Global Inference for Sentence Compression: An Integer Linear Programming Approach

James Clarke

Doctor of Philosophy

Institute for Communicating and Collaborative Systems

School of Informatics

University of Edinburgh

2008

Abstract

In this thesis we develop models for sentence compression. This text rewriting task has recently attracted a lot of attention due to its relevance for applications (e.g., summarisation) and simple formulation by means of word deletion. Previous models for sentence compression have been inherently local and thus fail to capture the long range dependencies and complex interactions involved in text rewriting. We present a solution by framing the task as an optimisation problem with local and global constraints and recast existing compression models into this framework. Using the constraints we instill syntactic, semantic and discourse knowledge the models otherwise fail to capture. We show that the addition of constraints allow relatively simple local models to reach state-of-the-art performance for sentence compression.

The thesis provides a detailed study of sentence compression and its models. The differences between automatic and manually created compression corpora are assessed along with how compression varies across written and spoken text. We also discuss various techniques for automatically and manually evaluating compression output against a gold standard. Models are reviewed based on their assumptions, training requirements, and scalability.

We introduce a general method for extending previous approaches to allow for more global models. This is achieved through the optimisation framework of Integer Linear Programming (ILP). We reformulate three compression models: an unsupervised model, a semi-supervised model and a fully supervised model as ILP problems and augment them with constraints. These constraints are intuitive for the compression task and are both syntactically and semantically motivated. We demonstrate how they improve compression quality and reduce the requirements on training material.

Finally, we delve into document compression where the task is to compress every sentence of a document and use the resulting summary as a replacement for the original document. For document-based compression we investigate discourse information and its application to the compression task. Two discourse theories, Centering and lexical chains, are used to automatically annotate documents. These annotations are then used in our compression framework to impose additional constraints on the resulting document. The goal is to preserve the discourse structure of the original document and most of its content. We show how a discourse informed compression model can outperform a discourse agnostic state-of-the-art model using a question answering evaluation paradigm.

Acknowledgements

I would like to begin by thanking my supervisor, Mirella Lapata, for her guidance and continuous support throughout my graduate life. Mirella's ability to see the big picture while provide valuable detailed advice has certainly helped shape this thesis and keep me focused on the task at hand.

Many thanks also to my second supervisor, Steve Renals; everyone who has served on my committee, especially Jean Carletta and Frank Keller — who have all given useful, thoughtful advice and provided a different perspective on my work. I am grateful for the multiple discussions with Simone Teufel which helped shape the evaluation methodology. My office mate, Sebastian Riedel, has offered many insightful comments throughout the course of my PhD and has been an invaluable source of discussion on a wide range of topics, including those within AI and other less academic areas.

I am extremely grateful to everyone who has taken part in our online experiments and our annotators Vasilis Karaiskos, Beata Kouchnir, and Sarah Luger. I received many helpful comments and suggestions from the anonymous reviewers during the submission process and the audience at conference presentations for ACL-2006, EMNLP-CoNLL-2007 and JAIR. In particular the JAIR reviewers' comments and suggestions have helped substantially improve the presentation of Chapter 6. Thank you to Jon Oberlander and Dan Roth for agreeing to serve on my thesis committee and providing a thoroughly interesting discussion during the viva.

I also want to thank my office mates: Abhishek, Sebastian and Ruken, for making the office a fascinating place to spend the day (and night!). Many thanks to Chloe for her encouragement and support during the difficult times. She has an incredible knack of chasing away the blues and provided a much sort after escape from the PhD.

Finally, thank you to my parents who have provided the love, support and encouragement necessary for me to be where I am today.

Declaration

I declare that this thesis was composed by myself, that the work contained herein is my own except where explicitly stated otherwise in the text, and that this work has not been submitted for any other degree or professional qualification except as specified.

(*James Clarke*)

Table of Contents

List of Figures

List of Tables

Chapter 1

Introduction

This thesis is concerned with the task of sentence compression. Sentence compression is often considered a subtask within automatic summarisation. In its simplest form it can be viewed as producing a summary of a single sentence. This chapter presents motivation for the task and how it differs from other summarisation tasks. The chapter concludes with a summary of the thesis and its main contributions.

1.1 Automatic Summarisation

The field of automatic summarisation has traditionally been dominated by extractive summarisation. Extract summaries are summaries consisting entirely of material copied from a source document. This is in contrast to abstract summaries where at least some of the material is not present in the source document. Abstracts tend to contain paraphrases and offer higher degrees of condensation: a short abstract may contain more information than a longer extract.

In extractive summarisation the units of text that are deemed most representative of the document are selected and then concatenated verbatim together to form a summary. Sentences are typically used as the unit of text, however it is possible to use paragraphs or clauses too (Mani 2001). A large body of work has focused on the selection process using features such as: position in document, keyword frequency, sentence length and sentence similarity or dissimilarity within the document (see Mani (2001) for an overview).

Performing sentence extraction alone can lead to incoherent and fragmented summaries, as the context of each sentence is not considered during the extraction process;

this is especially true in multi-document summarisation[1] where sentences from many documents may be concatenated. The problem of producing a coherent summary given a set of extracted sentences has received some attention in summarisation. For example, Mani et al. (1999) have looked at revising single document extracts with the aim of making them more readable. By way of rendering the summary less repetitive they remove extraneous constituents such as relative clauses and prepositional phrases. Another example are Jing and McKeown (1999) who propose to smooth the extracts with operations such as sentence compression, sentence combination and syntactic transformations. Redundancy also poses challenges to multi-document summarisation. Again systems often include a component that deals especially with this problem (Barzilay et al. 1999).

Generating abstractive summaries is a complex and difficult task. Abstractive summarisation systems often perform sentence compression, not only to help produce a coherent summary, but also to remove any redundancy from the summary. This is typically done through manually written rules for compression (Barzilay et al. 1999; Mani et al. 1999). Thus recently research emphasis has shifted towards sentence compression which is an integral part of summarisation systems. The problem is studied in its own right which removes the other factors of summarisation (i.e., sentence selection). Sentence compression is considerably simpler than full abstraction but still provides many of the same challenges facing document summarisation.

1.2 Sentence Compression

Sentence compression can be viewed as producing a summary of a single sentence. Instead of being given a document, or collection of documents, and asked to produce a summary (either extract or abstract) we are given a sentence to compress. The compressed sentence should retain the most important information and remain grammatical while using fewer words than the original source sentence. Although compressing a sentence may seem a relatively trivial task, performing it automatically is not.

Sentence (1-a) can be compressed to form sentence (1-b), while (2-b) and (2-c) are two possible compressions of (2-a).

(1) a. Prime Minister Tony Blair today insisted the case for holding terrorism suspects without trial was “absolutely compelling” as the government pub-

[1] Mutli-document summarisation is concerned with creating a single summary using multiple documents about the same event or topic.

lished new legislation allowing detention for 90 days without charge.

b. Tony Blair has insisted there is a “compelling” case for newly published legislation allowing terror suspects to be held without trail.

(2) a. David Cameron’s bid for the Conservative leadership received a double boost today in the form of endorsements from the party’s most senior woman, Theresa May, and Bernard Jenkin, a figure from the Tory right.

b. Theresa May and Bernard Jenkin endorsed David Cameron’s bid for Tory leadership.

c. David Cameron’s bid for Tory leadership gets support from party’s most senior woman and figure from Tory right.

These two examples demonstrate that sentence compression involves determining what information is important and how to convey it. This can involve complex text rewriting operations which include: word reordering, deletion, substitution and insertion. Ideally a sentence compression algorithm will have all these operations at its disposal. However, much of the current research in the sentence compression literature has simplified the problem to the removal of words from the original sentence. Examples of this can be seen in sentences (3-b) and (4-b).

(3) a. Prime Minister Tony Blair today insisted the case for holding terrorism suspects without trial was “absolutely compelling” as the government published new legislation allowing detention for 90 days without charge.

b. Tony Blair insisted the case for holding terrorism suspects without trail was “compelling”.

(4) a. David Cameron’s bid for the Conservative leadership received a double boost today in the form of endorsements from the party’s most senior woman, Theresa May, and Bernard Jenkin, a figure from the Tory right.

b. David Cameron’s bid for leadership received a boost in the form of endorsements from Theresa May and Bernard Jenkin.

Over the last few years there have been numerous papers published on sentence compression, however the task itself remains poorly defined. Much of the current work in the literature focuses on one particular instantiation of the compression task — word deletion. Given an input source sentence of words $\mathbf{x} = x_1, x_2, \ldots, x_n$, a compression is formed by dropping any subset of these words (Knight and Marcu 2002). Good

compressions are those which:

- use fewer words than the source sentence,
- retain the most important information from the source sentence,
- remain grammatical.

A similar definition is provided by Jing (2000), who states that the goal of sentence compression is "to reduce without major loss". This entails removing as many extraneous phrases as possible from a sentence without detracting from the main idea the sentence conveys. In this definition the notion of importance is dependent on the topic of the sentence.

Our own definition of sentence compression is broader. Following from Sparck-Jones's (1998) definition of a summary, we formulate *sentence compression* as a *transformation of a source sentence through information reduction and/or paraphrasing with respect to what is important in the source*. The information that is important in the source is a very subjective concept. Assuming that sentence compressions are generated with a user in mind, the notion of information content will depend upon: (1) the user's background knowledge, (2) their information need, and (3) their compression requirements.

Background Knowledge Background knowledge is one of the most important factors influencing how to compress a sentence. For example, in sentences (3-a) and (4-a) if the user is aware that Tony Blair is the Prime Minster and David Cameron is a Conservative then we can produce compressions (3-b) and (4-b) respectively, with little or no information loss.

The users's background knowledge can vary from general (i.e., the knowledge we assume an average person has accumulated from life experiences) to domain specific knowledge. Another form of background knowledge is the information gained while reading a document. For example, documents tend to contain redundant information. On the first mention of a novel fact we may decide not to remove it, however on subsequent mentions, we can consider it redundant information. Thus, the document's content will also influence compression.

Information Need The information need of a user provides us with an idea of which information to present in the compression. For example, the user could require that

compressions contain information related to certain key events or people they are interested in. This would be similar to query-focused summarisation[2]. Another scenario would be presenting compressions of one document in relation to a reference document. For example, the reference document may be a news article on an event, and here, when a compression system is presented a new document the compressions it generates should present new information not found in the reference document.

There are many different configurations of information need. Perhaps the most general information need concerns the document as a whole and how compressed sentences relate to its main topic.

Compression Requirements The final aspect affecting the compression are the requirements which are not user specific. For example, a hypothetical compression system may be faced with physical limitations. In such a case, if we are compressing sentences to be displayed on small screens, a strict length limit may be imposed which must not be exceeded. Other compression requirements may be more general such as transforming complex wordy and technical sentences into shorter sentences which are simpler and less technical.

Exploring all these different compression factors is beyond the scope of this thesis. Here, we limit ourselves to the simple instantiation of sentence compression as word deletion. We will assume a hypothetical user requires a compression that takes into account general background knowledge and will not specifically account for the individual user's information need. Therefore, we aim to create compressions from a document that relate to the main topic or topics of the document.

1.3 Applications of Sentence Compression

Thus far we have motivated sentence compression from an automatic summarisation standpoint. Beyond summarisation, sentence compression has a wide variety of useful applications on its own. Subtitles for television programmes can not be typically created using speech transcripts verbatim as the rate of speech is usually much higher than the rate at which words can be displayed on the screen while keeping the text and picture in synchronisation. Thus, sentence compression can be used to automatically

[2]The goal of query-focused summarisation is to produce a summary which is directed by a user's query expressing their information need.

generate subtitles (Vandeghinste and Pan 2004), in which redundant or less important information can be missed while retaining the main argument or premise of the programme.

One of the first applications proposed for sentence compression was audio scanning devices for the blind (Grefenstette 1998). Sighted readers can easily and quickly scan over a page or document and understand the topic being discussed. Blind readers, who read documents via a reading machine which produces audio output, cannot easily navigate a document quickly. To do this, a blind reader can only speed up or slow down the audio. If reading machines contained a sentence compression module, the amount of compression could be controlled via a knob and a blind reader could scan a document in a similar manner to sighted readers.

Another application is compressing text to be displayed on small screens (Corston-Oliver 2001). In many cases reading full documents or email messages on a small screen such as a mobile phone or PDA is impractical.

1.4 Contributions

This thesis contributes to the sentence compression task in the following ways:

- We study the compression task by assessing how humans compress sentences. Previous work has concentrated on automatically generated compression corpora. We focus on human authored compressions of spoken and written text and show that these compressions are radically different to those obtained automatically.

- We reformulate and extend three compression models in the Integer Linear Programming (ILP) framework which allows us to examine how constraints influence the compression task. The three models cover the spectrum of learning paradigms: unsupervised, semi-supervised and fully supervised. ILP provides us with exact inference even in the face of constraints.

- Under the ILP framework, we introduce several novel and intuitive constraints for the compression task. The constraints instill additional syntactic, semantic and discourse knowledge the models otherwise fail to capture. We show that the constraints allow relatively simple models to reach state-of-the-art performance for sentence compression.

- We extend our module from sentence compression to document compression. In order to perform document compression we formulate a robust method for automatically annotating discourse information using two theories of discourse, Centering Theory and lexical chains. Using this information we instill discourse information into a compression model through ILP constraints. Our discourse enhanced model conserves the core content of documents when performing document compression better than state-of-the-art discourse agnostic compression systems.

- Finally, we assess the evaluation of sentence compressions. Specifically we describe two judgement elicitation studies for comparing system compressions. The first considers sentences in isolation where judges rate compressions in two dimensions: grammaticality and importance. The second is concerned with document compression evaluation and follows a question-answering paradigm where the content of the compressions is evaluated without reference to the original document material. We also study automatic evaluation measures and show that F-score over the grammatical relations between gold standard and system compressions can be used since it correlates reliably with human judgements.

1.5 Thesis Overview

The remainder of this thesis is structured as follows:

- Chapter 2 introduces previous approaches to the sentence compression task. We give details of several fully supervised methods which can be split broadly into generative models and discriminative models. Semi-supervised, unsupervised and less data intensive models are also examined. We summarise previous evaluation studies which give us insight into the performance of the models and motivate why current approaches are not completely satisfactory.

- Chapter 3 focuses on the analysis of human authored sentence compressions. Specifically, we show that automatically collected compression corpora differ significantly from human authored compression corpora. To undertake this analysis we create two human authored compression corpora on spoken and written news text. These corpora differ from those previously available as they contain compressed sentences of complete documents or news stories thus allowing us build models that compress entire documents.

- Chapter 4 is concerned with methods for evaluating sentence compression systems and fleshes out the evaluation techniques we adopt in this thesis. We present a variety of methods for automatically and manually evaluating compressions. We outline the problems of current elicitation studies and provide a more rigorous paradigm for evaluating compressed sentences in isolation. We also assess whether the proposed automatic evaluation measures are suitable for the task by correlating their scores with human judgements. Finally, we present a method for evaluating document compressions through a question-answering paradigm.

- Chapter 5 introduces the frameworks of linear programming (LP) and integer linear programming (ILP). These are two flexible frameworks for modelling various optimisation problems. We provide an overview of how ILP has previously been used within natural language processing as a motivating factor for choosing the framework.

- Chapter 6 reformulates and extends three sentence compression models in the ILP framework. We introduce a set of linguistically and semantically motivated constraints which are designed to bring less local syntactic knowledge into the models and help preserve the meaning of the source sentence in the compression. We investigate the influence of our constraint set across models and learning paradigms; in particular how the performance of supervised, semi-supervised and unsupervised models is impacted by constraint-based inference.

- Chapter 7 is concerned with document compression (where all sentences within a document are compressed). Central to our approach is the use of discourse-level information which we annotate automatically. Our annotation algorithms are robust and complementary. They are inspired by two linguistic theories relating to local coherence, Centering Theory and lexical cohesion; and provide our model with important information for document (as opposed to sentence) compression. This information is instilled into our model using a set of discourse constraints designed to preserve coherence of the original document and also provide information about which entities are important.

- Chapter 8 summarises the major contributions of this work and discusses future research directions.

1.6 Published Work

Some of the material presented in this thesis has been previously published. Chapter 3 and Chapter 4 expands on the material in Clarke and Lapata (2006b) by providing more details of: the corpus annotation method, variations across corpora and analysis of the nature of compressions.

Chapter 5 and Chapter 6 is related to the work in Clarke and Lapata (2006a, 2008). In particular, Chapter 5 contains a more detailed introduction to Integer Linear Programming, how it relates to other methods such as constraint programming and reranking; and its previous use within natural language processing. Chapter 6 contains additional information on parameter estimate and Chapter 4 covers the evaluation method in more detail.

Finally, some of the work in Clarke and Lapata (2007) is described in Chapters 4 and 7. The former covers the evaluation methodology in more detail, and the latter discusses various approaches to incorporating discourse information into models, as well as providing full details of how we obtain our discourse annotations automatically.

Chapter 2

Overview of Compression Models

In this chapter we examine the computational treatment of sentence compression. A wide variety of methods have been proposed in the literature. We review these methods concentrating on the training requirements of each approach. Some methods require rich linguistic annotations of sentences such as parse trees and dependency trees; while other methods rely on very little linguistic knowledge such as part-of-speech tags or merely the lexical items alone.

Current approaches are split into two broad classes: data intensive and data lean. The data intensive approaches usually follow a supervised learning paradigm and require a parallel corpus of (source sentence, compressed sentence) pairs which are used to learn the rules of compression. Data lean approaches usually have some generalisation of compressed sentences so that without learning specific rules, they incorporate knowledge with respect to compression. The data lean algorithms are typically unsupervised and applied with little or no prior learning from a parallel corpus.

Supervised learning aims to learn a function that maps from inputs $x \in \mathcal{X}$ to outputs $y \in \mathcal{Y}$. Many natural language processing tasks can be framed using this mapping. For example, in machine translation x could be a French sentence and y is the English translation. In sequence labelling tasks such as part-of-speech tagging, x is a sentence and y is the corresponding part-of-speech sequence. In the case of sentence compression $x \in \mathcal{X}$ is a source sentence and $y \in \mathcal{Y}$ its corresponding compression. Here the task is viewed as developing a mapping from $\mathcal{X}$ to $\mathcal{Y}$ that retains important information from the source sentence and provides a grammatical compression.

We first introduce data intensive treatments of the compression task. Next we progress to more data lean methods and conclude the chapter with a discussion of previous compression models.

2.1 Generative Approaches

Generative approaches have received a considerable amount of attention in the compression literature. The generative models typical estimate the joint probability $P(x,y)$ of a source sentence x having the target compression y. One appealing aspect of these models is their simplicity to train. Parameters are estimated using simple functions of counts of various compression operations obtained from a parallel corpus of source sentence and target compression pairs.

The initial generative models were inspired by the models used in machine translation. Machine translation has natural parallels with the compression task. In translation the goal is to translate a document from a *source* language into another *target* language. The machine translation community focus on producing models that translate between sentences. Probabilistic models are trained on aligned sentence-sentence pairs from which the model must learn word or phrase alignments and word or phrase translations.

Sentence compression can be viewed as a machine translation problem where instead of translating between languages we are translating between original source sentences and target compressed sentences. Thus the same probabilistic approach and model can be applied — the noisy-channel model.

Next we review the noisy-channel model in general and how it has been applied to sentence compression (Knight and Marcu (2002); Turner and Charniak (2005)). We then discuss some of the shortcomings of this model.

2.1.1 The Noisy-channel Model

The noisy-channel model has been used successfully in a variety of natural language processing applications including speech recognition (Jelinek 1997), part-of-speech tagging (Church 1988) and machine translation (Brown et al. 1993).

Rather than directly modelling probability of the target compression given the source sentence, $P(y|x)$, the noisy channel model breaks the probability into $P(y) \cdot P(x|y)$, the goal to find the best compression, y then becomes $y^* = \arg\max_y P(y) \cdot P(x|y)$.

This corresponds to three components (see Figure 2.1 for visual representation):

- The channel model $P(x|y)$ – the conditional probability of the source sentence given the target compression. This is responsible for capturing the operations

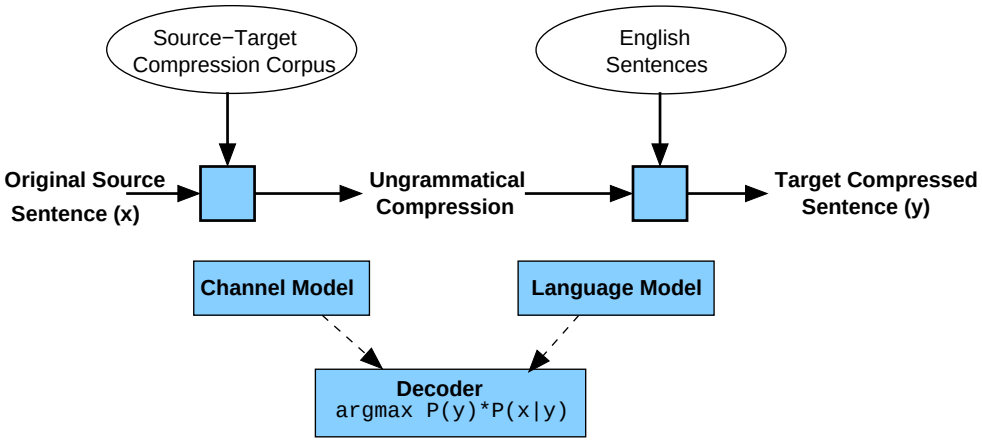

Figure 2.1: The noisy-channel model.

which transform the compression into the source sentence.

- The language model $P(y)$ – gives us the probability of the compression occurring. In this model we want grammatical compressions to score higher than ungrammatical ones.

- The decoder — searches for the best compressed sentence given the source sentence by maximising $P(y) \cdot P(x|y)$.

Sentence compression within the noisy-channel framework can be viewed as follows: given a source sentence we must imagine that it was once a compressed sentence which has had additional (and optional) text added to it. The noise in the model corresponds to the additional text present in the long string.

A parallel corpus is required to learn the probability estimates of the channel model ($P(x|y)$).

Knight and Marcu (2002) first proposed using the noisy-channel approach for sentence compression and since then it has been extended by Turner and Charniak (2005).

2.1.2 Knight and Marcu's Compression Model

Knight and Marcu (2002) propose a probabilistic approach using the noisy-channel model for sentence compression. Their source and channel models act on parse trees rather than words, and this differs from previous work using statistical channel models for caption generation which are solely word-based (Witbrock and Mittal 1999). Their goal is to take a large tree and rewrite it into a smaller tree while retaining the word

```
y = S ((NP John)
       (VP (VB saw)
           (NP Mary)))
```

is assigned the score:

$$
\begin{aligned}
P(y) \;=\; & P_{cfg}(\text{TOP} \rightarrow \text{S} \mid \text{TOP}) \cdot P_{cfg}(\text{S} \rightarrow \text{NP VP} \mid \text{S}) \cdot \\
& P_{cfg}(\text{NP} \rightarrow \text{John} \mid \text{NP}) \cdot P_{cfg}(\text{VP} \rightarrow \text{VB NP} \mid \text{VP}) \cdot \\
& P_{cfg}(\text{VB} \rightarrow \text{saw} \mid \text{VB}) \cdot P_{cfg}(\text{NP} \rightarrow \text{Mary} \mid \text{NP}) \cdot \\
& P_{bigram}(\text{John} \mid \text{EOS}) \cdot P_{bigram}(\text{saw} \mid \text{John}) \cdot \\
& P_{bigram}(\text{Mary} \mid \text{saw}) \cdot P_{bigram}(\text{EOS} \mid \text{Mary})
\end{aligned}
$$

Figure 2.2: Example of the source model as introduced by Knight and Marcu (2002)

ordering of the source tree. The language model is concerned with creating target compressions, *y*, that look grammatical; while the channel model has the task of preserving the important meaning between source sentence, *x* and compressed sentences.

For the language model, a good compressed tree is one that has a normal-looking parse structure and a high bigram score. Thus $P(y)$ is computed using a combination of a probabilistic context-free grammar (PCFG) score (which is computed over the grammar rules that yielded the tree *y* from *x*) and the bigram score for the leaves of the tree. Knight and Marcu (2002) note that the probability assignments made by the source model do not sum to one as they are counting the cost of each word twice. Figure 2.2 shows the score for the target compressed sentence “John saw Mary”.

The channel model performs minimal operations on the compressed sentence to produce the source sentence. The model probabilistically chooses an expansion template, which are synchronous context free grammar (SCFG) rules for each internal node in *y*, based on the labels of the node and its children. For example, given the structure S $\rightarrow$ NP VP, the channel model may grow this into S $\rightarrow$ NP VP PP with the probability of $P_{exp}(\text{S} \rightarrow \text{NP VP PP} \mid \text{S} \rightarrow \text{NP VP})$. It could also choose not to grow it at all, with probability $P_{exp}(\text{S} \rightarrow \text{NP VP} \mid \text{S} \rightarrow \text{NP VP})$ or grow it into another structure with a probability framed in a similar way. If a new node is grown, the subtree is also grown with probabilities given by the PCFG factorisation shown in Figure 2.2 (without taking the bigram probabilities into account; only P_{cfg}).

A parallel corpus is used to train the models' parameters. Each side of the corpus is parsed with Collins's (1997) parser. The parses are used to identify corresponding syntactic nodes which provide a frequency count of joint events, such as (S $\rightarrow$ NP VP, S $\rightarrow$ NP VP PP). These joint counts can be normalised to provide P_{exp}. The PCFG and bigram language model are estimated from the Penn Treebank and unannotated Wall Street Journal respectively.

The decoder selects the trees with the best combination of the source and channel scores. This is achieved by creating a packed parse forest of all possible compressions that are grammatical according to the Penn Treebank. If a compression has zero expansion probability with respect to the training data it is assigned a very small probability. A tree extractor then collects the sentences with the highest $P(y|x)$ score. It returns a list of trees that correspond to the best compression for each possible compression length with their corresponding log-probabilities. Knight and Marcu (2002) observed that if they rely on the log-probability to select the best compression, they almost always select the shortest compression. To avoid this the log-probabilities are normalised by the compression length, thus rewarding longer compressions.

Their noisy-channel based approach was tested on the Ziff-Davis corpus (details of which are provided in Chapter 3) and gives a compression rate of approximately 70% compared to a human authored compression rate of 53%. A baseline system using a bigram language model provides a compression rate of 64%. When evaluated by human judges the noisy-channel model's compressions significantly outperformed the baseline compression but proved to be significantly worse than the human authored compressions.

2.1.3 Turner and Charniak's Extensions

Turner and Charniak (2005) extend the noisy-channel model proposed by Knight and Marcu (2002) by modifying the language model and channel model. Their most significant change is the substitution of the language model. Recall that the latter consists of a probabilistic context free grammar (PCFG) score that determines if the parse structure is normal looking and a bigram language model. This language model is substituted with a syntax-based language model following Charniak (2001). More specifically the language model is an "immediate-head" parser that conditions all events below a constituent c upon the head of c.

A slight modification to the channel model is made giving $P(x|y) = P_{exp}(x|y) \cdot$

$P_{deleted}$, where $P_{deleted}$ is the probability of adding the deleted subtrees back into the compression to give the original source sentence. This is also estimated using the syntax-based language model. Turner and Charniak (2005) do not require a packed parse forest for the decoding process as they limit their system to only generate compressions for original sentences for which they have rules. Thus if they have never seen the original sentence they do not generate a compression. While Knight and Marcu (2002) required a parameter to discourage compression Turner and Charniak (2005) found the opposite true; their system did not naturally produce compressions therefore a parameter was added to encourage compression.

One of the biggest problems with the noisy-channel approach to sentence compression is the lack of training data. To alleviate this, Turner and Charniak (2005) added manually crafted rules and approximated other rules (from different corpora). They added a selection of *special rules* which could not be modelled using the simple channel model. These rules are structurally more complicated such as the rule NP(1) $\rightarrow$ NP(2) CC NP(3), where the parent has at least one child with the same label as itself; then the resulting compression is one of the matching children, for example NP(2). Constraints were also added to never allow the deletion of a complement without its syntactic parent. A similar constraint was applied to noun phrases.

It is possible to estimate the channel model without a parallel corpus. In this unsupervised version $P_{exp}(x|y)$ is estimated from the first section of the Penn Treebank while P_{delete} remains the same as it is obtained from the language model. This is achieved by matching the PCFG expansions with similar rules occurring in the Penn Treebank. A rule must be a *svo* (shorter version of) the PCFG expansion for it to be considered a match. Where *svo* is defined as:

svo r_1 svo r_2 if and only if the right hand side of r_1 is a subsequence of the right hand side of r_2.

This unsupervised version is then restricted to generating compressions provided the head of any subtree is not deleted; thus reducing the number of poor compressions.

