeled edges such as the inferred edges from Section 3.2.4. A downside of this method is that the inferred edges may only represent certain types of coherence relations.

3.5.3 Coherence versus Cohesion

We have proposed G-FLOW as a system for *coherent* multi-document summarization, but in some ways G-FLOW is better tuned for *cohension* than coherence. (Halliday and Hasan, 1976) define cohesion as follows, “[The concept of cohesion] refers to relations of meaning that exist within the text, and that define it as a text. Cohesion occurs where the interpretation of some element in the discourse is dependent on that of another.” While coherence requires global flow of the summary, cohesion relates to the interconnectedness of the individual sentences.

Our discourse graph is inherently approximate, and, in some ways, lends itself more to cohesion than coherence. For example, if the sentences share a coreference, there is no guarantee that the story will progress between the two, but there is cohesion between the sentences. Additionally, our algorithm relies on pairwise connections between sentences. Thus, while each sentence could flow from the previous sentence if the discourse graph is built correctly, there is no guarantee the the fully summary will exhibit coherence. Future research could investigate both of these issues.

3.6 Conclusion

In this chapter, we present G-FLOW, a multi-document summarization system aimed at generating coherent summaries. While previous MDS systems have focused primarily on salience and coverage but not coherence, G-FLOW generates an ordered summary by jointly optimizing coherence and salience. G-FLOW estimates coherence by using an approximate discourse graph, where each node is a sentence from the input documents and each edge represents a discourse relationship between two sentences.

We performed manual evaluations that demonstrate that G-FLOW generates substantially better summaries than a pipeline of state-of-the-art sentence selection and reordering components. ROUGE scores, which measure summary coverage, show that G-FLOW sacrifices a small amount of coverage for overall readability and coherence. Comparisons to gold standard summaries show that G-FLOW must improve in coverage to equal the quality of manually written summaries.

The discourse graph proposed in this chapter will serve as the basis for our coherence metrics in the following chapters on hierarchical summarization. Beyond the work in this disseration, however, this research has applications to other areas of summarization such as update summarization and query based summarization, as well as work in dialogue systems, natural language generation, and machine translation. We are interested in investigating these topics in future work.

Chapter 4

HIERARCHICAL SUMMARIZATION: SCALING UP MULTI-DOCUMENT SUMMARIZATION

The explosion in the number of documents on the Web necessitates automated approaches that organize and summarize large document collections on a complex topic. Existing methods for multi-document summarization (MDS) are designed to produce short summaries of 10-15 documents.[1] MDS systems do not scale to datasets ten times larger and proportionately longer summaries: they either cannot run on large input or produce a disorganized summary that is difficult to understand.

In this chapter, we introduce a novel MDS paradigm, *hierarchical summarization*, which operates on large document collections, creating summaries that organize the information coherently. It mimics how someone with a general interest in a complex topic would learn about it from an expert – first, the expert would provide an overview, and then more specific information about various aspects. By arranging the information hierarchically, users can read as much as they wish to and browse areas of interest. Hierarchical summarization has the following novel characteristics:

- The summary is hierarchically organized along one or more organizational principles such as time, location, entities, or events.

- Each non-leaf summary is associated with a set of child summaries where each gives details of an element (e.g. sentence) in the parent summary.

- A user can navigate within the hierarchical summary by clicking on an element of a parent summary to view the associated child summary.

[1] In the DUC evaluations, summaries have a budget of 665 bytes and cover 10 documents.

For example, given the topic, "1998 embassy bombings," the first summary (Figure 1.2) might mention that the US retaliated by striking Afghanistan and Sudan. The user can click on this information to learn more about these attacks. In this way, the system can present large amounts of information without overwhelming the user, and the user can tailor the output to their interests.

In the next section, we formalize hierarchical summarization, and in the following two chapters, we describe fully implemented systems to perform hierarchical summarization for news and scientific documents respectively.

4.1 Hierarchical Summarization

We propose a new task for large-scale summarization called *hierarchical summarization*. Input to a hierarchical summarization system is a set of related documents D and a budget b for each summary within the hierarchy (in bytes, words, or sentences). The output is the hierarchical summary H, which we define formally as follows.

Definition A *hierarchical summary* H of a document collection D is a set of summaries X organized into a hierarchy. The top of the hierarchy is a summary X_1 representing all of D. Each summary X_i consists of summary units $x_{i,j}$ (*e.g.* the jth sentence of summary i) that point to a child summary, except at the leaf nodes of the hierarchy.

A child summary adds more detail to the information in its parent summary unit. The child summary may include sub-events or background and reactions to the event or topic in the parent.

We define several metrics in Chapter 5 for a well-constructed hierarchical summary. Each summary should maximize coverage of *salient* information; it should minimize *redundancy*; and it should have *intra-cluster coherence* as well as *parent-to-child coherence*.

4.2 Advantages of Hierarchical Summarization

Hierarchical summarization has several important strengths in the context of large-scale summarization.

Organized Output For any long summary, organization is of paramount importance. Human generated documents nearly always provide some system of organization after a certain length is reached, whether it is by paragraph, section, or chapter. Likewise, machine generated summaries should provide systems of organization. Without any organization, the user will likely become lost and will not be able to form a clear view of the information.

Tailored Output Length One important consequence of large output size is the possiblity of overwhelming users. For example, imagine a system that knows nothing about the user's desired amount of output. In flat multi-document summarization, the system chooses some budget of output that may or may not be the amount the user wants to see. In hierachical multi-document summarization, the information presented at the start is small and grows only as the user directs it, so as not to overwhelm the user. An obvious alternative solution is for the user to preselect the amount of information they wish to view. However, we would argue that when first learning about a topic, the user may not know how much they want to view. Hierarchical summarization allows the user to iteratively refine the level of detail and information they wish to learn.

Personalization Each user directs his or her own experience, so a user interested in one aspect need only explore that section of the data without having to view or understand the entire summary. The parent-to-child links provide a means for a user to navigate, drilling down for more details on topics of interest.

For broad topics that cover a wide range of information such as areas of scientific research or complex events that develop over several months, this quality is especially important. For example, a user who begins broadly interested in multi-document summarization does not need to read about all aspects of multi-document summarization, but instead only the parts that interest him or her. If abstractive summarization is more interesting to that user than extractive summarization, the user does not need to read about different extractive methods.

Interaction Lastly, users interact with the summarization system by clicking on sentences of interest and exploring the summaries on their own terms. Interactive systems provide a more engaging experience for users, and potentially allow for better knowledge retention.

By providing an interactive interface, we also allow the possibility of using the interactions as user feedback in future work. These interactions could potentially provide valuable information on what information is most interesting *etc.*

4.3 Organizing Principles

The hierarchy can be organized along any possible principle – by date, entity, location, event, task, method, *etc.* Ideally, the hierarchy will be organized in a way that is most illuminating to the user. For example, if a hurricane strikes several islands in the Caribbean, the best organization may be by location. Likewise, if a crime is committed and then an arrest is made, the best organization may be by time. For scientific topics, methodology (for example semi-supervised methods as opposed to supervised or unsupervised) is likely to be much more useful than time or location.

A system may select different organization for different portions of the hierarchy, for example, organizing first by location or prominent entity and then by date for the next level. Most importantly, the organizing principle should be clear to the user so that he or she can easily navigate the hierarchy to identify information of interest.

4.4 Challenges

Hierarchical summarization has all the challenges of flat multi-document summarization, including identifying the most important information in the input documents, avoiding redundant information, and building a coherent narrative. However, added to these challenges are two very important additions:

1. **Organizational Strategy** As described above, different organizational strategies will be most appropriate in different circumstances. Identifying a strategy which is illuminating to the reader is of paramount importance because without an obvious organization, the hierarchy loses its meaning to users.

2. **Top-Down Organization** The most important and most general information should appear at the top of the hierarchy and details of least importance should be at the leaves of the hierarchy. The system will need to be capable of distinguishing fine-grained differences in salience.

3. **Coherence between Parents and Children** Lastly, the summary should include sentences that have coherent relationships between parent sentences and the child summaries to which they lead. Without a coherent relationship, the users will be extremely confused by the summary produced when a sentence is clicked on, and will have no way of navigating the hierarchy to find information important to them. The system must be able to maintain local and global (topical) coherence between parent sentences and child summaries.

In the next chapters, we describe approaches to these problems targeted at the news and scientific document domains.

4.5 Conclusion

In this chapter, we introduced a new paradigm for large-scale summarization called hierarchical summarization, which allows a user to navigate a hierarchy of relatively short summaries. Hierarchical summarization scales up to orders of magnitude larger document collections than current multi-document summarization (MDS) systems. Hierarchical summarization provides organization, tailored output lengths, personalization, and interaction, all of which are important qualities for large-scale summarization.

In the following two chapters, we present approaches to hierarchical summarization for two important domains: (1) news documents and (2) scientific documents. We also describe experiments which demonstrate the effectiveness of hierarchical summarization in large-scale summarization tasks, as well as the strengths and limitations of these first two systems.

Chapter 5

HIERARCHICAL SUMMARIZATION FOR NEWS

Having introduced the new paradigm of hierarchical summarization in the previous chapter, in this chapter, we describe Summa, the first hierarchical summarization system for multi-document summarization. Summa operates on a corpus of related news articles. Summa hierarchically clusters the sentences by time, and then summarizes the clusters using an objective function that jointly optimizes salience, parent-to-child coherence, and intra-summary coherence while minimizing redundancy.

We conducted an Amazon Mechanical Turk (AMT) evaluation in which AMT workers compared the output of Summa to that of timelines and flat summaries. Summa output was judged superior more than three times as often as timelines, and users learned more in twice as many cases. Users overwhelmingly preferred hierarchical summaries to flat summaries (92%) and learned just as much.

Our main contributions are as follows:

- We present Summa, the first hierarchical summarization system. Summa operates on news corpora.

- Summa summarizes over an order of magnitude more documents than traditional MDS systems, producing summaries an order of magnitude larger.

- We present a user study which demonstrates the value of hierarchical summarization over timelines and flat multi-document summaries in learning about a complex topic.

- We release our system Summa to the research community. Summa is available at `http://summa.cs.washington.edu`.

In the next section, we describe our methodology to implement the Summa hierarchical summarization system: hierarchical clustering in Section 5.2 and creating summaries based on that clustering in Section 5.3. We discuss our experiments in Section 5.4, and conclusions in Section 5.6.

5.1 Task Definition and Overview

In this section, we define the task and give an outline of our system SUMMA and our experiments.

Task: SUMMA takes as input a set of related documents and produces as output a hierarchical summary as defined in Section 4.1. SUMMA is designed for the news domain. In the next chapter, we discuss another hierarchical summarization system, SCISUMMA, which is designed for scientific document summarization, but in this chapter we focus exclusively on news. SUMMA requires no background knowledge.

Method Overview: The problem of hierarchical summarization has all of the requirements of MDS, and additional complexities of inducing a hierarchical structure, processing an order of magnitude bigger input, generating a much larger output, and enforcing coherence between parent and child summaries. To simplify the task, we decompose it into two steps: hierarchical clustering and summarizing over the clustering (see Figure 5.1 for an example). A hierarchical clustering is a tree in which if a cluster g_p is the parent of cluster g_c, then each sentence in g_c is also in g_p. This organizes the information into manageable, semantically-related sections and induces a hierarchical structure over the input.

The hierarchical clustering serves as input to the second step – summarizing given the hierarchy. The hierarchical summary follows the hierarchical structure of the clustering. Each node in the hierarchy has an associated flat summary, which summarizes the sentences in that cluster. Moreover, the number of sentences in a flat summary is exactly equal to the number of child clusters of the node, since the user will click a sentence to get to the child summary. See Figure 5.1 for an illustration of this correspondence.

Evaluation: We evaluate the hierarchical summaries produced by SUMMA through a user study which measures user preference and also knowledge acquistion. Lastly, we perform a final set of evaluations to determine the coherence and informativeness of the summaries produced.

5.2 Hierarchical Clustering

In this section, we describe the process of hierarchical clustering, the output of which serves as the input to the summarization process (see Figure 5.1).

We perform clustering to identify a structure for our hierarchical summary. Each cluster in the hierarchical clustering should represent a set of sentences which would make sense to summarize together. In future work we intend to design a system that dynamically selects the best organizing principle for each level of the hierarchy. In this first implementation, we have opted for temporal organization, since this is generally the most appropriate for news events.

Because we are interested in *temporal* hierarchical summarization, we hierarchically cluster all the sentences in the input documents by time. Unfortunately, neither agglomerative nor divisive clustering is suitable, since both assume a binary split at each node (Berkhin, 2006). The number of clusters at each split should be what is most natural for the input data. We design a recursive clustering algorithm that automatically chooses the appropriate number of clusters at each split.

Before clustering, we timestamp all sentences. We use SUTime (Chang and Manning, 2012) to normalize temporal references, and we parse the sentences with the Stanford parser (Klein and Manning, 2003) and use a set of simple heuristics to determine if the timestamps in the sentence refer to the root verb. If no timestamp is given, we use the article date.

5.2.1 Temporal Clustering

After acquiring the timestamps, we must hierarchically cluster the sentences into sets that make sense to summarize together. Since we wish to partition along the temporal dimension, our problem reduces to identifying the best dates at which to split a cluster into subclusters. We identify these dates by looking for bursts of activity.

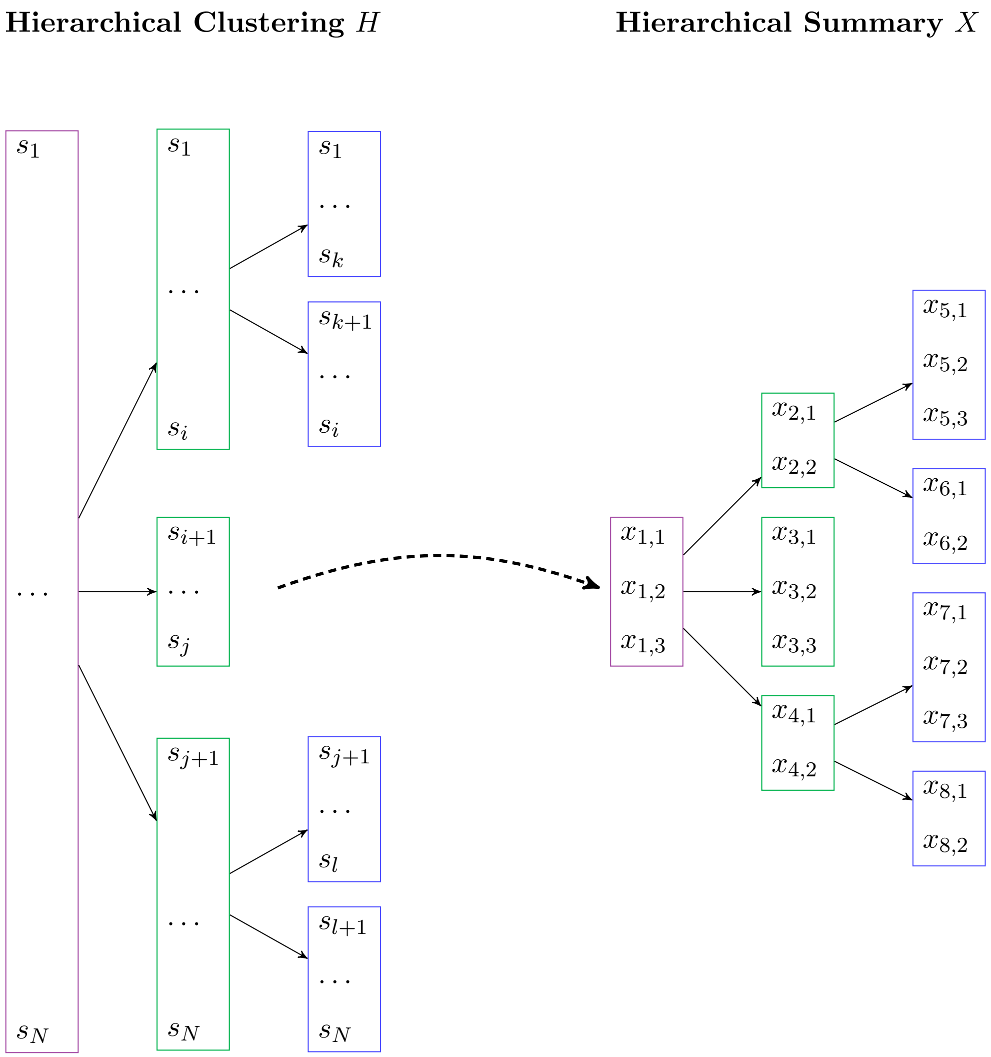


Figure 5.1: Examples of a hierarchical clustering and a hierarchical summary, where the input sentences are $s \in S$, the number of input sentences is N, and the summary sentences are $x \in X$. The hierarchical clustering determines the structure of the hierarchical summary.

News tends to be *bursty* – many articles on a topic appear at once and then taper out (Kleinberg, 2002). For example, Figure 5.2 shows the number of articles per day related related to the death of Pope John Paul II and the election of Pope Benedict XVI in the New York times (identified using a key word search). Figure 5.3 shows the number of articles per day related to Daniel Pearl's abduction, death, and the video of his murder. The figures shows a correspondence between major events and news spikes.

Ideal splits for these examples would occur just before each spike in coverage. However, when there is little differentiation in news coverage, we prefer clusters evenly spaced across time. We thus choose clusters $C = \{c_1, \ldots, c_k\}$ as follows:

$$\underset{C}{\text{maximize}} \quad B(C) + \alpha E(C) \tag{5.1}$$

where C is a clustering, $B(C)$ is the burstiness of the set of clusters, $E(C)$ is the evenness of the clusters, and α is the tradeoff parameter.

$$B(C) = \sum_{c \in C} burst(c) \tag{5.2}$$

$burst(c)$ is the difference in the number of sentences published the day before the first date in c and the average number of sentences published on the first and second date of c. We average the number of sentences published in the first two days because events sometimes occur too late in the day to show the spike in coverage until the day after:

$$burst(c) = \frac{pub(d_i) + pub(d_{i+1})}{2} - pub(d_{i-1}) \tag{5.3}$$

where d is a date indexed over time, such that d_j is a day before d_{j+1}, and d_i is the first date in c. $pub(d_i)$ is the number of sentences published on d_i. The evenness of the split is measured by:

$$E(C) = \min_{c \in C} size(c) \tag{5.4}$$

where $size(c)$ is the number of dates in cluster c.