All these changes are merged together to produce a variety of models. They present a series of models, one of which uses the special rules and constraints when appropriate and only relies on the unsupervised compression probabilities if there are no probabilities under the supervised model. This model outperforms the original model by Knight and Marcu (2002) and provides a compression rate of 81% (compared to 70% for Knight and Marcu's model).

2.1.4 Discussion of the noisy-channel model

Turner and Charniak (2005) point out a fundamental problem with the noisy-channel model as discussed above for sentence compression. The problem is the following: the probability of a constituent being deleted is far lower than that of the constituent being left in. Thus, the most probable compression should be a sentence which is barely compressed if at all. To make this assertion firmer we will follow the reasoning made in Turner and Charniak (2005).

If we state the noisy-channel model more explicitly as Equation (2.1):

$$\begin{aligned} P(x) &= \arg\max_{c} P(y, L = y | x, L = x) \\ &= \arg\max_{c} P(y, L = y) \cdot P(x, L = x | y, L = y) \end{aligned} \tag{2.1}$$

where the events $L = y$ and $L = x$ explicitly state that the sentence is target compression or original source respectively. Then in order to give the equation $P(y) \cdot P(x|y)$ in the current formulation, we must assume:

$$P(y, L = y) = P(y) \tag{2.2}$$

$$P(x, L = x | y, L = y) = P(x|y) \tag{2.3}$$

Thus we are assuming that the probability of y as a target compression is simply its probability of being a sentence. This should not be the case in sentence compression. Ideally the probability of y being a compression should be calculated with respect to a set of compressed sentences rather than the set of all English sentences. Whereas the probability of x being a source sentence should be calculated against the set of all English sentences (of which compressed sentences will be a subset) as any sentence can be considered a source sentence. However, we do not have a large enough corpus of compressed sentences to estimate $P(y, L = y)$ reliably thus we must assume that Equations (2.2) and (2.3) hold.

These assumptions eventually undermine the whole compression process; i.e., the probability of deleting constituents is far lower than leaving them in (Turner and Charniak 2005). Thus, a weighting factor to aid compression is added when the source model is a syntax language model. If the source model consists of a PCFG probability model and a bigram language model, then this weighting factor is not required, as we are paying the probabilistic price twice for including a word (once in the PCFG and once in the bigram).

2.1.5 Lexicalized Markov Grammars

Galley and McKeown (2007) present another generative approach to sentence compression which addresses some of the deficiencies of previous models, most notably the sparseness issues encountered through lack of training data. This is achieved through a head-driven Markovization of synchronous context-free grammar (SCFG) compression rules. The Markovization provides several benefits including the ability to condition deletions on a flexible amount of syntactic context, to treat head-modifier dependencies independently, and to lexicalize SCFG productions. These benefits lead to more robust probability estimates.

Similarly to the noisy-channel models, Galley and McKeown's (2007) model is generative. However they estimate the joint probability $P(x,y)$ directly rather than breaking it down into $P(y) \cdot P(x|y)$. If $\tau(x,y)$ is the set of all SCFG compression rules that yield (f,c) and π_x is a parse of x then the joint probability can be estimated using:

$$P(x,y) = \sum_{(\pi_x,\pi_y)\in\tau(x,y)} P(\pi_x,\pi_y)$$

One of the problems encountered by previous SCFG approaches to compression is unreliable probability estimates for rules. This is mainly due to the use of Penn Treebank (PTB) tree structures for estimation. PTB tree structures are relatively flat which leads to sparse probabilities. For example, Galley and McKeown (2007) found that over half of SCFG compression productions only occurred once in the training set.

Instead of using PTB structures to estimate SCFG compression rules, the PTB structures are annotated to provide additional information. The first type of annotation added to each syntactic category is the category's lexical head and head part-of-speech. This annotation allows the model to determine if prepositional phrases are adjunct or complements. The second type of annotation added to syntactic categories is the *parent annotation* (Johnson 1998) which is used to break unreasonable context-free assumptions.

Galley and McKeown (2007) compare their lexicalized markov grammar model against the noisy channel of Knight and Marcu (2002) on the 32 test sentences from the Ziff-Davis corpus. However, their model is trained on a much larger training set (15,554 sentence pairs compared to 823 sentence pairs). Their system compresses with an average compression rate of 62.7% and according to human judgements outperforms the noisy channel model's compressions. Human authored compressions are favoured over either system by judges.

2.2 Discriminative Approaches

In Section 2.1 we saw how sentence compression can be framed in a generative paradigm. Two problems of the generative approach are that its simplicity is achieved by making strong statistical independence assumptions, and that training does not optimise any notion of the quality of compression.

Discriminative approaches attempt to alleviate these problems. In the discriminative paradigm a model can use a large and rich set of features to help disambiguate many natural language phenomena. Unlike in the generative approach, these features are not required to be independent and thus multiple overlapping features can be engineered. The parameters of the model are set discriminatively by minimising the error rate on the training data. Discriminatively trained models have been exploited in other areas of natural language processing and have provided state-of-the-art results, such as parsing (McDonald et al. 2005b), entity extraction (Sang and Meulder 2003) and relation extraction (Zelenko et al. 2003).

2.2.1 Decision-based Sentence Compression

Converting a source parse tree, *x*, into a target compression parse tree, *y*, can be viewed as a rewriting problem (Knight and Marcu 2002). The rewriting process can be decomposed into a sequence of shift-reduce-drop actions that follow an extended shift-reduce parsing paradigm.

The rewriting process starts with an empty stack and an input list that is built from the source sentence's parse tree. Words in the input list are labelled with the name of all the syntactic constituents in the original sentence that start with it. Each stage of the rewriting process is an operation that aims to reconstruct the compressed tree. There are four types of operations that can be performed on the stack:

- SHIFT operations transfers the first word from the input list onto the stack.
- REDUCE pops the syntactic trees located at the top of the stack, combines them into a new tree and then pushes the new tree onto the top of the stack. This can be used to derive the syntactic structure in the compressed sentence.
- DROP, deletes from the input list subsequences of words that correspond to a syntactic constituent.

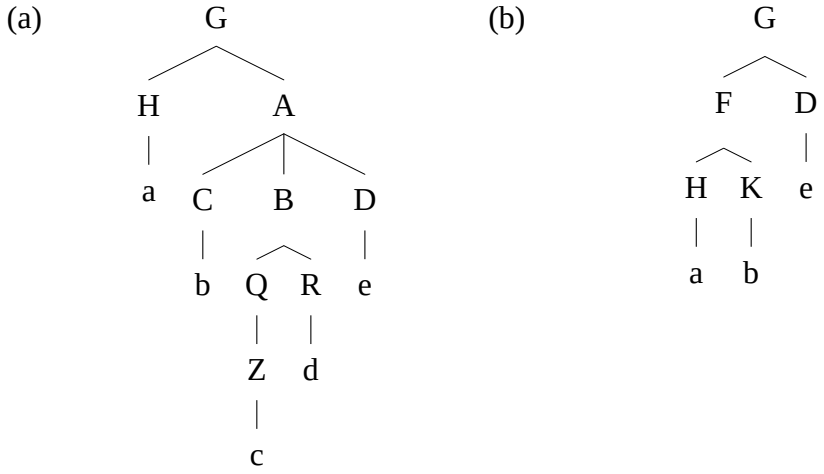

Figure 2.3: Parse trees of a source sentence (a) and its target compression (b)

- ASSIGNTYPE operations can change the label of the trees at the top of the stack (i.e, the POS tag of words can be changed).

An example of rewriting the tree in Figure 2.3 (a) into (b) is shown in Figure 2.4.

Learning cases are automatically generated from a parallel corpus. Each learning case performs one of the four possible operations for a given stack and input list. The operations represent 210 distinct operations, for example, there are distinct ASSIGNTYPE operations for each part-of-speech tag.

Each learning case is represented by 99 features that fall under two categories: operational features that reflect the current state of the input list, stack and previous operations; and source-tree-specific features that consider the tree before any operations have been applied. Using these 99 features the decision-tree model is automatically learnt using the C4.5 program (Quinlan 1993). The model tries to determine what operation should be performed on a parse given a set of features.

The decision-based model is applied to a parsed source sentence in a deterministic fashion. First an input list is built from the source sentence parse, this list contains each word and the syntactic constituents they 'begin'. The features for the current state are extracted and the classifier is queried for the next operation to perform. This is repeated until the input list is empty and the stack contains only one item (this corresponds to the parse for the compressed tree). The compressed sentence is recovered by traversing the leaves of the tree in order.

At test time the decision-based model was compared against the noisy channel model of Knight and Marcu (2002) and human authored compressions on the Ziff-

Stack	Input List	Operation
	G H a *G* A C b *G A B* Q Z c *G A B* R d *G A* D e	SHIFT ASSIGNTYPE H
H | a	*G* A C b *G A B* Q Z c *G A B* R d *G A* D e	SHIFT ASSIGNTYPE K
H K | | a b	*G A B* Q Z c *G A B* R d *G A* D e	REDUCE 2 F
F H K | | a b	*G A B* Q Z c *G A B* R d *G A* D e	DROP B
F H K | | a b	*G A* D e	SHIFT ASSIGNTYPE D
F D H K e | | a b		REDUCE 2 G
G F D H K e | | a b		

Figure 2.4: Example of Decision Tree process (italics denotes parents of nodes)

Davis corpus. The decision-based model is much more aggressive than the noisy channel approach, providing a compression rate of 57.19% on average compared to 70.37%. This is much closer to the human compression rate of 53.33%. Human ratings on grammaticality and importance show that the decision-based and noisy channel models perform comparably, however, compressions produced by both systems are significantly worse than the human authored compressions.

Nguyen et al. (2004b) extend the decision-based model using probabilistic support vector machines (SVM). They propose a two-stage method with pairwise coupling to remove the deterministic constraint of the original model. Given the probabilistic model, the score of a target compression tree, y, is obtained through its derivation, $d(y) = a_1, a_2, \ldots, a_d$ where a_i are the actions performed on the original tree to reach the compression. The score of y is the product of the conditional probabilities of the individual actions in the derivation:

$$\text{Score}(y) = \prod_{a_i \in d(y)} p(a_i|c_i) \tag{2.4}$$

where c_i is the context in which a_i was applied. A heuristic search is then used to find the best compressed tree, y^*. The SVM variant of the decision-based algorithm performed comparably to the original formulation on the Ziff-Davis corpus.

2.2.2 Maximum Entropy Reranking

The previous approaches have relied on the lexical items and parse trees to learn rules for compression. Riezler et al. (2003) use a richer sentence representation. Specifically their approach uses a Lexical-functional Grammar (LFG) parser combined with a set of rules for sentence compression learnt from a parallel corpus. Their method is supplemented with a maximum entropy model which selects the best compression.

The LFG parser (Riezler et al. 2002) produces a set of functional (f-)structures and constituent (c-)structures for a given sentence in a packed format. For sentence compression only f-structures are used, and these encode the predicate-argument structure of the sentence. A transfer component (based on one used previously in machine translation (Frank 1999)) is used to produce reduced f-structures by modifying the packed format. It rewrites one f-structure into another using an ordered set of rewriting rules. These rules include: adding, deleting and changing individual facts; all of which can be obligatory or optional. For example the optional deletion of all intersective adjuncts can transform a sentence like “He slept in the bed.” to “He slept.” However, “He did

not sleep." cannot become "He slept."

The transfer rules are independent of the grammar and thus do not always produce sentences, thus a generator is used to remove structures that do not have a lexical form. The remaining f-structures correspond to candidate compressions which are weighted by a maximum entropy model. The model is trained on a parallel corpus of source sentence and target compression f-structure pairs. The f-structures were manually selected from the candidate compression for their suitability as compressions. Around 13,000 features were used falling into three categories:

- Property-functions that indicate attributes, attribute-combinations or attribute-value pairs for f-structure attributes.
- Property-functions that indicate co-occurrences of verb stems and sub-categorisation frames.
- Property-functions indicating transfer rules used to arrive at the reduced f-structure.

The two-stage LFG system was tested on the Ziff-Davis corpus and provided an average compression rate of approximately 60%. Using a human judgement evaluation it was found the system performs comparably to the noisy channel and decision-based systems of Knight and Marcu (2002). The authors note that this result may seem disappointing consider the more complex machine employed. However they believe this is due to the limited variation possible in word deletion.

2.2.3 Online Large-Margin Learning

Thus far all previous approaches have relied heavily on various parse-trees for compression. While parse-trees are a rich source of linguistic information and allow for compression decisions to be generalised, they can suffer from noise. The previous work has treated the syntactic information as gold truth; unfortunately this is not always the case. McDonald (2006) present a discriminative approaching using a large-margin learning framework. The model has a rich feature set defined over compression bigrams which includes part-of-speech, parse-tree and dependency information. The discriminative learning algorithm learns to only trust features that are good discriminators of compression and not rely on noisy data or features that do not discriminate between compressions.

Assume we have a source sentence $\mathbf{x} = x_1, \ldots, x_n$ with a target compression $\mathbf{y} = y_1, \ldots, y_m$ where each y_j occurs in $\mathbf{x}$. The function $L(y_i) \in \{1 \ldots n\}$ maps word y_i in the

target compression to the index of the word in the source sentence, **x**. We also include the constraint that $L(y_i) < L(y_{i+1})$ which forces each word in **x** to occur at most once in the compression **y**. Let the score of a compression **y** for a sentence **x** be:

$$s(\mathbf{x}, \mathbf{y}) \tag{2.5}$$

This score is factored using a first-order Markov assumption on the words in the target compression to give:

$$s(\mathbf{x}, \mathbf{y}) = \sum_{j=2}^{|\mathbf{y}|} s(\mathbf{x}, L(y_{j-1}), L(y_j)) \tag{2.6}$$

The score function is defined to be the dot product between a high dimensional feature representation and a corresponding weight vector:

$$s(\mathbf{x}, \mathbf{y}) = \sum_{j=2}^{|\mathbf{y}|} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, L(y_{j-1}), L(y_j)) \tag{2.7}$$

Decoding in this model amounts to finding the combination of bigrams that maximises the scoring function in (2.7). McDonald (2006) uses a dynamic programming approach where the maximum score is found in a left-to-right manner. The algorithm is an extension of Viterbi for the case in which scores factor over dynamic sub-strings (McDonald et al. 2005a; Sarawagi and Cohen 2004). This allows back-pointers to be used to reconstruct the highest scoring compression as well as the *k*-best compressions.

Features The computation of the compression score crucially relies on the dot product between a high dimensional feature representation and its corresponding weight vector (see Equation (2.7)). McDonald (2006) employs a rich feature set defined over adjacent and individual parts of speech, dropped words and phrases from the original sentence, and dependency and syntactic structures (also of the original sentence). These features are designed to mimic the information presented in the previous noisy-channel and decision-tree models of Knight and Marcu (2002). Features over adjacent words are used as a proxy to the source model of the noisy-channel. Unlike other models, such as the noisy-channel and decision-tree models, which treat the parses as gold standard, McDonald (2006) uses the dependency and syntactic information as another form of evidence. Faced with noisy parses, the learning algorithm can reduce the weighting given to those features, based on the parses, if they prove poor discriminators on the training data. Thus the model should be much more robust and portable across different domains and training corpora.

Learning The weight vector, $\mathbf{w}$ is learnt using the Margin Infused Relaxed Algorithm (MIRA, Crammer and Singer (2003)) a discriminative large-margin online learning technique (McDonald et al. 2005b). This algorithm learns by compressing each sentence and comparing the result with the gold standard. The weights are updated so that the score of the correct compression (the gold standard) is greater than the score of all other compressions by a margin proportional to their loss. The loss function of McDonald (2006) is the number of words falsely retained or dropped in the incorrect compression relative to the gold standard. A source sentence will have exponentially many compressions and thus exponentially many margin constraints. To render learning computationally tractable, McDonald et al. (2005b) create constraints only on the k compressions that currently have the highest score, $\text{best}_k(\mathbf{x};\mathbf{w})$.

McDonald (2006) provided an evaluation on the Ziff-Davis corpus, in which he compared his model's output against the decision tree model (Knight and Marcu 2002) and human authored compressions. Human judges were asked to rate compressions for grammaticality and importance. They judged that McDonald's system provided more grammatical and informative compressions than the decision tree; however human authored compressions tended to be more grammatical. McDonald found that his model is more robust than the decision tree model which sometimes fails to produce reasonable compressions, for example on a handful of sentences the decision tree compressions were a single word or noun-phrase.

2.2.4 Example-Based Sentence Compression

The noisy-channel model is not the only model to be used within the machine translation paradigm. Example-based machine translation is another corpus based method of automatic translation. It can be adapted to the sentence compression problem using translation-template learning (TTL) (Nguyen et al. 2004a). The previous noisy-channel approaches relied on having a parse of the sentence available to perform compression. With a template-learning algorithm the sentences need not be represented by their parse. TTL uses examples of source and target sentences to automatically generate template rules.

Rules for template reduction map from the source sentence language to the target compression language and have the form of Equation (2.8) where S_i's and $T_j's$ are either constants or variables in the source and target language respectively.

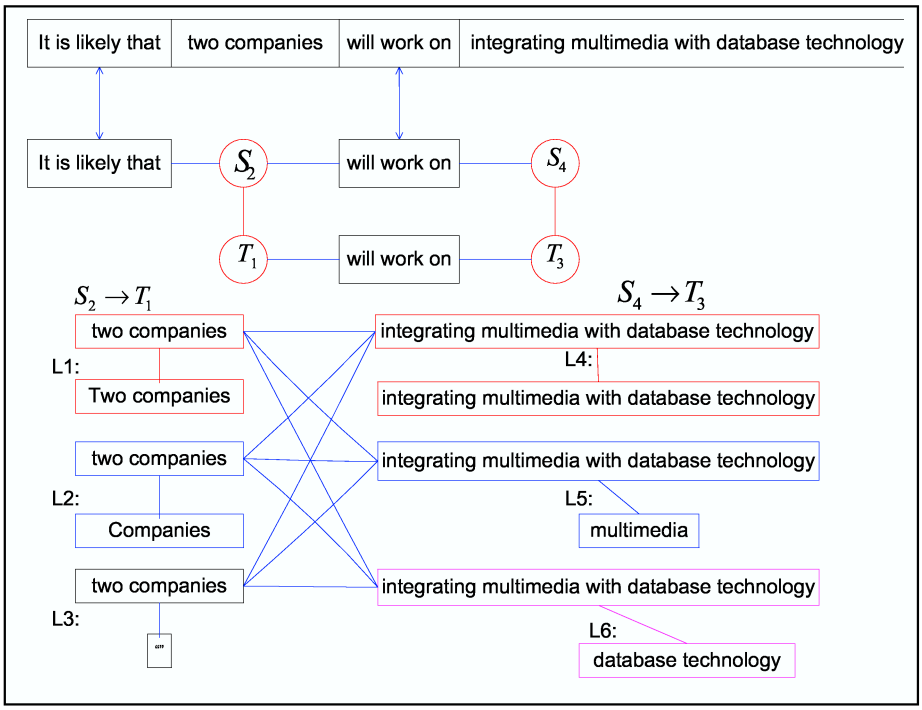

Figure 2.5: Example-based reduction for the sentence “It is likely that two companies will work on integrating multimedia with database technology”

$$S_1, S_2 \ldots S_i \ldots S_N \leftrightarrow T_1, T_2 \ldots T_j \ldots T_M \tag{2.8}$$

The template reduction rules are learnt using an unparsed parallel corpus. Pairs of examples are compared against one another to find similarities between the constituents of the two example pairs. In this case a constituent is considered to be a subsequence of lexical items. If there are no similar constituents, then a template reduction rule cannot be learnt; however, when there are similarities a match sequence is generated. TTL then aligns each side of the match sequence to form template rules. Thus all template reduction rules can be learnt automatically using only the lexical items of the sentences.

Figure 2.5 shows how the sentence “It is likely that two companies will work on integrating multimedia with database technology” can be compressed using the template rules. The two phrases “It is likely that” and “will work on” are matched to a template rule. Lexical rules are then applied to the remainder of the sentence which generate alternatives for “two companies” and “integrating multimedia with database technology”. These are shown in L_1 to L_6. An HMM is used to select a combination of rules that result in the best compression.

2.3 Data Lean Methods

The previous two sections (Sections 2.1 and 2.2) have shown how parallel corpora can be used to automatically learn the rules for sentence compression. One of the main problems of relying entirely on parallel corpora is that all the compression rules must be estimated solely from the training corpus[1] without any other knowledge about words or compressions. Thus the estimates suffer from data sparseness and are consequently unreliable. These poor estimates can eventually accumulate and result in unsatisfactory compressions. One method of improving these systems is to simply create more training data, however this is expensive and time consuming. Instead one could rely less on training data for learning and start to incorporate domain specific knowledge about sentence compression. Another approach would be to remove the parallel corpus all together and move to an unsupervised approach.

This section reviews methods for sentence compression that do not rely solely on a parallel corpus or at all.

2.3.1 Knowledge Rich Compression

Jing (2000) uses multiple knowledge sources to determine which phrases in a sentence can be removed. These knowledge sources are combined with a small amount of parallel data to select nodes of a parse tree for removal and include:

- multiple lexical resources that together form a rich lexicon (Jing and McKeown 1998). These consist of a subcategorisation lexicon for over 5000 verbs and also include: the COMLEX syntactic dictionary (Grishman et al. 1994), English verb classes (Levin 1993) and WordNet (Miller 1995). The lexicon is used to identify the obligatory arguments of verb phrases.

- lexical relations between words such as synonymy, entailment and causation are identified using WordNet and provide information about the focus of the local context as determined by the number of relations between words (i.e., words with more links to other words are important for the local context).

- a parse tree of the sentence with thematic roles of phrases (such as object or subject) using the English Slot Grammar (ESG) parser (McCord 1989)

[1] Although Turner and Charniak (2005) do propose an unsupervised method which they recommend using in combination with rules obtained from a parallel corpus.

- a small collection of 500 source-target sentence pairs are used for training and testing purposes. These were gathered automatically from a news service provided by the Benton Foundation[2] consisting of news reports on telecommunication related issues.

The compression algorithm works in five stages. The first stage involves parsing the sentence with the ESG parser; this provides a base parse tree that later stages annotate with additional information.

Stage two involves determining which components of the sentence must not be deleted in order for the sentence to remain grammatical. Each node in the parse tree is traversed and its children are marked if they are grammatically obligatory with respect to their parent. Simple linguistic rules determine which words should be marked; such as the head noun of a noun phrase, and the main verb, subject and object of a sentence if they are present. A second method is also used which relies on the lexicon. This stage results in each node of the parse tree being annotated with a value indicating whether it is grammatically obligatory (relative to its parent node).

The next stage takes contextual information into account. Words in the sentence are linked to words within the local context, which is assumed to also be the sentence. Words can be linked in a variety of ways through repetition, morphological relations or WordNet's lexical relations; there are nine such relations in total. The more often a word occurs in the local context (the sentence) the more important it is. The nine relations used can be weighted according to how strongly the relation holds. For example, repetition and inflectional relations are considered more important and thus given a higher weight than the hypernym relation. These word scores are then used to score the phrases within the sentence with different relations contributing a different weight to the overall score.

Stage four involves corpus evidence gathered from a parallel corpus of sentence pairs. This includes probabilities on the removal of clauses given their head noun or main verb, the reduction of a phrase or clause (where the phrase is altered but not removed entirely), the phrase being unchanged.

The final stage decides which phrases should be dropped or reduced given all the scores of the previous steps. A phrase will be dropped if it is not grammatically obligatory, not the focus of the local context and there is *reasonable* past evidence that it would be removed by humans. If there is no previous corpus evidence for a drop

[2] http://www.benton.org

the system uses the lexical and context information to determine how to compress a sentence.

The compression system draws on large amount of knowledge about the characteristics of language. These knowledge sources are united through rules and handcrafted scores. This makes the approach difficult to port to new domains or languages.

Jing (2000) tests her system on a corpus of 100 compressed sentences against a baseline system which removes all preposition phrases, clauses, *to* infinitives and gerunds. Phrase removal probabilities were calculated from a corpus of 400 sentences. For evaluation, Jing defines a *success rate* automatic measure which calculates the percentage of system compression decisions that agree with human decisions (see Chapter 4 for details). Her system achieves a success rate of 81.3% which considerably outperforms the baseline (success rate 43.2%). In terms of compression rate her system on averages compresses to 67.3% whereas the human authored compressions were approximately 58%.

2.3.2 Word-based Compression

All previous approaches have used parallel corpora to different degrees to learn what a compressed sentence should look like or when to perform compression. In contrast, Hori and Furui (2003) propose an unsupervised method for sentence compression. It is part of an automatic speech summarisation system that compresses individual sentences and then joins them together to form a summary.

A set of words are extracted from a sentence according to a summarisation score. We could equally term this score a compression score, which must be maximised for a fixed and prescribed compression ratio. This approach goes beyond simple word extraction as it not only selects the important words in the sentence but also ensures function words are selected which lead to a grammatical output.

The summarisation score (see Equation (2.9)) is a combined measure of the appropriateness of the compressed sentence; it consists of individual scores that measure word significance (I), word confidence (C), linguistic likelihood (L) and word concatenation likelihood (T). The lambdas ($\lambda_L, \lambda_C, \lambda_T$) are used as weighting factors to adjust the contribution of each score.

$$S(\mathbf{y}) = \sum_{i=1}^{m} \{ \quad I(y_i) + \lambda_L L(y_i|\ldots y_{i-1}) + \lambda_C C(y_i) + \lambda_T T(y_{i-1}, y_i) \} \tag{2.9}$$

The sentence $\mathbf{y}$ (of m words) that maximises the score $S(\mathbf{y})$ is the best compression for an original sentence consisting of n words ($m < n$).

We now introduce each measure individually, giving details of how a value is derived for each word.

Word significance score The word significance score I measures the relative importance of a word in a document. This is similar to the tf-idf score (Salton 1988) that is popular in the information retrieval community; it is given by Equation (2.10).

$$I(w_i) = f_i \log \frac{F_A}{F_i} \tag{2.10}$$

Where w_i is the topic word of interest, f_i is the frequency of w_i in the document, F_i is the corpus frequency of w_i and F_A is the sum of all topic word occurrences in the corpus ($\sum_i F_i$). Topic words are defined as nouns and verbs. A flat score is assigned to non-topic words and repeated topic words within the sentence.

Linguistic score The linguistic score's $L(w_m|\ldots w_{m-1})$ responsibility is to select function words thus ensuring the compressions remain grammatical. It also controls which topic words can be placed together. The score is measured by the *n*-gram probability of the compressed sentence.

Confidence score A confidence score C is taken from the output of an automatic speech recogniser (ASR). This measures how certain the recogniser is that the acoustics for the given word match the output. This is necessary when working with ASR output rather than transcribed speech. The argument is that words which the ASR predicts with little confidence should not be included in the compression as they will introduce errors into the compression.

Word concatenation score The linguistic score alone is not powerful enough to stop the concatenation of topic words that make linguistic sense but cause semantic differences between the original and compressed sentences. For example, sentence (2) is a grammatical compression of (1), however it is semantically incorrect as *beautiful* modifies *cherry blossoms* and not *Japan*. The word concatenation score is designed to alleviate this problem.

(1) The beautiful cherry blossoms in Japan

(2) The beautiful Japan

The score is calculated based on the sum of the dependency probabilities between two words (w_i, w_j) and between w_i and each of $w_{j+1}, \ldots, w_n$. The dependency probabilities are estimated from a Stochastic Dependency Context Free Grammar (SDCFG) (Hori et al. 2003).

Maximising the summarisation score The summarisation score (Equation (2.9)) is maximised for a given compression length m using dynamic programming. We can break the problem down into smaller sub-problems that compute the optimal substructures for the compression. Firstly, the summarisation score for sub-sentences consisting of one word are calculated. Sub-sentence hypotheses for two words are then calculated using the optimal substructures for sub-sentences consisting of one word and so on. This is recursively done until we have calculated the optimal summarisation score for the compression of m words using the previous sub-sentences. A backtracking process is then performed that selects the correct word sequence $\mathbf{y} = y_1, \ldots, y_m$ that maximises the summarisation score.

This algorithm has been extended to provide a summary of multiple sentences (Hori et al. 2003), by compressing each sentence at varying compression ratios and then selecting the best combination of compressed sentences according to an overall compression ratio for the set of sentences. The first stage is performed using the process described above, while the second stage is done using another dynamic programming process.

Hori and Furui (2004) evaluate their compression method at fixed compression rates of 40% and 70% against a baseline which randomly removes words until the desired compression rate is reached. Fifty utterances from CNN TV news broadcasts in English were used for evaluation purposes. Seventeen annotators compressed the sentences and the compressions were merged to form a word network to use in automatic evaluation (see Chapter 4 for more details). Their compression methods performed better than the baseline in all tests.