We perform hierarchical clustering top-down, at each point solving for Equation 5.1. α was set using a grid-search over a development set. The development set was relatively easy to make as each instance simply consists of a set of articles (which can be automatically identified) and a small set of the best dates to split on. These dates are easily chosen by listing the most important dates and events in the article set.

5.2.2 Choosing the number of clusters

We cannot know *a priori* the number of clusters for a given topic. However, when the number of clusters is too large for the given summary budget, the sentences will have to be too short, and when the number of clusters is too small, we will not use enough of the budget.

We set the maximum number of clusters k_{max} and minimum number of clusters k_{min} to be a function of the budget b and the average sentence length in the cluster s_{avg}. k_{max} multiplied by the average sentence length should not exceed b:

$$k_{max} = \lfloor b/s_{avg} \rfloor \tag{5.5}$$

And k_{min} multiplied by s_{avg} should be at least half of b:

$$k_{min} = \lceil b/(2 \cdot s_{avg}) \rceil \tag{5.6}$$

Given a maximum and minimum number of clusters, we must determine the appropriate number of clusters. At each level, we cluster the sentences by the method described above and choose the number of clusters k according to the gap statistic (Tibshirani et al., 2000). Specifically, for each level, the algorithm will cluster repeatedly with k varying from the minimum to the maximum. The algorithm will return the k that maximizes the gap statistic:

$$Gap_n(k) = E_n^*\{\log(W_k)\} - log(W_k) \tag{5.7}$$

where W_k is the score for the clusters computed with Equation 5.1, and E_n^* is the expectation under a sample of size n from a reference distribution.

Ideally, the maximum depth of the clustering would be a function of the number of sentences in each cluster, but in our implementation, we set the maximum depth to three, which works well for the size of the datasets we use (300 articles).

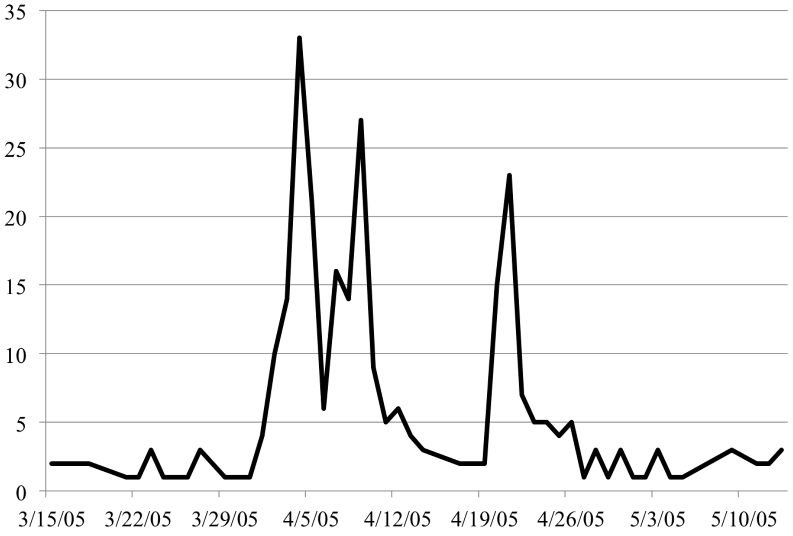


Figure 5.2: News coverage by date for Pope John Paul II's death, his funeral, and Pope Benedict XVI's election. Spikes in coverage correspond to the major events.

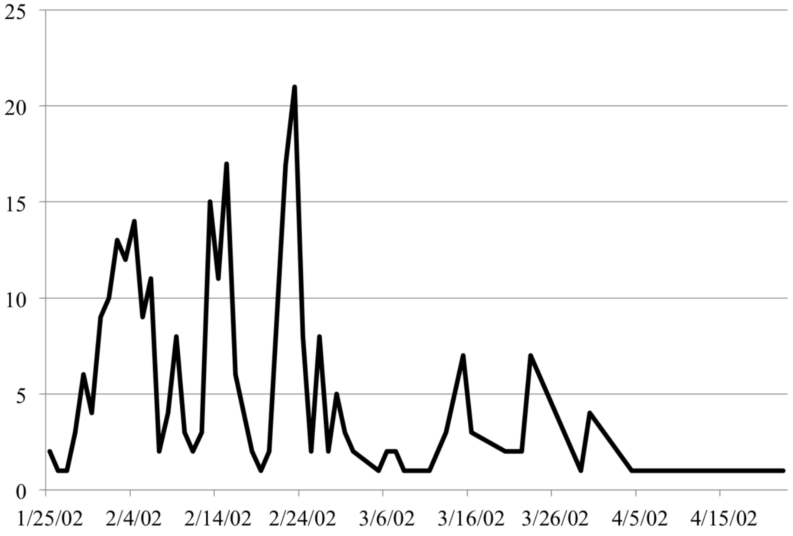


Figure 5.3: News coverage by date for Daniel Pearl's abduction, death, and the video of his murder. Spikes in coverage correspond to the major events.

5.3 Summarizing within the Hierarchy

After the sentences are clustered, we have a structure for the hierarchical summary that dictates the number of summaries and the number of sentences in each summary. We also have the set of sentences from which each summary is drawn. Our task is now to fill in each of the slots in the structure with the best sentence for that slot.

Intuitively, each cluster summary in the hierarchical summary should convey the most **salient** information in that cluster. Furthermore, the hierarchical summary should not include **redundant** sentences. A hierarchical summary that is only salient and nonredundant may still not be suitable if the sentences within a cluster summary are disconnected or if the parent sentence for a summary does not relate to the child summary. Thus, a hierarchical summary must also have **intra-cluster coherence** and **parent-to-child coherence**.

5.3.1 Salience

Salience is the value of each sentence to the topic from which the documents are drawn. We measure salience of a summary ($Sal(X)$) as the sum of the saliences of individual sentences:

$$Sal(X) = \sum_i Sal(x_i) \tag{5.8}$$

Following our work in coherent MDS from Chapter 3, we computed individual saliences using a linear regression classifier trained on ROUGE scores over the DUC'03 dataset (Lin, 2004). The classifier uses surface features designed to identify sentences that cover important concepts. For example, the classifier finds a sentence more salient if it mentions nouns or verbs that are present in more sentences across the documents. The highest ranked features are the number of other sentences that mention a noun or a verb in the given sentence.

In preliminary experiments, we noticed that many sentences that were *reaction* sentences were given a higher salience than *action* sentences. For example, the reaction sentence, "President Clinton vowed to track down the perpetrators behind the bombs that exploded outside the embassies in Tanzania and Kenya on Friday," would have a higher score than the *action* sentence, "Bombs exploded outside the embassies in Tanzania and Kenya on Friday." This problem occurs because the first sentence has a higher ROUGE score (it covers more important words than the second sentence).

Our first thought was to compute salience as the result of the salience classifier divided by the length of the sentence. This solution works poorly in practice because most important sentences in news articles are quite long with many clauses (for example, "A critical roadblock to the path to peace was removed on Tuesday under the watch of President Clinton after hundreds of Palestinians, many of them former guerrilla fighters, voted to remove clauses calling for the destruction of Israel from their organization's charter.").

To adjust for this problem, we use only words identified in the main clause (heuristically identified via the parse tree) to compute our salience scores.

5.3.2 Redundancy

In Chapter 3, we identified redundant sentences by looking at the Open Information Extraction tuples extracted from each sentence (Mausam et al., 2012), but we found that this methodology was insufficient for this task and the increased budget which presented far more opportunties for redundancy.

Instead, we identify redundant sentences using a linear regression classifier trained on a manually labeled subset of the DUC'03 sentences. The features include shared noun counts, sentence length, TF*IDF cosine similarity, timestamp difference, and features drawn from information extraction such as number of shared tuples in Open IE (Mausam et al., 2012).

5.3.3 Summary Coherence

For a hierarchical summary to be understandable to users, it must have coherence. In hierarchical summarization, we require two types of coherence: coherence between the parent and child summaries and coherence within each summary X_i.

We rely on the approximate discourse graph (ADG) that was described in Chapter 3 as the basis for measuring coherence. Each node in the ADG is a sentence from the dataset. An edge from sentence s_i to s_j with positive weight indicates that s_j may follow s_i in a coherent summary, *e.g.* continued mention of an event or entity, or coreference link between s_i and s_j. A negative edge indicates an unfulfilled discourse cue or co-reference mention.

Parent-to-Child Coherence: Users navigate the hierarchical summary from parent sentence to child summary, so if the parent sentence bears no relation to the child summary, the user will be understandably confused. The parent sentence must have positive evidence of coherence with the sentences in its child summary.

For example, the parent sentence:

s_2 *US missiles struck targets in Afghanistan.*

would connect well with the child summary:

s_4 *US issued warnings following the strikes.*

s_5 *Congress rallied behind Clinton.*

s_6 *Osama bin Laden survived the attack.*

We estimate parent to child coherence as the coherence between a parent sentence and each sentence in its child summary:

$$PCoh(X) = \sum_{c \in C} \sum_{i=1..|X_c|} w_{G+}(x_c^p, x_{c,i}) \tag{5.9}$$

where x_c^p is the parent sentence for cluster c and $w_{G+}(x_c^p, x_{c,i})$ is the sum of the positive edge weights from x_c^p to $x_{c,i}$ in the ADG G.

Intra-cluster Coherence: In traditional MDS, the documents are usually quite focused, allowing for highly focused summaries. In hierarchical summarization, however, a cluster summary may span hundreds of documents and a wide range of information. For this reason, we may consider a summary acceptable even if it has limited positive evidence of coherence in the ADG, as long as there is no negative evidence in the form of negative edges. For example, the following is a reasonable summary for events spanning two weeks:

s_1 *Bombs exploded at two US embassies.*

s_2 *US missiles struck in Afghanistan and Sudan.*

Our measure of intra-cluster coherence minimizes the number of *missing references*. These are coreference mentions or discourse cues where none of the sentences read before (either in an ancestor summary or in the current summary) contain an antecedent:

$$CCoh(X) = -\sum_{c \in C} \sum_{i=1..|X_c|} \#missingRef(x_{c,i}) \tag{5.10}$$

5.3.4 Objective Function

Having estimated salience, redundancy, and two forms of coherence, we can now put this information together into a single objective function that measures the quality of a candidate hierarchical summary.

Intuitively, the objective function should maximize both salience and coherence. Furthermore, the summary should not contain redundant information and each cluster summary should honor the given budget, i.e., maximum summary length b. We treat redundancy and budget as hard constraints and coherence and salience as soft constraints. Lastly, we require that sentences are drawn from the cluster that they represent and that the number of sentences in the summary corresponding to each non-leaf cluster c is equivalent to the number of child clusters of c. We optimize:

$$\begin{array}{rc} \textbf{maximize:} & F(x) \triangleq Sal(X) + \beta PCoh(X) + \gamma CCoh(X) \\ s.t. & \forall c \in C : \sum_{i=1..|X_c|} len(x_{c,i}) < b \\ & \forall x_i, x_j \in X : \text{redundant}(x_i, x_j) = 0 \\ & \forall c \in C, \forall x_c \in X_c : x_c \in c \\ & \forall c \in C : |X_c| = \#children(c) \end{array}$$

The tradeoff parameters β and γ were set manually based on a development set. Unfortunately, manually generating extractive hierarchical summaries to automatically set the tradeoff parameters is too expensive and time consuming to be a viable option.

5.3.5 Algorithm for Approximation

Optimizing this objective function is NP-hard, so we approximate a solution by using beam search over the space of partial hierarchical summaries. Notice the contribution from a sentence depends on individual salience (Sal), coherence ($CCoh$) based on sentences visible

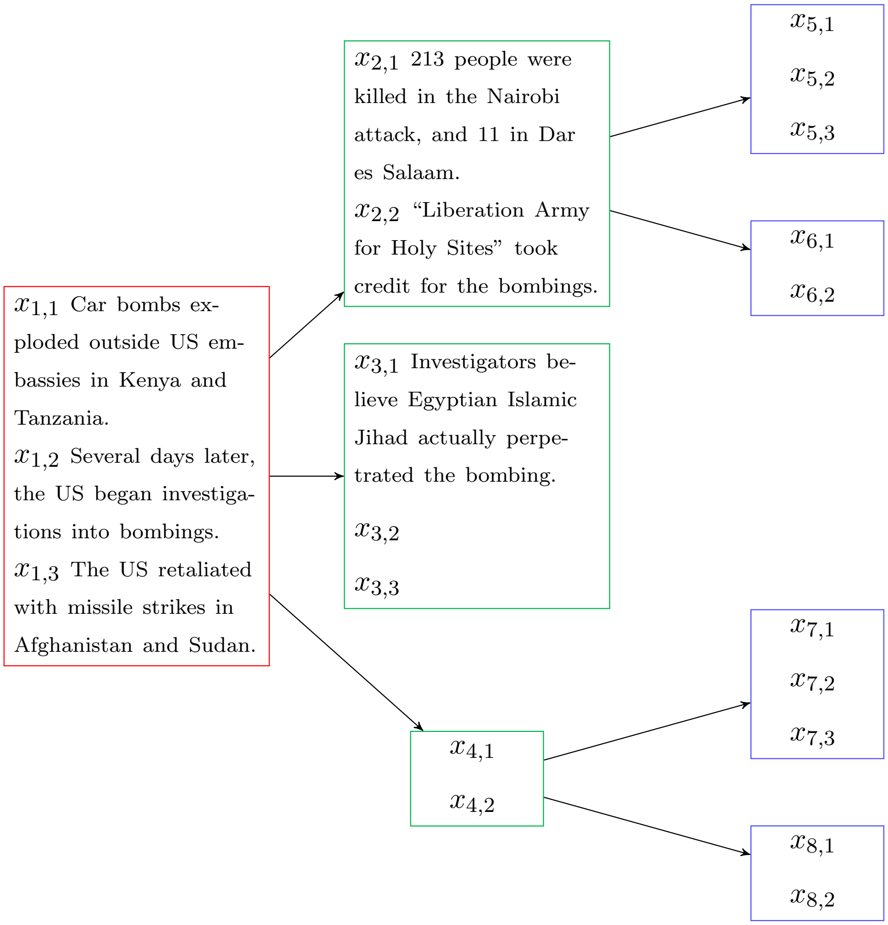

$x_{1,1}$ Car bombs exploded outside US embassies in Kenya and Tanzania.
$x_{1,2}$ Several days later, the US began investigations into bombings.
$x_{1,3}$ The US retaliated with missile strikes in Afghanistan and Sudan.
$x_{2,1}$ 213 people were killed in the Nairobi attack, and 11 in Dar es Salaam.
$x_{2,2}$ "Liberation Army for Holy Sites" took credit for the bombings.
$x_{3,1}$ Investigators believe Egyptian Islamic Jihad actually perpetrated the bombing.
$x_{3,2}$
$x_{3,3}$
$x_{4,1}$
$x_{4,2}$
$x_{5,1}$
$x_{5,2}$
$x_{5,3}$
$x_{6,1}$
$x_{6,2}$
$x_{7,1}$
$x_{7,2}$
$x_{7,3}$
$x_{8,1}$
$x_{8,2}$

Figure 5.4: An example of a hierarchical summary partially filled in.

```
function fillInClusterSummary
    Inputs:
      beam limit $B$
      partial summaries to return limit $K$
      beam of partial hierarchical summaries $H = \{X^1, \ldots, X^B\}$
      cluster summary index $i$ to be filled in of the hierarchical summary
    Output:
      beam of partial hierarchical summaries $H = \{X^1, \ldots, X^B\}$
    for $j = 1, \ldots, M$ do // For each slot in the current cluster summary
        $\hat{H} = \{\}$ // $\hat{H}$ will store the partial hierarchical summaries with slot $j$ filled in
        for $b = 1, \ldots, B$ do // For each partial hierarchical summary in the beam
            $x_{i,j} = X_i^b$ // Summary slot to fill in $x_{i,j}$ of current partial cluster summary $X_i^b$
            $P =$ getTopKSummaries$(X_i^b, x_{i,j}, K)$ // Get the $K$ best partial summaries with $x_{i,j}$ filled in
            $\hat{H} =$ addToQueue$(P)$ // Add the returned partial hierarchical summaries to the priority queue
        end for
        $H = getTopN(\hat{H}, B)$ // Beam is the top $B$ partial hierarchical summaries identified with slot $j$ filled in
    end for
    if numberOfChildren$(X_i) == 0$ then
        return $H$
    end if
    for $j = 1, \ldots, M$ do // For each of the children of cluster summary $X_i$
        $l =$ getChildIndex$(H, i, j)$ // Get the index of child $j$ of cluster summary $X_i$
        $H =$ fillInClusterSummary$(B, K, H, l)$ // Get the top partial hierarchical summaries with $X_l$ filled in
                                                  // The beam is now those partial summaries
    end for
    return $H$
end function
```

Figure 5.5: Summa's algorithm for approximating a solution to the objective function.

on the user path down the hierarchy to this sentence, and coherence ($PCoh$) based on its parent sentence and its child summary. Since most of the sentence's contributions depend on the path from the root to the sentence, we build our partial summary by incrementally adding a sentence top-down in the hierarchy and from first sentence to last within a cluster summary (see Figure 5.4 for an example).

When considering a sentence for inclusion, we need to consider its Sal and $CCoh$, both of which are available to us based on the sentences already in the summary. However, we must also consider the sentences's $PCoh$ contribution with respect to its child summary, which is not available at the search node. To account for this problem, we estimate the contribution of the sentence by jointly identifying its best child summary. However, we do not fix the child summary at this time – we simply use it to estimate $PCoh$ when using that sentence. Fixing the child summary would be a poor choice because each sentence in the child summary should be chosen with respect to its $PCoh$ as well, which necessitates identifying its child summary, requiring a recursive child summary identification. Instead, we simply use the child summary to estimate the contribution of the parent sentence. Since computing the best child summary is also intractable, we approximate a solution by a local search algorithm over the child cluster.

Overall, our algorithm is a two level nested search algorithm – beam search in the outer loop to search through the space of partial summaries and local search (hill climbing with random restarts) in the inner loop to pick the best sentence to add to the existing partial summary. We use a beam of size ten in our implementation. See Figure 5.5 for pseudocode of this algorithm.

5.4 Experiments

Our experiments are designed to evaluate how effective hierarchical summarization is in summarizing a large, complex topic and how well this helps users learn about the topic. We ran a user study evaluation to address the following questions:

- Do users prefer hierarchical summaries for topic exploration? (Section 5.4.4)

- Are hierarchical summaries more effective than other methods for learning about complex events? (Section 5.4.5)

- How informative are the hierarchical summaries compared to the other methods? (Section 5.4.6)

- How coherent is the hierarchical structure in the summaries? (Section 5.4.7)

5.4.1 Comparison Systems

We compared Summa against two baseline systems which represent the main NLP methods for large-scale summarization: an algorithm for creating timelines over sentences (Chieu and Lee, 2004),[1] and a state-of-the-art flat MDS system (Lin and Bilmes, 2011).

Chieu and Lee (2004) generate timelines by first identifying sentences relevant to a given query, resolving the dates of those sentences, ranking the sentences, removing duplicate sentences, and ordering the top sentences (given the budget) by time. Sentences are ranked according to their 'interest,' which is defined as the number of sentences that report the same events as the current sentence. Chieu and Lee (2004) measure whether two sentences reported the same event by their cosine similarity. Like many multi-document summarization papers, this paper focuses on choosing the set of sentences with highest coverage, excluding redundant sentences.

Lin and Bilmes (2011) present a sentence selection algorithm for multi-document summarization which maximizes coverage of the corpus and diversity of the summary sentences. The objective function that they maximize is a monotone nondecreasing submodular function and thus a greedy algorithm can approximate the solution. This characteristic is especially desirable given the large input and output setting of our experiments. Lin and Bilmes (2011)'s algorithm also has state-of-the-art ROUGE scores, but, like other existing MDS systems, does not account for coherence.

We do not compare against G-Flow, because our implementation does not scale to large enough inputs and large enough outputs. The branching factor of the search space is

[1] Unfortunately, we were unable to obtain more recent timeline systems from authors of the systems.

simply too large.

5.4.2 Budget

Recall that for hierarchical summarization, the budget is per cluster summary, rather than for the entire hierarchical summary. Thus, we give Summa 665 bytes per cluster summary (the traditional MDS budget), then calculate the total budget that Summa was allowed by multiplying 665 bytes by the number of cluster summaries in the hierarchical summary for the given topic. The other two systems were then given the total budget as their total budget (over 10 times the traditional MDS budget).

5.4.3 Datasets

We evaluated the questions on ten news topics, representing a range of tasks. The full list of topics is displayed in Table 5.1. We chose topics containing a set of related events that unfolded over several months and were prominent enough to be reported in at least 300 articles. We drew our articles from the Gigaword corpus, which contains articles from the New York Times and other major newspapers. For each topic, we automatically identified the 300 documents that had the highest tf*idf match with a key word search between two dates which specified the start and end of the event timespan. This method represented a simple and inexpensive way of gathering large datasets. One could potentially create even better datasets using more sophisticated methodology like topic models. Research in this area is orthogonal to the work we describe here.

We deliberately selected topics which were between five and fifteen years old so that evaluators would have relatively less pre-existing knowledge about the topic. Less pre-existing knowledge is desirable because, as part of our evaluations, we test users' knowledge gain from the different summaries and timelines.

5.4.4 User Preference

In our first experiment, we simply wished to evaluate which system users most prefer. We hired Amazon Mechanical Turk (AMT) workers and assigned two topics to each worker. We paired up workers such that one worker would see output from Summa for the first topic and a competing system for the second and the other worker would see the reverse. For quality control, we asked workers to complete a qualification task first, in which they were required to write a short summary of a news article. We also manually removed spam from our results. In Chapter 3, we showed that AMT workers' summary evaluations have high correlations with expert ratings. Five workers were hired to view each topic-system pair.

We asked the workers to choose which format they preferred and to explain why. The results are as follows:

Summa	**76%**	Timeline	24%
Summa	**92%**	Flat-MDS	8%

Users preferred the hierarchical summaries three times more often than timelines and over ten times more often than flat summaries. When we examined the reasons given by the users, we found that the people who preferred the hierarchical summaries liked that they gave a big picture overview and were then allowed to drill down deeper. Some also explained that it was easier to remember information when presented with the overview first. Typical responses included, "Could gather and absorb the information at my own pace," and, "Easier to follow and understand." When users preferred the timelines, they usually remarked that it was more familiar, i.e. "I liked the familiarity of the format. I am used to these timelines and they feel comfortable." Users complained that the flat summaries were disjointed, confusing, and very frustrating to read.

5.4.5 Knowledge Acquisition

Evaluating how much a user learned is inherently difficult, more so when the goal is to allow the user the freedom to explore information based on individual interest. For this reason, instead of asking a set of predefined questions, we assess the knowledge gain by following the methodology of Shahaf et al. (2012b) – asking users to write a paragraph summarizing the information learned.

Using the same setup as in the previous experiment, for each topic, five AMT workers spent three minutes reading through a timeline or summary and were then asked to write

	Topic	Time Covered in Months
1	Pope John Paul II's death and the 2005 Papal Conclave	2
2	The 2001 US presidential election and Bush v. Gore	1.5
3	The Tulip Revolution	2
4	Daniel Pearl's kidnapping and murder and the trial of his kidnappers	3
5	The Lockerbie bombing handover of suspects	5
6	The Kargil War	2.5
7	NATO's bombing of Yugoslavia in 1999	2.5
8	Pinochet's arrest in London and the subsequent legal battle	3
9	The 2005 London bombings, investigations, and arrests	1
10	The crash and investigation of SwissAir Flight 111	4

Table 5.1: Topics and amount of time covered per topic in the test set for our news hierarchical summarization experiments. All clusters have 300 documents total.

a description of what they had learned. Workers were not allowed to see the timeline or summary while writing. We collected five descriptions for each topic-system combination.

We then asked other AMT workers to read and compare the descriptions written by the first set of workers. Each evaluator was presented with a corresponding Wikipedia article and descriptions from a pair of users (timeline vs. Summa or flat MDS vs. Summa). The descriptions were randomly ordered to remove bias. The workers were asked which user appeared to have learned more and why. For each pair of descriptions, four workers evaluated the pair. We then took the majority vote for each pair. If the workers were tied, we marked the pair as indifferent.

To ensure that workers provided quality data, we added the following checks: (1) we restricted the task to workers who have an overall approval rating of over 95% on AMT, (2) have completed at least 100 tasks, and (3) were performing the task from inside the United States. The results of this experiment are as follows:

Prefer		Indiff.	Prefer	
Summa	**58%**	17%	Timeline	25%
Summa	**40%**	22%	Flat-MDS	38%

Descriptions written by workers using Summa were preferred over twice as often as those from timelines. We looked more closely at those cases where the participants either preferred the timelines or were indifferent and found that this preference was most common when the topic was not dominated by a few major events, but was instead a series of similarly important events. For example, in the kidnapping and beheading of Daniel Pearl there were two or three obviously major events, whereas in the Kargil War there were many smaller important events. In latter cases, the hierarchical summaries provided little advantage over the timelines because it was more difficult to arrange the sentences hierarchically.

Since Summa was judged to be so much superior to flat MDS systems in Section 5.4.4, it is surprising that users' descriptions from flat MDS were preferred nearly as often as those from Summa. While the flat summaries were disjointed, they were good at including salient information, with the most salient tending to be near the start of the summary. Thus, descriptions from both Summa and Flat-MDS generally covered the most salient information.

5.4.6 Informativeness

In this experiment, we assess the salience of the information captured by the different systems, and the ability of SUMMA to organize the information so that more important information is placed at higher levels.

ROOUGE Evaluation: We first automatically assessed informativeness by calculating the ROUGE-1 scores of the output of each of the systems. For the gold standard comparison summary, we use the overview sections of the Wikipedia articles for the topics.[2] Note that there is no good translation of ROUGE for hierarchical summarization. Thus, we simply use the traditional ROUGE metric, which will not capture any of the hierarchical format. This score will essentially serve as a rough measure of coverage of the entire summary to the Wikipedia article. The scores for each of the systems are as follows:

	P	R	F1
SUMMA	0.25	**0.67**	0.31
TIMELINE	0.28	0.65	0.33
FLAT-MDS	**0.30**	0.64	**0.34**

None of the differences are significant. From this evaluation, one can gather that the systems have similar coverage of the Wikipedia articles.

Manual Evaluation: While ROUGE serves as a rough measure of coverage, we were interested in gathering more fine-grained information on the informativeness of each system. We performed an additional manual evaluation that assesses the recall of important events for each system.

We first identified which events were most important in a news story. Because reading 300 articles per topic is impractical, we asked AMT workers to read a Wikipedia article

[2] We excluded one topic (the handover of the Lockerbie bombing suspects) because the corresponding Wikipedia article had insufficient information.

on the same topic and then identify the three most important events and the five most important secondary events. We aggregated responses from ten workers per topic and chose the three most common primary and five most common secondary events.

One of the authors then manually identified the presence of these events in the hierarchical summaries, the timelines and the flat MDS summaries. Below we show event recall (the percentage of the events that were mentioned).

Events	Summa	Timeline	Flat-MDS
Primary	**96%**	74%	93%
Secondary	**76%**	53%	64%

The difference in recall between Summa and Timeline was significant in both cases, and the difference between Summa and Flat-MDS was not. In general, the flat summaries were quite redundant, which contributed to the slightly lower event recall. The timelines, on the other hand, were both incoherent and at the same time reported less important facts.

We also evaluated at what level in the hierarchy the events were identified for the hierarchical summaries. The event recall shows the percentage of events mentioned at that level or above in the hierarchical summary:

Events	Level 1	Level 2	Level 3
Primary	63%	81%	96%
Secondary	27%	51%	76%

81% of the primary events are present in the first or second level, and 76% of the secondary events are mentioned by the third level. While recognizing primary events is relatively simple because they are repeated frequently, identification of important secondary events often requires external knowledge. For example, the system has no way of distinguishing between two sentences that are identical except that one describes an unknown person's reaction and the other describes President Clinton's reaction.

5.4.7 Parent-to-Child Coherence

We next tested the hierarchical coherence. One of the authors graded how much each non-leaf sentence in a summary was coherent with its child summary on a scale of one to five, with one being incoherent and five being perfectly coherent. We used the coherence scale from DUC'04.[3]

	Level 1	Level 2
Coherence	3.8	3.4

We found that for the top level of the summary, the parent sentence generally represented the most important event in the cluster and the child summary usually expressed details or reactions of the event. The lower coherence scores were often the result of too few lexical connections or lack of a theme or story. While the facts of the sentences made sense together, the summaries sometimes did not read as if they were written by a human, but as a series of disparate sentences.

For the second level, the problems were more basic. The parent sentence occasionally expressed a less important fact that the child summary did not then expand on, or, more commonly, the child summary was not focused enough. This result stems from two problems in our algorithm. First, summarizing sentences are rare, making good choices for parent sentences difficult to find. The second problem relates to the difficulty in identifying whether two sentences are on the same topic. For example, suppose the parent sentence is, "A Swissair plane Wednesday crashed off the coast of Nova Scotia, Canada late Wednesday night." A very good child sentence is, "The airline confirmed that all passengers of the flight had died." However, based on their surface features, the sentence, "A plane made an unscheduled landing after a Swissair plane crashed off the coast of Canada," appears to be a better choice.

Even though there is scope for improvement, we find these coherence scores encouraging for a first algorithm for the task.

5.5 Discussion

In this section we discuss areas of Summa that still require improvement and future research.

[3] http://duc.nist.gov/duc2004/quality.questions.txt

5.5.1 Scalability

While we have designed Summa for document collections 30x that traditional multi-document summarization systems are designed for, Summa is still far from achieving the scalability that one would hope. Summa is limited in two ways. First, Summa cannot efficiently process much larger collections of documents or produce summaries of much larger budgets. As the budgets become larger, more combinations must be considered and the power of the beam search that Summa uses to find a solution will be limited. Likewise, as the number of input documents grows, Summa will need to be modified to efficiently identify the best candidate sentences or an entirely different algorithm will need to be considered.

The second problem with scalability is that Summa is limited by the generality of the sentences in the dataset. Because Summa is extractive, if all sentences in the dataset simply represent details rather than overarching statements, the resulting summary will be very limited. To overcome this problem, Summa must be an abstractive summarization system rather than an extractive system. Such problems are also likely to occur on small datasets in other domains. For example, extractively summarizing a set of text messages would be difficult because there are few overarching summary statements in conversational texts.

5.5.2 Generality and Parent to Child Coherence

In a hierarchical summary, two types of sentences should be pushed to the top of the hierarchy – sentences that are most important and sentences that are most general. For example, "Bombs exploded outside the embassies in Tanzania and Kenya on Friday," conveys very important information. And, "The US retaliated and Russia, Sudan, Pakistan, and Afghanistan condemned the retaliation," is a relatively general sentence, which covers a large amount of information. While Summa is directly targeted to identify highly salient sentences, we have done little to emphasize general statements.

One potential direction for future work is to investigate both the identification of these statements and also how to integrate that knowledge into the system. Automatic training data for identifying summary statements could be generated from news articles that link to other articles. Often the sentence that provides the link is a summary of the full article that is linked. For example, articles in the New York Times will sometimes provide background information for a story in a single sentence that summarizes a previous article. By gathering this data, one could potentially identify the properties that make a statement a summary of other sentences.

5.5.3 Organizational Structure

In this first implementation of a hierarchical summarization system, we have organized the hierarchy temporally. Temporal organization is a good proxy for events, which are likely to be a good organizational method for most news stories. However, one could imagine many other methods for organizing the information, some of which will perform better for different types of input. In particular, this implementation is designed for news events and when the domain of the input data changes, the organizational methodology will likely also need to change. In the next chapter, we explore different organizational structures for scientific documents.

5.5.4 Parameter Settings

Summa relies on two parameter settings, β and γ, which determine the tradeoff of the parent-to-child coherence and the within cluster coherence and the salience. As of now, we have no good way of determining the best setting for these parameters other than manually based on a development set. For G-Flow, we were able to identify the parameter settings by minimizing the difference between the score given the chosen summary and the score given a gold standard extractive summary (taken over a development set). Unfortunately, for Summa, there is no inexpensive, equivalent methodology. Manually creating hierarchical summaries is an extremely time consuming process. This difficulty leaves Summa open to a variety of problems. Future research could investigate alternative ways of balancing these scores.

5.6 Conclusion

In this chapter, we presented Summa, an implemented hierarchical news summarization system, and demonstrated its effectiveness in a user study that compares Summa with a timeline system and a flat MDS system. Summa creates hierarchical summaries by first splitting the task into two subparts: hierarchical clustering and summarizing over the clustering. Summarizing over the hierarchy is accomplished by maximizing salience and coherence. When compared to timelines, users learned more with Summa in twice as many cases, and Summa was preferred more than three times as often. When compared to flat summaries, users overwhelming preferred Summa and learned just as much.

Chapter 6

HIERARCHICAL SUMMARIZATION FOR SCIENTIFIC DOCUMENTS

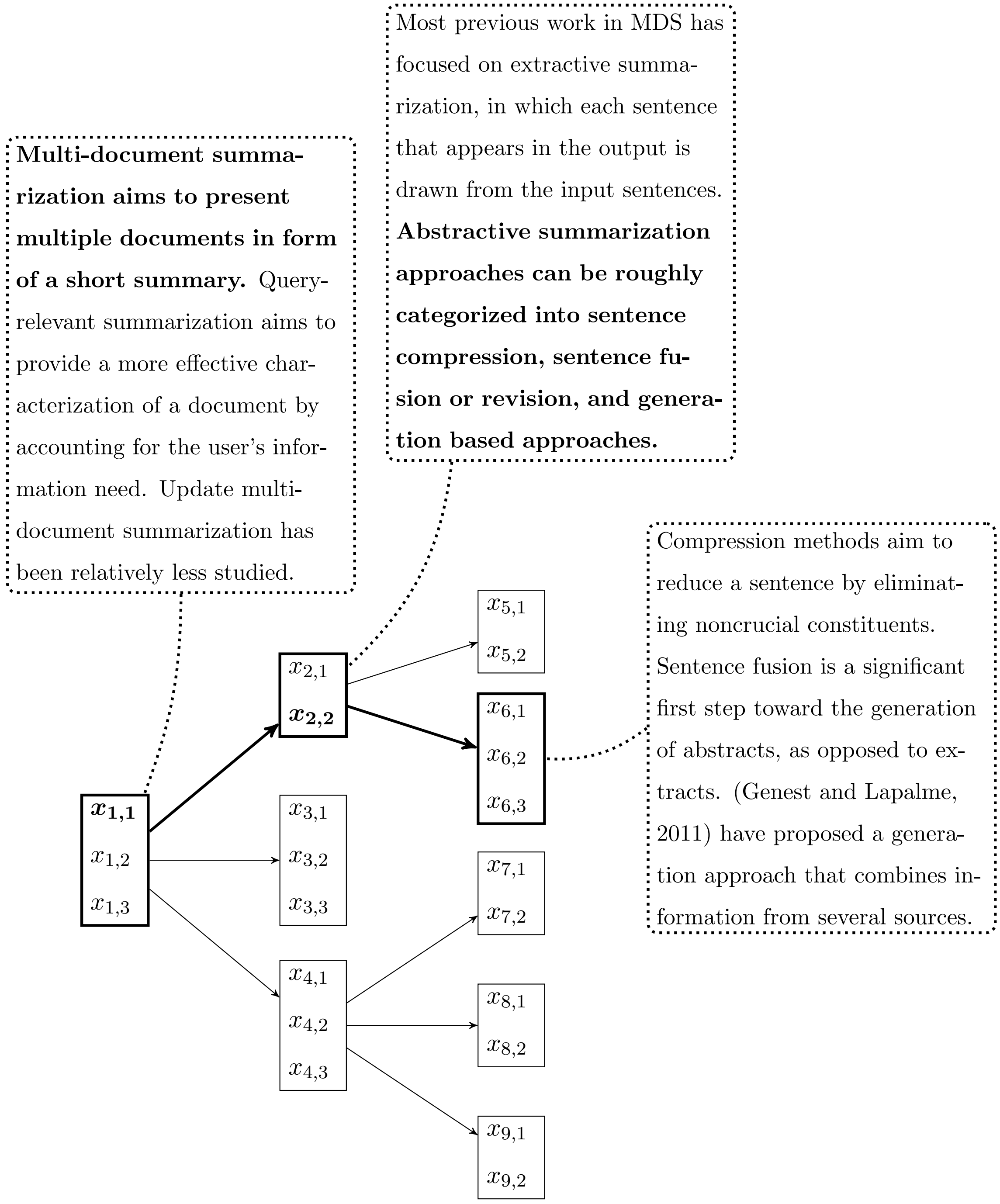
In this chapter, we describe a system designed to perform hierarchical summarization over scientific documents.