2.4 Discussion

Previous approaches to sentence compression model the process using *local* information. For instance, in order to decide which words to drop, they exploit information

about adjacent words or constituents. Local models can do a good job at producing grammatical compressions, however they are somewhat limited in scope since language has more *global* properties. The long range dependencies inherent in language mean many simple linguistic phenomena are difficult to capture with models that rely exclusively on local information. The desire to model more global properties in compression is apparent in the work of Turner and Charniak (2005) where they incorporate constraints on the rules they generate. These constraints are simple and allow for undesirable rules to be filtered from the whole rule set. However, such an approach is only applicable to models that map the compression task in a synchronous context-free grammar framework. It is desirable to have a general framework for modelling long range dependencies and linguistic phenomena which does not merely pre-process a selection of rules or decisions, or post-process the resulting compressions through editing or selection via an *n*-best list.

We will now provide some concrete examples of the long range and sentence level dependencies we wish to preserve from the source sentence when generating a compression.

- The compressed sentence should contain at least one verb, provided that the source sentence had one in the first place.
- When verbs are included in the compression their arguments should be preserved thus the semantics of the compressed sentence must be carried over from the source sentence.
- Dependencies between head words and modifiers should remain semantically valid in the compressed sentence. Examples of this include, permitting the removal of non-essential modifiers, not including modifiers if their head word has been removed and ensuring negations are held in the compression.
- In document compression, as opposed to isolated sentence compression, the discourse of a document should remain coherent. To achieve such a goal we need to ensure that the topic of the compression flows from one sentence to the next.

There maybe other properties of the generated compression which we may wish to capture but the current models are beyond learning; such properties need not only be linguistic or semantic but might be task or application specific. For example, an application which compresses text to be displayed on small screens would presumably

have a higher compression rate than a system generating subtitles from spoken text. These kind of properties are very difficult for the model to learn unless we have training data tailored to the task or application. Again a general method of incorporating such knowledge into the model is desirable.

Existing approaches do not model global properties of the compression problem, despite the potential benefits. This is for good reason. Finding the best compression for a long sentence given the space of all possible compressions[3] (this search process is often referred to as decoding or inference) can become intractable for too many constraints and overly long sentences. Typically, the decoding problem is solved efficiently using dynamic programming often in conjunction with heuristics that reduce the search space (e.g., Turner and Charniak 2005). Dynamic programming guarantees we will find the global optimum provided the principle of optimality holds. This principle states that given the current state, the optimal decision for each of the remaining stages does not depend on previously reached stages or previously made decisions (Winston and Venkataramanan 2003). However, we know this to be false in the case of sentence compression. For example, if we have included modifiers to the left of a noun in a compression then we should probably include the noun too, also if we include a verb its arguments should also be included. With a dynamic programming approach we cannot easily guarantee such global properties are enforced.

In later chapters we will begin to address the issue of modelling long range dependencies and other global and local properties in a manner that is applicable to many compression approaches.

2.5 Summary of Chapter

In this chapter we have examined the computational treatment of sentence compression and characterised the performance of each system.

The supervised systems that are comparable from the results presented can be summarised as follows: the decision-based systems (decision tree and two-stage SVM), the noisy-channel model of Knight and Marcu (2002) and Riezler et al.'s (2003) compression system all perform similarly according to human judgements, however the decision-based systems compressed much closer to the gold standard compression rate. These systems are outperformed by Turner and Charniak's (2005) noisy-channel model and Galley and McKeown's (2007) system using lexicalized markov grammars.

[3] There are 2^n possible compressions where n is the number of words in a sentence.

However, the former only compresses at 81.2% and the latter used considerably more training data. Finally, McDonald's (2006) system outperforms the decision-based systems using the same amount of training data and retains a similar compression rate to the gold standard. The other systems are difficult to compare because the evaluations have not contained a baseline system common in other evaluations.

We concluded the chapter with a discussion of some of the limitations of the current approaches and described various examples of long range and sentence level dependencies we would like to capture.

Chapter 3

Sentence Compression Analysis

The previous chapter introduced various models of the sentence compression task. Like many natural language processing techniques the majority of compression techniques fall under a supervised setting. The requirements on a supervised learning algorithm are that there is a training set of example input-output pairs for which to learn the model's parameters.

A set of training examples is usually termed a *parallel corpus* when text rewriting occurs between a source and target text (in our case sentences). Obtaining a parallel corpus is often a laborious task. Luckily in some text rewriting tasks, such as machine translation and summarisation, it has been possible to automatically collect a parallel corpus. For example, in machine translation, parallel corpora occur naturally within limited domains. They are often a by-product of governmental efforts to provide documents and proceedings to a multilingual populace. However, when large problems are split into sub-tasks training data may be difficult to obtain as the output of the sub-tasks may not be directly observable. The sentence compression task exhibits this problem. Compressions are not as naturally abundant as summaries or translations. Even rarer are compressions restricted to being formed by word deletion alone (see the definition in Chapter 1).

When a parallel corpus cannot be automatically acquired it is necessary to build one manually. In such situations a set of guidelines must be produced for annotators to follow when creating examples. This ensures the annotations made are consistent between annotators.

In this chapter we will discuss two methods for gathering compression corpora and motivate the approach we adopt in this thesis. Next, we will provide a detailed analysis of the compression task and highlight any differences between automatically gathered

Source	The *speakers notes and handouts are a by-product of the slide show process,* and add the professional polish to your presentation without extra effort.
Target	*Speakers notes and handouts are a by-product of the slide show process.*
Source	The *documentation is excellent* – it is clearly written with numerous drawings, cautions and tips, and includes an entire section on troubleshooting.
Target	*Documentation is excellent.*
Source	*The FTS 2000 acquisition strategy went beyond the* basic *objective of simply replacing the 25-year-old FTS.*
Target	*The FTS 2000 acquisition strategy went beyond the objective of replacing the 25-year-old FTS.*
Source	*The simplest topology is the daisy chain.*
Target	*The simplest topology is the daisy chain.*

Figure 3.1: Sentence compression examples from the Ziff-Davis corpus. Sentences marked Source are the original source sentences and Target the target compressions. Words in italics are shared between source and target.

and human authored corpora; and between spoken and written domains.

3.1 Compression Corpora

Automatically Created Corpora One method of automatically obtaining a parallel corpus of original *source* sentences and *target* compressed sentence pairs has been proposed and successfully employed by Knight and Marcu (2002). Given a collection of documents and corresponding abstracts we can automatically extract original source-target compression pairs. Assuming a document containing source sentences $D = s_1, s_2, \ldots, s_n$; we can search the document's abstract, $A = t_1, t_2, \ldots, t_n$ for a target compression t where the words in the compression are a subset of those in the original source sentence, s, and the words occur in the same order in both sentences.

Previous work on sentence compression has almost exclusively used the Ziff-Davis corpus for training and testing purposes. This corpus originates from a collection of news articles on computer products. The corpus was created automatically using the

previously described procedure. A training set and test set form the corpus consisting of 1035 sentences and 32 sentences respectively. Each source sentence is provided out-of-context. Figure 3.1 demonstrates some of the sentences found in the Ziff-Davis compression corpus.

Galley and McKeown (2007) note that the Ziff-Davis corpus contains over 4000 abstract-document pairs and the 1087 extracted sentence pairs represent a recall of only 1.84%. To gather additional training material they loosen the assumption that a target compression must only involve word deletion with respect to the source sentence and allow for substitutions and insertions. For example, Sentence (1-b) is now considered a valid compression of (1-a) as it includes the one-word substitution of *computer* with *unit.*

(1) a. The second computer started up and ran without incident.
b. The second unit ran without incident.

Although in this example *computer* and *unit* are meaning equivalent, Galley and McKeown's (2007) method considers substitutions without any knowledge about word relations. The resulting compressions may therefore contain noise (i.e., the substitutions performed may not always correspond to the same entities).

Relaxing the word deletion assumption allows for a richer set of compressions to be gathered automatically and allows for more varied compressions. Unfortunately, obtaining source-target compression pairs automatically when insertions and substitutions are permitted is not a trivial task especially when the number of non-deleting edits increases. The task is known to be NP-hard, however approximate algorithms exist which run in polynomial time (Zhang and Shasha 1989). Galley and McKeown (2007) gathered source-target compressions with up to six substitutions which resulted in 16,787 example pairs representing a recall of almost 25% of the abstract sentences in the Ziff-Davis corpus. They did not consider insertions as they adversely affected their compression model. It is important to note that although their model is trained on data containing substitutions it nevertheless still only performs word deletion during compression.

Manually Created Corpora Automatically constructed parallel corpora are created by matching sentences that occur in a document with sentences that occur in an abstract. The target abstract sentence must contain a subset of the words from the source sentence and the word order must remain the same. While this is a suitable method

when a parallel corpus is required cheaply and quickly it does have drawbacks.

One concern is the nature of the compressions. Although the compressions are valid with respect to our limited definition they may not be representative of human performance. Compressions derived from abstracts may contain artefacts of other tasks performed during the summarisation process rather than compression. Typically multiple factors are taken into consideration when creating a summary such as the information already placed in the summary, future summary content and the context of the information being summarised. These factors are required to ensure the summary flows and is coherent and truthfully represents the source material. It is difficult to imagine a coherent summary if compression is being performed in isolation on selected sentences.

Another concern is the limited scope of the compressions. Automatically derived compressions corpora are not suitable for investigating compression beyond isolated sentences, since it is unlikely that an abstract will be entirely composed of compressed sentences. This is an important issue. For example, in summarisation we may wish to compress a whole document prior to sentence extraction. In this case it would be beneficial to consider the discourse flow and document structure during compression. Another scenario is where sentence extraction first takes place and then a compression system compresses the extracted sentences to form an abstract (Lin 2003); again the dependencies between the abstract sentences should be taken into account. Without such wholly compressed documents or abstracts it is difficult to investigate the factors of compressing documents; such an omission would be unfortunate due to the relevance of document compression to applications.

Finally, although Knight and Marcu (2002) were able to create a compression corpus fairly easily using the Ziff-Davis corpus their recall of abstract sentences was extremely low. We have found that the technique does not yield as many compressions on other corpora of abstract-document pairs. Another problem is that abstract-document paired corpora only occur naturally within limited domains such as written news and scientific articles, in other domains such corpora are unavailable thus making it difficult to study compression.

Taking into consideration the previous points we manually created two compression corpora to investigate:

- Whether human compressions are similar to those obtained automatically.
- Whether there are any differences between compressions produced from differ-

ent domains.

- The range of compression phenomena within a document rather than focusing on isolated sentences.

We compiled two compression corpora to aid this investigation, a written and a spoken corpus. The appeal of written text is understandable since most summarisation work to date has focused on this domain. Speech data not only provides a natural test-bed for compression applications such as subtitle generation but also poses additional challenges. Spoken utterances can be ungrammatical, incomplete, and often contain artefacts such as false starts, interjections, hesitations, and disfluencies. Rather than focusing on spontaneous speech which is abundant in these artefacts, we conduct our study on the less ambitious domain of broadcast news. This lies in-between the extremes of written text and spontaneous speech as it has been scripted beforehand and is usually read off autocue. However, speech artefacts still arise in many places such as when presenters misread the autocue and during live segments and unscripted interviews.

Following the classification scheme adopted in the British National Corpus (BNC), we assume that our two corpora belong to the same genre (news) but to different domains (written and spoken). Our first corpus consists of news articles gathered from the BNC and the American News Text corpus. The articles originate form The LA Times, Washington Post, Independent, The Guardian and Daily Telegraph newspapers. Eighty-two articles, totalling 1,433 sentences, were selected for compression. The corpus was split into training, development and testing sets randomly on article boundaries. These sets contain 908, 63 and 462 sentences respectively. We refer to this corpus as the written corpus. The second corpus is a spoken corpus consisting of 50 broadcast news stories (1,370 sentences) taken from the HUB-4 1996 English Broadcast News corpus provided by the LDC. The HUB-4 corpus contains broadcast news stories from a variety of networks (CNN, ABC, CSPAN and NPR) which have been manually transcribed and split at the story and sentence level. Again the corpus has been divided into 882 training sentences, 78 development sentences and 410 testing sentences; each set contains full stories. We call this corpus the spoken corpus. Both corpora were automatically segmented at the sentence level and tokenised using the Robust Accurate Statistical Parsing (RASP, Briscoe and Carroll 2002) system.

Two annotators were asked to perform sentence compression by removing tokens from the original document. They were instructed to remove words while consider-

Source	President Boris Yeltsin has won the most votes in Russia's hotly contested presidential election, one watched around the world.
Annotator 1	Boris Yeltsin has the most votes in Russia's presidential election.
Annotator 2	Boris Yeltsin has won the most votes in Russia's presidential election.
Source	He became a power player in Greek politics in 1974, when he founded the socialist Pasok party.
Annotator 1	He became a power player in Greek politics in 1974, when he founded the socialist Pasok party.
Annotator 2	In 1974 he founded the socialist Pasok party.
Source	The number of people entitled to civil legal aid has fallen by more than 14 million since 1979, according to research published today.
Annotator 1	The number of people entitled to legal aid has fallen by 14 million since 1979.
Annotator 2	The number of people entitled to legal aid has fallen by 14 million since 1979.
Source	Some experts say that even if the eruption stopped today, the sheer pressure of lava piled up behind for six miles would bring debris cascading down on to the town anyway.
Annotator 1	Experts say even if the eruption stopped, the sheer pressure of lava piled up for miles would bring debris down on the town.
Annotator 2	Experts say even if the eruption stopped today, the pressure of lava piled up for six miles would bring debris on to the town.

Figure 3.2: Sentence compression examples from the two human authored compression corpora. The first two examples are taken from the spoken corpus and the last two from the written corpus. (Source: source sentence, Annotator 1: first annotator's compression, Annotator 2: second annotator's compression).

ing: (a) the most important information in the original sentence, and (b) the grammaticality of the compressed sentence. If they wished they could leave a sentence uncompressed by marking it as inappropriate for compression. They were not allowed to completely delete sentences even if they believed they contained no information content with respect to the document. This final constraint simplifies the task and ensures a boundary exists between compression and summarisation. See Appendix A.1 for the full instructions and examples given to our annotators.

Figure 3.2 demonstrates some example compressions created by our annotators. The first two sentences originate from the spoken corpus whereas the final two sentences come from the written corpus. These corpora demonstrate the variation possible and observed during compression.

3.2 Corpus Analysis

We first begin by providing an analysis of the three compression corpora: the automatically constructed Ziff-Davis corpus, and our two human authored compression corpora on spoken text and written text.

Compression Rate Compression rate is a measure of how terse a compression is and is given in Equation (3.1). A compression rate of 100% implies the sentence is left uncompressed.

$$\text{Compression Rate} = \frac{\text{Length of compression}}{\text{Length of source}} \tag{3.1}$$

We compute compression rate on a sentence-by-sentence basis as the task is defined over sentences. The compression rate for an entire document or a collection of compressions is calculated by taking the average of the compression rates for the collection.

Table 3.1 shows the average compression rate for each corpus and annotation method. The table displays a distinct difference between the human authored compression corpora and the automatically obtained corpus (Ziff-Davis). The Ziff-Davis is compressed much more aggressively than our human authored corpora. This maybe due to the methodology used in obtaining this corpus. We can see that the compressions created by our annotators are much more conservative in comparison and are similar across domains. The compression rate for our human authored corpora are within 5% of one another with an average rate of approximately 73%.

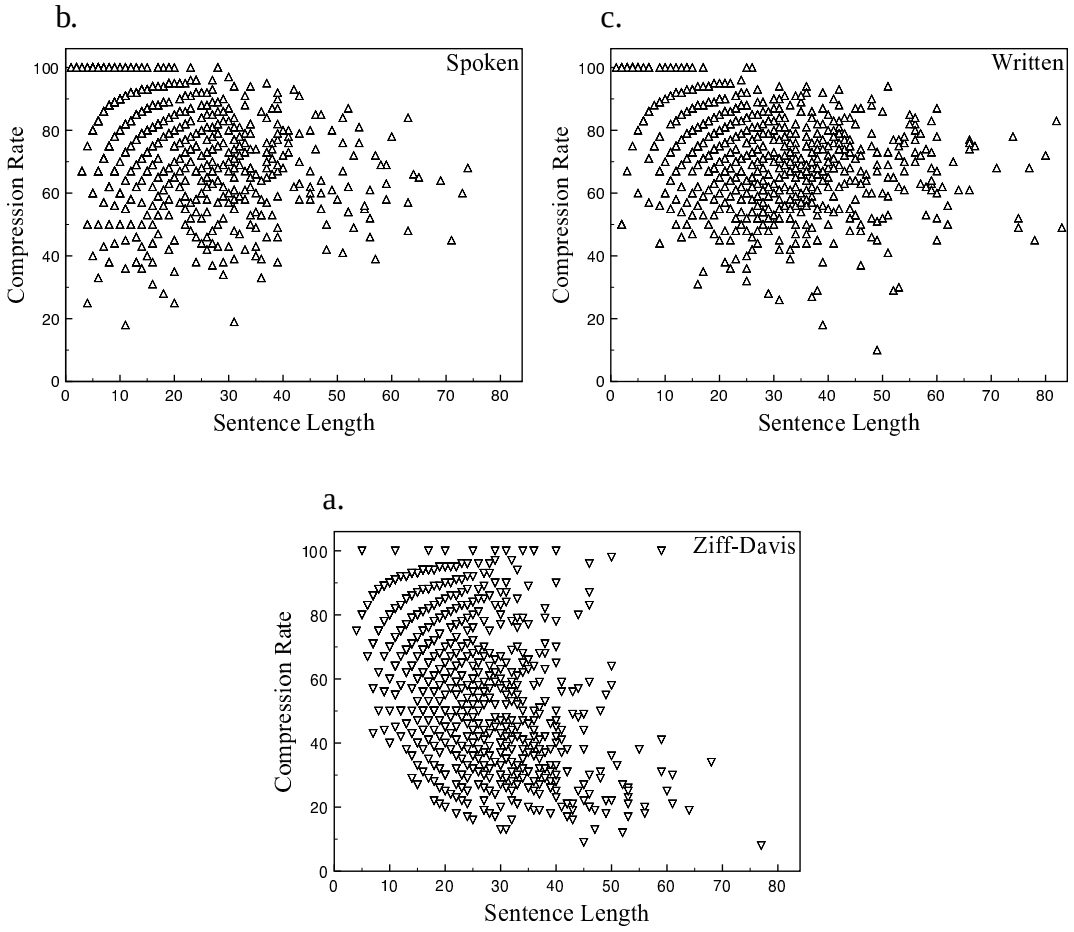

Figure 3.3: Scatter plots of source sentence length against compression rate for the three corpora (a. spoken corpus, b. written corpus, c. Ziff-Davis).

Corpus	Annotation	Length	Comp Rate
Spoken	Source	20.36	100%
Spoken	Human 1	14.67	75.2%
Spoken	Human 2	13.58	70.7%
Written	Source	27.83	100%
Written	Human 1	19.48	72.6%
Written	Human 2	20.57	74.2%
Ziff-Davis	Source	23.91	100%
Ziff-Davis	Automatic	12.74	58.1%

Table 3.1: Compression Rates for the two manually constructed corpora (spoken text and written text) and the automatically constructed Ziff-Davis corpus. Length: average sentence length; Comp Rate: average compression rate (where 100% implies uncompressed).

We next turn our attention to examining the relationship between sentence length and compression rate. Such an analysis may provide insight into selecting a compression rate for a given sentence. Figure 3.3 shows plots of the source sentence length against the compression rate for our spoken and written corpora and the Ziff-Davis corpus. All three plots are extremely scattered and demonstrate no correlation between source sentence length and compression rate. For example, if we examine sentences with a length of approximately 30 tokens, we see that the compression rate on the Ziff-Davis corpus (see Figure 3.3c) ranges from 12% to 100%, for the spoken corpus (Figure 3.3a) the range is 19% to 98% and 26% to 95% for the written corpus (Figure 3.3b). This suggests that compression rate is a function of a higher level property of the sentence, it does not depend on the sentence's surface features (such as length) but more likely is determined by its structure and information content. Thus the compression rate can not be assumed fixed throughout a corpus or even for sentences of a given length. Despite this, it may be desirable to specify a minimum, maximum or range of compression rates for various applications.

Although Figure 3.3 appears to display some regular patterns (see the arc shapes formed by the plot) such formations are due to the discrete nature of compression rate rather than any inherent pattern in the data. Compression rate is not a continuous measure: there are only a certain number of fixed compression rates possible for each sentence (i.e., n possible compression rates, where n is source sentence length).

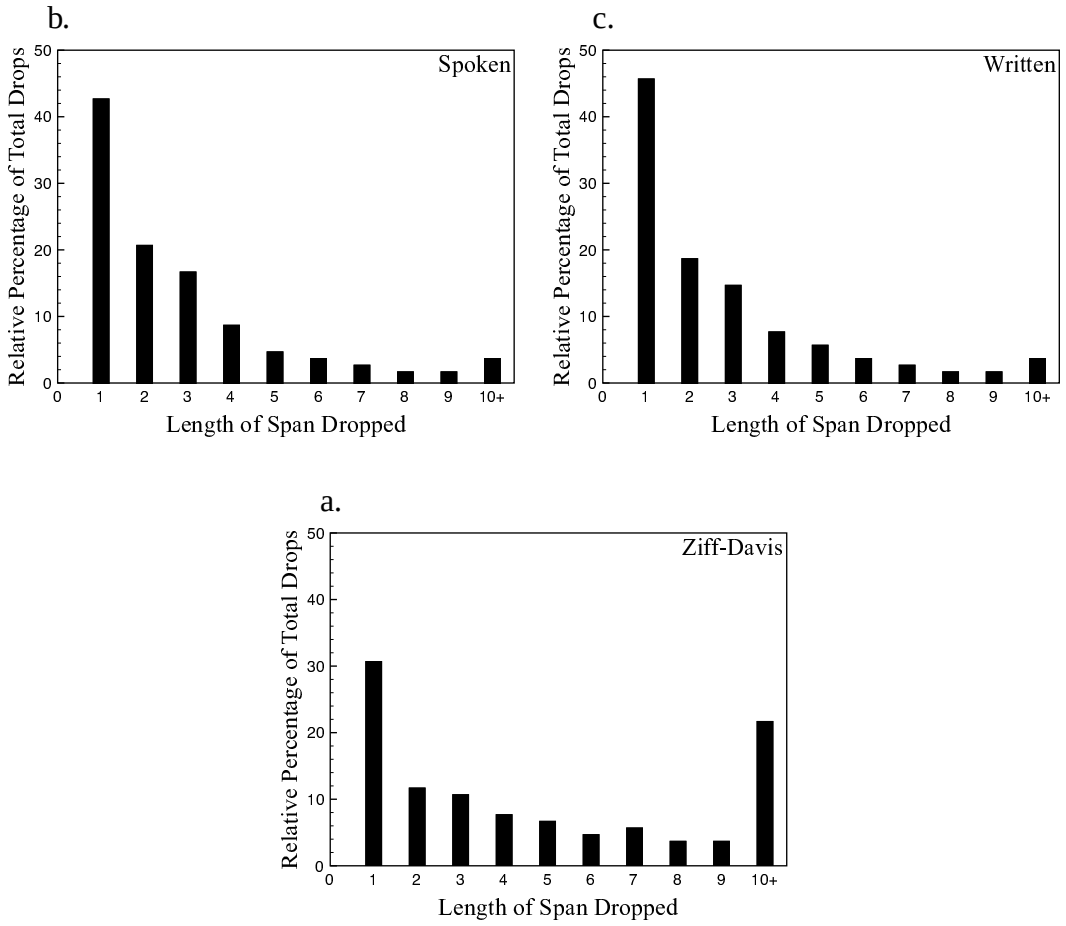

Figure 3.4: Distribution of spans of words dropped (a. spoken corpus, b. written corpus, c. Ziff-Davis)

The human annotators rarely compress sentences below 40% (removing over 60% of the words) as shown in Figures 3.3a and 3.3b. However, the Ziff-Davis corpus (Figure 3.3c) displays a large proportion sentences being compressed beyond 40%. It is also interesting to note that there is little difference in the plots between the domains of written and spoken text when compressions are being produced manually. The annotators also leave more sentences uncompressed in comparison to the Ziff-Davis corpus. This can be seen in the figures by the concentration of points at a compression rate of 100%.

Word Removal Analysis We also examined whether the three corpora differ with regard to the length of word spans being removed. Figure 3.4 shows how frequently word spans of varying lengths are being dropped. A word span is defined as a continuous sequence of tokens. As can be seen, a higher percentage of long spans (five

Constituent	Spoken		Written		Ziff-Davis	
	Total	% drop	Total	% drop	Total	% drop
NP	5892	18%	8678	19%	7114	38%
PP	1754	16%	2757	22%	2195	42%
SBAR	692	11%	863	8%	511	37%
WHNP	162	43%	247	34%	215	83%
VP	3302	8%	4320	7%	3379	27%
S	2324	6%	2734	5%	2227	21%
ADVP	564	57%	476	61%	421	64%
ADJP	305	14%	342	22%	402	35%

Table 3.2: Percentage of constituents dropped for the spoken corpus, written corpus and Ziff-Davis. Total refers to the frequency the constituent occurs in the source data. % drop is the percentage of times the constituent was dropped in forming the compression.

or more words) are dropped in the Ziff-Davis corpus. This suggests that the annotators are removing words rather than syntactic constituents. Closer examination shows there are no significant differences in the length of spans dropped between annotators on the same corpus. There are differences between the human authored corpora (spoken text and written text) and the automatically created corpus (Ziff-Davis), and these differences are significant at a level of $p < 0.01$ using the Wilcoxon Test. There is no statistically significant difference between the human authored written text and spoken text corpora.

We next investigate the deletion of syntactic units. We parsed our corpora using Roark's (2001) parser which provides Penn Treebank style annotations. Table 3.2 illustrates how often each constituent was dropped in the compression as a percentage. A higher percentage of constituents are being dropped in the Ziff-Davis. This is somewhat expected since the Ziff-Davis corpus is compressed at higher rate. A relatively high percentage of *wh*-noun phrases (WHNP) are dropped throughout all corpora; these typically introduce clauses and contain a *wh*-word, e.g. *who, which, whose abstract form, that, precisely what*, etc. It is interesting to note that in the two human authored corpora, clauses are not often dropped (SBAR, 10% and S, 5%) in comparison to the Ziff-Davis corpus (SBAR, 37% and S, 21%). This is most likely due to only part of the clause being dropped in the two human authored corpora. Adverbial phrases (ADVP) are dropped frequently throughout all corpora suggesting they are

Tag	Spoken		Written		Ziff-Davis	
	Total	% drop	Total	% drop	Total	% drop
AUX	1058	24%	1105	20%	801	36%
CC	510	40%	624	39%	805	50%
DT	1720	28%	2377	30%	2023	48%
IN	1929	32%	2862	35%	2162	53%
JJ/JJR/JJS	1072	32%	1796	47%	1945	45%
MD	175	21%	256	12%	281	34%
NN/NNS	3083	23%	4800	28%	5255	40%
NNP/NNPS	1559	16%	2642	25%	1612	50%
PRP/PRP$	932	30%	981	27%	308	68%
RB/RBR/RBS	922	55%	806	58%	663	63%
TO	446	25%	637	27%	481	46%
VB	497	20%	537	20%	533	37%
VBD	251	17%	697	14%	188	73%
VBG	315	19%	394	21%	318	51%
VBN	353	19%	688	25%	450	42%
VBP	165	53%	82	32%	147	42%
VBZ	151	22%	114	21%	417	39%
WDT/WP/WP$/WRB	232	41%	331	31%	237	81%

Table 3.3: Percentage of part-of-speech (POS) tags dropped for spoken, written and Ziff-Davis corpora. Total refers to the frequency the POS tag occurs in the source data. % drop is the percentage of times the POS tag was dropped in forming the compression.

usually superfluous. We see in our human authored corpora that sentences (S), clauses (SBAR) and verb phrases (VP) are very important and not often dropped whereas the Ziff-Davis compressions are more inclined to remove them.

We provide details of the grammatical categories being dropped in Table 3.3. The table shows various part-of-speech (POS) tags, their frequency in the source sentences from our training corpora, and the percentage of times each POS tag is dropped. When we examine Table 3.2 in conjunction with Table 3.3 we see that a naive baseline that removes all prepositional phrases, clauses, *to*-infinitives and gerunds will struggle to create high quality compressions; this has also been empirically observed (Jing 2000). The table also provides evidence against other naive methods such as dropping all

Tag	Spoken	Written	Ziff-Davis
AUX	5%	3%	3%
CC	4%	3%	4%
DT	10%	10%	10%
IN	12%	13%	11%
JJ/JJR/JJS	7%	11%	9%
NN/NNS	14%	18%	20%
NNP/NNPS	5%	9%	8%
PRP/PRP$	6%	4%	2%
RB/RBR/RBS	10%	6%	5%
TO	2%	2%	2%
VB*	8%	7%	7%

Table 3.4: Relative percentage of total part-of-speech (POS) tags dropped.

adjectives (JJ/JJR/JJS) as we observe less than 50% of all adjectives are dropped in forming compressions.