While news has received far more attention than scientific documents in the multi-document summarization community, scientific documents represent a challenging and important domain. Scientific documents are arguably more difficult and time consuming to summarize by hand than news articles, and very few up-to-date, manually generated summaries exist for many areas of research. Indeed, the creation of such manually generated summaries is so difficult that it often results in publication in the form of a surveys or book chapters. These manually generated summaries are then outdated within a few years.

Hierarchical summaries for scientific topics could be especially beneficial to undergraduates or first year graduate students, who are often interested in persuing a new topic, but lack guidance on the overarching problems and approaches. Hierarchical summaries could give such students the basic ideas of the topic and allow them to explore in more detail the areas that are of most interest to them. An example of a part of a hierarchical summary of topics in multi-document summarization is shown in Figure 6.1.

This chapter will also serve as a test of how easily the ideas proposed in the previous chapters can be transferred to a new domain. Thus far, we have focused exclusively on news articles, but ideally much of the previous work proposed here would apply to other domains as well.

In this chapter, we present an algorithm for generating hierarchical summaries of scientific documents, which we call SciSumma. SciSumma follows a similar design to Summa: we begin with hierarchically clustering the input and then summarize over the hierarchy. However, the key components of these steps – clustering and coherence modeling – must be



Multi-document summarization aims to present multiple documents in form of a short summary. Query-relevant summarization aims to provide a more effective characterization of a document by accounting for the user's information need. Update multi-document summarization has been relatively less studied.

Most previous work in MDS has focused on extractive summarization, in which each sentence that appears in the output is drawn from the input sentences. **Abstractive summarization approaches can be roughly categorized into sentence compression, sentence fusion or revision, and generation based approaches.**

Compression methods aim to reduce a sentence by eliminating noncrucial constituents. Sentence fusion is a significant first step toward the generation of abstracts, as opposed to extracts. (Genest and Lapalme, 2011) have proposed a generation approach that combines information from several sources.

Figure 6.1: An example of a hierarchical summary for multi-document summarization, with one branch of the hierarchy highlighted. Each rectangle represents a summary and each $x_{i,j}$ represents a sentence within a summary. The root summary provides an overview of some different types of multi-document summarization. When the last sentence is selected, a more detailed summary of generic multi-document summarization is produced, and when the last sentence of that summary is selected, a more detailed summary of abstractive approaches to multi-document summarization is produced.

substantially changed to fit this new domain. Temporal clustering is clearly a poor choice for scientific article clustering and while our discourse graph from Chapter 3 was well suited to the news domain, the indicators used to build it are insufficient for the scientific domain.

Both clustering and coherence modeling are designed around the observation that scientific articles are roughly composed of (1) problems and tasks and (2) methods and approaches. This observation enables the system to cluster all documents on abstractive summarization together, as well as to recognize that, "Compression methods aim to reduce a sentence's length by eliminating noncrucial constituents," is a good child sentence for, "Abstractive summarization approaches can be roughly categorized into sentence compression, sentence fusion or revision, and generation based approaches."

We performed user evaluations over eight topics in Natural Language Processing, which demonstrate the effectiveness of this approach. Our evaluations show that hierarchical summarization is preferred to existing methods for flat scientific document summarization, and that users learn more when they read hierarchical summaries.

The contributions of this chapter are as follows:

- We introduce SciSumma, a hierarchial summarization system for scientific documents. In addition to providing a hierarchical system for scientific documents, SciSumma demonstrates the applicability of Summa to an entirely different domain.

- SciSumma constructs a domain-independent graph of pairwise sentence ordering constraints, requiring no manual annotation. This graph is specifically designed for scientific articles.

- We evaluate SciSumma on the end-to-end task of summarizing research areas of NLP, and demonstrate its value over exiting methods.

- We release our system SciSumma to the research community. SciSumma is available at `http://scisumma.cs.washington.edu`.

The rest of this chapter is organized as follows. Section 6.1 describes previous work in this area. In Section 6.2, we give an overview of the approach taken by SciSumma.

Section 6.3 describes hierarchical clustering and in Section 6.4.1, we describe generation of the discourse graph, in Section 6.4, hierarchical summarization. We end with experiments in Section 6.5 and discussion and conclusions in Sections 6.6 and 6.7.

6.1 Related Work

Our work is focused on generating coherent, structured summaries of scientific research topics. There is very little previous work on generating multi-document summaries for scientific topics, and none designed for coherence or structure. However, this overview will identify some of the challenges in this area of research.

6.1.1 Generating Surveys

Relatively few papers have tackled the problem of automatically generating scientific literature surveys. Mohammad et al. (2009) studied the usefulness of different sections (*i.e.* abstracts versus citation text) in generating surveys and concluded that citation text is useful for survey generation. They also tested four summarization techniques to generate surveys. Jha et al. (2013) proposed a method based on expanding via the citation network for selecting the papers to be summarized given just a single query phrase.

Somewhat related to the task of generating surveys, Shahaf et al. (2012a) generated metro maps of scientific topics. These maps show relationships between *papers*. These relationships are designed to demonstrate developments in the area of research.

6.1.2 Citation Based Summarization

While few researchers have investigated generating surveys of research topics, the task of summarizing a scientific paper using its set of citation sentences (citation-based summarization) has received somewhat greater attention.

Citation-based summarization was introduced by Mei and Zhai (2008). Mei and Zhai (2008) proposed using language models to model the citation context and original content of the paper. They also revised the language models to include authority and proximity features. Elkiss et al. (2008) studied citation summaries and the extent to which they overlap with the original abstracts versus focus on different aspects.

Qazvinian and Radev (2008) introduced a system called C-LexRank, which performs citation-based summarization via a similarity network of the citation sentences. C-LexRank generates a summary by identifying a set of sentences that covers as much of the summarized information as possible using network analysis techniques. Qazvinian et al. (2010) improved upon this methodology by extracting important keyphrases from the set of citation sentences, and then identifying the set of sentences that covers as many keyphrases as possible.

Finally, Abu-Jbara and Radev (2011) focused on *coherent* citation-based summarization. They used three steps to generate summaries: preprocessing, extraction, and postprocessing. In preprocessing, sentences that depend on context or do not describe the work of the target paper are removed. In extraction, the sentences are first classified into functional categories (*e.g.* Background or Results), then clustered within categories, and finally the summary sentences are selected by their LexRank (Erkan and Radev, 2004) values. In postprocessing, the authors smooth the sentences to avoid reptition of authors' names and publication year.

6.1.3 Citations

Others have studied citations, modes of citation, and citation networks, providing insights and resources for work on summarization in this area. Bethard and Jurafsky (2010) proposed a model for literature search that learns the weights of factors important to researchers (*e.g.* recency of publication, topical similarity) through their citation patterns.

Other researchers have studied citation sentences to automatically determine their function (Nanba and Okumura, 1999; Nanba et al., 2000; Teufel et al., 2006). In Siddharthan and Teufel (2007), the authors introduce the task of deciding scientific attribution and show that scientific attribution improves results for the task of Argumentative Zoning. Qazvinian and Radev (2010) addressed the problem of identifying non-explicit citation sentences by modeling the sentences in an article as a markov random field and using belief propagation to detect context sentences.

The structure of citation networks was analyzed by Newman (2001). Newman (2001) demonstrated that collaboration networks are made up of small worlds, and pairs of scientisits are usually connected by short chains.

6.2 Overview of the Approach

Before giving details of our methodology, we define the task and outline the approach taken by SciSumma. Lastly, we will give an overview of the evaluations we perform.

Task: SciSumma is designed for the task of large scale summarization of scientific papers. Specifically, SciSumma takes as input a set of papers on a topic such as "semantic role labeling." The output of SciSumma is a hierarchical summary as defined in Section 4.1. SciSumma does not require any background knowledge.

Method Overview: Our basic approach for SciSumma is similar to that taken by Summa. We begin by hierarchically clustering all of the input documents. This clustering is the input for the second part of the algorithm, hierarchical summarization over the clustering. (See Figure 5.1 for an illustration of the input and output correspondance). Hierarchical summarization is performed by approximating a solution to an objective function which balances salience and coherence of the summary.

The differences in SciSumma and Summa lie primarily in (1) the clustering of the input text, (2) the generation of the discourse graph, and (3) the approximation algorithm for the summarization objective function. These differences represent the aspects of Summa which did not translate to a new domain. All other aspects, most notably the objective function and the basic setup of hierarchical clustering first and summarization second, transferred across domains, demonstrating the applicability of Summa.

Evaluation: Our evaluation for SciSumma is analogous to the evaluation we preformed for Summa. We compare SciSumma to a state-of-the-art system for scientific document multi-document summarization. We perform user evaluations on Natural Language Processing (NLP) topics, in which users use the output of SciSumma and that of the state-of-the-art system to learn about NLP topics. We evaluate user preference and knowledge acquisition.

6.3 Hierarchical Clustering

As for hierarchical summarization for news documents, we split this task into two parts: (1) hierarchical clustering and (2) hierarchical summarization. By first hierarchically clustering the input sentences, we gain valuable information on the structure the hierarchical summary should take. The hierarchical clustering provides the following information:

1. Each cluster in the hierarchical clustering represents one of the cluster summaries in the hierarchical summaries.

2. Each cluster summary in the hierarchical summary should summarize the information in the corresponding sentence cluster, and the sentences of the cluster summary should be drawn from the corresponding sentence cluster.

3. Each cluster summary should contain the same number of sentences as cluster children, because parent sentences are clicked to generate child summaries.

While news articles often cover multiple events, sometimes by giving background or relating associated news, research papers tend to cover just a single contribution. For this reason, we cluster *documents* rather than sentences. The only exception to this rule is for citation sentences. Sentences that cite other papers will be attached to each of those documents instead. The sentence will also be attached to the source document if it mentions a self-reference word or phrase such as ‘we,’ or ‘this paper’.

After examining several research topics and associated scientific papers, we identified two primary methods for organizing the documents: (1) by task or problem, and (2) by approach or method. For example, one could cluster all papers on multi-document summarization together and all papers on single document summarization together. Likewise, one could cluster all papers that use centroid-based methods together and all papers that use topic models together. Discoveries would be another way of organizing scientific documents, but we found discovery contributions to be uncommon in Computer Science, which is our focus for scientific article summarization.

The methodology described in Section 5.2 is unlikely to produce good clusters for scientific documents. Unlike news topics, research problems and methods are usually not organized by time. Instead of temporal clustering, we use a the citation graph, the tf*idf cosine similarity, and the co-citations to generate a distance function for documents.

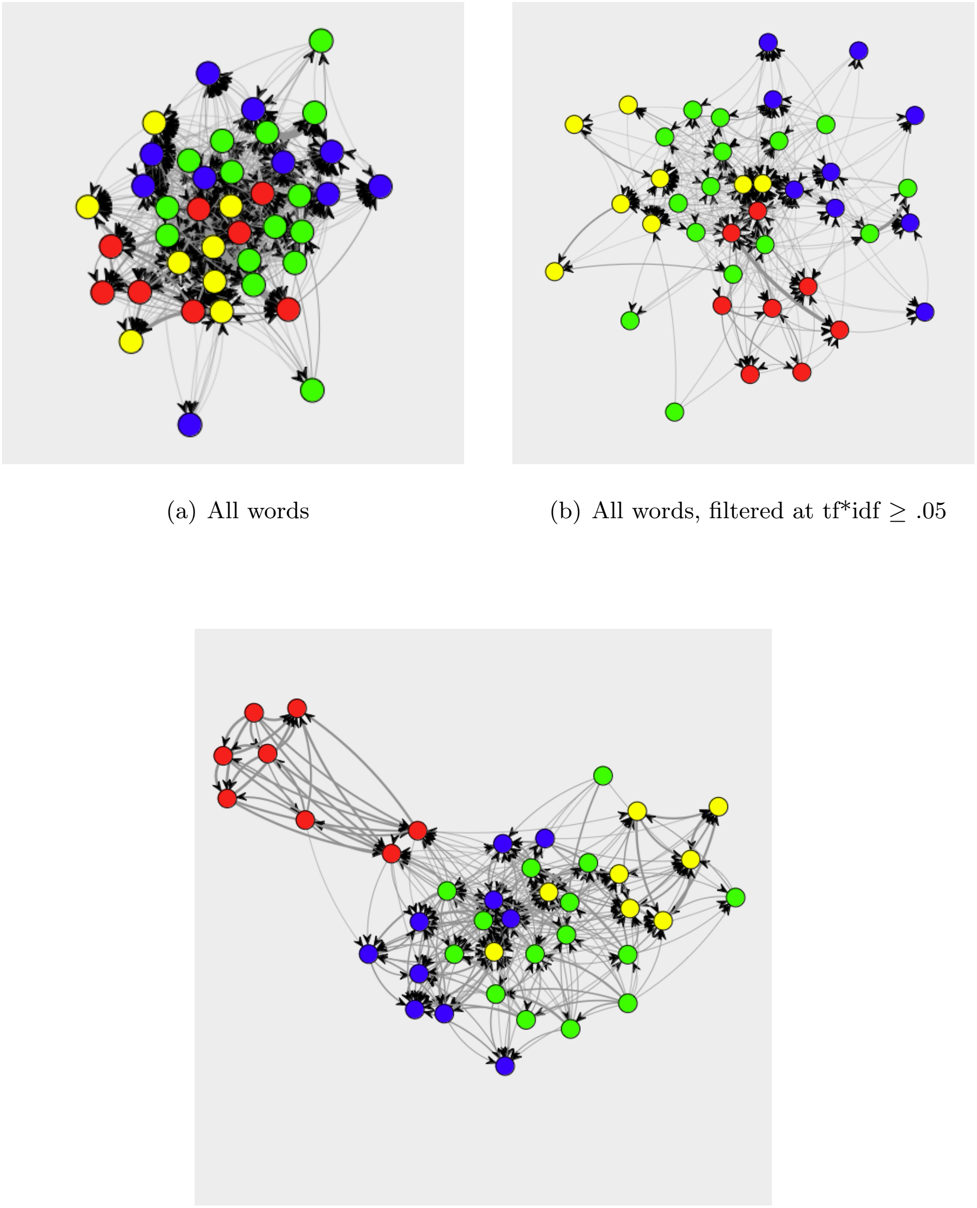
6.3.1 Distance Function

We use a linear combination of three features to measure the similarity between two documents. Our first feature is simply whether one of the papers cites the other. The second feature is co-citations (*i.e.* the number of times the two documents were cited within the same sentence in some document). Our final feature is the tf*idf cosine similarity of important words in the documents.

We began by using the tf*idf cosine similarity of the full documents, but quickly found that this method was far too noisy. Instead, we use our intuition that clustering should be across two dimensions – tasks and methods – and identify just those words from the documents. To identify these words, we first include the words found in each of the documents' titles, then add in task and method words found in the documents. We find these task and method words through patterns which are enumerated in Section 6.4.1. This set of words makes up our vocabulary and tf*idf cosine similarity is computed over only the words that appear in this vocabulary.

In Figure 6.2, we show the graph of the documents from our multi-document summarization development set using just the word similarity metric – first with tf*idf cosine similarity over all words, and then with tf*idf cosine similarity over only words found in the titles, the method words, and the task words. The documents are colored by their correct clustering (*i.e.* if two nodes (documents) share the same color, they belong in the same clustering). From this set of graphs, one can see that using only the filtered set of words, allows for a much more natural clustering, and indeed, the Rand index (Rand, 1971) when using the filtered set of words is much higher.

No single feature is a perfect distance metric. Instead, we combine these three features in a weighted sum for our final distance metric. We set the weights of the features by



(a) All words

(b) All words, filtered at tf*idf ≥ .05

(c) Title words, task words, and method words

Figure 6.2: Clustering of multi-document summarization papers using tf*idf with all words included versus just title words, task words, and method words. Documents are colored according to their correct cluster.

maximizing the Rand index of the clustering over our development set. The development set consisted of several sets of scientific articles (*e.g.* a set of articles on MDS), each of which we manually clustered into semantically related sets of articles (*e.g.* articles on generic MDS versus update MDS).

6.3.2 Clustering Algorithm

After defining the distance metric, we must cluster the documents. We choose to use the Markov Cluster Algorithm (MCL) (van Dongen, 2000). MCL is a fast and scalable graph clustering algorithm. MCL performs clustering by simulating flow across the graph with two algebraic operations: expansion and inflation. Expansion models the spreading of flow and inflation models the contraction of flow. MCL simulates flow spreading out within the natural clusters in the graph and contracting between the diffent clusters. MCL has the additional advantage of not requiring the number of clusters to be specified beforehand.[1]

Occasionally, the clustering algorithm fails to generate a balanced clustering (one or two documents only are present in one cluster, with all other documents in the other cluster). In these cases, the distance metric rather than the clustering algorithm is generally at fault. When the system detects an unbalanced clustering, we resort to temporal clustering, simply splitting the documents into equal sized sets, organized by date of publication.

6.4 Hierarchical Summarization

After hierarchically clustering the documents, SciSumma performs summarization over the clustering. As in G-Flow and Summa, SciSumma emphasizes three characteristics: (1) coherence, (2) salience, and (3) redundancy. We discuss how we measure each of these characteristics in the next sections.

6.4.1 Coherence in Scientific Document Summarization

Once again, coherence is measured via a discourse graph, however, building the discourse graph for scientific documents is quite different than building it for news articles. While

[1] We use Stijn van Dongen's implementation available from `http://micans.org/mcl/`

Pattern	Example Sentence
.* (goal|objective) of .* is .*	The goal of relation extraction is to detect semantic relations . . .
.* aim[s]? (to|at) .*	MDS systems aim to summarize sets of related articles.
.* is (a|an|the) (task|area) .*	Why-qustion answering is the task of answering why questions.
.* involves .*	Textual entailment involves identifying inferences in text.
.* \([A-Z][A-Z]+\) .*	In Sentiment Analysis (SA), systems seek to determine the . . .

Table 6.1: Patterns used to identify definitional sentences in scientific articles.

news articles were dominated by events (such as an attack or an airplane crash), scientific articles are dominated by abstract concepts (such as semantic parsing). Our coherence graph should reflect this difference. Instead of indicators like deverbal noun references, we will look for indicators that show a progression of an abstract concept. To build the graph, we recognize six types of coherence indicators, related below. For an example of a coherence graph over scientific papers, see Figure 6.3.

Citations We exploit the citations in scientific documents for edges in the discourse graph. Specifically, if sentence s_i contains a citation to paper p, then we will add an edge to sentence s_j if either of the following conditions holds: (1) s_j also references p or (2) s_j is drawn from p and contains self-referencing words or phrases:

s_1 *Mei and Zhai (2008) introduced citation-based summarization.*

s_2 *Mei and Zhai (2008) used language models to model the citation context and original content of the paper.*

Definition and Topic Continuation We also include edges between sentences in which the first sentence s_i is a definitional sentences and the second sentence s_j is related to the topic being defined.

To identify definitional sentences, we use a small set of patterns enumerated in Table 6.1.

s_3 *The goal of **MDS** is to produce quality summaries of collections of related documents.*

s_4 *(Aker et al., 2010) proposed an A * search approach for the task of **MDS**.*

Taxonomy and Topic Continuation Taxonomy sentences are particularly useful in hierarchical summarization. These sentences list a set of methods for a task or subtasks within a larger task:

s_5 *Approaches to abstractive summarization include sentence compression, sentence fusion or revision, and generation based approaches.*

s_6 *Compression methods aim to reduce a sentence by eliminating noncrucial constituents.*

We identify taxonomy sentences by looking for comma separated lists of simple noun phrases (potentially with a citation following each noun phrase), and identify the destination sentences simply by matching the phrases that appear in the taxonomy list.

Method/Task Continuation Like entity linking, from Section 3.2.2, in scientific documents, we consider two sentences to have a coherence relation if they relate to the same method or task.

Both tasks and methods are identified by the definitional patterns. Additionally, SciSumma identifies methods simply by looking for adjectives (*e.g.* linear) or past-participles (*e.g.* unsupervised) before the words 'approach,' 'method,' or 'model' (we apply a stop list to filter common words such as 'traditional,' or 'similar'). SciSumma also identifies tasks by finding sentences that contain 'problem of,' 'task of', and 'system for.' This forms the vocabulary of methods and tasks.

s_7 *Sentence compression methods were introduced by Knight and Marcu (2000).*

s_8 *Cohn and Lapata (2009) proposed a tree-to-tree transduction method for sentence compression.*

Co-referent Mentions As for news articles, any sentences sharing the same co-reference mention should have an edge in our discourse graph.

s_9 *In this paper, we propose a submodular method for document summarization.*

s_{10} *We apply this method to the DUC 2004 data and show statistically significant results.*

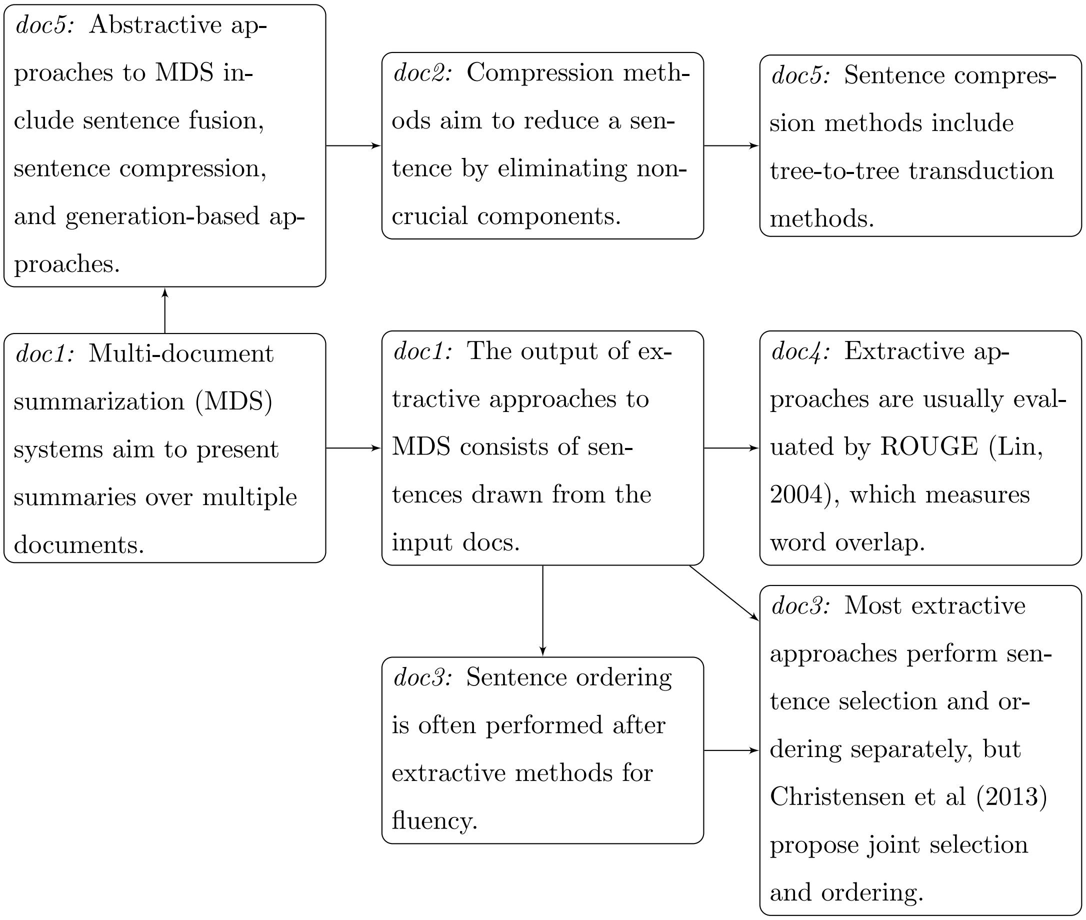

doc5: Abstractive approaches to MDS include sentence fusion, sentence compression, and generation-based approaches.
doc2: Compression methods aim to reduce a sentence by eliminating non-crucial components.
doc5: Sentence compression methods include tree-to-tree transduction methods.
doc1: Multi-document summarization (MDS) systems aim to present summaries over multiple documents.
doc1: The output of extractive approaches to MDS consists of sentences drawn from the input docs.
doc4: Extractive approaches are usually evaluated by ROUGE (Lin, 2004), which measures word overlap.
doc3: Sentence ordering is often performed after extractive methods for fluency.
doc3: Most extractive approaches perform sentence selection and ordering separately, but Christensen et al (2013) propose joint selection and ordering.

Figure 6.3: An example of a scientific document discourse graph covering research methods in multi-document sumamrization (MDS). Each node represents a sentence from the original documents and is labeled with the source document id. A coherent summary should begin with MDS and then describe various approaches. Sentences are abbreviated for compactness.

Discourse Markers We use the same discourse markers from Section 3.2.3 to identify adjacent sentences connected by explicit discourse cues:

s_{11} *Recently, several papers have proposed methods for citation based summarization.*

s_{12} *However, generating surveys from scientific articles has been largely ignored.*

6.4.2 Parent-to-child Coherence

Parent-to-child coherence represents how coherent the child summary is in the context of the parent sentence, or in other words, how well the child summary flows from the parent sentence. Parent-to-child coherence is especially important because users navigate the hierarchy by clicking on parent sentences and retrieving child summaries. The parent sentence must have positive evidence of coherence with the sentences in its child summary.

We originally estimated parent to child coherence as the coherence between a parent sentence and each sentence in its child summary exactly as we did for news hierarchical summaries:

$$PCoh(X) = \sum_{c \in C} \sum_{i=1..|X_c|} w_{G+}(x_c^p, x_{c,i}) \tag{6.1}$$

where x_c^p is the parent sentence for cluster c and $w_{G+}(x_c^p, x_{c,i})$ is the sum of the positive edge weights from x_c^p to $x_{c,i}$ in the ADG G. However, this formulation presents an important problem for scientific papers.

Citation links should become less valuable the more the same citation is used in a given child summary. Otherwise, all sentences in the child summary could be from the same citation, despite the parent sentence including multiple different citations:

p_1 *Generative topic models for MDS have been studied by Haghighi and Vanderwende (2009), Celikyilmaz and Hakkani-Tur (2010), and Li et al. (2011).*

s_{13} *Celikyilmaz and Hakkani-Tur (2010) introduced a two-step hybrid model for MDS.*

s_{14} *Celikyilmaz and Hakkani-Tur (2010) score sentences based on their latent characteristics via a hierarchical topic model.*

To account for this problem, we adjust our calculation of parent-to-child coherence to provide a backoff for citation edges. Each additional citation edge from the same citation is weighted by half as much as the previous edge.

6.4.3 Intra-cluster Coherence

In the previous chapter, we discussed how the primary criteria for good intra-cluster coherence is a lack of negative coherence. While in the previous chapter we measured intra-cluster coherence as minimizing the number of *missing references*, we will add to that undefined tasks in this chapter. References are missing and tasks are undefined if none of the sentences read before (either in an ancestor summary or in the current summary) contain an antecedent or definition:

$$CCoh(X) = -\sum_{c \in C} \sum_{i=1..|X_c|} \#missing(x_{c,i}) \tag{6.2}$$

6.4.4 Salience

Coherence is clearly not enough for good summaries. Good summaries must also cover the most important information within the input documents. To measure the salience of the entire summary, we simply sum up our estimation of the salience of each individual sentence in the summary:

$$Sal(X) = \sum_{i} Sal(x_i) \tag{6.3}$$

This methodology is the same as that used for generating short summaries in Chapter 3 and news hierarchical summaries in Chapter 5

For news articles, we measured the salience of a sentence using a classifier we trained on the DUC'03 dataset. Each sentence in the input documents served as a training instance x_i and the sentence's label y_i was generated automatically using ROUGE scores over the manually written summaries.

Unfortunately, there is no existing comparable corpus of collections of scientific papers and corresponding manually written summaries. Instead, we automatically create a corpus using the ACL anthology (Radev et al., 2009). We first search the documents for 50 related work sections that contain at least 10 references to documents also in the ACL anthology. We collect the documents that each of the related work sections cited. For our training data, each sentence from a cited paper serves as training instance, and the label is once again automatically generated using ROUGE scores, this time over the corresponding related work section.

We train a simple linear regression classifier, and supplement the features used for news salience regression to account for differences in scientific papers. The full list of features is shown in Table 6.2.

The most important features are the overlap with the words in the titles, the number of sentences the common nouns appear in, the number of sentences the proper nouns appear in, and the number of references in the sentence. We tried many other features which were not useful, including the position in the document, the citation count of the paper the sentence was drawn from, whether the sentence appeared to be a definitional sentence, and whether the sentence was a "summary" statement, *i.e.* included words like 'propose,' 'introduce', or 'present.'

6.4.5 Redundancy

Lastly, we use a logisitic regression classifier to determine if two sentences are redundant. This classifier is identical to that used in Chapter 5 for hierarchical summaries of news articles.

6.4.6 Objective Function

Having changed the components to fit this new domain, we can use the same objective function that we used for hierarchical summarization of news. We maximize the salience and the coherence of the hierarchical summary, and each of the constraints from before apply (each cluster summary must fit within the given budget, no redundant sentences,

weight	feature
-0.1239	contains money
0.0313	sentence length > 20
0.3492	number of sentences common nouns appear in
0.1219	number of sentences proper nouns appear in
0.1616	number of sentences verbs appear in
0.1756	number of references in the sentence
-0.034	if the sentences was from the abstract
0.0276	if the sentence was from the related work section
0.4855	the word overlap with the words contained in the titles

Table 6.2: Linear regression features and learned weights for salience. Sentences classified were drawn from the Abstract, Introduction, Related Work, and Conclusion sections.

each summary sentence must be drawn from the cluster it corresponds with, and each cluster summary must have the same number of sentences as the corresponding cluster has children).

$$\begin{aligned} \textbf{maximize:} \quad & F(x) \triangleq Sal(X) + \beta PCoh(X) + \gamma CCoh(X) \\ s.t. \quad & \forall c \in C : \textstyle\sum_{i=1..|X_c|} len(x_{c,i}) < b \\ & \forall x_i, x_j \in X : \text{redundant}(x_i, x_j) = 0 \\ & \forall c \in C, \forall x_c \in X_c : x_c \in c \\ & \forall c \in C : |X_c| = \#children(c) \end{aligned}$$

6.4.7 Algorithm for Approximation

Once again we approximate a solution to the objective function listed above. After experimenting with the algorithm proposed for hierarchical summarization of news from Section 5.3.5, we identified a problem with this framework for scientific papers. By filling in the

```
function fillInClusterSummary

    Inputs:
     beam limit $B$
     partial summaries to return limit $K$
     beam of partial hierarchical summaries $H = \{X^1, \ldots, X^B\}$
     cluster summary index $i$ to be filled in of the hierarchical summary

   Output:
       beam of partial hierarchical summaries $H = \{X^1, \ldots, X^B\}$

   for $j = 1, \ldots, M$ do // For each slot in the current cluster summary

      // Find the best partial hierarchical summaries with the current slot filled in
      $\hat{H} = \{\}$ // $\hat{H}$ will store the partial hierarchical summaries with slot $j$ filled in
      for $b = 1, \ldots, B$ do // For each partial hierarchical summary in the beam
         $x_{i,j} = X_i^b$ // Summary slot to fill in $x_{i,j}$ of current partial cluster summary $X_i^b$
         $P =$ getTopKSummaries($X_i^b, x_{i,j}, K$) // Get the $K$ best partial summaries with $x_{i,j}$ filled in
         $\hat{H} =$ addToQueue($P$) // Add the returned partial hierarchical summaries to the priority queue
      end for
      $H =$ getTopN($\hat{H}, B$) // Beam is the top $B$ partial hierarchical summaries identified with slot $j$ filled in

      // Find the best partial hierarchical summaries with the current slot's child summary filled in
      if isParentSentence($x_{i,j}$) then
         $l =$ getChildIndex($H, i, j$) // Get the index of child $j$ of cluster summary $X_i$
         $H =$ fillInClusterSummary($B, K, H, l$) // Get the top partial summaries with $X_l$ filled in
                                                  // The beam is now those partial hierarchical summaries
      end if

   end for
   return $H$
end function
```

Figure 6.4: SciSumma's algorithm for approximating a solution to the objective function. This approximation algorithm differs from Summa's in that Summa fills in the summary slots in level-order and SciSumma fills in the slots in pre-order.

summary slots in level-order, the choice of each sentence is in part determined by its sibling sentences because the beam allows for backtracking after looking at sibling sentences.

While sibling sentences are useful for choosing current sentences in the news domain (major events are referenced even months later), in scientific papers, the children and grandchildren are better indicators than the siblings. Recall that each sentence is chosen after approximating its value by jointly choosing its child summary. Thus by filling in the children after the parent sentence, the grandchildren also have influence. Definitional sentences are far more likely to be chosen when the children and grandchildren have more influence because these sentences are likely to rely on those definitions. Sibling sentences may not require those definitions, because clusters are often separated based on task or method. Thus, we fill in the slots in pre-order rather than level-order. Otherwise, the algorithm is identical to that described in Section 5.3.5. See Figure 6.4 for pseudocode of the algorithm.

6.4.8 Postprocessing of Sentences

Finally, we perform postprocessing of the sentences in the hierarchical summary to resolve self-references such as 'we' or 'this paper.' We use simple pattern matching to identify these instances and change them to standard references. For example, this procedure would resolve the following:

s_{14} *In this paper, we introduce a submodular approach to MDS that builds on our work in (Lin and Bilmes, 2010).*

s_{15} *Lin and Bilmes (2011) introduce a submodular approach to MDS that builds on their work in (Lin and Bilmes, 2010).*

6.5 Experiments

In this section, I describe experiments for our scientific document hierarchical summarization system. Specifically, we would like to answer two questions:

1. **User Preference** Do people prefer the hierarchical format of SciSumma for multi-document scientific article summarization than other state-of-the-art methods?

2. **Knowledge Aquisition** Do people learn more with SciSumma's hierarchical format than with other state-of-the-art methods?

We investigated these questions through a series of user studies analogous to those described in 5.4 for hierarchical summarization of news articles.

6.5.1 Systems

To the best of our knowledge, there are no existing options for large-scale summarization of scientific articles, and timelines will likely be inappropriate for scientific document summarization. For this reason, we only compared SciSumma to one other system – a system that performs multi-document summarization of scientific articles called C-LexRank (Qazvinian and Radev, 2008).

C-LexRank was first proposed by Qazvinian and Redev (Qazvinian and Radev, 2008) for citation-based summarization and was later applied to multi-document summarization of scientific articles (Mohammad et al., 2009). C-LexRank creates a fully connected network in which the sentences are nodes and the edges represent the cosine similarity of two sentences. The graph is pruned by applying a cutoff value of 0.1, after which the largest connected component is extracted and clustered. C-LexRank uses LexRank to calculate the most salient sentences of each cluster, which are then extracted in decreasing order of salience until the summary budget is reached.

C-LexRank is designed for short, flat summaries, but for our experiments, we increase the budget to be equal to that we give to SciSumma.

6.5.2 Budget

In hierarchical summarization, the budget is per cluster summary, rather than for the entire hierarchical summary. SciSumma is given a budget of 665 bytes for each cluster summary (the traditional MDS budget). C-LexRank is given the same total budget that SciSumma was allowed (multiply 665 bytes by the number of cluster summaries in the hierarchical summary for the given topic). This total budget comes to over 10 times the traditional MDS budget.

Topic	Number of Articles	Number of Sentences
Coreference Resolution	46	1779
Dialogue Systems	81	4068
Textual Entailment	60	2225
Opinion Mining	52	2531
Semantic Parsing	52	2000
Sentiment Analysis	56	3789
Semantic Role Labeling	36	1786
Twitter	60	2600

Table 6.3: Topics and number of articles and sentences per topic in the test set for our scientific document experiments. Only sentences in the abstracts, introductions, conclusions, and related work sections are counted. Overall, the clusters had many fewer articles than in our news hierarchical summarization experiments, but the number of sentences is similar because scientific articles are much longer.

6.5.3 Datasets

We used eight clusters of scientific articles for our experiments. We manually generated each cluster through a series of keyword searches and pruning in order to achieve clusters that covered a wide range of important research in each area.

All articles were drawn from the ACL Anthology Network Corpus (Radev et al., 2013). In addition to the pdfs of the papers, the ACL Anthology Network also provides a citation graph of the documents included in the corpus. The corpus contains papers from the ACL venues, which are focused on research in Natural Language Processing, so each of our clusters represents an area of Natural Language Processing. The clusters that we chose are displayed in Table 6.3.