Table 3.4 provides a slightly different look at the grammatical categories being dropped. The table shows the proportions of the total drops accounted for by each part-of-speech (POS) tag. This shows that the frequently occurring tags tend to account for a larger percentage of total drops and similar proportions are observed across all corpora.

3.3 Summary of Chapter

In this chapter we have presented a novel and detailed analysis of the sentence compression task. We examined various methods for data acquisition. This has resulted in the creation of two new publicly available compression corpora[1] in the spoken and written domains. Upon examining the corpora we have found that the compressions produced by our annotators differ to those obtained automatically from the Ziff-Davis corpus. The annotators' compressions are much more conservative than those automatically acquired (approximate 70% compression rate compared to 50% of the Ziff-Davis). Our annotators were asked to perform sentence compression explicitly as an isolated task rather than indirectly as part of the broader task of abstracting, which

[1]The data can be downloaded from `http://homepages.inf.ed.ac.uk/s0460084/data`

we can assume is the case with the Ziff-Davis corpus. The framing of the task may have been a contributing factor in the differences observed between the corpora. For example, the Ziff-Davis compressions may not be trying to retain all the important information in a sentence, instead only retaining the information which is relevant to the rest of the abstract sentences. This suggests that the Ziff-Davis corpus may be more representative of an abstracting task rather than a pure compression task.

Our compression analysis affords several conclusions regarding the task. Setting a fixed compression rate is inappropriate unless the application imposes a certain compression rate. Methods which remove prepositional phrases, clauses, to infinitives and gerunds will prove to be weak baselines; our word removal analysis demonstrates that the sentence as a whole plays a role in compression not just its linguistic units in isolation. Also the notion of how much to compress a sentence goes beyond the surface features of the source sentence and is more likely related to the information content of the sentence.

Chapter 4

Evaluation Techniques

Evaluation is an important aspect of any natural language processing task. Without systematic evaluation it is impossible to assess the quality of an NLP system and compare performance against other systems. Many NLP tasks (e.g., parsing, named entity recognition, chunking and semantic role labelling) can be automatically evaluated using standard precision and recall measures. However, this is not always applicable to text generation tasks such as summarisation, machine translation and sentence compression where there is no unique gold standard against which to evaluate the system's output. For example, in machine translation there are multiple possible translations of the source sentence which can be considered correct. The same is true for summarisation and sentence compression. Our annotators do not always compress a source sentence identically (Figure 3.2 demonstrates some of the differences between compressions produced by our annotators), however we still consider compressions as gold standard. The nature of text generation tasks is such that often system output must be evaluated by human judges. Human evaluations consider different aspects of the automatically generated texts such as grammaticality, fluency, readability and content selection.

Although manual evaluations provide essential feedback on the quality of system output, they are costly and time consuming to run. During development, evaluations must be performed quickly and frequently and is thus impractical to elicit human judgements for development purposes. Due to this, researchers seek methods for automatically evaluating system output without any human input. Unfortunately it can be difficult to find suitable automatic evaluation measures for text generation tasks. A lot of research is devoted to finding suitable automatic evaluation measures for summarisation and machine translation. In this chapter we will explore some of the automatic

evaluation methods proposed for sentence compression and seek to find one that correlates with human judgements. First we concentrate on previous manual evaluation studies and their design. We conclude the chapter by considering how to evaluate compressed documents rather than sentences.

4.1 Manual Evaluation

Almost all previous approaches to sentence compression evaluation have focused on intrinsic[1] human judgements. Knight and Marcu (2002) provided the first intrinsic human judgement evaluation. Their experimental setup consisted of four judges being given 32 source sentences coupled with four different compressions (three system compressions and one gold standard). The judges were told that all compressions had been generated automatically and the order they were presented was randomised across judges. The evaluation was broken down into two stages, in the first stage the judges were asked to rate on a five point scale how well the systems did at selecting the most important information with respect to the source sentence. In the second experiment the judges rated how grammatical the outputs were on a five point scale. This experimental setup has been adopted in most sentence compression work (Galley and McKeown 2007; McDonald 2006; Nguyen et al. 2004b; Turner and Charniak 2005).

Our experiments will follow a modified version of Knight and Marcu's (2002) evaluation setup but allow for a greater range of significance tests to be performed. Our changes also allow us to more reliably measure the differences between system compressions. In Knight and Marcu's design each judge (or subject) is presented with $n \times k$ compressions, where n is the number of sentences and k is the number of system configurations (including the gold standard). This requires subjects to judge a large number of compressions which may become a burden on the subject. Another problem is that subjects directly judge the difference between compressions on the same sentences; such a design can lead to inaccurate judgements as the comparison of compressions is done on a per sentence basis rather than a per system basis. We modify the experimental design to use a Latin square which prevents subjects from seeing two different compressions of the same sentence. This results in subjects seeing n compressions rather than $n \times k$. Obviously with such a change more subjects are

[1] Intrinsic evaluations test the system in and of itself; for example, they determine the quality of a system's compression, whereas extrinsic evaluations test the system in relation to a task (Sparck-Jones et al. 1996).

	Treatment 1	Treatment 2	Treatment 3
Item 1	A	B	C
Item 2	C	A	B
Item 3	B	C	A
Item 4	A	B	C
Item 5	C	A	B
Item 6	B	C	A

Figure 4.1: Example Latin Square design. Different treatments are represented by columns and items as rows. Subjects are split into three sets (A, B, C) and only see one treatment of each item.

required to obtain the same number of judgements per compression. However, as we are only dealing with *n* compressions the time required to complete an evaluation is much shorter and thus it is easier to elicit volunteers via the Internet.

Our human evaluation setup is outlined as follows. Volunteers are recruited through mailing lists (typically student mailing lists) to participate in our evaluation. The experiment is conducted via the Internet using a custom made web interface. Before the subjects participate in the experiment they are presented with a set of instructions detailing sentence compression and their task with the aid of example compressions; good compressions and poor compressions[2]. They are informed that all compressions are generated automatically and asked to provide some personal details such as the country they grew up in. These details are used to ensure that our subjects are native speakers and have an adult's grasp of English. Each subject is presented with *n* source sentences and *n* compressions (one compression per sentence). They are asked to first read the source sentence and then press a button to reveal the compression and ratings interface. A Latin Square design is used and the order of the sentences is randomised. The Latin Square design ensures that subjects do not see two different treatments (i.e., compressions) of the same sentence. For example, if we have three compression systems (treatments) and six source sentences (items) each subject (participant) will see one of three possible set of compressions (see Figure 4.1, sets A, B, C). It is important to ensure we have the same number of participants for each set; this is taken care of by the evaluation interface which selects the appropriate set to show each subject.

[2]Appendix A.2 contains a typical set of instructions for our elicitation study.

Subjects are asked to rate how grammatical the compression is and how well the compression preserves the most important information from the source sentence. Both ratings are on a five point scale, with a score of one being poor and five being excellent.

4.2 Automatic Evaluation

Although human evaluations provide valuable feedback, it is not practical to conduct them repeatedly during system development. It is thus desirable to have automatic evaluation measures for gauging how different factors influence a system's or model's performance without always resorting to manual evaluation which is admittedly time consuming and expensive. There are three criteria which are desirable from an automatic evaluation measure (Lapata 2006). First, it should measure the numerical similarity or closeness of the system output with respect to one or several gold standards. Second, the measure should be robust and domain and language independent; we want to be able to use the same measure across different corpora. Finally, correlation with human judges is an important aspect. The measure should reflect the results observed in human evaluations.

Automatic evaluation of sentence compression has been less studied in the literature, although several automatic measures have been proposed; these include calculating deletion decisions on a syntactic tree, considering similarity to a gold standard or multiple gold standards and computing F-scores on grammatical structures between system output and gold standard.

Jing (2000) proposed the first automatic measure for compression: she defines a *success rate* measure that evaluates systems based on the decisions made at the syntactic level compared to those required to reach a gold standard compression. The compression process is considered as a series of decisions along the edge of a sentence's parse tree. Each node can be either kept or removed and the agreement between system and human are computed. The success rate ranges from zero to one and is defined as the ratio of the number of edges that the human and system make the same decision on to the number of edges which the human and system have made decisions upon. Thus, this measure only concentrates on the edges that both human and system perform a decision on.

For example, consider the tree in Figure 4.2. If the human keeps edge A-C and A-B but the computer drops A-C and keeps A-B, the edges in $C \rightarrow DE$ will not be considered, thus it may be possible to get an artificially high score for a poor compression by

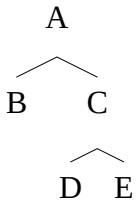

Figure 4.2: An example parse tree

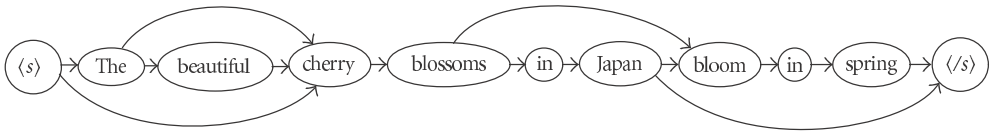

Figure 4.3: An example word network formed by multiple compressions of the same sentence.

only considering a few of the 'top' nodes.

The word network of Hori et al. (2004) combines multiple human compressions into a word lattice. The lattice contains many more sentences than the number of human compressions, thus Hori et al. compare their system against the compression from the word lattice that gives them the highest evaluation score (based on substitutions, insertions and deletions). A typical word network is shown in Figure 4.3. The use of a word lattice forces artificially high scores, because some sentences obtained within the lattice will be poor compressions. For example in Figure 4.3 the compression "Cherry blossoms in Japan" would be considered 100% correct even though it has very little information content in comparison to the other compressions. Also the evaluation procedure assumes there are multiple gold standard compressions available, but in practice these are difficult to obtain.

Simple String Accuracy (SSA, Bangalore et al. 2000) has been proposed as a baseline evaluation measure in natural language generation. It is based on a normalised string edit distance between a generated sentence and its gold standard. The measure consists of the number of insertion (I), deletion (D) and substitution (S) errors between the two strings. Equation (4.1) defines the SSA score where R is the length of the gold standard string. The measure has an upper limit of 1 (when the compression matches the gold standard), however it can fall below 0 when the number of edits required is greater than the length of the gold standard string.

$$\text{Simple String Accuracy} = (1 - \frac{I+D+S}{R}) \tag{4.1}$$

A parsing-based evaluation measure has been proposed by Riezler et al. (2003). They compare the grammatical relations found in the system compressions against those found in a gold standard. This provides a means to measure the semantic aspects of summarisation quality in terms of grammatical-functional information. The standard measures of precision, recall and F-score can then be used to quantify the quality of system output against a gold standard. Precision is the number of relations in the system compression that match the gold standard over the total number of relations in the system compression. Recall is the number of relations in the system compression that occur in the gold-standard over the total number of relations in the gold-standard. The F-score is then defined in terms of precision and recall in Equation (4.2).

$$\text{F-score} = \frac{2 * P * R}{P + R} \tag{4.2}$$

The matching of grammatical relations is shown in Figure 4.4 in which we have a gold standard compression and its relations compared against the system output and its relations. The precision and recall in this example are 0.75 and 1.0 respectively, this results in an F-score of 0.86.

We ran a judgement elicitation study in order to investigate which of the above mentioned measures correlates reliably with human judgements. This is a prerequisite for employing automatic measures in large scale evaluations. In our experiment we consider the Simple String Accuracy (SSA, Bangalore et al. 2000) and grammatical relation F-score (Riezler et al. 2003). A set of 40 sentences were taken from the Ziff-Davis and spoken corpus (split evenly, 20 sentences each) along with their corresponding gold standard compressions. We used Knight and Marcu's (2002) deterministic shift-reduce-drop decision tree system and Hori and Furui's (2004) word-based model to obtain system compressions of the 40 sentences. The models were chosen due to their previously published results on written and spoken text respectively. The compressions were presented to sixty volunteers, all self reported native English speakers. The study followed our evaluation setup outlined in Section 4.1 but differed in one aspect. Automatic evaluation scores conflate the two ratings of our human judgements (grammaticality and importance). In order to correlate human ratings with automatic scores we must also conflate both criteria into a single score. Participants were asked to rate the compressions on a five point scale taking into account the information retained by the compression and how grammatical it is. This provided a score to correlate against automatic evaluation measures.

Along with performing correlations we also wanted to examine one other aspect

Gold standard	"Mother Catherine, 82, the mother superior, will attend." (ncsubj, attend, Catherine) (ncmod, Catherine, Mother) (ncmod, superior, mother) (detmod, superior, the) (ncmod, Catherine, 82) (aux, attend, will)
System compression	"Mother Catherine, 82, the mother superior, will attend the hearing." (ncsubj, attend, Catherine) (dobj, attend, hearing) (ncmod, Catherine, Mother) (ncmod, superior, mother) (detmod, superior, the) (ncmod, Catherine, 82) (detmod, hearing, the) (aux, attend, will)

Figure 4.4: Grammatical relations obtained from RASP for gold standard and hypothetical system compression.

of the evaluation setup, the mode of presentation. The study was thus split into two conditions. In the first condition participants were presented with the source sentence while the compression was hidden. They were asked to read the source sentence before revealing the compression (via a button on the web interface). They then read the compression and gave it a score. For the second condition the compression was displayed first with the source sentence hidden. These two conditions were designed to investigate if there is any variation in judgements depending on the order the source sentence and compression are presented. Analysis of Variance (ANOVA) tests revealed that there was no significant difference in the ratings obtained when the judges were presented with the source first, then the compression and visa versa. The results of the two studies were then concatenated to perform correlation measures.

We next examined the degree to which the automatic evaluation measures correlate with human ratings using simple string accuracy (SSA Bangalore et al. 2000) and relation F-score (Riezler et al. 2003). Our results are shown in Table 4.1 using Pear-

Measure	Ziff-Davis	Spoken
SSA	0.171	0.348*
F-score	0.575**	0.532**
$*p < 0.05$ $**p < 0.01$		

Table 4.1: Correlation (Pearson's *r*) between evaluation measures and human ratings. Stars indicate level of statistical significance.

son's *r* correlation. Pearson's *r* reflects the linear degree to which the two variables (human judgement and automatic measure) are related. It ranges from +1 to -1. A correlation of +1 means there is a perfect positive linear relationship between the variables whereas -1 implies a negative linear relationship. We find that SSA does not correlate on both corpora with human judgements; it thus seems to be an unreliable measure of compression performance. However, the F-score correlates significantly with human ratings, yielding a correlation coefficient of $r = 0.575$ on the Ziff-Davis corpus and $r = 0.532$ on the spoken corpus. To get a feeling for the difficulty of the task, we assessed how well our participants agreed in their ratings using leave-one-out resampling (Weiss and Kulikowski 1991). The technique correlates the ratings of each participant with the mean ratings of all the other participants. The average agreement is $r = 0.679$ on the Ziff-Davis corpus and $r = 0.746$ on the spoken corpus. This result indicates that F-score's agreement with the human data is not far from the human upper bound.

4.3 Document-level Evaluation

The evaluation criteria we have considered thus far have focused exclusively on intrinsic evaluation; judging a compression in relation to the source sentence. This has allowed us to define evaluation procedures for sets of isolated sentences. However, many potential applications will operate at the document rather than sentence level. For example a compression system that shortens text to be displayed on devices with small screens will generate compressed documents. Although the individual sentences can be evaluated, we may also wish to evaluate the document as a whole. It is possible, for example, for the individual compressions to be good compressions but for the document to be incoherent. In Chapter 7 we present a document-based compression model which will require a document specific evaluation methodology. Our evaluation

methodology is motivated by two questions: (1) are the compressed documents readable? and (2) how much key information is preserved between the source document and its target compression? These are similar to the sentence-level questions presented in Section 4.1; however, now we are considering compressed documents.

The readability of a document is fairly straightforward to measure using a rating on a scale. Measuring how much information is preserved in the compressed document is more involved. Under the assumption that the compressed document is to function as a replacement for the source document, we can design a question-answering paradigm to find answers for questions which have been derived from the source document and are representative of its core content. Thus, the overall objective of our Q&A task is to determine how accurate each document (generated by different compression systems) is at answering questions derived from the source document. The Q&A paradigm has been used previously to evaluate summaries and text comprehension (Mani et al. 2002b; Morris et al. 1992).

Morris et al. (1992) performed one of the first question-answering evaluations to investigate the degree to which documents could be summarised before reading comprehension diminished. Their corpus consisted of four passages randomly selected from a set of sample Graduate Management Aptitude Test (GMAT) reading comprehension tests. The text covered a range of topics including: medieval literature, 18th-century Japan, minority-operated businesses and Florentine art. Accompanying each text were eight multiple-choice questions, each containing five possible answers. The questions were provided by the Educational Testing Service and were designed to measure the subject's reading comprehension. Subjects were given various textual treatments: the full text, a human authored abstract, three system generated extracts and a final treatment where merely the questions were presented without any text. The questions only treatment was used as a control to investigate if subjects could answer questions without any source material. Subjects were instructed to read the passage (if provided) and answer the multiple choice questions.

The advantage of using standardised tests, such as the GMAT reading comprehension test, is that question-answer pairs are provided along with a method for scoring answers (i.e., which answer from the multiple choice question is correct). However, our corpora do not contain ready prepared question-answer pairs thus we require a methodology for constructing questions, constructing answers and scoring documents against the answers. One such methodology is presented in the TIPSTER Text Summarization Evaluation (SUMMAC Mani et al. 2002a). SUMMAC was concerned with

What is posing a threat to the town? (lava)
What hindered attempts to stop the lava flow? (bad weather)
What are the Army attempting to block to halt the lava flow? (underground conduits)
What did the Army do first to stop the lava flow? (detonate explosives)
What other experiments are planned? (using concrete slabs)
Do the experts agree over what to do next? (no)

Figure 4.5: Example questions with answer key in brackets for document in Figure 4.6.

summarising TREC topics, and for the Q&A evaluation three topics were selected. For each topic, 30 relevant documents were chosen as source texts to generate a single summary. One annotator per topic crafted no more than five questions relating to the *obligatory aspects* of the topic. An *obligatory aspect* of a topic was defined as information that must be present in the document for the document to be relevant to the topic. The annotators then created an answer key for their topic by annotating the passages and phrases from the documents which provided the answers to the questions. In the SUMMAC evaluation, the annotator for each topic was also given the task of scoring system summaries. Systems were scored against the answer key (annotated passages from the source documents) using scoring criteria that involved judging if the summary provided a *Correct*, *Partially Correct* or *Missing* answer. If a summary contained an answer key and sufficient context the summary was deemed to be 'correct', however, summaries would be rewarded 'partially correct' if the answer key was present but with insufficient context. If context was completely missing, misleading or the answer key was absent then the summary was judged 'missing'.

Our methodology for constructing Q&A pairs and for scoring documents is inspired by the SUMMAC methodology (Mani et al. 2002a). Rather than creating questions for document sets (or topics) our questions are derived from individual documents. Two annotators were independently instructed to read the documents from our written corpus (test set, 31 documents) and create Q&A pairs. Each annotator drafted no more than ten questions and answers per document, related to its content. Annotators were asked to create factual-based questions which required an unambiguous answer; these were typically who, what, where, when, how style questions. The purpose of using two annotators per document was to allow annotators to compare and revise their question-answer pairs; this process was repeated until a common agreed

Snow, high winds and bitter disagreement yesterday further hampered attempts to tame Mount Etna, which is threatening to overrun the Sicilian town of Zafferana with millions of tons of volcanic lava.

The wall of molten lava has come to a virtual halt 150 yards from the first home in the town, but officials said yesterday that its flow appeared to have picked up speed further up the slope. A crust appears to have formed over the volcanic rubble, but red-hot lava began creeping over it yesterday and into a private orchard. Bad weather dashed hopes of attempts to halt the flow during what was seen as a natural lull in the lava's momentum.

Some experts say that even if the eruption stopped today, the sheer pressure of lava piled up behind for six miles would bring debris cascading down on to the town anyway. Some estimate the volcano is pouring out one million tons of debris a day, at a rate of 15ft per second, from a fissure that opened in mid-December.

The Italian army yesterday detonated nearly 400lb of dynamite 3,500 feet up Mount Etna's slopes. The explosives, which were described as nothing more than an experiment, were detonated just above a dam built in January and breached last week. They succeeded in closing off the third of five underground conduits formed beneath the surface crust and through which red-hot magma has been flowing. But the teams later discovered that the conduit was dry, suggesting that the lava had already found a new course.

Rumours have been circulating that experts are bitterly divided over what to do. But in another experiment 50 two-ton concrete slabs are to be chained together and dumped from a huge tilting steel platform about 6,750ft above sea level. It is hoped the slabs will block the conduit from which the main force of the lava is said to be bearing down "like a train", causing it to break up and cool. High winds and snowfalls have, however, grounded at a lower level the powerful US Navy Sea Stallion helicopters used to transport the slabs.

Prof Letterio Villari, a noted vulcanologist, said yesterday he had "absolutely no faith whatsoever" in the plan. If Zafferana was saved from the lava, which could flow for a year or more, it would be "a complete fluke", he said.

Figure 4.6: Sample document from the written test set.

upon set of questions was reached. Revisions typically involved merging and simplifying questions to make them clearer, and in some cases splitting a question into multiple questions. Rewording is another important revision. The questions must not contain too much information about the content of the document, and were revised until concise. Documents for which too few questions were agreed upon and for which the questions and answers were deemed too ambiguous by the annotators were removed. From a test set of 31 documents this left an evaluation set of six documents with between five to eight concise questions per document[3]. Figure 4.5 shows the questions and answers our annotators created for the document in Figure 4.6.

For scoring our documents we adopt a more objective method than asking the annotator who constructed the questions to check the document compressions for the answers. We recruit naive human subjects to answer the questions using the compressed documents alone. The compressed document and questions are presented to participants who are asked to answer the questions as best they can. At no point during the evaluation is the source document shown to the subject; thus if the compression is difficult to read, the participant has no point of reference to help them understand the compression. This is a departure from previous evaluations within text generation tasks, where the source text is available at judgement time; in our case only the system output is available.

We now present the details of our evaluation setup. The evaluation is conducted remotely over the Internet using a custom built web interface. Participants are recruited through student mailing lists and the Language Experiments website[4]. Upon loading the web interface, participants are presented with a set of instructions that explain the Q&A task and provide examples[5]. Subjects are first asked to read the compressed document and then rate its readability on a seven point scale where seven is excellent, and one is terrible. Once a rating has been obtained questions are presented one at a time (the order of which is defined by the annotators) and participants are asked to consult the document for the answer. Answers are written directly into a textfield on the web interface which allows for free form text to be submitted. Once a participant provides an answer and confirms the answer, the interface locks the answer to ensure it is not modified later. This is necessary because later questions may reveal information which could help answer previous questions. A Latin square design is used to prevent

[3] Appendix B contains the full documents and question answer pairs.

[4] `http://www.language-experiments.org`

[5] Full instructions and examples can be found in Appendix A.3.

participants from seeing multiple treatments (compressions) of the same document thus removing any learning effect.

The answers provided by the participants are scored against an answer key. Each answer is marked with a score of one for a correct answer and zero for a incorrect answer. In cases where two answers are required a score of 0.5 is awarded for each correct answer. The score for a compressed document is the average of its individual question scores.

4.4 Summary of Chapter

In this chapter we have presented a variety of methods for automatically and manually evaluating sentence compression systems. We have outlined some of the problems of current sentence compression elicitation studies and presented a more rigorous paradigm for evaluating isolated sentences. Furthermore, we reviewed earlier proposed measures for automatic evaluation and assessed whether these are appropriate for the compression task. Our results show that grammatical relation-based F-score (Riezler et al. 2003) correlates reliably with human judgements and thus can be used to measure compression performance automatically. This is especially useful during system development for assessing quickly and effectively how different system configurations impact compression performance. Another advantage of a reliable automatic measure is that much larger tests set can be used than the 32 sentences used in previous studies allowing for significance tests to be performed.

We have also presented a method for evaluating document compressions through a question-answering paradigm. This includes a methodology for creating question-answer pairs, presenting document compressions and question-answer pairs to subjects, and scoring the subjects' answers. We have managed to create question-answer pairs for six documents from our written compression corpus each containing between five to eight concise questions and answers. The Q&A evaluation study allows us to determine how well our compression systems preserve the most important information from the source documents and whether the resulting compressed document is understandable by naive human judges.

Chapter 5

Integer Linear Programming

Before we present our compression models (Chapter 6), we will provide a brief introduction to the integer linear programming framework which we will adopt in later chapters.

Mathematical Programming encompasses a set of tools for solving optimisation problems. This chapter concentrates on two types of mathematical programming frameworks: Linear Programming and Integer Linear Programming. Many practical optimisation problems in operations research can be expressed as linear programming problems; consequently considerable research has been devoted to the efficient solving of linear programs. An example of a business application would be maximising profit in a factory that manufactures a number of different products from the same raw material using the same resources (in fact we will use such an example in the next section to describe the concepts of linear programming). Integer linear programming is an extension to linear programming which allows us to model a wider range of real world problems. For example, the Travelling Salesman Problem (TSP) can be formulated as an integer linear programming problem.

We begin the chapter by introducing the terminology of linear and integer linear programming, along with the algorithms required to solve these problems. We then demonstrate why these frameworks, in particular integer linear programming, are beneficial for NLP.

5.1 Linear Programming

Linear programming (LP) problems are optimisation problems with constraints. They consist of three parts:

- A linear function (the *objective function*). This is the function we wish to minimise or maximise. This can be a linear combination of many such functions.

- Decision variables. These are variables under our control which influence the result of the objective function. These are the variables we must optimise to maximise (or minimise) the objective function.

- Constraints. The ability to include constraints is one of the main strengths of the LP framework. Most problems will only allow the decision variables to take certain values. These restrictions are modelled by the constraints.

These terms are best demonstrated with a simple example taken from Winston and Venkataramanan (2003).

Telfa Example Imagine the Telfa Corporation manufactures tables and chairs. To produce a table 1 hour of labour and 9 square board feet of wood is required. Chairs require 1 hour of labour and 5 square board feet of wood. Telfa have 6 hours of labour and 45 square board feet of wood available. The profit made from each table is 8 GBP and 5 GBP for chairs. Determine the number of tables and chairs that should be manufactured to maximise Telfa's profit.

First we must determine the *decision variables*. These must represent the decisions that need to be made. In our case we define:

$$\begin{aligned} x_1 &= \text{number of tables manufactured} \\ x_2 &= \text{number of chairs manufactured} \end{aligned}$$

Our *objective function* is the value we wish to maximise or minimise – the profit.

$$\text{Profit} = 8x_1 + 5x_2$$

The coefficient of a variable in the objective function is referred to as the *object function coefficient* of the variable.

There are two constraints in this problem: we must not exceed 6 hours of labour and no more than 45 square board feet of wood must be used. Also we can not create a negative amount of chairs or tables.

$$\begin{array}{lrcrcl}
\text{Labour constraint} & x_1 & + & x_2 & \leq & 6 \\
\text{Wood constraint} & 9x_1 & + & 5x_2 & \leq & 45 \\
\text{Variable constraints} & & & x_1 & \geq & 0 \\
& & & x_2 & \geq & 0
\end{array}$$

Once the decision variables, objective function and constraints have been determined we can express the LP model:

$$\max z = 8x_1 + 5x_2 \quad \text{(Objective function)}$$

subject to (s.t.)

$$\begin{array}{rcrcll}
x_1 & + & x_2 & \leq & 6 & \text{(Labour constraint)} \\
9x_1 & + & 5x_2 & \leq & 45 & \text{(Wood constraint)} \\
& & x_1 & \geq & 0 & \\
& & x_2 & \geq & 0 &
\end{array}$$

5.1.1 Solving LP models

Two of the most basic concepts involved in solving LP problems are the *feasibility region* and *optimal solution*. The optimal solution is one in which all the constraints of the LP problem are satisfied and the objective function is minimised or maximised. A specification of the value for each decision variable is referred to as a *point*. The feasibility region for a LP is a region consisting of the set of all points that satisfy all the constraints of the LP. The optimal solution lies within this feasibility region, it is the point with the minimum or maximum objective function value.

A set of points satisfying a single linear inequality (in our case a constraint) is a *half-space*. The feasibility region is defined by a the intersection of m (for m linear inequalities) half-spaces and forms a *polyhedron*. Our Telfa example forms a polyhedron set (a polyhedral convex set) from the intersection of our four constraints. Figure 5.1a shows the feasible region (the polyhedron enclosed by points A, B, C, D) for the Telfa example.

To find the optimal solution we graph a line (or hyperplane) on which all points have the same objective function value. In maximisation problems it is called the *isoprofit line* and in minimisation problems the *isocost line*. One isoprofit line is represented by the dashed black line in Figure 5.1a. Once we have one isoprofit line we can find all other isoprofit lines by moving parallel to the original isoprofit line.

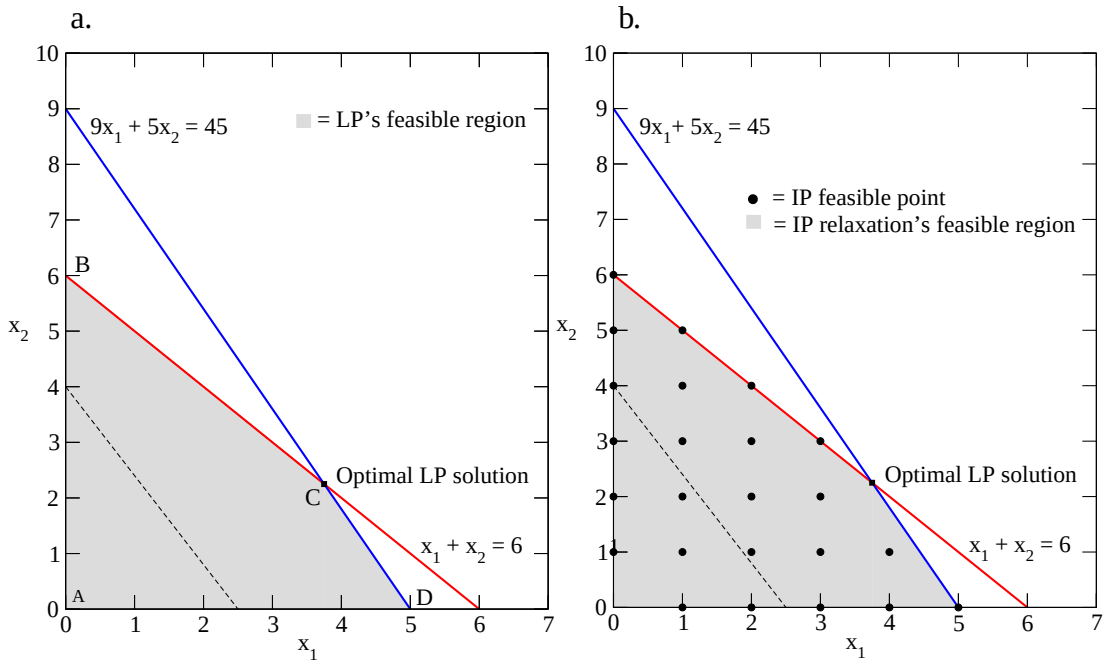

Figure 5.1: Feasible region for the Telfa example

The *extreme points* of the polyhedral set are defined as the intersections of the lines that form the boundaries of the polyhedral set (points A, B, C and D in Figure 5.1a). It can be shown that any LP that has an optimal solution, has an extreme point that is globally optimal. Another important property of LPs is that there are only a finite number of extreme points, which is proportional to the number of variables and constraints. These two properties reduce the search space of the optimisation problem to finding the extreme point with the highest profit or lowest cost.

Algorithms such as the simplex method (Dantzig 1963) are used to find the optimal solutions of LPs. The simplex method starts by computing an initial extreme point and tests its optimality. If some optimality condition is verified the algorithm terminates. Otherwise the simplex method identifies *adjacent extreme points* (extreme points that lie on the same line segment) with a better objective function value. Optimality of this new solution is tested again, and the entire method is repeated until an optimal extreme point is found. As there are only a finite number of extreme points for a given LP, it follows that the simplex method will terminate in a *finite* number of iterations.

Many LP solvers (both commercial and free) rely on a the simplex algorithm to solve large scale linear programs. This is despite its poor worst-case behaviour. It is possible to construct a LP for which the simplex algorithm will take an exponential (in the problem size) number of steps to solve. However, in practice the algorithm is very

efficient and is found in many solvers providing solutions as efficiently as worst-case polynomial-time algorithms (e.g., interior points methods Vanderbei 2001).

The optimal solution for the Telfa example is $z = \frac{165}{4}$, $x_1 = \frac{15}{4}$, $x_2 = \frac{9}{4}$. Thus to achieve a maximum profit of 41.25 GBP, they must build 3.75 tables and 2.25 chairs. This is obviously impossible as we would not expect people to buy fractions of tables and chairs. We therefore want to be able to constrain the problem such that the decision variables can only take integer values. This can be done with Integer Linear Programming, described in the following section.

5.2 Integer Linear Programming

Integer linear programming (ILP) problems are LP problems in which some or all of the variables are required to be non-negative integers. They are formulated in a similar manner to LP problems but have the additional constraint that all the decision variables must take non-negative integer values. Many different types of real world problems such as *scheduling problems* and the *travelling salesman problem* can be modelled as ILPs.

Let us now return to the Telfa problem which also requires an integer solution. To formulate it as an ILP model we merely add the constraints that x_1 and x_2 must be integer. This gives:

$$\max z = 8x_1 + 5x_2 \quad \text{(Objective function)}$$

subject to (s.t.)

$$\begin{aligned} x_1 + x_2 &\leq 6 && \text{(Labour constraint)} \\ 9x_1 + 5x_2 &\leq 45 && \text{(Wood constraint)} \\ x_1 &\geq 0; x_1 \text{ integer} \\ x_2 &\geq 0; x_2 \text{ integer} \end{aligned}$$

In the LP model it can be proved that the optimal solution lies on an extreme point of the feasible region. This gave us two real numbers as the optimal solution to the Telfa problem. When we define this problem as an ILP we only wish to consider the points that are integer values. These points are shown in Figure 5.1b as dots.

5.2.1 Solving ILP problems

One might think that solving ILP problem would not be much harder than solving linear programming problems. Unfortunately this is not the case: solving ILP problems is NP-hard (Cormen et al. 2000).

A number of techniques have been developed to find a global optimal solution to an ILP problem. Two such techniques are the cutting planes method (Gomory 1960) and the branch-and-bound method (Land and Doig 1960). Both of these are guaranteed to find a global optimal solution. The cutting planes method adds extra constraints to slice parts of the feasible region until it contains only extreme points that are integer points — however reducing the feasible region until it contains only extreme integer points can be a difficult or impossible process (Nemhauser and Wolsey 1988).

The branch-and-bound method involves solving a (potentially) large number of (related) linear programming problems to find the optimal integer solution. This involves relaxing the constraints that variables must be integral and solving the resulting linear program in the hope that the solution contains integer solutions. The linear programming problem obtained from relaxing these constraints is called the *LP relaxation*. If all the variables assume integer values then the solution is also optimal for the ILP; however, if this is not the case, the resulting solution provides an upper bound on the optimal solution for the ILP. Typically the solution to the LP relaxation contains non-integer variables. One naive strategy of obtaining an integer solution would be to round the variables to their nearest integer value. Unfortunately, this strategy is a poor choice as the integer solutions it yields might not even be feasible. The branch-and-bound strategy (Land and Doig 1960) is a cleverer approach. It uses the non-integer solutions obtained from the relaxation to divide the ILP into several LP sub-problems. This creates an enumeration tree in which the original relaxation is the root node and the first sub-problems are child nodes. Sub-problems are created based on the non-integer solution for one variable at a time.

For example, the LP relaxation to the Telfa problem returns a solution of $\frac{15}{4}$ for variable x_1, in this case two sub-problems are created, one with the constraint that $x_1 \leq 3$ and the other with the constraint $x_1 \geq 4$. Figure 5.2 shows the enumeration tree resulting from dividing the LP relaxation into two sub-problems.

These sub-problems are solved and the process is repeated until:

- the LP sub-problem returns integer solutions for all variables. The first integral solution found becomes the candidate solution and provides a lower bound for

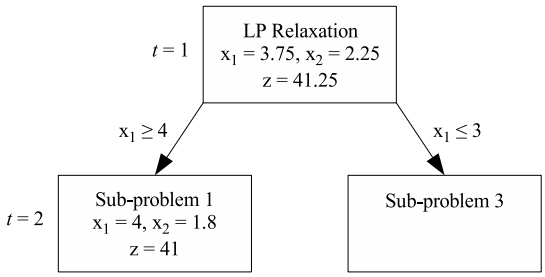

Figure 5.2: Telfa's First Enumeration Tree; t represents the solving iteration.

the ILP. If a subsequent integer solution has a higher objective value than the lower bound, it becomes the new candidate solution (and the lower bound is modified accordingly).

- the LP sub-problem is infeasible; or
- the objective function value is less than the currently optimal candidate (the lower bound).

The search method for the enumeration tree is typically depth-first as this allows us to find feasible solutions to the ILP early, these give a lower bound to the problem and can be used to prune nodes from the tree. Figure 5.3 shows the final enumeration tree for the Telfa problem. The t value represents the iteration for which each sub-problem was solved. Using the branch-and-bound method we find the first candidate solution on iteration five, which is then replaced by the solution found on the sixth iteration. The sixth iteration solution is optimal; however, its optimality cannot be proved until the seventh iteration; after which all sub-problems in the enumeration tree have been exhausted. The final solution to the Telfa problem is $z = 40$, $x_1 = 5$, $x_2 = 0$; thus to achieve a maximum profit of 40 GBP, Telfa must manufacture 5 tables and 0 chairs.

For the full details of the cutting planes and branch-and-bound methods see Winston and Venkataramanan (2003), Vanderbei (2001), or Nemhauser and Wolsey (1988).

Special Cases In general, the branch-and-bound methods have proved to be the most successful in solving practical ILP problems. However, there are special cases of problems which have more efficient solving strategies or specialised structure that simplifies solving.

0–1 ILP problems are a special case of ILP problem in which all variables are binary. Such problems can be solved using a simplified branch-and-bound technique

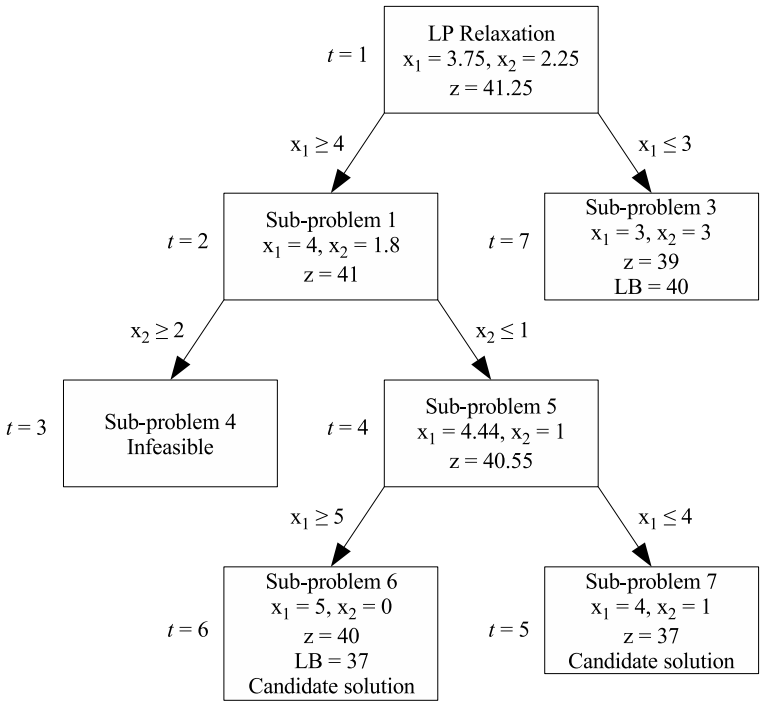

Figure 5.3: Telfa's Final Enumeration Tree; t represents the solving iteration, LB represents lower bound value.

called implicit enumeration. This simplifies both the branching and bounding components and can efficiently identify when a node in the enumeration tree is infeasible by exploiting the binary nature of the variables. Variables in implicit enumeration can either be *fixed* variables where their value is specified, or *free* variables whose value is unspecified. For any node in the enumeration tree, a specification of the values of the free variables is known as a *completion* of the node. For example, if a problem consists of three variables x_1, x_2, x_3 and we are at a node in the enumeration tree where x_1 is fixed at 0, then one completion of the node is $x_2 = 1$, $x_3 = 1$, another completion is $x_2 = 0$, $x_3 = 1$. The different completions are the various combinations the free variables can form. The previous branch-and-bound technique relied on relaxing the ILP to the corresponding LP without integer constraints to solve the node whereas during implicit enumeration the relaxation is not required. By exploiting the fact that the variables are binary it is possible to efficiently compute the best completion of a node and determine its feasibility.

First, the best completion of the node is found by setting the free variables to the value (0 or 1) which makes the objective function largest (in max problems) or smallest (in min problems). If this completion is feasible (no constraints are violated) then it is the best feasible completion of the node and no further branching of the node is

required. If the best completion is not feasible then the completion gives us an upper bound for the node. The bound can be used to eliminate the node from consideration (i.e., if the bound is lower than the current candidate score, in the max case).

Next we determine if all the completions of the node are infeasible. For each constraint in the problem we assign the free variables to the best value for satisfying the constraint. If the constraint is not satisfied by this most feasible completion we can deduce the node has no feasible completion. For example, if our node has the fixed variable $x_1 = 1$ and free variables x_2, x_3 and the problem has a constraint:

$$9x_1 - 1x_2 + 3x_3 \leq 3$$

By setting $x_2 = 1$, $x_3 = 0$ the left side of the constraint as small as possible. If this completion does not satisfy the constraint then no completion of the node can. In this case $9 - 1 + 0 \leq 3$ does not hold, so the node can be eliminated from consideration. In general, if even one constraint can not be satisfied by its most feasible completion, then the node has no feasible completion and can be eliminated.

At this point if the node's best completion is infeasible and there exists feasible completions for each constraint, we cannot deduce if the node has a feasible completion or is infeasible until more variables are fixed. Thus, we select a free variable x_i to branch on, creating two sub-nodes: one with $x_i = 1$ and another with $x_i = 0$. The process then repeats itself by computing the best completion of a new node and determining its feasibility.

Another special case of ILP problem are ones which have a *totally unimodular* constraint matrix. These ILP problems can be solved directly by the LP relaxation as the relaxation is guaranteed to result in an integer solution. This removes the need to perform branch-and-bound as the problem can be treated as an LP. A matrix A is totally unimodular if every square sub-matrix of A has its determinant equal to 0, $+1$ or -1. Unfortunately the definition of total unimodularity does not help us detect if a constraint matrix has the property. Evaluating the determinant of every square sub-matrix is computationally prohibitive. However, it is known that problems which can be formulated as the minimum cost network flow problem (MCNFP) have totally unimodular constraint matrix. Generally, the more the constraint matrix looks totally unimodular the easier the problem is to solve by branch-and-bound methods. Therefore, it is good practice to formulate an ILP in which as many variables as possible have coefficients of 0, $+1$ or -1 in the constraints.

5.2.2 Uses of Discrete Variables

We have shown how linear programming models can be converted to integer linear programming models in which all the variables must take integer values. The obvious use of discrete variables is when we want to represent discrete quantities (e.g., the number of chairs to build), however, for NLP this is not the most applicable use. In NLP it is more useful to be able to determine if different actions or decisions should be taken rather than determining numeric quantities of variables. For example, we may wish to know what part-of-speech tag should be assigned for a given word (each decision would be a part-of-speech assignment) or if a word should be included in a compression.

Integer variables are frequently used in ILP to represent which decision should be made. Typically these variables are constrained to take the two values, zero or one. Such variables are known as 0–1 variables. For example, in compression a variable could be used to represent if a certain word should be in the compression where a value of 0 would represent the word being dropped from the compression whereas a value of 1 would indicate the word is included in the compression. Although decision variables are usually 0–1 variables they need not be, for example the variable could be constrained to take a value from zero, one or two.

Indicator variables are another use for discrete variables. These are used in cases where extra conditions must be imposed on a model. To accomplish this it maybe necessary to introduce additional 0–1 variables which are linked to other variables to indicate certain states. For example, it would be possible to introduce a single 0–1 variable to represent a combination of two words being included in the compression.

Having introduced these 0–1 variables it is now possible to represent logical connections between different decisions through linear constraints involving the variables. Many different types of logical condition can be imposed using constraints. Table 5.1 lists some useful conditions which can be modelled.

Another useful condition to express is transitivity, i.e., ‘z if and only if x and y’. It is often thought that such a logical condition can only be expressed as a polynomial expression of 0–1 variables.

$$xy = z$$

However it is possible to replace this polynomial expression with the linear constraints (Williams 1999):

Condition	Statement	Constraint
Implication	if x then y	$y-x \geq 0$
Iff	x if and only if y	$x-y=0$
OR	x or y or z	$x+y+z \geq 1$
XOR	x xor y xor z	$x+y+z=1$
AND	x and y	$x=1; y=1$
NOT	not x	$1-x=1$

Table 5.1: How to represent various logical conditions using 0–1 variables and constraints in ILP. x, y, z are 0–1 variables.

$$(1-z)+x \geq 1$$
$$(1-z)+y \geq 1$$
$$z+(1-x)+(1-y) \geq 1$$

This can be extended easily to model an indicator variable which represents if a set of 0–1 variables take certain values.

The ability to incorporate logical conditions through constraints will allow us to instill more linguistic and semantic knowledge into our compression models. This will be shown in Chapter 6.

5.2.3 Constraint Programming

Constraint Programming (CP) and Integer Linear Programming share many similarities, in particular both allow the modelling of constraints over a set of variables. In CP each variable has a finite domain of possible values, this could be integers, real numbers or even sets of values. Constraints are used to restrict the possible combinations of values the variables can take much like in ILP. However, the constraint language of CP is much richer than the linear constraints in ILP; constraints need not be expressed only as linear inequalities. A greater variety of constraint operators and relations are available (Williams and Wilson 1998), such as $=$, $\geq$, $\leq$, $>$, $<$, $*$, $\setminus$, *subset*, *superset*, *union*, *intersection*, *member*, *all_different*, *OR*, *AND* and *XOR*. It is possible to model these CP constraints in ILP by transforming them, with varying degrees of difficultly, to linear constraints involving integer variables. However, CP provides a greater expressive power as it allows for these constraints directly. This representation can make

problems easier to model and sometimes easier to solve due to the more concise representation, that is problems can be expressed with fewer constraints and variables than in ILP.

Constraint Programming is often used when a quick feasible solution to a problem is required rather than a provably optimal solution. A feasible solution is found using *constraint propagation* whereby the values of variables (or domains of variables) is reduced using information from the constraints. For example (taken from Williams and Wilson (1998)), let x, y, z be integer variables defined in the range $[1,10]$ and let the constraints be $y < z$ and $x = y + z$. We can deduce that $y < 10$ and $z > 1$ from the constraint $y < z$. Combining this with $x = y + z$ we can deduce that $x > 2$, $y < 9$ and $z < 10$. Thus the domains of the variables can be revised to $x \in [3,10]$, $y \in [1,8]$, $z \in [2,9]$.

It is also possible to specify an objective function. When an objective function is introduced solutions are usually found via a tree search similar to the branch-and-bound used in ILP. This depends on having a good bound on the objective function (Lustig and Puget 2001). For example, assume that our objective is to minimise the function $g(x_1, x_2, \ldots, x_n)$ and we know the lower bound, L. Before optimisation begins, we must first find a feasible solution (ignoring the objective function) which determines the upper bound U of the objective function. These two bounds provide us with a range the optimal objective function must fall within. Using a binary search on the objective function we can find the optimal value. The procedure computes the midpoint $M = (U + L)/2$ of the bounds and then solves a CP by taking the original problem (without objective function) and adding the constraint $g(x_1, x_2, \ldots, x_3) < M$. If a feasible solution is found then the upper bound is updated and the search continues with a new midpoint. If the problem is infeasible then the lower bound is updated and search continues with a new midpoint. The main difference between this procedure and the brand-and-bound procedure of ILP is that in CP we stress the search for feasible solutions, whereas branch-and-bound procedures usually emphasise improving the lower bound (Lustig and Puget 2001).

Although CP may appear more desirable than ILP due to its expressive power and the ability to incorporate an objective function, it is not without its problems. When constraints contain many variables, constraint propagation becomes ineffective; this stems from the history of the CP community who mainly consider problems in which the constraints only contain two variables each (Hooker 2002). Objective functions tend to contain many variables as they represent the cost or reward incurred by making

different decisions. Thus CP is less desirable for modelling problems which involve large objective functions.

5.3 Integer Linear Programming in NLP

Our presentation of ILP has been very general with little reference to NLP. In this section we describe how ILP has been used in NLP and the benefits it can bring.

Integer Linear Programming has recently attracted much attention in the NLP community. ILP techniques have been applied to several tasks, including machine translation (Germann et al. 2004), relation extraction (Roth and Yih 2004), semantic role labelling (Punyakanok et al. 2004), the generation of route directions (Marciniak and Strube 2005), temporal link analysis (Bramsen et al. 2006), set partitioning (Barzilay and Lapata 2006), syntactic parsing (Riedel and Clarke 2006), multi-document summarisation (McDonald 2007) and coreference resolution (Denis and Baldridge 2007).

Most of these approaches use ILP to model problems in a more global manner. Capturing the global properties of a problem can improve a model's accuracy as it is able to represent the long-range dependencies of the problem. In this section we give an overview of previous NLP work using ILP and describe the tasks to which ILP has been applied.

It is important to clarify what we mean by *global* since it has multiple senses depending on its context. When referring to optimisation, a *global* method seeks to find the globally best solution of the model in the presence of multiple local optima. By contrast, global in the sense of *global* models means being able to perform decisions based on evidence beyond the local scope (i.e., beyond adjacent words, part-of-speech, constituents). In this section we will use the term *exact inference* to refer to a technique which finds the globally best solution under the model (rather than global optimisation). And global will be reserved to the modelling sense, therefore *global inference* refers to being able to incorporate more global information during inference.

Recall that many tasks in natural language processing can be framed as mapping from inputs $x \in \mathcal{X}$ to outputs $y \in \mathcal{Y}$. For example, in the case of sentence compression $x \in \mathcal{X}$ is a source sentence and $y \in \mathcal{Y}$ its corresponding compression.

5.3.1 Global Constraints with ILP

In the sequence labelling setting (e.g., part-of-speech tagging and chunking) there are existing techniques for training local models under the global structure of the problem which provide exact and tractable inference. Two popular techniques are hidden Markov Models (HMMs) and conditional random fields (CRFs, Lafferty et al. 2001). These models benefit from being trained under the global structure of the problem, by this we mean the sequential (or structural) constraints of the tasks are enforced. For example, each word must be assigned exactly one part-of-speech tag or chunk sequences cannot overlap. However, HMMs and CRFs force us to make strong assumptions of conditional independence between variables. This is due to the widely used Viterbi algorithm which provides efficient and exact inference. The assumptions are not always justifiable as many real world problems exhibit complex structure and long range dependencies which we are unable to capture with local models. In both HMMs and CRFs low-order Markov assumptions (typically first-order) are made on the output structure, thus limiting us to consider the output sequence locally, i.e., that the current part-of-speech tag for a word only depends on the previous tag; longer histories can be difficult to encode. In many tasks and domains there are hard or soft global constraints on the output sequence that are easily motivated by common sense typically through linguistic understanding or domain knowledge. One example of this may be, that a sentence must contain a verb. It is, therefore, desirable to have models which can look more globally at the output structure.

Reranking Reranking is one method that has been used to incorporate global constraints on the output (Collins 2000; Shen et al. 2004). Its motivation is to provide a method of incorporating constraints which would otherwise be awkward to encode in existing models.

In order to perform reranking, a base model, which is typically local (e.g., HMM, CRF or perceptron), produces a set of candidate solutions for each input. Accompanying these solutions are their respective probabilities or model scores which define a natural ranking on the solutions. This ranked list of solutions is termed the n-best list. A second model (the reranker) then attempts to improve on this initial ranking using more global features or constraints over the output space. The task of the reranker is to pick the best global solution from the n-best list.

The advantage of reranking is that it provides a quick method for incorporating global constraints on the output and is relatively easy to implement. It has been shown

to improve parsing accuracy (Collins 2000) and machine translation quality (Shen et al. 2004). However it is not without its drawbacks. Reranking is sensitive to the size of the *n*-best list. The performance of the reranker is limited by the base model. If the *n*-best list does not contain good solutions then the reranker will not be able to find any better solutions. This is very task dependent, in parsing 27 candidate parses needed to be considered on average to see a notable improvement in accuracy whereas in machine translation at least 1000 candidates must be considered to see an increase in performance. Generally, the size of n grows with the size of the output space $\mathcal{Y}$.

Another problem is that finding a solution using reranking replaces a single process with a two stage process. In the first stage, the base model must generate the n candidate solutions thus pruning the search space (or solution space); the second stage then selects the best candidate according to the reranking model. Thus this method has multiple approximations; it is possible that good solutions according to the reranker will be pruned in the first stage. Reranking relies on the base model being able to produce *n*-best solutions which contain globally good answers. If this isn't the case then the reranker will not be able to find better solutions.

Integer Linear Programming Another method of incorporating global constraints is by reformulating the inference procedure as an ILP. This is favourable to the reranking method as global constraints can be incorporated into the model in a single inference process rather than as part of a two stage process. ILP allows us to introduce new local and global constraints that act over the output space and model the long-range dependencies of the problem in a natural and systematic fashion.

In the context of CRFs, Roth and Yih (2005) reformulate the Viterbi algorithm as an ILP which allows them to extend CRF models to support general constraint structures (i.e., more global and non-sequential constraints). Specifically inference in CRFs (and HMMs) can be viewed as a minimum cost network flow problem (MCNFP) and can be easily formulated as an ILP. This formulation also results in a totally unimodular constraint matrix and thus can be solved by the LP relaxation without resorting to specific ILP solving methods (see Section 5.2.1 for details).

Roth and Yih (2005) test their ILP inference in CRFs using the Semantic Role Labelling (SRL) task which attempts to discover the verb-argument structure of a sentence. A *semantic role* is the relationship that a syntactic constituent has with a predicate. Example of roles arguments include Agent, Patient, Instrument, Locative, Temporal, Manner and Cause. Sentence (1) shows the semantic roles annotation for

Label	Description	Label	Description
V	Verb	**A3**	Attribute
A0	Acceptor	**AM-MOD**	Modal
A1	Thing accepted	**AM-NEG**	Negation
A2	Accepted-from		

Table 5.2: Semantic Role Labels for the verb *accept* as defined by the PropBank Frame Scheme.

the predicate *accept,* taken from the PropBank corpus.

(1) [$_{A0}$ He] [$_{AM\text{-}MOD}$ would] [$_{AM\text{-}NEG}$ n't] [$_{V}$ **accept**] [$_{A1}$ anything of value] *from* [$_{A2}$ those he was writing about].

The labels in the case of *accept* are defined in the PropBank Frame scheme according to Table 5.2.

When the task is framed as a sequence labelling problem we can view identifying the segments (arguments) as attempting to label consecutive words as being part of a segment or not. This can be represented at the word level using a `BIO` representation. In this case, we label each word as either beginning (`B`) a text segment, being inside (`I`) a text segment but not the first word of the segment, or being outside (`O`) a text segment. The `O` label is assigned all words we are not interested in. In the case of semantic role labelling where we must not only segment the sentence but also label the segments we can append the label to the `BIO` representation. For example, to signify an `A1` argument we would have the labels `B-A1` and `I-A1`. When adjacent text segments cannot share the same label we can simplify the representation to the `IO` representation.

Using this representation, Roth and Yih (2005) reformulate the Viterbi algorithm used in CRFs as an ILP to incorporate linguistically motivated constraints. Examples of these constraints include: no duplicate verb arguments (i.e., a verb cannot have two `A1` arguments), all verbs must have at least one argument, and disallowing invalid arguments for a verb among others. The constraints can be used both during training and testing, however, Roth and Yih observe the best performance when they apply the additional constraint at test time after training without the constraints.

Riedel and Clarke (2006) augment a dependency parsing model with linguistically motivated constraints to create a *more* global model by reformulating the inference process as an ILP. Dependency parsing is the task of attaching words to their arguments,

for example verbs are attached to their objects and subjects. The output of a dependency parser is a dependency tree in which each word depends on exactly one parent or the dummy root symbol. Inference in dependency parsing models is made tractable by assuming a first-order factorisation (Eisner 1996) in which attachment decisions are made independently of one another. McDonald et al. (2005c) show that using a first-order factorisation inference can be performed globally and exactly by solving the maximum spanning tree. The resulting tree is guaranteed to be a structurally valid dependency tree and the best solutions with respect to the model. The factorisation, however, often causes the output to contain linguistic errors as attachment decisions are made independently of one another. For example, a verb could be attached to more than one subject or nouns and verbs may be coordinated. Such properties cannot be modelled with a first-order factorisation.