In our experiments, we used only sentences drawn from the abstracts, introductions, related work sections, and conclusion sections.

6.5.4 User Preference Experiment

In this first experiment, our goal is to evaluate user preference. In previous chapters, we have described Amazon Mechanical Turk experiments, but scientific article summaries require some technical background to fully understand. Accordingly, we asked nine Computer Science graduate students and five Computer Science undergraduate students to take part in the study. The full user study took about an hour, and students were compensated for their time with a $15 gift card for Amazon.com.

Each student read about each of the eight topics alternating between hierarchical summaries and flat summaries. Half of the students read the summaries beginning with a hierarchical summary and the other half began with a flat summary. Each student was allowed four minutes to read through each summary.

After reading the summary, the student was asked to write a short description of what he or she had learned. We encouraged the students to take only three minutes or less to write the description, but did not set a hard limit. Students were not allowed to view the summary while writing the description, primarily to prevent them from copying information over. Table 6.4 shows examples of the descriptions that students wrote. Students were also asked to rate their prior knowledge of the topic. See Figure 6.5 for an example of the user interface for reading one of the hierarchical summaries.

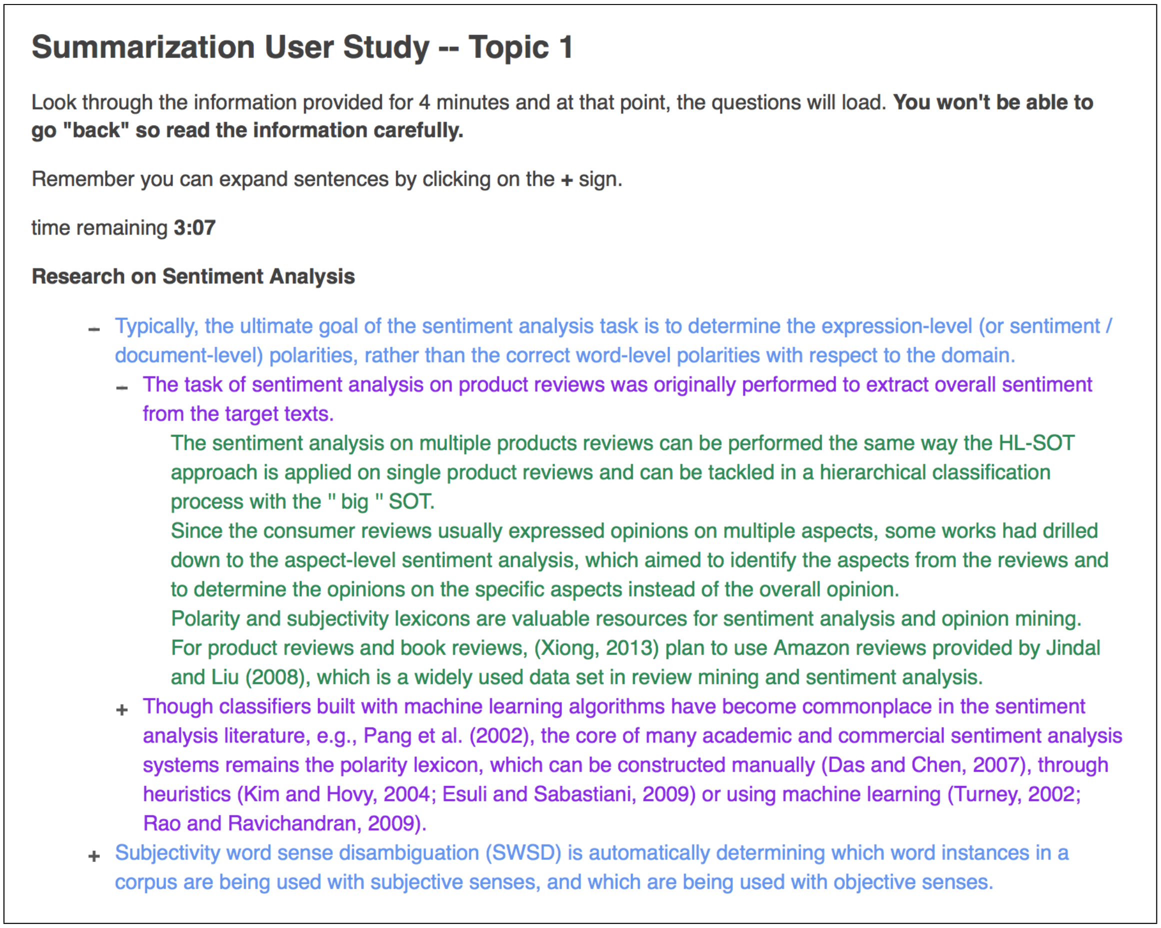
After reading eight summaries and writing eight descriptions, the students were asked to describe what they liked or disliked about each system and to choose which system they preferred. Below we show the precentage of students who chose each system:

User Preference			
SciSumma	**71%**	C-LexRank	29%

Overall, SciSumma was preferred more often than the flat summaries. Students liked that the information was organized and that the most important information and overview information was towards the top of the hierarchy. Some students complained that they had to click through the information to reach the lower levels. This problem is partly an artifact of the setup of the experiments. Hierarchical summaries are intended for settings in which we do not know how much the user wishes to read, however in this evaluation, the students

	Example descriptions written for SciSumma	Example descriptions written for C-LexRank
Sentiment Analysis	The primary goal of sentiment analysis is to automatically determine the sentiment of expressions and documents such as book and product reviews holistically, rather than individual words. Traditionally, such analysis rely on the use of individual words classified in a 'polarity lexicon' (learned automatically or manually). A technique called SWSD can help determine which word instances are used in a subjective sense, and which ones are used objectively. Research shows SWSD algorithms help improve the sentiment analysis correctness measures at the expression level.	Identifying sentiments in tweets is challenging. To determine if a sentience has a positive or a negative sentiment, one not only has to consider the positive or negative subjectives in the sentence, but also the context of the whole sentence. Some sentences have many positive words yet the overall sentiment becomes negative examing in the context.
Semantic Parsing and Grounded Language Learning	Traditional NLP semantic parsing learning algorithms rely extensively on a pre-annotated corpus. These techniques often use rule-based systems and pattern matching words to grammatical roles (Who, what, when, why). A newer approach (based on grounded theory) seems to be emerging (a bit less established though), which considers the 'state of the observed world' – much as a child learning a language would use social feedback to learn.	Semantic parsing is a very difficult task. A good semantic parser demands supervised learning methods and annotated training data. However, gaining annotated training examples are expensive and time-consuming. A new algorithm is proposed to learn the semantic parser in an unsupervised way, which enables the researcher to build a new semantic parser using the semi-supervised manner.
Textual Entailment	Textual entailment is the act of determining whether natural language text T entails (implies) the hypothesis H or not. This is an important problem, as it is used in tasks such as text summarization and question answering. The TE problem may be monolingual or H and T may be in different languages. Recognizing Text entailment (RTE) is difficult as modeling the syntax and semantics are hard. Positive and negative examples, RTE entailment graphs, bag of word probabilistic models, and automatic rule learning can assist with this task.	Semantic Entailment Recognition is an emerging field of research. When human agree that the meaning of H can be inferred by the meaning of T, we can say T entails H. It's an very interesting task if a NLP algorithm can recognize entailment relations in the natural language. SER in multiple languages is even difficult task. Machine learning algorithms have been applied to solve this SER problem and show promising results.

Table 6.4: Examples of descriptions written by students who read the system output.



Summarization User Study -- Topic 1

Look through the information provided for 4 minutes and at that point, the questions will load. **You won't be able to go "back" so read the information carefully.**

Remember you can expand sentences by clicking on the **+** sign.

time remaining **3:07**

Research on Sentiment Analysis

- − Typically, the ultimate goal of the sentiment analysis task is to determine the expression-level (or sentiment / document-level) polarities, rather than the correct word-level polarities with respect to the domain.
  - − The task of sentiment analysis on product reviews was originally performed to extract overall sentiment from the target texts.
    - The sentiment analysis on multiple products reviews can be performed the same way the HL-SOT approach is applied on single product reviews and can be tackled in a hierarchical classification process with the " big " SOT.
    - Since the consumer reviews usually expressed opinions on multiple aspects, some works had drilled down to the aspect-level sentiment analysis, which aimed to identify the aspects from the reviews and to determine the opinions on the specific aspects instead of the overall opinion.
    - Polarity and subjectivity lexicons are valuable resources for sentiment analysis and opinion mining.
    - For product reviews and book reviews, (Xiong, 2013) plan to use Amazon reviews provided by Jindal and Liu (2008), which is a widely used data set in review mining and sentiment analysis.
  - + Though classifiers built with machine learning algorithms have become commonplace in the sentiment analysis literature, e.g., Pang et al. (2002), the core of many academic and commercial sentiment analysis systems remains the polarity lexicon, which can be constructed manually (Das and Chen, 2007), through heuristics (Kim and Hovy, 2004; Esuli and Sabastiani, 2009) or using machine learning (Turney, 2002; Rao and Ravichandran, 2009).
- + Subjectivity word sense disambiguation (SWSD) is automatically determining which word instances in a corpus are being used with subjective senses, and which are being used with objective senses.

Figure 6.5: User interface for reading the hierarchical summaries during the evaluation. Users can click on the sentences to expand the hierarchy.

necessarily want to read as much as possible.

Students also remarked that the flat summaries lacked organization, and that they had to read through many details to get to the important information, making the summaries more confusing and difficult to read. This result underscores the importance of coherence and organization in summarization, and the need for systems which have both of these qualities.

Interestingly, of the nine graduate students who took part in the study, eight preferred the hierarchical format, but only two of the five undergraduates preferred the hierarchical summaries to the flat summaries. The undergraduates generally had much more trouble with the study, and it may be that the organization of the hierarchical summaries was far less clear to students who have little experience in reading about research topics.

6.5.5 Knowledge Acquisition Experiment

As in our experiments for hierarchical summarization for news articles, user preference is not the only important factor. We are also interested in how much people can learn from each of the systems. We once again followed (Shahaf et al., 2012b) and compared the descriptions written by students in the user preference experiment to see whether the students learned more when they read the hierarchical summaries or the flat summaries. Due to time constraints, ten of the fourteen students were evaluated in this part of the study.

We paired up each of the students so that one student started with a hierarchical summary and the other started with a flat summary. By pairing up the students, we were able to control for ability (or enthusiasm for the task). We also deliberately paired up graduate students with other graduate students and undergraduates with other undergraduates.

Four experts in Natural Language Processing performed a blind, randomly-ordered comparison of the descriptions written by each pair of students and chose which person they believed had learned more. For each pair of descriptions, we then used the majority vote as the preferred description or “indifferent” if the annotators were tied.

The results for this experiment are as follows:

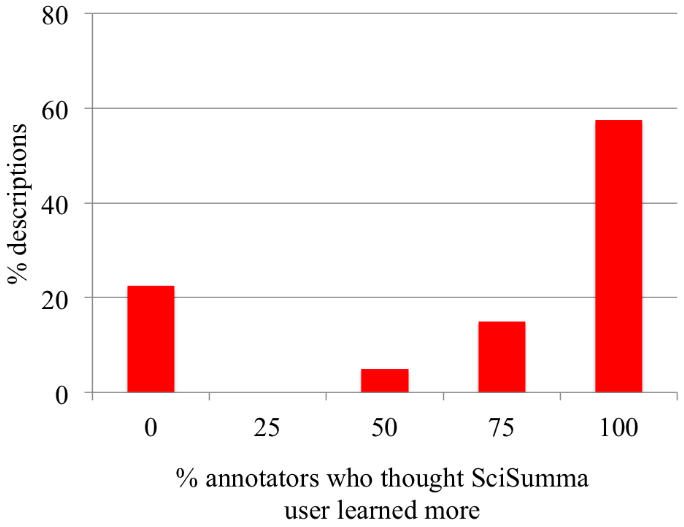


Figure 6.6: The percentage of description pairs that had each percentage of annotator votes. Annotators chose the description they thought demonstrated a better understanding of the topic. Here we show the distribution of votes. In 80% of cases, the annotators were in full agreement.

Knowledge Acquisition					
SciSumma	**72.5%**	Indifferent	5%	C-LexRank	22.5%

Figure 6.5.5 shows the distribution of votes from the annotators. Generally, people learned more with the hierarchical summaries. The difference was greater than what we observed against flat summaries and timelines in the previous chapter for news articles. This result appears to be due to the complexity of learning about scientific research. The importance of correct organization likely grows with the complexity of the information to be summarized.

We also analyzed how often the annotators agree. If the annotators rarely agree, one can assume that the differences in the amount learned are small, whereas if the annotators often agree, the differences should be larger. In Figure 6.5.5, we show the percentage of annotators who thought that the SciSumma user learned more versus the percentage of descriptions. In around 22% of the descriptions, all annotators believed the C-LexRank user learned more, and in about 58% of the descriptions, all annotators believed the SciSumma user learned more. In 80% of cases, the annotators were in full agreement. In 15% of cases, three quarters of the annotators thought the SciSumma user learned more, and in 5% of cases, half the annotators voted for the C-LexRank user and half voted for the SciSumma user. These numbers suggest that it was often quite clear which user had learned more to the annotators.

6.6 Discussion

In this section, we discuss topics related to SciSumma that require future research and development.

6.6.1 Beyond Computer Science Documents

While we have aimed this section at scientific documents as a whole, we have only tested on documents drawn from Computer Science research. We believe that the basic methodology will apply across domains, but the specific heuristics we have used will not. In science, papers attack problems (*e.g.* how to parse a sentence or how stars form), and either provide a solution (*e.g.* methodology for parsing) or a discovery (*e.g.* older stars are less centrally concentrated than younger stars).[2]

In some ways, Computer Science articles are relatively easy to hierarchically summarize because they often clearly spell out their problems and findings. Many problems are well defined and can be referred to by a name (Parsing, Semantic Role Labeling, Coreference Resolution). Likewise, solutions have a variety of well known properties (supervised, machine learning, approximate). These qualities allow us to group documents together much more easily than in a domain with less well defined problems, solutions, and discoveries. Similarly, the redundancy of terms in Computer Science documents allows for easier calculations of saliency (Semantic Role Labeling will be repeated many times in papers on SRL).

To apply our ideas to other scientific domains, we will first need to be able to organize the articles into meaningful clusters. The lack of redundancy of terms makes this more

[2] Computer Science papers rarely report discoveries, so we have not focused on identifying discoveries in this paper.

challenging. Secondly, the discourse graph must be modified to account for the new domain. Once again the lack of well defined problems and approaches will create difficulties. Perhaps a fuzzier notion of problem and approach will be useful (*i.e.* instead of relying on specific names, a collection of terms can be used). Additionally, the notion of discovery must also be included in the discourse graph. Finally, salience metrics will likely require in domain training data (although the methods we proposed in this chapter of automatically generating training data should translate across scientific domains), and features specifically targeted at the current domain should be included.

6.6.2 Generating the Input Set

For news articles, we were able to simply select the input document sets by automatically choosing the articles with the highest tf*idf cosine similarity with respect to a query. For scientific articles, such simple methods do not produce good results. Part of the problem is the distribution of good or relevant sentences in scientific articles as compared to news. While many news articles will contain sentences relevant to popular queries (*e.g.* many articles will contain important information about the 1998 embassy bombings), there are many scientific articles that simply confuse the system. For example, querying for "semantic role labeling" brings up many relevant articles, but also many articles that relate to side issues, such as how Semantic Role Labeling can be used for Coreference Resolution. News is more likely to contain many highly concentrated articles around the query topic, while scientific articles will contain lots of articles that are only slightly related, but appear to be highly related.

For this first implementation, we have manually selected our datasets through a combination of queries. We relied on our own intuition of relevance which took into account the title and the citation count, and occasionally the abstract as well. Research in document clustering is orthogonal to the research proposed here, and thus we have not attempted a more sophisticated methodology. Nonetheless, for SciSumma to be useful to people as an end-to-end system, this problem must be solved satisfactorily.

6.7 Conclusions

In this chapter, we presented a method for building a discourse graph for scientific articles, and introduced a hierarchical summarization system, SciSumma, that leverages this graph to build summaries. Our system is a first step towards building long summaries of research on scientific topics. In experiments, we demonstrate that students prefer SciSumma to learn about research topics and that they learn more from reading SciSumma summaries. SciSumma represents a test of how well the ideas proposed in the previous chapter apply across domains. We found that the basic ideas of Summa were applicable to the scientific domain, but that many of the details, including clustering and coherence graph generation, required domain-specific modifications.

Chapter 7

CONCLUSION AND FUTURE WORK

This dissertation introduced a new task aimed at large scale summarization, which we call *hierarchical summarization.* Hierarchical summarization is designed for situations in which the user has a general interest in a complex topic that covers a wide range of information. We have focused on the problem of identifying relevant information in a set of documents and collating it into a coherent whole.

Hierarchical summarization mimics how a user with a general interest would interact with a human expert. It first starts with a general overview summary, and then users can click on sentences to learn more about areas of interest. Hierarchical summarization allows for structured output, custom output length, personalization, and interaction. Chapter 2 discussed how current approaches are insufficient for large scale summarization, how multi-document summarization approaches do not provide structure or coherence to summaries, and how other solutions such as timelines lack the ability to extend to domains beyond news.

In this dissertation, I presented four technical contributions to the problem of large scale summarization. My first contribution is G-Flow, a system for *coherent multi-document summarization.* G-Flow is a first step to hierarchical summarization, as well as a state-of-the-art multi-document summarization system. This system has at its core a discourse graph that can be used to estimate the coherence of a candidate summary. The discourse graph's nodes represent sentences from the input and the discourse graph's edges represent pairwise ordering constraints between sentences. This system performs joint sentence selection and sentence ordering, maximizing coherence and salience. In experiments, we found that users substantially preferred the summaries produced using this framework.

As my second contribution, I introduced the paradigm of hierarchical summarization. My third contribution is the first system for hierarchical summarization of news articles, Summa. This system uses the discourse graph from Chapter 3, and identifies the structure of the summary using hierarchical clustering. Manual evaluations demonstrated that users substantially preferred hierarchical summaries to flat summaries and timelines.

My final contribution is a system for hierarchical summarization of scientific articles, SciSumma. I discussed what ideas translated across domains and what ideas need to be modified for this new domain. I described how to create a discourse graph for scientific articles. Manual evaluations once again showed that SciSumma was preferred over other solutions for large scale summarization. Additionally, we found that SciSumma users learned more over three times as often as those who used a flat multi-document summary.