Riedel and Clarke (2006) reformulate the maximum spanning tree problem as an ILP and introduce additional linguistic constraints which act over the output. The resulting model uses these constraints to disallow dependency trees which do not conform to certain linguistic constraints. For example one constraint states: "heads are not allowed to have more than one outgoing edge labelled l for all l in the set of labels". This constraint covers situations such as ensuring that verbs have no more than one subject. The additional constraints ensure the resulting dependency trees are linguistically correct and contain fewer errors.

5.3.2 ILP for Arbitrary Problem Structure

In some problems the task is not easily mapped into sequence labelling or other settings with well understood and efficient decoding algorithms such as Viterbi or the maximum spanning tree. However, ILP is capable handling a variety of problems provided the objective and constraints can be expressed as linear functions.

Rather than tackling the whole problem of mapping $\mathcal{X}$ into $\mathcal{Y}$, many approaches break the mapping into a series of isolated decisions. This makes the task amenable to a variety of "simple" learning algorithms. Local classifiers, such as the perceptron (Rosenblatt 1988) and support vector machines (Vapnik 1998), can be used to predict each "simple" decision without knowledge of the broader task. However, these local classifiers have no knowledge of the global structure of the problem and thus their collective output can be inconsistent and contradictory. Using ILP, it is possible to perform global inference over the classifier's output, thus ensuring that the output's

structure is consistent and correct.

We will demonstrate this using the aggregation task from natural language generation (Barzilay and Lapata 2006). In sentence generation a content planner provides a set of entities which must appear in the generated document. A aggregation component takes these entities as input and clusters (or aggregates) them such that each cluster corresponds to a sentence. Entities from the same cluster will be mentioned in same generated sentence. The aggregation is formulated as a set partitioning task where the goal is to find a cluster of the input entities that maximises a global utility function. This can be viewed as mapping from a set of entities $x \in \mathcal{X}$ to a partitioning of non-empty subsets $y \in \mathcal{Y}$ such that each entity appears in exactly one subset. They model this task as a series of local decisions in which a local classifier predicts whether two entities should be aggregated based on their similarity. Efficient exact algorithms do not exist for solving set partitioning problems (Cormen et al. 2000), which are NP-complete and thus typically solved approximately without imposing structure on the output. Solving this problem greedily (without imposing any structure) using solely a local model can lead to an inconsistent partitioning. For example, transitivity may not hold; entity *A* could be aggregated with entity *B* which in turn is aggregated with entity *C*, however entities *A* and *C* are not aggregated. Barzilay and Lapata (2006) alleviate these problems by using ILP to perform inference over the local classifier model with a set of structural constraints which ensure that the global partitioning is consistent (i.e., transitivity holds). This provides exact and global inference over the output which was not possible in the greedy case.

5.3.3 Combining Multiple Classifiers with ILP

Traditionally, NLP applications are implemented using a cascade of classifiers in which the output representation is built incrementally and the output of one classifier serves as input to the next. This process is repeated until the output representation is reached. For example a simple pipeline for relation extraction might involve performing named entity recognition and then using the result as input to a relation extraction module. A major problem with the pipeline approach is that classifiers must blindly trust the output of earlier classifiers even when there may be evidence to the contrary — this is especially true if the tasks being performed are strongly correlated with one another.

Consider for instance sentence (2):

(2) John Doe has worked for many airports throughout his life, he is currently

employed by JFK.

Here, a hypothetical named entity recogniser may label *John Doe* and *JFK* as *person*. This labelling is then passed on to a relation extraction module which from the sentence alone has strong evidence that a *work for* relation exists between *John Doe* and *JFK*. However, despite this evidence, it must label the entity pair differently as a *work for* relation takes *person* and *location* as its arguments and not two *persons*. This problem is common in pipeline architectures; they suffer hugely from error propagation as later classifiers have no means of informing earlier classifiers of possible errors or inconsistencies.

Integer Linear Programming has been used to combine local classifiers in a global manner thus removing the reliance on the pipeline. Such an approach has been used for generating route directions (Marciniak and Strube 2005) and relation extraction (Roth and Yih 2004). Roth and Yih (2004) use ILP to combine the output of a named entity identifier and relation identifier. Given a sentence, ILP provides global and exact inference over all possible classifications that could in the sentence. Taking sentence (2) as an example, the objective function would contain the sum of the scores for all possible labellings for *John Doe* and *JFK* plus the scores of all possible relations between those labellings. A set of constraints help model the structure of the problem to ensure the output is valid. This disallows labelling both entities as *person* and selecting the relation *work for*. Modelling the problem in this manner allows the relation extraction scores to help resolve the ambiguity in labelling *JFK* which will have relatively high scores for being labelled either *person* or *location*.

5.3.4 ILP for Exact Inference

While most uses of ILP in NLP focus on combining the output of multiple local classifiers to find global solutions or producing more global models, some work has used ILP solely for inference without extending the model with linguistic or other constraints. Germann et al. (2004) and McDonald (2007) have compared approximate inference algorithms against the exact solution provided by ILP for two NP-hard problems: machine translation and multi-document summarisation. In these cases the sheer scale of the problem makes inference intractable and approximate algorithms are often used to ensure speed and tractability. ILP is a prime candidate for these kinds of problems since it is suited to solving NP-hard problems exactly, although solve time is sacrificed to reach optimality.

Germann et al. (2004) compare the speed and output quality of three decoders for statistical machine translation (SMT) using the IBM translation Model 4 (Brown et al. 1993). SMT is framed as the Travelling Salesman Problem (TSP) where choosing a good word order for the output is likened to determining a good TSP tour. They find that the ILP becomes intractable due to the sub-tour elimination strategy employed which requires an exponential number of constraints be added to avoid subtours.

The multi-document summarisation problem is framed as selecting sentences from multiple related documents to form a general summary. The desirable properties for a summary are that: it is relevant for its purpose, it contains no redundant sentences and that its length is bounded by some upper limit. McDonald (2007) treats the task as optimising all these properties jointly, and proves that global inference is NP-hard. The output of three global inference algorithms are compared. These are a greedy approximate method, a dynamic programming approach based on solutions to the knapsack problem, and an ILP formulation of the knapsack problem. He shows that the dynamic programming approach provides near optimal results and scales much better than the exact ILP method which is feasible for smaller problems.

5.3.5 ILP in Other Scenarios

Dras (1999) develops a document paraphrasing model using ILP. The key premise of his work is that in some cases one may want to rewrite a document so as to conform to some global constraints such as length, readability, or style. The proposed model has three ingredients: a set of sentence-level paraphrases for rewriting the text, a set of global constraints, and an objective function which quantifies the effect incurred by the paraphrases. Under this formulation, ILP can be used to select which paraphrases to apply so that the global constraints are satisfied. The constraints are focused on: length, readability, lexical density (Halliday 1985) and variety in sentence structure. Paraphrase generation falls outside the scope of the ILP model – sentence rewrite operations are mainly syntactic and provided by a module based on synchronous tree adjoining grammar (S-TAG). Unfortunately, only a proof-of-concept is presented; implementation and evaluation of this module are left to future work.

5.4 Summary of Chapter

In this chapter we have presented linear programming (LP) and integer linear programming (ILP). LP and ILP are flexible frameworks for modelling various optimisation problems. ILP problems consist of three parts: a linear objective function, a set of decision variables, and a set of linear constraints. The constraints are flexible, easy to formulate and can represent a variety of real world properties.

We provided an overview of the application of ILP to various NLP problems. This outlined some of the properties which make ILP an appealing framework, in particular how constraints can be used to enforce global structure on problems.

Chapter 6

ILP for Compression

Sentence compression has been expressed in a variety of formulations using either lexical information, syntactic information or both. Despite differences in formulation, all these approaches model the compression process using *local* information. For instance, in order to decide which words to drop, they exploit information about adjacent words or constituents. Local models can do a good job at producing grammatical compressions, however they are somewhat limited in scope since they cannot incorporate *global* constraints on the compression output. Such constraints consider the sentence as a whole instead of isolated linguistic units (words or constituents). To give a concrete example we may want to ensure that each target compression has a verb, provided that the source had one in the first place. Or that verbal arguments are present in the compression. Or that pronouns are retained. Such constraints are fairly intuitive and can be used to instill not only linguistic but also task specific information into the model. For example, an application which compresses text to be displayed on small screens would presumably have a higher compression rate than a system generating subtitles from spoken text. A global constraint could force the former system to generate compressions with a fixed rate or a fixed number of words.

Existing approaches do not model global properties of the compression problem for a good reason. The decoding process for finding the best compression for a source sentence given the space of all possible compressions can become intractable for too many constraints and overly long sentences. In cases where the decoding problem cannot be solved efficiently using dynamic programming, an approximate search is used. For example, in the noisy-channel approach of Turner and Charniak (2005), the decoder first searches for the best combination of rules to apply. As it traverses the list of compression rules, it removes sentences outside the 100 best compressions

(according to the channel model). This list is eventually truncated to 25 compressions.

In this chapter we propose a novel framework for sentence compression that takes into account local and global constraints and finds an optimal solution. Our formulation uses many of the integer linear programming (ILP) techniques discussed in Chapter 5. Specifically, we show how previously proposed models can be recast as integer linear programs. We extend these models with constraints which we express as linear inequalities. Decoding amounts to finding the best solution given a linear (scoring) function and a set of linear constraints that can be either global or local. Our constraints are syntactically and semantically motivated and designed to bring less local knowledge into the model and help preserve the meaning of the source sentence. Previous work has identified several important features for the compression task (Knight and Marcu 2002; McDonald 2006); however, the use of constraints during the search process is novel to our knowledge.

Although ILP has been used in previous work (see Chapter 5), its application to generation is not widespread. Barzilay and Lapata (2006) use ILP for aggregation, a subtask within generation. Our work however tackles the whole generation pipeline of sentence compression including content selection and surface realisation. Our ILP systems are end-to-end systems in which the input is an uncompressed sentence and the output is a compressed sentence. Contrary to most previous work (Roth and Yih (2005) are an exception) we do joint inference and learning within the ILP framework including learning in the presence of our constraints.

We present three compression models from Chapter 2 recast in the ILP framework. These models are representative of an unsupervised, semi-supervised and fully supervised learning approach. This allows us to perform a comparison across learning paradigms and assess the impact of our constraints on these models.

Finally, we introduce a series of constraints designed to ensure the compressions are structurally and semantically valid. The first set of constraints are concerned with relations between modifier and head words. We then look at sentence wide constraints such as verb argument structure. Our final set of constraints concern discourse information and are explored in more detail in the next chapter.

6.1 Compression Models

In this section we recap three compression models from Chapter 2 which we reformulate as integer linear programs; and present our compression-related global constraints

in Section 6.2. All the constraints are derived from the logical condition identities presented in Section 5.2.2; these allow us to build simple logical conditions into our models through 0–1 variables.

The first model is a simple language model which has been used as a baseline in previous research (Knight and Marcu 2002). Our second model is based on Hori and Furui (2004); it combines a language model with a corpus-based significance scoring function (we omit here the confidence score derived from the speech recogniser since our models are applied to text only). This model requires a small amount of parallel data to learn weights for the language model and the significance score.

Our third model uses a discriminative large margin framework (McDonald 2006), is fully supervised and trained on a larger parallel corpus.

6.1.1 Language Model

A language model is perhaps the simplest model that springs to mind. It does not require a parallel corpus (although a relatively large monolingual corpus is necessary for training), and will naturally prefer short sentences to longer ones. Furthermore, a language model can be used to drop words that are either infrequent or unseen in the training corpus. Knight and Marcu (2002) use a bigram language model as a baseline against their noisy-channel and decision-tree models.

Let $\mathbf{x} = x_1, x_2, \ldots, x_n$ denote a source sentence for which we wish to generate a target compression. We introduce a decision variable for each word in the source and constrain it to be binary; a value of 0 represents a word being dropped, whereas a value of 1 includes the word in the target compression. Let:

$$\delta_i = \begin{cases} 1 & \text{if } x_i \text{ is in the compression} \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in [1 \ldots n]$$

If we were using a unigram language model, our objective function would maximise the overall sum of the decision variables (i.e., words) multiplied by their unigram probabilities (all probabilities throughout this chapter are log-transformed):

$$\max \sum_{i=1}^{n} \delta_i \cdot P(x_i) \tag{6.1}$$

Thus, if a word is selected, its corresponding δ_i is given a value of 1, and its probability $P(x_i)$ according to the language model will be counted in our total score.

A unigram language model will probably generate many ungrammatical compressions. We therefore use a more context-aware model in our objective function, namely

a trigram model. Dynamic programming would be typically used to decode a language model by traversing the sentence in a left-to-right manner. Such an algorithm is efficient and provides all the context required for a conventional language model. However, it can be difficult or impossible to incorporate constraints into such a model as decisions on word inclusion cannot extend beyond a three word window. By formulating the decoding process for a trigram language model as an integer linear program we are able to take into account constraints that affect the compressed sentence more globally. This process is a much more involved task than in the unigram case where there is no context, instead we must now make decisions based on word sequences rather than isolated words. We first create some additional decision variables:

$$\alpha_i = \begin{cases} 1 & \text{if } x_i \text{ starts the compression} \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in [1 \dots n]$$

$$\beta_{ij} = \begin{cases} 1 & \text{if sequence } x_i, x_j \text{ ends} \\ & \text{the compression} \quad \forall i \in [0 \dots n-1] \\ 0 & \text{otherwise} \quad \forall j \in [i+1 \dots n] \end{cases}$$

$$\gamma_{ijk} = \begin{cases} 1 & \text{if sequence } x_i, x_j, x_k \quad \forall i \in [0 \dots n-2] \\ & \text{is in the compression} \quad \forall j \in [i+1 \dots n-1] \\ 0 & \text{otherwise} \quad \forall k \in [j+1 \dots n] \end{cases}$$

Our objective function is given in Equation (6.2). This is the sum of all possible trigrams that can occur in all compressions of the source sentence where x_0 represents the 'start' token and x_i is the ith word in sentence $\mathbf{x}$. Equation (6.3) constrains the decision variables to be binary.

$$\begin{aligned} \max z \quad = \quad & \sum_{i=1}^{n} \alpha_i \cdot P(x_i|\text{start}) \\ & + \sum_{i=1}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} \gamma_{ijk} \cdot P(x_k|x_i, x_j) \\ & + \sum_{i=0}^{n-1} \sum_{j=i+1}^{n} \beta_{ij} \cdot P(\text{end}|x_i, x_j) \end{aligned} \tag{6.2}$$

subject to:

$$\delta_i, \alpha_i, \beta_{ij}, \gamma_{ijk} = 0 \text{ or } 1 \tag{6.3}$$

The objective function in (6.2) allows any combination of trigrams to be selected. This means that invalid trigram sequences (e.g., two or more trigrams containing the

'end' token) could appear in the target compression. We avoid this situation by introducing *sequential constraints* (on the decision variables δ_i, γ_{ijk}, α_i, and β_{ij}) that restrict the set of allowable trigram combinations.

Constraint 1 Exactly one word can begin a sentence.

$$\sum_{i=1}^{n} \alpha_i = 1 \tag{6.4}$$

Constraint 2 If a word is included in the sentence it must either start the sentence or be preceded by two other words or one other word and the 'start' token x_0.

$$\delta_k - \alpha_k - \sum_{i=0}^{k-2} \sum_{j=1}^{k-1} \gamma_{ijk} = 0 \qquad \forall k : k \in [1 \ldots n] \tag{6.5}$$

Constraint 3 If a word is included in the sentence it must either be preceded by one word and followed by another or it must be preceded by one word and end the sentence.

$$\delta_j - \sum_{i=0}^{j-1} \sum_{k=j+1}^{n} \gamma_{ijk} - \sum_{i=0}^{j-1} \beta_{ij} = 0 \qquad \forall j : j \in [1 \ldots n] \tag{6.6}$$

Constraint 4 If a word is in the sentence it must be followed by two words or followed by one word and then the end of the sentence or it must be preceded by one word and end the sentence.

$$\delta_i - \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} \gamma_{ijk} - \sum_{j=i+1}^{n} \beta_{ij} - \sum_{h=0}^{i-1} \beta_{hi} = 0 \qquad \forall i : i \in [1 \ldots n] \tag{6.7}$$

Constraint 5 Exactly one word pair can end the sentence.

$$\sum_{i=0}^{n-1} \sum_{j=i+1}^{n} \beta_{ij} = 1 \tag{6.8}$$

The sequential constraints described above ensure that the second order factorisation (for trigrams) holds and are different from our compression-specific constraints which are presented in Section 6.2.

Unless normalised by sentence length, a language model will naturally prefer one-word output. This normalisation is however non-linear and cannot be incorporated into

our ILP formulation. Instead, we impose a constraint on the length of the compressed sentence. Equation (6.9) below forces the compression to contain at least b tokens.

$$\sum_{i=1}^{n} \delta_i \geq b \tag{6.9}$$

Alternatively, we could force the compression to be exactly b tokens (by substituting the inequality with an equality in (6.9)) or to be less than b tokens (by replacing $\geq$ with $\leq$).[1] The constraint in (6.9) is language model-specific and is not used elsewhere.

6.1.2 Significance Model

The language model just described has no notion of which content words to include in the compression and thus prefers words it has seen before. But words or constituents will be of different relative importance in different documents or even sentences.

Inspired by Hori and Furui (2004), we add to our objective function (see Equation (6.2)) a significance score designed to highlight important content words. In Hori and Furui's original formulation each word is weighted by a score similar to unnormalised $tf * idf$. The significance score is not applied indiscriminately to all words in a sentence but solely to topic-related words, namely nouns and verbs. Our score differs in one respect. It combines document-level with sentence-level significance. So in addition to $tf * idf$, each word is weighted by its level of embedding in the syntactic tree.

Intuitively, in a sentence with multiply nested clauses, more deeply embedded clauses tend to carry more semantic content. This is illustrated in Figure 6.1 which depicts the clause embedding for the sentence "*Mr Field has said he will resign if he is not reselected, a move which could divide the party nationally*". Here, the most important information is conveyed by clauses S_3 (*he will resign*) and S_4 (*if he is not reselected*) which are embedded. Accordingly, we should give more weight to words found in these clauses than in the main clause (S_1 in Figure 6.1). A simple way to enforce this is to give clauses weight proportional to the level of embedding. Our modified significance score becomes:

$$I(x_i) = \frac{l}{N} \cdot f_i \log \frac{F_a}{F_i} \tag{6.10}$$

where x_i is a topic word, f_i and F_i are the frequency of x_i in the document and corpus respectively, F_a is the sum of all topic words in the corpus, l is the number of clause

[1] Compression rate can be also limited to a range by including two inequality constraints.

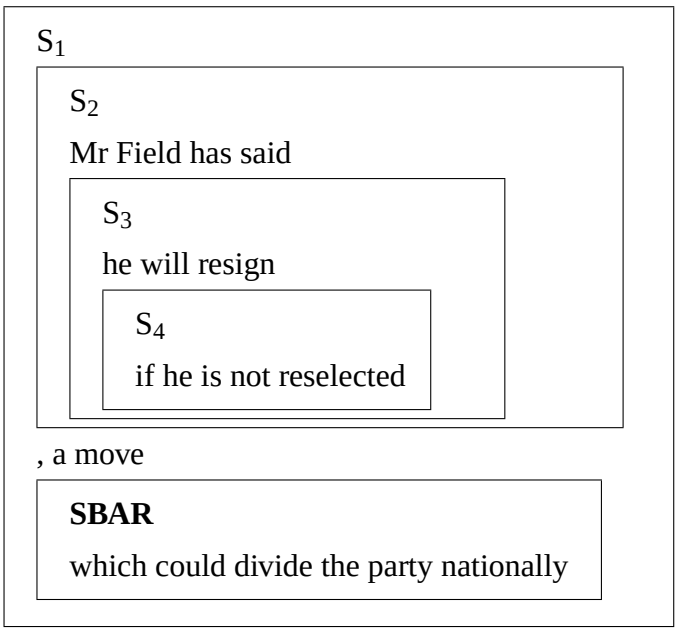

Figure 6.1: The clause embedding of the sentence "*Mr Field has said he will resign if he is not reselected, a move which could divide the party nationally*"; nested boxes correspond to nested clauses.

constituents above x_i, and N is the deepest level of clause embedding. F_a and F_i are estimated from a large document collection, f_i is document-specific, whereas $\frac{l}{N}$ is sentence-specific. So, in Figure 6.1 the term $\frac{l}{N}$ is 1.0 (4/4) for clause S_4, 0.75 (3/4) for clause S_3, and so on. Individual words inherit their weight from their clauses.

The modified objective function with the significance score is given below:

$$\begin{aligned} \max z \;=\; & \sum_{i=1}^{n} \delta_i \cdot \lambda I(x_i) + \sum_{i=1}^{n} \alpha_i \cdot P(x_i|\text{start}) \\ & + \sum_{i=1}^{n-2} \sum_{j=i+1}^{n-1} \sum_{k=j+1}^{n} \gamma_{ijk} \cdot P(x_k|x_i,x_j) \\ & + \sum_{i=0}^{n-1} \sum_{j=i+1}^{n} \beta_{ij} \cdot P(\text{end}|x_i,x_j) \end{aligned} \tag{6.11}$$

We also add a weighting factor (λ) to the objective, in order to counterbalance the importance of the language model and the significance score. The weight is tuned on a small parallel corpus. The sequential constraints from Equations (6.4)–(6.8) are again used to ensure that the trigrams are combined in a valid way.

6.1.3 Discriminative Model

For our discriminative model we use the model presented by McDonald (2006). This model uses a large-margin learning framework coupled with a feature set defined on compression bigrams and syntactic structure.

We briefly recap the model. Full details of the model including the features and learning algorithm are discussed in Section 2.2.3. Let $\mathbf{x} = x_1, \ldots, x_n$ denote a source sentence with a target compression $\mathbf{y} = y_1, \ldots, y_m$ where each y_j occurs in $\mathbf{x}$. The function $L(y_i) \in \{1 \ldots n\}$ maps word y_i in the target compression to the index of the word in the source sentence, $\mathbf{x}$. We also include the constraint that $L(y_i) < L(y_{i+1})$ which forces each word in $\mathbf{x}$ to occur at most once in the compression $\mathbf{y}$. Let the score of a compression $\mathbf{y}$ for a sentence $\mathbf{x}$ be:

$$s(\mathbf{x}, \mathbf{y}) \tag{6.12}$$

This score is factored using a first-order Markov assumption on the words in the target compression to give:

$$s(\mathbf{x}, \mathbf{y}) = \sum_{j=2}^{|\mathbf{y}|} s(\mathbf{x}, L(y_{j-1}), L(y_j)) \tag{6.13}$$

The score function is defined to be the dot product between a high dimensional feature representation and a corresponding weight vector:

$$s(\mathbf{x}, \mathbf{y}) = \sum_{j=2}^{|\mathbf{y}|} \mathbf{w} \cdot \mathbf{f}(\mathbf{x}, L(y_{j-1}), L(y_j)) \tag{6.14}$$

Decoding in this model amounts to finding the combination of bigrams that maximises the scoring function in (6.14). McDonald (2006) uses a dynamic programming approach where the maximum score is found in a left-to-right manner. The algorithm is an extension of Viterbi for the case in which scores factor over dynamic sub-strings (McDonald et al. 2005a; Sarawagi and Cohen 2004). This allows back-pointers to be used to reconstruct the highest scoring compression as well as the k-best compressions.

Again this is similar to the trigram language model decoding process (see Section 6.1.1), except that here a bigram model is used. Consequently, the ILP formulation is slightly simpler than that of the trigram language model. Let:

$$\delta_i = \begin{cases} 1 & \text{if } x_i \text{ is in the compression} \\ 0 & \text{otherwise} \end{cases} \quad (1 \leq i \leq n)$$

We then introduce some more decision variables:

$$\alpha_i = \begin{cases} 1 & \text{if } x_i \text{ starts the compression} \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in [1 \dots n]$$

$$\beta_i = \begin{cases} 1 & \text{if word } x_i \text{ ends the compression} \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in [1 \dots n]$$

$$\gamma_{ij} = \begin{cases} 1 & \text{if sequence } x_i, x_j \text{ is in the compression} \quad \forall i \in [1 \dots n-1] \\ 0 & \text{otherwise} \quad \forall j \in [i+1 \dots n] \end{cases}$$

The discriminative model can be now expressed as:

$$\begin{aligned} \max z \quad = \quad & \sum_{i=1}^{n} \alpha_i \cdot s(\mathbf{x}, 0, i) \\ & + \sum_{i=1}^{n-1} \sum_{j=i+1}^{n} \gamma_{ij} \cdot s(\mathbf{x}, i, j) \\ & + \sum_{i=1}^{n} \beta_i \cdot s(\mathbf{x}, i, n+1) \end{aligned} \tag{6.15}$$

Constraint 1 Exactly one word can begin a sentence.

$$\sum_{i=1}^{n} \alpha_i = 1 \tag{6.16}$$

Constraint 2 If a word is included in the sentence it must either start the compression or follow another word.

$$\delta_j - \alpha_j - \sum_{i=1}^{j} \gamma_{ij} = 0 \quad \forall j : j \in [1 \dots n] \tag{6.17}$$

Constraint 3 If a word is included in the sentence it must be either followed by another word or end the sentence.

$$\delta_i - \sum_{j=i+1}^{n} \gamma_{ij} - \beta_i = 0 \quad \forall i : i \in [1 \dots n] \tag{6.18}$$

Constraint 4 Exactly one word can end a sentence.

$$\sum_{i=1}^{n} \beta_i = 1 \tag{6.19}$$

Again, the sequential constraints in Equations (6.16)–(6.19) are necessary to ensure that the resulting combination of bigrams are valid.

The current formulation provides a single optimal compression given the model. However, McDonald's (2006) dynamic programming algorithm is capable of returning the *k*-best compressions; this is useful for their learning algorithm described later. In order to produce *k*-best compressions, we must rerun the ILP with extra constraints which forbid previous solutions. In other words, we first formulate the ILP as above, solve it, add its solution to the *k*-best list, and then create a set of constraints that forbid the configuration of δ_i decision variables which form the current solution. The procedure is repeated until *k* compressions are found.

6.2 Constraints

We are now ready to describe our constraints. The models presented in the previous sections contain only sequential constraints and are thus equivalent to their original formulations. Our constraints are linguistically and semantically motivated in a similar fashion to the grammar checking component of Jing (2000). However, they do not rely on any additional knowledge sources (such as a grammar lexicon or WordNet) beyond the parse and grammatical relations of the source sentence. While grammatical relations are a general concept, for our purposes we obtain them from RASP (Briscoe and Carroll 2002), a domain-independent, robust parsing system for English. However, any other parser with a broadly similar output (e.g., Lin (2001)) could also serve our purposes. Our constraints revolve around modification, argument structure, and discourse related factors.

In presenting our constraints it is useful to have real world examples to understand how each constraint affects compression output. Table 6.1 provides example sentences and their corresponding compressions given by the trigram language model, it also shows the improvements made over the trigram model as we add different styles of constraints (described below).

We can see from Table 6.1 that the language model (rows marked b) does a reasonable job of modelling local word dependencies, but it is unable to capture syntactic

1a. He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.
1b. He became a player in the Pasok.
1c. He became a player in the Pasok Party.
1d. He became a player in politics.
2a. She was in a Canadian hospital last night suffering form exhaustion.
2b. She was a night.
2c. She was suffering from exhaustion.
2d. She was in a hospital suffering.
3a. We took these troubled youth who don't have fathers, and brought them into the room to Dads who don't have their children.
3b. We don't have, and don't have children.
3c. We don't have them don't have their children.
3d. We took these youth and brought them into the room to Dads.

Table 6.1: Compression examples (a: source sentence, b: compression with the trigram model, c: compression with LM and modifier constraints, d: compression with LM, Mod and argument structure constraints).

dependencies that could potentially allow for more meaningful compressions. For example, in sentence (1b) it is unable to capture the object-verb dependency between *Pasok Party* and *founded.*

Modifier Constraints Modifier constraints ensure that relationships between head words and their modifiers remain grammatical in the compression:

$$\delta_i - \delta_j \geq 0 \qquad (6.20)$$

$$\forall i, j : x_j \in x_i\text{'s } \texttt{ncmods}$$

$$\delta_i - \delta_j \geq 0 \qquad (6.21)$$

$$\forall i, j : x_j \in x_i\text{'s } \texttt{detmods}$$

Equation (6.20) guarantees that if we include a non-clausal modifier[2] (`ncmod`) in the compression (such as an adjective or a noun) then the head of the modifier must also be included; this is repeated for determiners (`detmod`) in (6.21). In Table 6.2 we illustrate

[2]Clausal modifiers (`cmod`) are adjuncts modifying entire clauses. In the example "*he ate the cake because he was hungry*", the *because*-clause is a modifier of the sentence "*he ate the cake*".

4a.	He became a power player in Greek Politics in 1974, when he founded the socialist Pasok Party.
4b.	*He became a power player in Greek Politics in 1974, when he founded **the Pasok**.
5a.	We took these troubled youth who don't have fathers, and brought them into a room to Dads who don't have their children.
5b.	*We took these troubled youth who **do have** fathers, and brought them into a room to Dads who **do have** their children.
5c.	*We took these troubled youth who don't have fathers, and brought them into a room to Dads who don't **have children**.
6a.	The chain stretched from Uganda to Grenada and Nicaragua, since the 1970s.
6b.	***Stretched from Uganda** to Grenada and Nicaragua, since the 1970s.
6c.	***The chain from Uganda** to Grenada and Nicaragua, since the 1970s.
6d.	*The chain **stretched Uganda** to Grenada and Nicaragua, since the 1970s.
6e.	*The chain **stretched from** to Grenada and Nicaragua, since the 1970s.
6f.	*The chain stretched from **Uganda to Grenada Nicaragua**, since the 1970s.

Table 6.2: Examples of compressions disallowed by our set of constraints.

how these constraints disallow the deletion of certain words (starred sentences denote compressions that would not be possible given our constraints). For example, if the modifier word *Pasok* from sentence (4a) is in the compression, then its head *Party* will also included (see (4b) as a counter example).

We also want to ensure that the meaning of the source sentence is preserved in the compression, particularly in the face of negation. Equation (6.22) implements this by forcing *not* in the compression when the head is included (see sentence (5b) in Table 6.2). A similar constraint is added for possessive modifiers (e.g., *his*, *our*), including genitives (e.g., *John's gift*), as shown in Equation (6.23). An example of the possessive constraint is given in sentence (5c) in Table 6.2.

$$\delta_i - \delta_j = 0 \qquad (6.22)$$

$$\forall i, j : x_j \in x_i\text{'s } \texttt{ncmods} \wedge x_j = \text{not}$$

$$\delta_i - \delta_j = 0 \qquad (6.23)$$

$$\forall i, j : x_j \in x_i\text{'s possessive } \texttt{mods}$$

Compression examples with the addition of the modifier constraints are shown in Table 6.1 (see rows labelled c). Moving from the language model (rows labelled b) to the compressions with modifier constraints also demonstrates the interaction of the constraints during inference. We see that sentence (1c) and (3c) change little from the language model but are certainly improvements. Unlike post-processing or reranking, which could be used to 'fix' the output, the constraints play a role in inference and thus the optimal compression according to the model and constraints will always be found. Thus we see the differences between sentence (2c) and (2b) are much greater.

Although the compressions created with the use of modifier constraints are grammatical (see the inclusion of *Party* due to the modifier *Pasok*), they are not entirely meaning preserving.

Argument Structure Constraints We also define a few intuitive constraints that take the overall sentence structure into account. The first constraint (Equation (6.24)) ensures that if a verb is present in the compression then so are its arguments, and if any of the arguments are included in the compression then the verb must also be included. We thus force the program to make the same decision on the verb, its subject, and object (see sentence (6b) in Table 6.2).

$$\delta_i - \delta_j = 0 \tag{6.24}$$
$$\forall i, j : x_j \in \text{subject/object of verb } x_i$$

Our second constraint forces the compression to contain at least one verb provided the source sentence contains one as well:

$$\sum_{i:x_i \in verbs} \delta_i \geq 1 \tag{6.25}$$

The constraint entails that it is not possible to drop the main verb *stretched* from sentence (6a) (see also sentence (6c) in Table 6.2).

Other sentential constraints include Equations (6.26) and (6.27) which apply to prepositional phrases and subordinate clauses. These constraints force the introducing term (i.e., the preposition, or subordinator) to be included in the compression if any word from within the syntactic constituent is also included. By subordinator we mean *wh*-words (e.g., *who, which, how, where*), the word *that*, and subordinating conjunctions (e.g., *after, although, because*). The reverse is also true, i.e., if the introducing term is included, at least one other word from the syntactic constituent should also be

included.

$$\delta_i - \delta_j \geq 0 \quad (6.26)$$
$$\forall i, j : x_j \in \texttt{PP/SUB}$$
$$\wedge x_i \text{ starts } \texttt{PP/SUB}$$

$$\sum_{i:x_i \in \texttt{PP/SUB}} \delta_i - \delta_j \geq 0 \quad (6.27)$$
$$\forall j : x_j \text{ starts } \texttt{PP/SUB}$$

As an example consider sentence (6d) from Table 6.2. Here, we cannot drop the preposition *from* if *Uganda* is in the compression. Conversely, we must include *from* if *Uganda* is in the compression (see sentence (6e)).

We also wish to handle coordination. If two head words are conjoined in the source sentence, then if they are included in the compression the coordinating conjunction must also be included:

$$(1 - \delta_i) + \delta_j \geq 1 \quad (6.28)$$
$$(1 - \delta_i) + \delta_k \geq 1 \quad (6.29)$$
$$\delta_i + (1 - \delta_j) + (1 - \delta_k) \geq 1 \quad (6.30)$$
$$\forall i, j, k : x_j \wedge x_k \text{ conjoined by } x_i$$

Consider sentence (6f) from Table 6.2. If both *Uganda* and *Nicaragua* are present in the compression, then we must include the conjunction *and.*

Table 6.1 illustrate the compression output when sentential constraints are added to the model (see rows labelled with d). In sentence (1d) we see that *politics* is forced into the compression due to the presence of *in.* Sentences (2d) and (3d) change quite considerably from those with only the modifier constraints (sentence (2c) and (3c)).

Finally, Equation (6.31) disallows anything within brackets in the source sentence from being included in the compression. This is a somewhat superficial attempt at excluding parenthetical and potentially unimportant material from the compression.

$$\delta_i = 0 \quad (6.31)$$
$$\forall i : x_i \in \text{ bracketed words (inc parentheses)}$$

Discourse Constraints Discourse constraints will be fully investigated in Chapter 7, however, for the time being we include a naive approximation in our model.

Our discourse constraint concerns personal pronouns. Specifically, Equation (6.32) forces personal pronouns to be included in the compression. The constraint is admittedly more important for generating coherent documents (as opposed to individual sentences). It nevertheless has some impact on sentence-level compressions, in particular when verbal arguments are missed by the parser. When these are pronominal, constraint (6.32) will result in more grammatical output since some of the argument structure of the source sentence will be preserved in the compression.

$$\delta_i = 1 \qquad (6.32)$$
$$\forall i : x_i \in \text{ personal pronouns}$$

We should note that some of the constraints described above would be captured by models that learn synchronous deletion rules from a corpus. For example, the noisy-channel model of Knight and Marcu (2002) learns not to drop the head when the latter is modified by an adjective or a noun, since the transformations $DT\,NN \rightarrow DT$ or $AJD\,NN \rightarrow ADJ$ are almost never seen in the data. Similarly, the coordination constraint (Equations (6.28)–(6.30)) would be enforced using Turner and Charniak's (2005) special rules — they enhance their parallel grammar with rules modeling more structurally complicated deletions than those attested in their corpus. In designing our constraints we aimed at capturing appropriate deletions for many possible models, including those that do not rely on a training corpus or do not have an explicit notion of a parallel grammar (e.g., McDonald 2006). The modification constraints would presumably be redundant for the noisy-channel model, which could otherwise benefit from more specialised constraints, e.g., targeting sparse rules or noisy parse trees; however we leave this to future work.

Another feature of the modelling framework presented here is that deletions (or non-deletions) are treated as unconditional decisions. For example, we require not to drop the noun in adjective-noun sequences if the adjective is not deleted as well. We also require to always include a verb in the compression if the source sentence has one. These hardwired decisions could in some cases prevent valid compressions from being considered. For instance, it is not possible to compress the sentence "*this is not appropriate behaviour*" to "*this is not appropriate*" or"*Bob loves Mary and John loves Susan*" to "*Bob loves Mary and John Susan*". Admittedly we lose some expressive power, yet we ensure that the compressions will be broadly grammatical, even for unsupervised or semi-supervised models. Furthermore, in practice we find that our models consistently outperform non-constraint-based alternatives, without extensive

constraint engineering.

6.3 Experimental Setup

Our evaluation experiments were motivated by three questions: (1) Do compression models with constraints deliver performance gains? We expect better compressions for the model variants which incorporate constraints. (2) Are there differences among constraint-based models? Here, we would like to investigate how much compression quality is improved with the additional modelling power gained through constraints. For example, it may be the case that a state-of-the-art model like McDonald's (2006) does not benefit much from the addition of constraints. And that the effect of these constraints is much bigger for a less sophisticated model. (3) How do the models port across domains? In particular, we are interested in assessing whether the models and proposed constraints are general and robust enough to produce good compressions for both written and spoken texts.

We next describe the data sets on which our models were trained and tested, explain how model parameters were estimated, discuss the solve times of our ILPs and present our evaluation setup. We discuss our results in Section 6.4.

Corpora Our intent was to assess the performance of the models just described on written and spoken text. The appeal of written text is understandable since most summarisation work today focuses on this domain. Speech data not only provides a natural test-bed for compression applications (e.g., subtitle generation) but also poses additional challenges. Spoken utterances can be ungrammatical, incomplete, and often contain artefacts such as false starts, interjections, hesitations, and disfluencies. Rather than focusing on spontaneous speech which is abundant in these artefacts, we conduct our study on the less ambitious domain of broadcast news transcripts. This lies in-between the extremes of written text and spontaneous speech as it has been scripted beforehand and is usually read off on autocue.

We use the two manually compressed corpora introduced in Section 3.1; a written text corpus and a spoken text corpus. The written text corpus comprises of 82 newspaper articles (1,433 sentences) from the British National Corpus (BNC) and the American News Text corpus. The corpus is split into training, development and testing sets randomly on article boundaries. The sets contain 908, 63 and 462 sentences respectively. The spoken text corpus consists of 50 broadcast news stories (1,370 sentences)

taken from the HUB-4 1996 English Broadcast News corpus provided by the LDC. Again the corpus is divided into 882 training sentences, 78 development sentences and 410 testing sentences; each set contains full stories.

Parameter Estimation McDonald's (2006) model was trained on the full training set on both corpora. The training is required to learn the feature weights, **w**. Our implementation uses the same features as McDonald (2006) (see Section 2.2.3 for details). The only difference is that our phrase structure and dependency features are extracted from the output of Roark's (2001) parser. McDonald uses Charniak's (2000) parser which performs comparably.

A loss function is required to inform the learning algorithm on the quality of a compression hypothesis. McDonald's (2006) loss is a measure of the number of words falsely retained or dropped from the compression (i.e., the number of false positives and false negatives). During development we observed that this loss function did not compress aggressively enough on our corpora (typically around 85% compression rate). To alleviate this we introduced a new loss function:

$$L(x,y) = f_p + f_n + \lambda LP \tag{6.33}$$

where f_p is the number of words falsely retained in the compression, f_n is the number of words falsely dropped from the compression and LP is the length penalty as defined below in Equation (6.34). Here r is the length of the gold standard compression and c is the length of the candidate compression. The λ parameter controls how strongly a candidate compression is penalised for exceeding the length of the gold standard compression. Using a line search on the development data λ was set to 3.

$$LP = \begin{cases} c - r & \text{if } c > r \\ 0 & \text{otherwise} \end{cases} \tag{6.34}$$

Recall that two of our models require a trigram language model (see Sections 6.1.1 and 6.1.2). This was estimated from 25 million tokens of the North American corpus using the CMU-Cambridge Language Modeling Toolkit (Clarkson and Rosenfeld 1997) with a vocabulary size of 50,000 tokens and Good-Turing discounting. The significance score was calculated using 25 million tokens from the American News Text corpus. In one of our models this score is combined with a language model (see Equation (6.11)) and both terms are weighted. We optimised the weights using a small subset of the training data (three documents in each case). The optimisation followed

Powell's method (Press et al. 1992) with a loss function based on the F-score of the grammatical relations found in the original sentence and its compressed version (see Chapter 4 for details).

Solving the ILP As we mentioned in the previous chapter (Chapter 5) solving ILPs is NP-hard. In cases where the coefficient matrix is unimodular, it can be shown that the optimal solution to the linear program is integral. Although the coefficient matrix in our problems is not unimodular, we obtained integral solutions for all sentences we experimented with (approximately 3,000, see Section 6.3 for details). We conjecture that this is due to the fact that all of our variables have 0, $+1$ or -1 coefficients in the constraints and therefore our constraint matrix shares many properties of a unimodular matrix. We generate and solve an ILP for every sentence we wish to compress. Solve times are less than a second per sentence (including input-output overheads) for all models presented here.

Evaluation Method Previous studies rely almost exclusively on human judgements for assessing the assessing the well-formedness of automatically derived compressions. We followed the evaluation procedure outlined in Chapter 4 by evaluating the output of our models in two ways. First, we present results using an automatic evaluation measure comparing the relations found in the system compression against those found in the gold standard (Riezler et al. 2003). This allows us to measure the semantic aspects of summarisation quality in terms of grammatical-functional information and can be quantified using F-score. Since our test corpora are fairly large (over 400 sentences in each corpus) differences among systems can be highlighted using significance testing.

Our implementation of the F-score measure uses the grammatical relations annotations provided by RASP (Briscoe and Carroll 2002). This parser is particularly appropriate for the compression task since it provides parses for both full sentences and sentence fragments and is generally robust enough to analyse semi-grammatical compressions. We calculated F-score over all the relations provided by RASP (e.g., subject, direct/indirect object, modifier; 15 in total).

In line with previous work we also evaluate our models by eliciting human judgements. In the first experiment participants were presented with a source sentence and its target compression and asked to rate how well the compression preserved the most important information from the source sentence. In the second experiment, they were asked to rate the grammaticality of the compressed outputs. In both cases they used

a five point rating scale where a high number indicates better performance. We randomly selected 21 sentences from the test portion of each corpus. These sentences were compressed automatically by the three models presented in this paper with and without constraints. We also included gold standard compressions. Our materials thus consisted of 294 ($21 \times 2 \times 7$) source-target sentences. A Latin square design ensured that subjects did not see two different compressions of the same sentence. We collected ratings from 42 unpaid volunteers, all self reported native English speakers. Both studies were conducted over the Internet. Examples of our experimental items are given in Table 6.3.

6.4 Results

Let us first discuss our results when compression output is evaluated in terms of F-score. Tables 6.4 and 6.5 illustrate the performance of our models on the written and spoken corpora, respectively. We also present the compression rate for each system. In all cases the models with the constraints (+Constr) yield better F-scores than those without. The difference is starker for the semi-supervised model (Sig). On the written corpus the constraint-based model outperforms the original model by 17.2% and on the spoken corpus by 18.3%. We further examined whether performance differences among models are statistically significant, using the Wilcoxon test. On the written corpus all constraint enhanced models significantly outperform the models without constraints. The same tendency is observed on the spoken corpus except for McDonald's (2006) model which performs comparably with and without constraints.

We also wanted to establish which is the best constraint model. On both corpora we find that the language model performs worst, whereas the significance model and McDonald perform comparably (i.e., the F-score differences are not statistically significant). To get a feeling for the difficulty of the task, we calculated how much our annotators agreed in their compression output. The inter-annotator agreement (F-score) on the written corpus was 65.8% and on the spoken corpus 73.4%. The agreement is higher on spoken texts since it consists of many short utterances (e.g., *Okay*, *That's it for now*, *Good night*) that can be compressed only very little or not all. Note that there is a marked difference between the automatic and human compressions. Our best performing systems are inferior to human output by more than 20 F-score percentage points.

Differences between the automatic systems and the human output are also observed

Source	The aim is to give councils some control over the future growth of second homes.
Gold	The aim is to give councils control over the growth of homes.
LM	The aim is to the future.
LM+Constr	The aim is to give councils control.
Sig	The aim is to give councils control over the future growth of homes.
Sig+Constr	The aim is to give councils control over the future growth of homes.
McD	The aim is to give councils.
McD+Constr	The aim is to give councils some control over the growth of homes.
Source	The Clinton administration recently unveiled a new means to encourage brownfields redevelopment in the form of a tax incentive proposal.
Gold	The Clinton administration unveiled a new means to encourage brownfields redevelopment in a tax incentive proposal.
LM	The Clinton administration in the form of tax.
LM+Constr	The Clinton administration unveiled a means to encourage redevelopment in the form.
Sig	The Clinton administration unveiled a encourage brownfields redevelopment form tax proposal.
Sig+Constr	The Clinton administration unveiled a means to encourage brownfields redevelopment in the form of tax proposal.
McD	The Clinton unveiled a means to encourage brownfields redevelopment in a tax incentive proposal.
McD+Constr	The Clinton administration unveiled a means to encourage brownfields redevelopment in the form of a incentive proposal.

Table 6.3: Example compressions produced by our systems (Source: source sentence, Gold: gold-standard compression, LM: language model compression, LM+Constr: language model compression with constraints, Sig: significance model, Sig+Constr: significance model with constraints, McD: McDonald's (2006) compression model, McD+Constr: McDonald's (2006) compression model with constraints).

Models	CompR	F-score
LM	46.2	18.4
Sig	60.6	23.3
McD	60.1	36.0
LM+Constr	41.2	28.2*
Sig+Constr	72.0	40.5*†
McD+Constr	63.7	40.8*†
Gold	70.3	—

Table 6.4: Results on the written text corpus; compression rate (CompR) and grammatical relation F-score (F-score); *: constraint-based model is significantly different from model without constraints; †: significantly different from LM+Constr.

Models	CompR	F-score
LM	52.0	25.4
Sig	60.9	30.4
McD	68.6	47.6
LM+Constr	49.5	34.8*
Sig+Constr	78.4	48.7*†
McD+Constr	68.5	50.1†
Gold	76.1	—

Table 6.5: Results on the spoken text corpus; compression rate (CompR) and grammatical relation F-score (F-score); *: constraint-based model is significantly different from model without constraints.; †: significantly different from LM+Constr.

Models	Grammar	Importance
LM	$2.25^{\dagger\$}$	$1.82^{\dagger\$}$
Sig	$2.26^{\dagger\$}$	$2.99^{\dagger\$}$
McD	$3.05^{\dagger}$	$2.84^{\dagger}$
LM+Constr	$3.47^{*\dagger}$	$2.37^{*\dagger\$}$
Sig+Constr	3.76^{*}	3.53^{*}
McD+Constr	$3.50^{\dagger}$	$3.17^{\dagger}$
Gold	4.25	3.98

Table 6.6: Results on the written text corpus; average grammaticality score (Grammar) and average importance score (Importance) for human judgements; *: model is significantly different from model without constraints; †: significantly different from gold standard; $^{\$}$: significantly different from McD+Constr.

with respect to the compression rate. As can be seen the language model compresses most aggressively, whereas the significance model and McDonald tend to be more conservative and closer to the gold standard. Interestingly, the constraints do not necessarily increase the compression rate. The latter increases for the significance model but decreases for the language model and remains relatively constant for McDonald. It is straightforward to impose the same compression rate for all constraint-based models (e.g., by forcing the model to retain b tokens $\sum_{i=1}^{n} y_i = b$). However, we refrained from doing this since we wanted our models to regulate the compression rate for each sentence individually according its specific information content and structure.

We next consider the results of our human study which assess in more detail the quality of the generated compressions on two dimensions, namely grammaticality and information content. F-score conflates these two dimensions and therefore in theory could unduly reward a system that produces perfectly grammatical output without any information loss. Tables 6.6 and 6.7 show the mean ratings[3] for each system (and the gold standard) on the written and spoken corpora, respectively. We first performed an Analysis of Variance (ANOVA) to examine the effect of different system compressions. The ANOVA revealed a reliable effect on both grammaticality and importance for each corpus (the effect was significant by both subjects and items ($p < 0.01$)).

We next examine the impact of the constraints (+Constr in the tables). In most cases we observe an increase in ratings for both grammaticality and importance when

[3]All statistical tests reported subsequently were done using the mean ratings.

Models	Grammar	Importance
LM	$2.20^{\dagger\$}$	$1.56^{\dagger}$
Sig	$2.29^{\dagger\$}$	$2.64^{\dagger}$
McD	$3.33^{\dagger}$	$3.32^{\dagger}$
LM+Constr	$3.18^{*\dagger}$	$2.49^{*\dagger\$}$
Sig+Constr	$3.80^{*\dagger}$	$3.69^{*\dagger}$
McD+Constr	$3.60^{\dagger}$	$3.31^{\dagger}$
Gold	4.45	4.25

Table 6.7: Results on the spoken text corpus; average grammaticality score (Grammar) and average importance score (Importance) for human judgements; *: model is significantly different from model without constraints; †: significantly different from gold standard; $^{\$}$: significantly different from McD+Constr.

a model is supplemented with constraints. Post-hoc Tukey tests reveal that the grammaticality and importance ratings of the language model and significance model significantly improve with the constraints ($\alpha < 0.01$). By contrast, McDonald's system sees a numerical improvement with the addition of constraints, but this difference is not statistically significant. These tendencies are observed on the spoken and written corpora.

Upon closer inspection, we can see that constraints influence considerably the grammaticality of the unsupervised and semi-supervised systems. Tukey tests reveal that LM+Constr and Sig+Constr are as grammatical as McD+Constr. In terms of importance, Sig+Constr and McD+Constr are significantly better than LM+Constr ($\alpha < 0.01$). This is not surprising given that LM+Constr is a very simple model without a mechanism for highlighting important words in a sentence. Interestingly, Sig+Constr performs as well as McD+Constr in retaining the most important words, despite the fact that it requires minimal supervision. Although constraint-based models overall perform better than models without constraints, they generally receive lower ratings (for grammaticality and importance) in comparison to the gold standard. And the differences are significant in most cases.

In summary, we observe that constraints boost performance. This is more pronounced for compression models that are either unsupervised or use small amounts of parallel data. For example, a simple model like Sig yields performance comparable to McDonald (2006) when constraints are taken into account. This is an encouraging

result suggesting that ILP can be used to create good compression models with relatively little effort (i.e., without extensive feature engineering or elaborate knowledge sources). Performance gains are also obtained for competitive models like McDonald's (2006) that are fully supervised. But these gains are smaller, presumably because the initial local model does a good job at producing grammatical output. Finally, our improvements are consistent across corpora and evaluation paradigms.

6.5 Summary of Chapter

In this chapter we have presented a novel method for automatic sentence compression. A key aspect of our approach is the use of integer linear programming for inferring globally optimal compressions in the presence of linguistically motivated constraints. We have shown how previous formulations of sentence compression can be recast as ILPs and extended these models with constraints ensuring that the compressed output is structurally and semantically well-formed. Contrary to previous work that has employed ILP solely for decoding, our models integrate learning with inference in a unified framework.

Our experiments have demonstrated the advantages of the approach. Constraint-based models consistently bring performance gains over the same models without constraints. These improvements are more impressive for models that require little or no supervision. A case in point here is the significance model discussed above. The unconstrained incarnation of this model performs poorly and considerably worse than McDonald's (2006) state-of-the-art model. The addition of constraints improves the output of this model so that its performance is indistinguishable from McDonald. Note that the significance model requires a small amount of training data (50 parallel sentences), whereas McDonald is trained on hundreds of sentences. It also presupposes little feature engineering, whereas McDonald utilises thousands of features. Some effort is associated with framing the global constraints; however these are created once and are applied across models and corpora. We have also observed small performance gains for McDonald's system when the latter is supplemented with constraints. Larger improvements are possible with more sophisticated constraints; however our intent was to devise a set of general constraints that are not tuned to the mistakes of any specific system in particular.

Chapter 7

Document Compression

Throughout this thesis we have focused on a simple instantiation of the sentence compression task. First we assume that compression occurs without any rewriting operations besides word removal. Secondly, compression is performed on isolated sentences without taking their surrounding context into account.

In this chapter we address the latter simplification and present a compression model that makes use of discourse-level information. Performing compression on isolated sentences is at odds with most of its applications which aim to create a shorter document rather than a single sentence. For example, compressing a document to display text on PDA requires the resulting document to not only be grammatical but also coherent in order to be easily read and understood. However, this cannot be guaranteed without knowledge of how the discourse progresses from sentence to sentence. To give a simple example, a contextually aware compression system could drop a word or phrase from the current sentence simply because it is not mentioned anywhere else in the document. Or it could decide to retain the word or phrase due to previous references. Neglecting to incorporate discourse-level information into our compression models may lead to documents fraught with coherence violations (e.g., dangling anaphora) and thus difficult to understand. The discourse information will provide a much richer view of the document than can be gained from the surface form and sentence parse trees. It can be viewed as another form of linguistic evidence and can complement the representations used in earlier models, such as parse trees and grammatical relations, to provide a better interpretation of the document.

Knowledge of the discourse will not only help maintain coherence but can also notify our models of what information is important thus providing improvements in sentence-level compressions. The task is admittedly complex and the topic of much

research in document summarisation. A number of factors have been identified as signalling what information is important in a document. These include the discourse topic, whether the sentence introduces new entities or events that have not been mentioned before, and the reader's background knowledge. Evidence that contextual cues are strong indicators of importance stems from professional summarisation. Abstractors often rely on contextual cues and a discourse-level representation which they piece together to form the *theme* of the document, while creating summaries (Endres-Niggemeyer 1998). The contextual cues are shallow sentence-level features whereas the theme is a structured mental representation of what the document is about. It links textual elements together in a similar way to a rhetorical-level analysis of the document's content. For example, two passages may be linked if one is a restatement, exemplifies, or is a cause/effect of the other passage.

For the remainder of this chapter we will use the term document compression to refer to a document whose sentences have been compressed. More formally, given a document, D, consisting of sentences, $D = S_1, \ldots, S_m$, our goal is to compress each sentence $S = w_1, \ldots, w_n$ by deleting words from the original. The compressed document should retain the most important information, remain grammatical and coherent. We could simply tackle the task by compressing each sentence sequentially using our sentence compression systems from Chapter 6. However, in this chapter we will show that a discourse aware model is better suited to this task.

7.1 Related Work

In this section we review some of the previous work on incorporating discourse-level information into summarisation and compression models.

Jing (2000) uses information from the local context as evidence for and against the removal of phrases during compression. Her model assumes that the local context provides information about the main topic being discussed and phrases in the sentence which are most related to the main topic should not be dropped. The topic is not explicitly identified, instead the importance of each phrase is determined by the number of lexical links within the local context. Words which are more connected have a higher chance of being the focus of the local context and thus related to the main topic.

Her model links two words if they are repetitions, morphologically related or associated in WordNet (Miller 1995) through a lexical relation (e.g., hyponymy, synonymy). This leads to nine possible relational links. Different types of links are con-

sidered more important, for example, repetition and inflections are more important than hypernyms. A context weight is calculated for each word based on the number of links to the local context and the importance of each relation type. Phrases are scored by the sum of their children's context scores. The decision to drop a phrase from a sentence are based partly on the local context and other factors such as the phrase's grammatical role and previous evidence from a parallel corpus (see Section 2.3.1 for more details).

Although Jing (2000) incorporates discourse information into a compression approximately through the local context, the number of 'free' parameters in her method poses some problems. Firstly, determining the size of the local context is non-trivial and in the worst-case will be set arbitrarily. In her experiments the size of the local context is not mentioned. Also weights must be given to each type of lexical relation to signify how good each relation is at determining the focus of the local context. Ideally, we want a method for supplementing our compression models with discourse information which requires as few 'free' parameters as possible.

Daumé III and Marcu (2002) present a summarisation system that uses the syntactic structure of each sentence and the overall discourse structure of the input document. The system uses a statistical hierarchical model of text production in order to drop syntactic and discourse units from a document deemed to be unimportant, this in turn generates a coherent and grammatical summary. The task is framed in a similar manner to the sentence compression task. Given a document $D = w_1, w_2, \ldots, w_n$ the goal is to produce a summary, S, by dropping any subset of words from D.

Their model is an extension of the noisy-channel model for sentence compression (Knight and Marcu 2002). Recall that the noisy-channel has two components: a language model and a channel model. In the sentence compression instantiation both models act on probabilistic context-free grammar (PCFG) representations. Daumé III and Marcu (2002) supplement this representation with a discourse representation that connects sentences within the document in the form of a tree structure. For this purpose they select the Rhetorical Structure Theory (RST Mann and Thompson 1988) of discourse structure to model the relationships between sentences.