This final chapter describes open problems and ideas for future work in this area.

7.1 Refining the Discourse Graph

All of the systems proposed in this dissertation rely heavily on the discourse graph. However, the indicators that are used to build the graph are quite noisy. An important extension to this work would be to refine the discourse graph either by how the edges are chosen or how the edges are weighted.

One way of weighing the edges is to use the final output summaries as a form of weak training data. One could compare human ratings of output summaries and train the weights using this data. A downside of this methodology is that the training data will necessarily be sparse, meaning the system may produce good summaries that are not labeled.

7.2 Hierarchical Summarization in Real Time

We have not prioritized processing time, and many of the algorithms proposed here are relatively slow (5-10 minutes to generate a hierarchical summary covering 300 documents). One possible extension is to make the systems work in real-time. A real-time system must not only be able to quickly process data, but should be able to react to streaming input data and quickly alter the current summary with respect to the new data. This framework may pose a problem for the two-step nature of our system design. Perhaps the data will only need to be reclustered under certain conditions and only the clusters effected by the incoming data will need to be summarized again.

Related to this problem is relaxing the assumption of the dataset size. Ideally, the systems would be able to handle much larger datasets. These datasets are arguably more interesting and a better use case for this problem than those investigated here because they are much more tedious for users to research using current technology.

7.3 User Interface

Another extension is to refine the user interface. There is extensive research on news reading interfaces (*e.g.* (Gabrilovich et al., 2004; Wagner et al., 2009)), and an interesting project would be to investigate possible platforms and build a working demo.

One element missing from hierarchical summarization is a way to quickly visualize interactions between related events. Indeed, a few users complained that the relationship between the parent sentences and the child summaries was not always clear or that it was not obvious from the parent sentences what the child summaries would cover.

We could consider potential ways to combine the summarization ideas in this dissertation with a system such as Metro Maps (Shahaf et al., 2012b). Metro Maps are structured sets of documents analogous to the metro maps used to navigate subways. See Figure 7.1 for an example of a Metro Map. These maps visualize how threads of documents relate to one another. While Metro Maps work at the document level and show interactions, hierarchical summarization works at the sentence (or phrase) level and describes a topic in increasing detail. Because these techniques target different, distinct areas of information overload, a system which combines both could be quite useful.

7.4 Query Focused Hierarchical Summarization

Hierarchical summarization systems are designed to identify relevant information in a set of documents and produce a coherent summary of that information. An obvious extension to this work is to generate focused summaries that respond to a user's query. At present, the system is limited by its summarization-style approach. Instead of targeting the response to the user's query, the system simply attempts to summarize what is in the input documents.

This problem could be approached on two sides. The first is in the selection of the input data. A preprocessing step could simply filter out any sentences not focused on the question

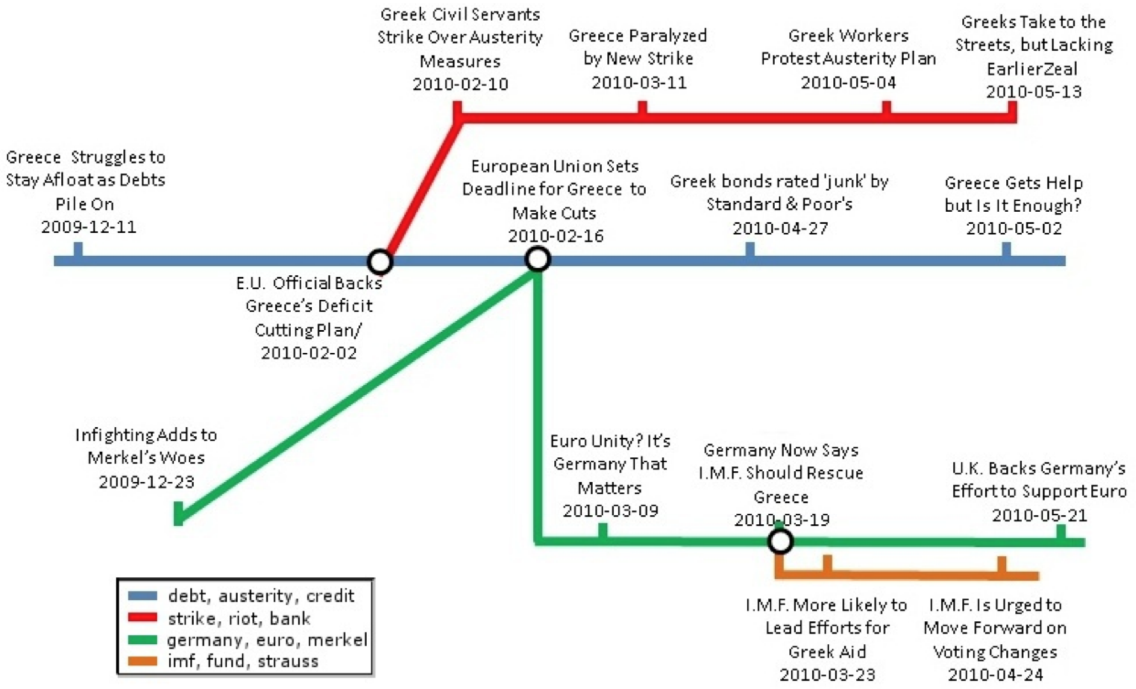

Greek Civil Servants Strike Over Austerity Measures 2010-02-10
Greece Paralyzed by New Strike 2010-03-11
Greek Workers Protest Austerity Plan 2010-05-04
Greeks Take to the Streets, but Lacking EarlierZeal 2010-05-13
Greece Struggles to Stay Afloat as Debts Pile On 2009-12-11
European Union Sets Deadline for Greece to Make Cuts 2010-02-16
Greek bonds rated 'junk' by Standard & Poor's 2010-04-27
Greece Gets Help but Is It Enough? 2010-05-02
E.U. Official Backs Greece's Deficit Cutting Plan/ 2010-02-02
Infighting Adds to Merkel's Woes 2009-12-23
Euro Unity? It's Germany That Matters 2010-03-09
Germany Now Says I.M.F. Should Rescue Greece 2010-03-19
U.K. Backs Germany's Effort to Support Euro 2010-05-21
I.M.F. More Likely to Lead Efforts for Greek Aid 2010-03-23
I.M.F. Is Urged to Move Forward on Voting Changes 2010-04-24
🟦 debt, austerity, credit
🟥 strike, riot, bank
🟩 germany, euro, merkel
🟧 imf, fund, strauss

Figure 7.1: An example of a metro map produced by Shahaf et al. (2012b). This map illustrates the development of the Greek debt crisis. The storylines show the austerity plans, the strikes and riots, and the role of Germany and the role of the IMF.

at hand. Alternatively, the additional sentences may provide some amount of insight into the organization of the documents or the salience of the sentences. They may also provide background information indirectly related to the query. It may be more advantageous to use the query when hierarchically clustering the sentences and when estimating salience. For specific types of questions, such as biographies, the system could also leverage the expected structure of the infomarion. See Section 2.1.3 for a discussion of previous work on query focused summarization.

7.5 Global Coherence

In this dissertation, we have made extensive use of our discourse graph which identifies pairwise ordering constraints of the input sentences. However, this measure is of *local* coherence, not *global* coherence. While each pair of adjacent sentences in a summary should be coherent (assuming the discourse graph is accurate), there is no guarantee at all of global coherence across all sentences. An important next step in this work is to identify a way of measuring global coherence. This step will likely require a more advanced modeling of the information in the sentences and the development of the information across the sentences. One could incorporate ideas from frame induction (Chambers and Jurafsky, 2009; Cheung et al., 2013), which models event sequences, for news summarization in particular.

7.6 Abstractive Summarization

We have avoided any inclusion of abstractive summarization beyond refining the references in our work on scientific document hierarchical summarization. However, one complaint that we have received from users of our systems is that the sentences are quite long, and often include less important information in clauses.

We are very interested in testing sentence compression and fusion methods on our system. Potentially, we may be able to use insights from the disourse graph to assist with identifying which elements of the sentences are necessary and which are expendable.

7.7 Joint Clustering and Summarization

Both SUMMA and SCISUMMA first cluster the input information and then summarize over the clustering. Decomposing the problem into two steps simplifies the task substantially, but is also problematic. If the clustering is performed poorly, the system will not be able to recover.

The system could instead jointly cluster and summarize the information. The summarization step could provide feedback for the clustering step, possibly enabling a much more effective clustering. The largest stumbling block for this methodology is the processing time necessary. By first clustering, the system was able to substantially decrease the summary search space (the number of sentences in a summary was known and the set of potential sentences for each slot in the summary was much smalller).

7.8 User Feedback for Hierarchical Summarization

Finally, the interactive nature of the hierarchical summaries provides a natural feedback loop. Users click on sentences to expand the child summary and click again to collapse the summary. One could monitor users' interactions, and from those interactions learn what information is most valuable (what information is most often clicked), and what sentences are possibly misleading (if a user quickly collapses a summary). This data could then be integrated in the system to learn better salience metrics and better coherence relations.

BIBLIOGRAPHY

Amjad Abu-Jbara and Dragomir R. Radev. Coherent citation-based summarization of scientific papers. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 500–509, 2011.

Amr Ahmed, Qirong Ho, Jacob Eisenstein, Eric Xing, Alexander J. Smola, and Choon Hui Teo. Unified analysis of streaming news. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 267–276, 2011.

C. G. Akcora, M. A. Bayir, M. Demirbas, and H. Ferhatosmanoglu. Identifying breakpoints in public opinion. In *Proceedings of the First Workshop on Social Media Analytics*, SOMA '10, pages 62–66, 2010.

Ahmet Aker, Trevor Cohn, and Robert Gaizauskas. Multi-document summarization using A* search and discriminative training. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP'10, 2010.

Michele Banko, Vibhu O. Mittal, and Michael J. Witbrock. Headline generation based on statistical translation. In *Proceedings of the 38th Annual Meeting on Association for Computational Linguistics*, ACL '00, pages 318–325, 2000.

Michele Banko, Michael Cafarella, Stephen Soderland, Matt Broadhead, and Oren Etzioni. Open information extraction from the web. In *Proceedings of the 20th International Joint Conference on Artifical Intelligence*, IJCAI'07, pages 2670–2676, 2007.

Regina Barzilay and Michael Elhadad. Using lexical chains for text summarization. In *Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, pages 10–17, 1997.

Regina Barzilay and Mirella Lapata. Modeling local coherence: An entity-based approach. *Computational Linguistics*, 34(1):1–34, 2008.

Regina Barzilay and Lillian Lee. Catching the drift: Probabilisitic content models, with applications to generation and summarization. In *Human Language Technologies: The 2004 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '04, pages 113–120, 2004.

Regina Barzilay and Kathleen R. McKeown. Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–328, 2005.

Regina Barzilay, Noemie Elhadad, and Kathleen R McKeown. Sentence ordering in multidocument summarization. In *Proceedings of the First International Conference on Human Language Technology Research*, HLT '01, pages 1–7, 2001.

Berkhin Berkhin. A survey of clustering data mining techniques. *Grouping Multidimensional Data*, pages 25–71, 2006.

Steven Bethard and Dan Jurafsky. Who should I cite: Learning literature search models from citation behavior. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 609–618, 2010.

Fadi Biadsy, Julia Hirschberg, and Elena Filatova. An unsupervised approach to biography production using Wikipedia. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, ACL '08, pages 807–815, 2008.

Sasha Blair-Goldensohn, Kathleen R. McKeown, and Andrew Hazen Schlaikjer. Defscriber: A hybrid system for definitional qa. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 462–462, 2003.

Danushka Bollegala, Naoaki Okazaki, and Mitsuru Ishizuka. A bottom-up approach to sentence ordering for multi-document summarization. *Information Process Management*, 46(1):89–109, 2010.

Orkut Buyukkokten, Hector Garcia-Molina, and Andreas Paepcke. Seeing the whole in parts: Text summarization for web browsing on handheld devices. In *Proceedings of the 10th International Conference on World Wide Web*, WWW '01, pages 652–662, 2001.

Jaime Carbonell and Jade Goldstein. The use of MMR, diversity-based reranking for reordering documents and producing summaries. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 335–336, 1998.

Asli Celikyilmaz and Dilek Hakkani-Tur. A hybrid hierarchical model for multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 815–824, 2010.

Asli Celikyilmaz and Dilek Hakkani-Tür. Discovery of topically coherent sentences for extractive summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 491–499, 2011.

Nathanael Chambers and Dan Jurafsky. Unsupervised learning of narrative schemas and their participants. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 2 - Volume 2*, ACL '09, pages 602–610, 2009.

Angel Chang and Christopher Manning. SUTime: A library for recognizing and normalizing time expressions. In *Proceedings of the Eight International Conference on Language Resources and Evaluation*, LREC '12, 2012.

Jackie C.K. Cheung, Hoifung Poon, and Lucy Vanderwende. Probabilistic frame induction. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, NAACL-HLT '13, pages 837–846, 2013.

Hai Leong Chieu and Yoong Keok Lee. Query based event extraction along a timeline. In *Proceedings of the 27th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '04, pages 425–432, 2004.

James Clarke and Mirella Lapata. Discourse constraints for document compression. *Computational Linguistics*, 36(3):411–441, 2010.

Trevor Cohn and Mirella Lapata. Sentence compression beyond word deletion. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 137–144, 2008.

John Conroy, Judith Schlesinger, and Jade Goldstein Stewart. CLASSY query-based multi-document summarization. In *Proceedings of Document Understanding Conference*, DUC '05, 2005.

John M. Conroy and Dianne P. O'leary. Text summarization via hidden markov models. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 406–407, 2001.

Hoa Trang Dang and Karolina Owczarzak. Overview of the TAC 2008 Update Summarization Task. In *Proceedings of the First Text Analysis Conference*, TAC '08, 2008.

Dipanjan Das and André F. T. Martins. A survey on automatic text summarization. Technical report, Literature Survey for the Language and Statistics II course at Carnegie Mellon University, 2007.

Hal Daumé, III and Daniel Marcu. Bayesian query-focused summarization. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual*

Meeting of the Association for Computational Linguistics, ACL '06, pages 305–312, 2006.

Jean-Yves Delort and Enrique Alfonseca. Dualsum: A topic-model based approach for update summarization. In *Proceedings of the 13th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '12, pages 214–223, 2012.

Quang Xuan Do, Wei Lu, and Dan Roth. Joint inference for event timeline construction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 677–687, 2012.

Pablo A. Duboue, Kathleen R. McKeown, and Vasileios Hatzivassiloglou. Progenie: Biographical descriptions for intelligence analysis. In *Proceedings of the 1st NSF/NIJ Conference on Intelligence and Security Informatics*, ISI'03, pages 343–345, 2003.

Aaron Elkiss, Siwei Shen, Anthony Fader, Güneş Erkan, David States, and Dragomir Radev. Blind men and elephants: What do citation summaries tell us about a research article? *Journal of the American Society for Information Science and Technology*, 59(1):51–62, January 2008.

Güneş Erkan and Dragomir R Radev. LexRank: Graph-based centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22(1):457–479, 2004.

Donghui Feng and Eduard Hovy. Handling biographical questions with implicature. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 596–603, 2005.

Katja Filippova and Michael Strube. Sentence fusion via dependency graph compression. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '08, pages 177–185, 2008.

Evgeniy Gabrilovich, Susan Dumais, and Eric Horvitz. NewsJunkie: Providing personalized newsfeeds via analysis of information novelty. In *Proceedings of the 13th International Conference on World Wide Web*, WWW '04, pages 482–490, 2004.

Michel Galley and Kathleen McKeown. Lexicalized markov grammars for sentence compression. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '07, pages 180–187, 2007.

Pierre-Etienne Genest and Guy Lapalme. Framework for abstractive summarization using text-to-text generation. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, MTTG '11, pages 64–73, 2011.

Pierre-Etienne Genest and Guy Lapalme. Fully abstractive approach to guided summarization. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ACL '12, pages 354–358, 2012.

Jennifer Gillenwater, Alex Kulesza, and Ben Taskar. Discovering diverse and salient threads in document collections. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 710–720, 2012.

Barbara Grosz and Candace Sidner. Attention, intentions, and the structure of discourse. *Computational Linguistics*, 12(3):175–204, 1986.

Aria Haghighi and Lucy Vanderwende. Exploring content models for multi-document summarization. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL '09, pages 362–370, 2009.

M.A.K Halliday and Ruqayia Hasan. *Cohesion in English.* Longman, 1976.

Po Hu, Minlie Huang, Peng Xu, Weichang Li, Adam K. Usadi, and Xiaoyan Zhu. Generating breakpoint-based timeline overview for news topic retrospection. In *Proceedings of the 11th International Conference on Data Mining*, ICDM '11, pages 260–269, 2011.

Rahul Jha, Amjad Abu-Jbara, , and Dragomir R. Radev. A system for summarizing scientific topics starting from keywords. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*, ACL '13, pages 572–577, 2013.

Hongyan Jing. Sentence reduction for automatic text summarization. In *Proceedings of the Sixth Conference on Applied Natural Language Processing*, ANLC '00, pages 310–315, 2000.

Maria Lucia Castro Jorge and Thiago Alexandre Salgueiro Pardo. *Multi-Document Summarization: Content Selection based on CST Model (Cross-document Structure Theory)*. PhD thesis, Núcleo Interinstitucional de Lingüística Computacional (NILC), 2010.

Remy Kessler, Xavier Tannier, Caroline Hagège, Véronique Moriceau, and André Bittar. Finding salient dates for building thematic timelines. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 730–739, 2012.

Dan Klein and Christopher D. Manning. Accurate unlexicalized parsing. In *Proceedings of the 41st Meeting of the Association for Computational Linguistics*, ACL '03, pages 423–430, 2003.

Jon Kleinberg. Bursty and hierarchical structure in streams. In *Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '02, pages 91–101, 2002.

Kevin Knight and Daniel Marcu. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107, July 2002.

Oleksandr Kolomiyets, Steven Bethard, and Marie-Francine Moens. Extracting narrative timelines as temporal dependency structures. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Long Papers - Volume 1*, ACL '12, pages 88–97, 2012.

Julian Kupiec, Jan Pedersen, and Francine Chen. A trainable document summarizer. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '95, pages 68–73, 1995.

Mirella Lapata. Probabilistic text structuring: Experiments with sentence ordering. In *Proceedings of the 41st Annual Meeting on Association for Computational Linguistics - Volume 1*, ACL '03, pages 545–552, 2003.

Dawn Lawrie, W. Bruce Croft, and Arnold Rosenberg. Finding topic words for hierarchical summarization. In *Proceedings of the 24th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '01, pages 349–357, 2001.

Dawn J. Lawrie. *Language models for hierarchical summarization.* PhD thesis, University of Massachusetts Amherst, 2003.

Heeyoung Lee, Yves Peirsman, Angel Chang, Nathanael Chambers, Mihai Surdeanu, and Dan Jurafsky. Stanford's multi-pass sieve coreference resolution system at the CoNLL-2011 shared task. In *CoNLL 2011 Shared Task*, 2011.

Peifeng Li, Guangxi Deng, and Qiaoming Zhu. Using context inference to improve sentence ordering for multi-document summarization. In *Proceedings of IJCNLP 2011*, pages 1055–1061, 2011a.

Peng Li, Jing Jiang, and Yinglin Wang. Generating templates of entity summaries with an entity-aspect model and pattern mining. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 640–649, 2010.

Peng Li, Yinglin Wang, Wei Gao, and Jing Jiang. Generating aspect-oriented multi-document summarization with event-aspect model. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 1137–1146, 2011b.

Chin-Yew Lin. Training a selection function for extraction. In *Proceedings of the Eighth International Conference on Information and Knowledge Management*, CIKM '99, pages 55–62, 1999.

Chin-Yew Lin. ROUGE: A package for automatic evaluation of summaries. In *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*, pages 74–81, 2004.

Hui Lin and Jeff Bilmes. A class of submodular functions for document summarization. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies - Volume 1*, HLT '11, pages 510–520, 2011.

Xiaojiang Liu, Zaiqing Nie, Nenghai Yu, and Ji-Rong Wen. Biosnowball: automated population of wikis. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 969–978, 2010.

Annie Louis and Ani Nenkova. Automatic summary evaluation without using human models. In *Proceedings of the 2009 Conference on Empirical Methods in Natural Language Processing*, EMNLP '09, pages 306–314, 2009.

Annie Louis, Aravind Joshi, Rashmi Prasad, and Ani Nenkova. Using entity features to classify implicit discourse relations. In *Proceedings of the 11th Annual Meeting of the Special Interest Group on Discourse and Dialogue*, SIGDIAL '10, pages 59–62, 2010.

Hans Peter Luhn. The automatic creation of literature abstracts. *IBM Journal of Research Development*, 2(2):159–165, 1958.

Inderjeet Mani. *Automatic Summarization.* John Benjamins Publishing Co, Amsterdam/Philadelphia, 2001.

William C. Mann and Sandra A. Thompson. Rhetorical structure theory: Toward a functional theory of text organization. *Text*, 8(3):243–281, 1988.

Daniel Marcu. From discourse structures to text summaries. In *Proceedings of the ACL Workshop on Intelligent Scalable Text Summarization*, pages 82–88, 1997.

Daniel Marcu. Improving summarization through rhetorical parsing tuning. In *Proceedings of the Sixth Workshop on Very Large Corpora*, 1998.

Daniel Marcu and Abdessamad Echihabi. An unsupervised approach to recognizing discourse relations. In *Proceedings of the 40th Annual Meeting on Association for Computational Linguistics*, ACL '02, pages 368–375, 2002.

Erwin Marsi and Emiel Krahmer. Explorations in sentence fusion. In *Proceedings of the European Workshop on Natural Language Generation*, pages 109–117, 2005.

Mausam, Michael Schmitz, Robert Bart, Stephen Soderland, and Oren Etzioni. Open language learning for information extraction. In *Proceedings of the 2012 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP-CoNLL '12, pages 523–534, 2012.

Kathleen McKeown and Dragomir Radev. Generating summaries of multiple news articles. In *Proceedings of the 18th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '95, pages 74–82, 1995.

Qiaozhu Mei and ChengXiang Zhai. Generating impact-based summaries for scientific literature. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics*, ACL '08, pages 816–824, 2008.

George A. Miller. Wordnet: A lexical database for english. *Commun. ACM*, 38(11):39–41, November 1995.

Saif Mohammad, Bonnie Dorr, Melissa Egan, Ahmed Hassan, Pradeep Muthukrishan, Vahed Qazvinian, Dragomir Radev, and David Zajic. Using citations to generate surveys of scientific paradigms. In *Proceedings of Human Language Technologies: The 2009 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, NAACL-HLT'09, 2009.

Ramesh Nallapati, Ao Feng, Fuchun Peng, and James Allan. Event threading within news topics. In *Proceedings of the Thirteenth ACM International Conference on Information and Knowledge Management*, CIKM '04, pages 446–453, 2004.

Hidetsugu Nanba and Manabu Okumura. Towards multi-paper summarization reference information. In *Proceedings of the 16th International Joint Conference on Artificial Intelligence - Volume 2*, IJCAI'99, pages 926–931, 1999.

Hidetsugu Nanba, Noriko Kando, and Manabu Okumura. Classification of research papers using citation links and citation types: Towards automatic review article generation. In *Proceedings of the 11th ASIS SIG/CR Classification Research Workshop*, 2000.

Ani Nenkova. Automatic text summarization of newswire: Lessons learned from the document understanding conference. In *Proceedings of the 20th National Conference on Artificial Intelligence - Volume 3*, AAAI'05, pages 1436–1441, 2005.

Ani Nenkova and Kathleen McKeown. References to named entities: A corpus study. In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology: Companion Volume of the Proceedings of HLT-NAACL 2003–short Papers - Volume 2*, NAACL-Short '03, pages 70–72, 2003.

Ani Nenkova and Kathleen McKeown. Automatic summarization. *Foundations and Trends in Information Retrieval*, 5(2-3):103–233, 2011.

Ani Nenkova and Lucy Vanderwende. The impact of frequency on summarization. Technical report, Microsoft Research, 2005.

Ani Nenkova, Rebecca Passonneau, and Kathleen Mckeown. The pyramid method: incorporating human content selection variation in summarization evaluation. *ACM Transactions on Speech and Language Processing*, 4(2), 2007.

Mark EJ Newman. The structure of scientific collaboration networks. *Proceedings of the National Academy of Sciences*, 98(2):404–409, 2001.

Chikashi Nobata and Satoshi Sekine. CRL/NYU summarization system at DUC-2004. In *Proceedings of the Document Understanding Conference*, DUC '04, 2004.

Naoaki Okazaki, Yutaka Matsuo, and Mitsuru Ishizuka. Improving chronological sentence ordering by precedence relation. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, pages 750–756, 2004.

Kenji Ono, Kazuo Sumita, and Seiji Miike. Abstract generation based on rhetorical structure extraction. In *Proceedings of the 15th Conference on Computational Linguistics - Volume 1*, COLING '94, pages 344–348, 1994.

Miles Osborne. Using maximum entropy for sentence extraction. In *Proceedings of the ACL-02 Workshop on Automatic Summarization - Volume 4*, AS '02, pages 1–8, 2002.

Jahna Otterbacher, Güneş Erkan, and Dragomir R. Radev. Using random walks for question-focused sentence retrieval. In *Proceedings of the Conference on Human Language Technology and Empirical Methods in Natural Language Processing*, HLT '05, pages 915–922, 2005.

Jahna Otterbacher, Dragomir Radev, and Omer Kareem. News to go: Hierarchical text summarization for mobile devices. In *Proceedings of the 29th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '06, pages 589–596, 2006.

You Ouyang, Wenji Li, and Qin Lu. An integrated multi-document summarization approach based on word hierarchical representation. In *Proceedings of the ACL-IJCNLP 2009 Conference Short Papers*, ACLShort '09, pages 113–116, 2009.

Emily Pitler, Annie Louis, and Ani Nenkova. Automatic evaluation of linguistic quality in multi-document summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 544–554, 2010.

Rashmi Prasad, Nikhil Dinesh, Alan Lee, Eleni Miltsakaki, Livio Robaldo, Aravind Joshi, and Bonnie Webber. The Penn Discourse TreeBank 2.0. In *Proceedings of the Sixth International Conference on Language Resources and Evaluation*, LREC '08, 2008.

Vahed Qazvinian and Dragomir Radev. Identifying non-explicit citing sentences for citation-based summarization. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 555–564, 2010.

Vahed Qazvinian and Dragomir R. Radev. Scientific paper summarization using citation summary networks. In *Proceedings of the 22Nd International Conference on Computational Linguistics - Volume 1*, COLING '08, pages 689–696, 2008.

Vahed Qazvinian, Dragomir R. Radev, and Arzucan Özgür. Citation summarization through keyphrase extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 895–903, 2010.

D. R. Radev and K. R. McKeown. Building a generation knowledge source using internet-accessible newswire. In *Proceedings of the Conference on Applied Natural Language Processing*, pages 221–228, 1997.

Dragomir R. Radev. LexRank: Graph-based lexical centrality as salience in text summarization. *Journal of Artificial Intelligence Research*, 22(1):457–479, 2004.

Dragomir R. Radev and Kathleen R. McKeown. Generating natural language summaries from multiple on-line sources. *Computational Linguistics*, 24(3):469–500, 1998.

Dragomir R. Radev and Daniel Tam. Single-document and multi-document summary evaluation via relative utility. In *Proceedings of the ACM Conference on Information and Knowledge Management*, pages 508–511, 2003.

Dragomir R. Radev, Hongyan Jing, Malgorzata Stys, and Daniel Tam. Centroid-based summarization of multiple documents. *Information Processing and Management*, 40 (6):919–938, 2004.

Dragomir R. Radev, Pradeep Muthukrishnan, and Vahed Qazvinian. The ACL anthology network corpus. In *Proceedings, ACL Workshop on Natural Language Processing and Information Retrieval for Digital Libraries*, Singapore, 2009.

Dragomir R. Radev, Pradeep Muthukrishnan, Vahed Qazvinian, and Amjad Abu-Jbara. The ACL anthology network corpus. *Language Resources and Evaluation*, pages 1–26, 2013.

W.M. Rand. Objective criteria for the evaluation of clustering methods. *Journal of the American Statistical Association*, 66(336):846–850, 1971.

G. J. Rath, A. Resnick, and R. Savage. The formation of abstracts by the selection of sentences: Part 1: sentence selection by man and machines. *American Documentation*, 2(12):139–208, 1961.

Horacio Saggion and Robert Gaizauskas. Multi-document summarization by cluster/profile relevance and redundancy removal. In *Proceedings of the Document Understanding Conference*, DUC '04, 2004.

Horacio Saggion and Guy Lapalme. Generating indicative-informative summaries with SumUM. *Computational Linguistics*, 28(4):497–526, 2002.

Christina Sauper and Regina Barzilay. Automatically generating Wikipedia articles: A structure-aware approach. In *Proceedings of the Joint Conference of the 47th Annual*

Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP: Volume 1 - Volume 1, ACL '09, pages 208–216, 2009.

Barry Schiffman, Inderjeet Mani, and Kristian J. Concepcion. Producing biographical summaries: Combining linguistic knowledge with corpus statistics. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*, ACL '01, pages 458–465, 2001.

Dafna Shahaf and Carlos Guestrin. Connecting the dots between news articles. In *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '10, pages 623–632, 2010.

Dafna Shahaf, Carlos Guestrin, and Eric Horvitz. Metro maps of science. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 1122–1130, 2012a.

Dafna Shahaf, Carlos Guestrin, and Eric Horvitz. Trains of thought: Generating information maps. In *Proceedings of the 21st International Conference on World Wide Web*, WWW '12, pages 899–908, 2012b.

Chao Shen and Tao Li. Multi-document summarization via the minimum dominating set. In *Proceedings of the 23rd International Conference on Computational Linguistics*, COLING '10, pages 984–992, 2010.

Advaith Siddharthan and Simone Teufel. Whose idea was this, and why does it matter? Attributing scientific work to citations. In *Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics*, HLT-NAACL '07, 2007.

Advaith Siddharthan, Ani Nenkova, and Kathleen McKeown. Syntactic simplification for improving content selection in multi-document summarization. In *Proceedings of the 20th International Conference on Computational Linguistics*, COLING '04, 2004.

H. Gregory Silber and Kathleen F. McCoy. Efficient text summarization using lexical chains. In *Proceedings of the 5th International Conference on Intelligent User Interfaces*, IUI '00, pages 252–255, 2000.

Josef Steinberger and Karel Jezek. Update summarization based on novel topic distribution. In *Proceedings of DocEng 2009*, 2009.

Krysta Svore, Lucy Vanderwende, and Christopher Burges. Enhancing single-document summarization by combining ranknet and third-party sources. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP'07, 2007.

Russell Swan and James Allen. Automatic generation of overview timelines. In *Proceedings of the 23rd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '00, pages 49–56, 2000.

Maite Taboada and William C. Mann. Applications of rhetorical structure theory. *Discourse Studies*, 8(4):567–588, 2006.

Kou Takahashi, Takao Miura, and Isamu Shioya. Hierarchical summarizing and evaluating for web pages. In *Proceedings of the 1st workshop on emerging research opportunities for Web Data Management*, EROW 2007, 2007.

Hiroya Takamura and Manabu Okumura. Text summarization model based on maximum coverage problem and its variant. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, EACL '09, pages 781–789, 2009.

Hideki Tanaka, Akinori Kinoshita, Takeshi Kobayakawa, Tadashi Kumano, and Naoto Kato. Syntax-driven sentence revision for broadcast news summarization. In *Proceedings of the 2009 Workshop on Language Generation and Summarisation*, UCNLG+Sum '09, pages 39–47, 2009.

Xuning Tang and Christopher C. Yang. TUT: A statistical model for detecting trends, topics and user interests in social media. In *Proceedings of the 21st ACM International Conference on Information and Knowledge Management*, CIKM '12, pages 972–981, 2012.

Simone Teufel, Advaith Siddharthan, and Dan Tidhar. Automatic classification of citation function. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*, EMNLP '06, pages 103–110, 2006.

Robert Tibshirani, Guenther Walther, and Trevor Hastie. Estimating the number of clusters in a dataset via the gap statistic. *Journal of the Royal Statistical Society, Series B*, 32 (2):411–423, 2000.

Kristina Toutanova, Chris Brockett, Michael Gamon, Jagadeesh Jagarlamudi, Hisami Suzuki, and Lucy Vanderwende. The PYTHY summarization system: Microsoft Research at DUC 2007. In *Proceedings of DUC 2007*, 2007.

Jenine Turner and Eugene Charniak. Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics*, ACL '05, pages 290–297, 2005.

Stijn van Dongen. *Graph Clustering by Flow Simulation.* PhD thesis, University of Utrecht, 2000.

Earl J. Wagner, Jiahui Liu, Larry Birnbaum, and Kenneth D. Forbus. Rich interfaces for reading news on the web. In *Proceedings of the 14th International Conference on Intelligent User Interfaces*, IUI '09, pages 27–36, 2009.

Xiaojun Wan and Jianwu Yang. Improved affinity graph based multi-document summarization. In *Proceedings of the Human Language Technology Conference of the NAACL, Companion Volume: Short Papers*, NAACL-Short '06, pages 181–184, 2006.

Dingding Wang and Tao Li. Document update summarization using incremental hierarchical clustering. In *Proceedings of the 19th ACM International Conference on Information and Knowledge Management*, CIKM '10, pages 279–288, 2010.

Fu Lee Wang, Christopher C. Yang, and Xiaodong Shi. Multi-document summarization for terrorism information extraction. In *Proceedings of the 4th IEEE International Conference on Intelligence and Security Informatics*, ISI'06, pages 602–608, 2006.

Yexin Wang, Li Zhao, and Yan Zhang. MagicCube: Choosing the best snippet for each aspect of an entity. In *Proceedings of the 18th ACM conference on Information and knowledge management*, CIKM '09, pages 1705–1708, 2009.

Michael J. Witbrock and Vibhu O. Mittal. Ultra-summarization (poster abstract): A statistical approach to generating highly condensed non-extractive summaries. In *Proceedings of the 22Nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '99, pages 315–316, 1999.

Florian Wolf and Edward Gibson. Representing discourse coherence: A corpus-based study. *Computational Linguistics*, 31(2):249–288, 2005.

Kristian Woodsend and Mirella Lapata. Automatic generation of story highlights. In *Proceedings of the 48th Annual Meeting of the Association for Computational Linguistics*, ACL '10, pages 565–574, 2010.

Rui Yan, Liang Kong, Congrui Huang, Xiaojun Wan, Xiaoming Li, and Yan Zhang. Timeline generation through evolutionary trans-temporal summarization. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, EMNLP '11, pages 433–443, 2011a.

Rui Yan, Liang Kong, Yu Li, Yan Zhang, and Xiaoming Li. A finegrained digestion of news webpages through event snippet extraction. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 157–158, 2011b.

Rui Yan, Xiaojun Wan, Jahna Otterbacher, Liang Kong, Xiaoming Li, and Yan Zhang. Evolutionary timeline summarization: A balanced optimization framework via iterative substitution. In *Proceeding of SIGIR 2011*, pages 745–754, 2011c.

Christopher C. Yang and Fu Lee Wang. Fractal summarization: summarization based on fractal theory. In *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval*, SIGIR '03, pages 391–392, 2003.

Conglei Yao, Xu Jia, Sicong Shou, Shicong Feng, Feng Zhou, and Hongyan Liu. Autopedia: Automatic domain-independent wikipedia article generation. In *Proceedings of the 20th International Conference on World Wide Web*, WWW '11, pages 161–162, 2011.

David Zajic, Bonnie J. Dorr, Jimmy Lin, and Richard Schwartz. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Information Processing and Management*, 43(6):1549–1570, November 2007.

Renxian Zhang, Li Wenjie, and Lu Qin. Sentence ordering with event-enriched semantics and two-layered clustering for multi-document news summarization. In *Proceedings of the 23rd International Conference on Computational Linguistics: Posters*, COLING '10, pages 1489–1497, 2010.

Zhu Zhang, Sasha Blair-Goldensohn, and Dragomir R. Radev. Towards CST-enhanced summarization. In *Proceedings of the 18th National Conference on Artificial Intelligence*, AAAI'02, pages 439–445, 2002.

Liang Zhou, Miruna Ticrea, and Eduard Hovy. Multi-document biography summarization. In *Proceedings of the 2004 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning*, EMNLP'04, 2004.