In the RST framework, a document is represented by a tree whose leaves correspond to text fragments. The fragments are the minimal units of the discourse and are termed *elementary discourse units* (EDUs). The internal nodes of the tree correspond to contiguous text spans and the nodes are labelled with a *rhetorical relation*. A crucial point made by RST is that most rhetorical relations between two segments in a text are asymmetric. The *nucleus* in the relation is the node which contains more essential

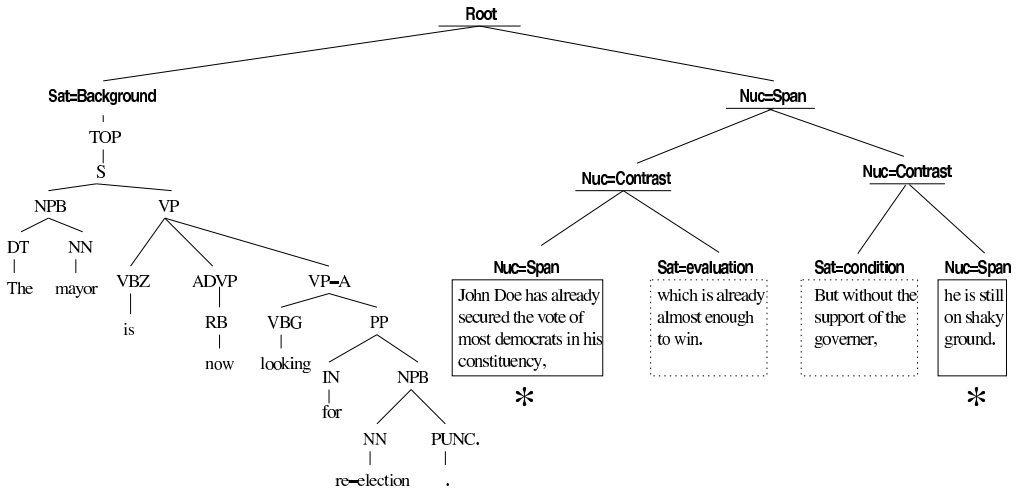

Figure 7.1: A DS-tree for text (1). The DS-tree depicts the full discourse and a partial syntactic parse (to save space).

information, while *satellite* nodes indicate supporting or background units of information. There are approximately 25 rhetorical relations in RST, examples of which include: background, contrast, purpose, motivation, circumstance and solutionhood.

Daumé III and Marcu's (2002) system works in a pipeline fashion. First discourse structures are generated using a decision-based discourse parser (Marcu 2000). This builds a RST discourse structure containing EDUs and rhetorical relations. The EDUs are then syntactically parsed using Collins's (1997) parser. The EDUs' syntactic trees are then merged with the discourse structure to form a discourse structure tree (DS-tree) which contains both discourse and syntactic information. The DS-tree acts as an input to the compression model. An example DS-tree for text (1), below, is given in Figure 7.1. The full parse of each EDU is omitted to save space.

(1) The mayor is now looking for re-election. John Doe has already secured the vote of most democrats in his constituency, which is already almost enough to win. But without the support of the governer, he is still on shaky ground.

Their compression performs compression by dropping either syntactic or discourse constituents from the DS-tree. The problem is framed as follows: given a document D, they wish to find the summary text S that maximises $P(S|D)$. They recast this formulation into the noisy-channel model thus maximising $P(D|S) \cdot P(S)$. It is intuitive to think of the compression process as: given a summary S what discourse and syntactic units can be added to S to yield the full document D.

The language model is tasked with assigning high $P(S)$ scores to summaries that

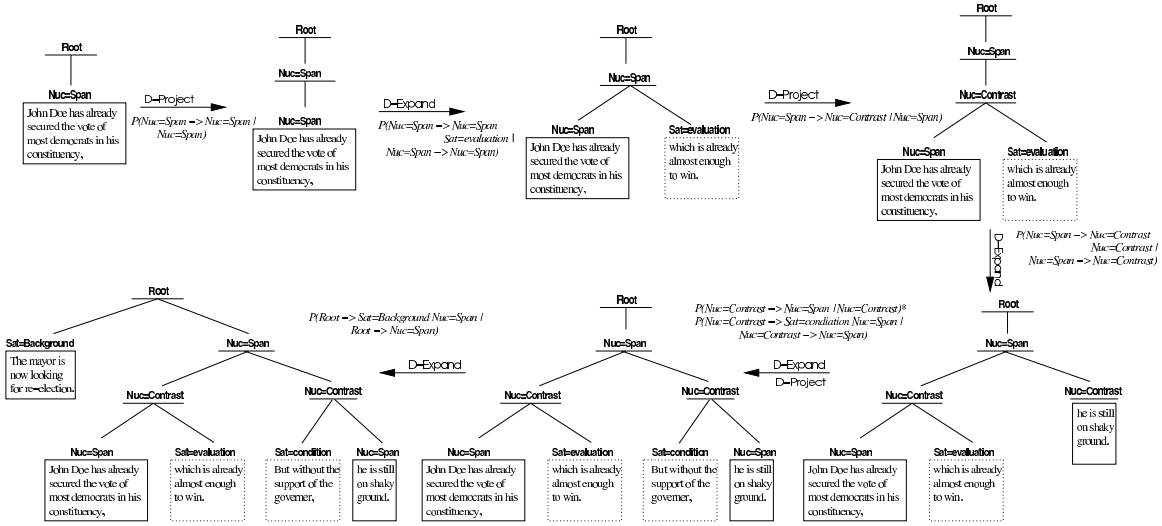

Figure 7.2: A sequence of discourse expansions for text (1) with probability factors.

contain grammatical sentences and are coherent. This is estimated using a bigram language model combined with non-lexicalised context-free grammar (PCFG) scores and context-free discourse probabilities, giving $P(S) = P_{bigram}(S) \cdot P_{PCFG}(S) \cdot P_{DPCFG}(S)$.

The channel model, $P(D|S)$ adds syntactic constituents or discourse units to the summary. Syntactic constituents are expanded in the same manner as Knight and Marcu (2002) (see Section 2.1.2 for details). For example, consider the text (2) as a summary of text (1). Through a sequence of discourse expansions it is possible to expand the summary (2) into the source text (1). The complete discourse expansion process is demonstrated in Figure 7.2.

(2) John Doe has already secured the vote of most democrats in his consistency.

The parameters for the language model, $P(S)$, require three corpora: a raw text corpus for P_{bigram}, a PCFG parsed corpus for P_{PCFG} and annotated discourse passages with their PCFG parse trees for P_{DPCFG}. Unfortunately annotated discourse passages are in short supply thus it is difficult to accurately estimate P_{DPCFG}. The parameters for the discourse portion of the language model, P_{DPCFG}, were estimated from an RST corpus of 385 Wall Street Journal articles from the Penn Treebank. Documents ranged from 31 to 2124 words with 458 words being the average.

The same corpus is also used to estimate the discourse parameter for the channel model, $P(D|S)$. 150 of the 385 documents were paired with extractive summaries and were manually annotated to mark the most important EDUs. Using these EDUs it is possible to examine the RST discourse tree and mark all descendants of the EDUs as

important. For example in Figure 7.1 if the annotators mark the two starred EDUs then all the parents are also considered important. This new annotation is then used to calculate the probability of dropping various nucleus and satellite nodes from certain relations, i.e., it is possible to estimate the probability $P(\text{Nuc} = \text{Span} \rightarrow \text{Nuc} = \text{Span Sat} = \text{Eval} | \text{Nuc} = \text{Span} \rightarrow \text{Nuc} = \text{Span})$.

Daumé III and Marcu (2002) test their system on two small data sets. The first is drawn from the Wall Street Journal (WSJ) portion of the Penn Treebank. It consists of 16 documents of between 41 and 87 words. The second, the MITRE set, originates from a collection of student compositions and contains five documents of between 64 and 91 words. They were unable to test the system on longer documents since the decoder (used to find the optimal summary) ran out of memory. Their system is compared against a baseline which randomly drops 50% of the words, the sentence compression system of Knight and Marcu (2002) in which each sentence in a document is compressed sequentially, and human authored summaries. Six human evaluators rated the systems according to three metrics: grammaticality, coherence and summary quality on a five point scale. Their results show that their system provides more grammatical (3.45 vs 3.30) and coherent (3.16 vs 2.98) summaries in comparison to the sentence compression system but there is no statistically significant difference between the quality of the summaries (2.88 vs 2.70) on the WSJ data set; a similar pattern is observed for the MITRE data. Another interesting note is that the system performs better on the MITRE data set due to its short sentences which can be parsed more accurately for discourse information. However, all systems perform significantly worse than the human authored summaries which score 4.65 for grammaticality, 4.48 for coherence and 4.53 for summary quality on the WSJ. Similar numbers are obtained for the MITRE data.

Discourse-information has also been incorporated into other summarisation methods including sentence extraction (Barzilay and Elhadad (1997), see section 7.2.2 for details) and content selection (Teufel and Moens 2002).

7.2 Discourse Representation

Obtaining an appropriate representation of discourse is the first step toward creating a compression model that exploits contextual information. Previous work has focused on theories of global discourse such as Rhetorical Structure Theory (RST Mann and

Thompson 1988), however RST parsers (Marcu 2000) tend to be unreliable[1] for most documents except those that are short and contain short, simple sentences. This is demonstrated by the results given by Daumé III and Marcu (2002) where they found that their discourse parser produced noisy parses for documents containing longer sentences. We strive for a more robust method for obtaining discourse representations and focus on local rather than global coherence. Models of local coherence are concerned with the way adjacent sentences bind together to form a larger discourse. Although these models do not explicitly capture the long distance relationships between sentences, local coherence is still an important prerequisite for maintaining global coherence. Our goal is to annotate our document automatically with discourse-level information which will subsequently be used to inform our compression procedure.

In this section we will examine two complementary theories of local coherence, namely Centering Theory (Grosz et al. 1995) and lexical chains. Both theories assume that coherence is achieved through the way discourse entities are introduced and discussed. We present a more detailed introduction in the following sections.

7.2.1 Centering Theory

Centering Theory (Grosz et al. 1995) is an entity-orientated theory of local coherence and salience. Its aim is to make cross-linguistically valid claims about which discourses are easier to process therefore it is best viewed as a *linguistic* theory rather than a computational one. The theory is presented in an abstract form and provides no specific algorithms computing the components required for centering.

The theory begins by assuming that a discourse is broken into 'utterances'. These can be phrases, clauses, sentences or even paragraphs. Centering characterises discourses as coherent because of the way discourse entities are introduced and discussed between utterances. The theory further distinguishes between salient entities and the rest. Specifically, although each utterance may contain several entities, it is assumed that a *single entity* is salient or "centered", thereby representing the current discourse focus. One of the main claims underlying centering is that discourse segments in which successive utterances contain common centers are more coherent than segments where the center repeatedly changes.

Each utterance U_j in a discourse has a list of *forward-looking centers*, $C_f(U_j)$ and a *unique backward-looking center*, $C_b(U_j)$. $C_f(U_j)$ represents a ranking of the entities

[1]Marcu's (2000) parser achieves the following F-scores for identification: 38.2 for EDUs, 50.0 for hierarchical spans, 39.9 for nuclearity and 23.4 for relation tagging.

invoked by U_j according to their salience. Thus, some entities in the discourse are deemed more important than others. The C_b of the current utterance U_j, is the highest-ranked element in $C_f(U_{j-1})$ that is also in U_j. The C_b thus links U_j to the previous discourse, but it does so *locally* since $C_b(U_j)$ is chosen from U_{j-1}. These concepts are demonstrated in passages (3-a)–(3-c) taken from Walker et al. (1998). Here we can see that utterances (3-a) and (3-b) have the forward-looking centers *Jeff*, *Dick* and *car* which are ranked according to their salience. To determine the backward-looking center of (3-b) we find the highest ranked entity in the forward-looking centers for (3-a) which also occurs in (3-b). The same procedure is applied for utterance (3-c).

(3) a. Jeff Helped Dick wash the car.
`CF(Jeff, Dick, car)`

b. He washed the windows as Dick waxed the car.
`CF(Jeff, Dick, car)`
`CB=Jeff`

c. He soaped a pane.
`CF(Jeff, pane)`
`CB=Jeff`

Centering Algorithm As noted, Centering is primarily considered a linguistic theory rather than a computation one. It is therefore not explicitly stated how the concepts of "utterance", "entities" and "ranking" are instantiated. A great deal of research has been devoted into fleshing these out and many different instantiations have been developed in the literature (see Poesio et al. 2004 for details). For our purposes, the instantiation will have a bearing on the reliability of the algorithm to detect centers. If the parameters are too specific then it may not be possible to accurately determine the center for a given utterance. Since our aim is to identify centers in discourse automatically, our parameter choice is driven by two considerations: robustness and ease of computation.

We therefore follow previous work (e.g., Miltsakaki and Kukich 2000) in assuming that the unit of an utterance is the sentence (i.e., a main clause with accompanying subordinate and adjunct clauses). This is a simplistic view of an utterance, however it is in line with our compression task which also operates over sentences. We determine which entities are invoked by a sentence using two methods. First, we perform named entity identification and coreference resolution on each document using LingPipe[2], a

[2]LingPipe can be downloaded from `http://www.alias-i.com/lingpipe/`.

publicly available system. Named entities are not the only type of entity to occur in our data, thus to ensure a high entity recall we add named entities and all remaining nouns[3] to the C_f list. Entity matching between sentences is required to determine the C_b of a sentence. This is done using the named entity's unique identifier (as provided by LingPipe) or by the entity's surface form in the case of nouns not classified as named entities.

Entities are ranked according to their grammatical roles; subjects are ranked more highly than objects, which are in turn ranked higher than other grammatical roles (Grosz et al. 1995); ties are broken using left-to-right ordering of the grammatical roles in the sentence (Tetreault 2001). We identify grammatical roles with RASP (Briscoe and Carroll 2002). Formally, our centering algorithm is as follows (where U_j corresponds to sentence j):

1. Extract entities from U_j.
2. Create $C_f(U_j)$ by ranking the entities in U_j according to their grammatical role (subjects $>$ objects $>$ others, ties broken using left-to-right word order of U_j).
3. Find the highest ranked entity in $C_f(U_{j-1})$ which occurs in $C_f(U_j)$, set the entity to be $C_b(U_j)$.

The above procedure involves several automatic steps (named entity recognition, coreference resolution, identification of grammatical roles) and will unavoidably produce some noisy annotations. So, there is no guarantee that the right C_b will be identified or that all sentences will be marked with a C_b. The latter situation also occurs in passages that contain abrupt changes in topic. In such cases, none of the entities realised in U_j will occur in $C_f(U_{j-1})$. Rather than accept that discourse information may be absent in a sentence, we turn to lexical chains as an alternative means of capturing topical content within a document.

7.2.2 Lexical Chains

Lexical cohesion refers to the degree of semantic relatedness observed among lexical items in a document. The term was coined by Halliday and Hasan (1976) who observed that coherent documents tend to have more related terms or phrases than incoherent ones. A number of linguistic devices can be used to signal cohesion; these range from

[3] As determined by the word's part-of-speech tag.

repetition, to synonymy, hyponymy and meronymy. Lexical chains are a representation of lexical cohesion as sequences of semantically related words (Morris and Hirst 1991). There is a close relationship between discourse structure and cohesion. Related words tend to co-occur within the same discourse. Thus, cohesion is a surface indicator of discourse structure and can be identified through lexical chains.

Lexical chains provide a useful means for describing the topic flow in discourse. For example, a document containing the chain {*house*, *home*, *loft*, *house*} will probably describe a situation involving a house. It is common for documents to contain many different lexical chains as multiple topics (or themes) occur throughout a document. However, some of these topics will only be asides and be represented by short lexical chains whereas the main topics will tend to represented by dense longer chains. Words participating in the latter chains are important for our compression task — they reveal what the document is about — and in all likelihood should not be deleted.

Barzilay and Elhadad (1997) describe a technique for building lexical chains for extractive text summarisation. In their approach chains of semantically related expressions are used to select sentences for inclusion in a summary. Their algorithm uses WordNet (Miller 1995) to build chains of nouns (and noun compounds). Words in WordNet are represented by senses which break a word into its possible meanings. Senses are represented relationally by synonym sets which are the sets of all the words sharing a common sense. Words belonging to the same category are linked through semantic relations. Generally, lexical chains are built using WordNet through a three stage procedure (Barzilay and Elhadad 1997):

1. Select a set of candidate words (typically all words that appear in WordNet).

2. For each candidate word, find the appropriate chain relying on a relatedness criterion among members of the chains.

3. If a chain is found, insert the word into the chain.

The crux of the problem lies in the disambiguation strategy for mapping words to their senses. If a weak strategy is chosen (for example, greedily disambiguate) and the senses are chosen wrongly, then chains obtained will not reflect the relationship between the word senses used in the document. It is on this issue that lexical chaining algorithms differ[4] .

[4]We refer the interested reader to Barzilay and Elhadad (1997) for details of their word sense disambiguation algorithm.

The lexical chains obtained by Barzilay and Elhadad (1997) are then used to perform text summarisation through sentence extraction. The chains are ranked heuristically by a score based on their length and homogeneity. A summary is produced by extracting sentences corresponding to *strong chains*, i.e., chains whose score is two standard deviations above the average score.

Like Barzilay and Elhadad (1997), we wish to determine which lexical chains indicate the most prevalent discourse topics. Our assumption is that terms belonging to these chains are indicative of the document's main focus and should therefore be retained in the compressed output. Barzilay and Elhadad's scoring function aims to identify sentences (for inclusion in a summary) that have a high concentration of chain members. In contrast, we are interested in chains that span several sentences. We thus score chains according to the number of sentences their terms occur in. For example, the hypothetical chain $\{house_3, home_3, loft_3, house_5\}$ (where $word_i$ denotes *word* occurring in sentence *i*) would be given a score of two as the terms only occur in two sentences. We assume that a chain signals a prevalent discourse topic if it occurs throughout more sentences than the average chain. The scoring algorithm is outlined more formally below:

1. Compute the lexical chains for the document.
2. $Score(Chain) = Sentences(Chain)$.
3. Discard chains for which $Score(Chain) < Average(Score)$.
4. Mark terms from the remaining chains as being the focus of the document.

We use the method of Galley and McKeown (2003) to compute lexical chains for each document.[5] It improves on Barzilay and Elhadad's (1997) original algorithm by providing better word sense disambiguation and linear runtime.

7.2.3 Annotation Method

Before compression takes place, all documents are processed using the centering and lexical chain algorithms described above. In each sentence we annotate the center $C_b(U_j)$ if one exists. Words (or phrases) that are present in the current sentence and function as the center in the next sentence $C_b(U_{j+1})$ are also flagged. Finally,

[5]The software is available from `http://www1.cs.columbia.edu/~galley/`.

Bad (weather) dashed hopes of attempts to halt the (flow$_1$) during what was seen as a lull in the ‖lava's‖ momentum. Experts say that even if the eruption stopped (today$_2$), the pressure of ‖lava‖ piled up behind for six (miles$_3$) would bring ‖debris‖ cascading down on to the (town) anyway. Some estimate the volcano is pouring out one million tons of ‖debris‖ a (day$_2$), at a (rate$_1$) of 15(ft$_3$) per (second$_2$), from a fissure that opened in mid-December.
The Italian Army (yesterday$_2$) detonated 400lb of dynamite 3,500 feet up Mount Etna's slopes.

Figure 7.3: Excerpt of document from our test set with discourse annotations. Centers are in double boxes; terms occurring in lexical chains are in oval boxes. Words with the same subscript are members of the same chain (e.g., *today*, *day*, *second*, *yesterday*)

words are marked if they are part of a prevalent chain. Examples of our discourse annotation are given in Figures 7.3, 7.4 and 7.5. As shown in the figures, the centering annotations tend to mark the most salient entities in each sentence. For example, in Figure 7.3 the centers are *lava* and *debris*, from this we can see that the document is related to volcanoes. Similarly Figure 7.4 is concerned with Mrs Allan (see the centers *Mrs Allan*, *her*, *she*). The centers of Figure 7.5 do not convey the salient topics of the document in the way the previous two examples did. In this example we can see that the lexical chains algorithm provides a better insight into the text. It shows that the centering algorithm was unable to fully annotate sentences, only finding *Peter Anderson* and *allotment*; however, using the lexical chains annotations we can see the text is about a policeman, a woman and her boyfriend.

7.3 Discourse Model

The foundation of our discourse model is the significance model presented in Section 6.1.2 along with the constraints from Section 6.2. We select this model for several reasons. First, it only requires little parallel data (50 sentences) and thus can be ported across domains and text genres, whilst delivering state-of-the-art results (see the results in Section 6.4 for details). Second, discourse-level information can be easily incorporated by augmenting the constraint set. This is not the case for other approaches

Mrs Allan was taken to nearby Kelowna General Hospital after the body$_1$ was found. Her husband, Stuart, 52, said yesterday he had been in daily contact with her since she flew to Canada last month$_2$ on the second pilgrimage to find her son." She is suffering from exhaustion but otherwise fine," he said."I spoke to her last night$_2$ and she is under strict orders to have complete rest."

Figure 7.4: Excerpt of document from our test set with discourse annotations. Centers are in double boxes; terms occurring in lexical chains are in oval boxes. Words with the same subscript are members of the same chain (e.g., *night*, *month*, *days*, *years*)

A policeman$_1$ was yesterday jailed for seven years for raping an 18-year-old woman$_2$ in his marked patrol car while he was on duty and in uniform.Sentencing constable Peter Anderson, 41, Mr Justice Jowitt told him he had done "great damage to the trust in police$_1$". Anderson, married with two children, attacked the woman$_2$ in a deserted allotment, after agreeing to give her and a boyfriend$_2$ a lift home from a discotheque.He first dropped the man$_2$ off and then drove to the allotment.

Figure 7.5: Excerpt of document from our test set with discourse annotations. Centers are in double boxes; terms occurring in lexical chains are in oval boxes. Words with the same subscript are members of the same chain (e.g., *police*, *policeman*, *officer*)

(e.g., those based on the noisy channel model) where compression is modelled by grammar rules indicating which constituents to delete in a syntactic context. Finally, the ILP framework provides exact inference even in the face of constraints thus avoiding approximations and heuristics during decoding.

The base model includes the objective function from Equation (6.11), the sequential constraints of Equations (6.4)–(6.8) to ensure valid combinations of trigrams are selected, and the syntactically and semantically motivated constraints from Equations (6.20)–(6.31). The latter constraints instill global linguistic information into the model and act on the modifier and argument structure of the sentence.

Recall that we have a 0–1 decision variable representing if a word is to be included in the compression.

$$\delta_i = \begin{cases} 1 & \text{if } x_i \text{ is in the compression} \\ 0 & \text{otherwise} \end{cases} \quad \forall i \in [1 \dots n]$$

This will be useful for building our new discourse constraints.

7.3.1 Discourse Constraints

We now turn our attention to incorporating discourse information into our compression model. Recall that we automatically annotate each sentence with its own center $C_b(U_j)$, the center $C_b(U_{j+1})$ of the sentence following it, and words that are members of high scoring lexical chains corresponding to the document's focus. Provided with this additional knowledge our compression model builds two new types of constraints to ensure that compressed documents preserve the flow and topic of the source documents.

Our first goal to is preserve the focus of each sentence. If the center, C_b is identified in the source sentence it must be retained in the compression. If present, the entity realised as the C_b in the following sentence should also be retained to ensure the entities in focus between sentences are preserved. Such a condition is easily captured with the following ILP constraint:

$$\delta_i = 1 \qquad (7.1)$$
$$\forall i : x_i \in \{C_b(U_j), C_b(U_{j+1})\}$$

Consider for example the discourse in Figure 7.3. The constraints generated from

Equation (7.1) will require the compression to retain *lava* in the first two sentences and *debris* in the second and third sentences.

The centering algorithm relies on NLP technology that is not 100% accurate (named entity detection and coreference resolution) therefore the algorithm can only approximate the center for each sentence. In some cases the algorithm is unable to identify the center. The lexical chains algorithm provides a complementary annotation of the topic or theme of the document using information which is not restricted to adjacent sentences. We thus require that words in topical lexical chains be retained in the compression.

$$\delta_i = 1 \tag{7.2}$$
$$\forall i : x_i \in \text{document topical lexical chain}$$

This constraint only applies to nouns that are members of lexical chains representing the focus of the document. See for instance the words *flow* and *rate* in Figure 7.3 which are members of the same chain (marked with subscript one). According to constraint (7.2) both words must be included in the compressed document. In the case of Figure 7.5 the chain relating to the police (*police*, *policeman*) and people (*woman*. *boyfriend*, *man*) would be retained in the compression.

Our final discourse constraint follows from our basic approximation of discourse in Section 6.2. It concerns personal pronouns. Specifically, we wish to include personal pronouns (whose antecedent may not always be identified). This is realised in constraint (7.3) repeated from Section 6.2.

$$\delta_i = 1 \tag{7.3}$$
$$\forall i : x_i \in \text{ personal pronouns}$$

The constraints just described ensure that the compressed document will retain the discourse flow of the source document and will preserve terms indicative of important topics. The discourse constraints will not only ensure that compressed documents are coherent but they will additionally benefit sentence-level compression. The discourse information is a deeper interpretation of the document and provides the compression model with strong evidence for including discourse relevant words in the compression. Words not marked as discourse relevant can be considered for removal. It is now possible to interpret what information is important through linguistic evidence as provided

Bad weather dashed hopes to halt the flow during what was seen as lull in lava's momentum. Experts say that even if eruption stopped, the pressure of lava piled would bring debris cascading. Some estimate volcano is pouring million tons of debris from fissure opened in mid-December. The Army yesterday detonated 400lb of dynamite.

Figure 7.6: Discourse ILP output on excerpt from Figure 7.3.

by the discourse rather relying solely on the surface level document characteristics (i.e., word frequencies).

7.3.2 Applying the Constraints

Our compression system is given a (sentence separated) source document as input. The model and constraints just presented are applied sequentially to all sentences to generate a compressed version of the source. We thus create and solve an ILP for every sentence. In our earlier formulation of the compression task, a significance score (see Section 6.1.2) was used to highlight which nouns and verbs to include in the compression. As far as nouns are concerned, our discourse constraints perform a similar task. Thus, when a sentence contains discourse annotations, we are inclined to trust them more and only calculate the significance score for verbs.

During development it was observed that applying all discourse constraints simultaneously (see Equations (7.3)–(7.2)) results in relatively long compressions. To counteract this, we employ these constraints using a back-off strategy that relies on progressively less reliable information. Our back-off model works as follows: if centering information is present, we apply the appropriate constraints (Equation (7.1)). If no centers are present, we back-off to the lexical chain information using Equation (7.2), and in the absence of the latter we back-off to the pronoun constraint (Equation (7.3)). Finally, if discourse information is entirely absent from the sentence, we default to the significance score. Sentential constraints are applied throughout irrespectively of discourse constraints. In our test data the centering constraint was used in 68.6% of the sentences. The model backed off to lexical chains for 13.7% of the test sentences, whereas the pronoun constraint was applied in 8.5%. Finally, the noun and verb significance score was used on the remaining 9.2%. Examples of our system's output for the texts in Figures 7.3, 7.4 and 7.5 are given in Figures 7.6, 7.7 and 7.8 respectively.

Mrs Allan was taken to Kelowna Hospital. Her husband, Stuart, said he had been in contact with her since she flew to last month to find her son. “She is suffering” he said. “I spoke to her last night and she is under orders to have rest.”

Figure 7.7: Discourse ILP output on excerpt from Figure 7.4.

Policeman was jailed for raping an woman while he was on duty and in uniform. Peter Anderson, Jowitt told him he had done “damage to trust”. Anderson, married with children, attacked the woman in allotment after agreeing to give her and a boyfriend a lift home. Drove to allotment.

Figure 7.8: Discourse ILP output on excerpt from Figure 7.5.

7.4 Experimental Set-up

In this section we present our experimental set-up. We briefly recap the model of McDonald (2006) which we use for comparison with our approach, henceforth Discourse ILP, and outline our parameter estimation strategy. Finally, we provide a summary of the evaluation methodology previously introduced.

Comparison with state-of-the-art An obvious evaluation experiment would involve comparing the ILP model without any discourse constraints against the discourse informed model presented in this work. Unfortunately, the two models obtain markedly different compression rates[6] which renders the comparison of their outputs problematic. To put the comparison on an equal footing, we evaluated our approach against a state-of-the-art model that achieves a compression rate similar to ours without taking discourse-level information into account. As discussed in Section 2.2.3, McDonald (2006) formalises sentence compression as a classification task in a discriminative large-margin learning framework: pairs of words from the source sentence are classified as being adjacent or not in the target compression. A large number of features are defined over words, parts of speech, phrase structure trees and dependencies. These are gathered over adjacent words in the compression and the words in-between which

[6]The discourse agnostic ILP model has a compression rate of 81.2%; when discourse constraints are include the rate drops to 65.4%. Recall that the ILP models from Chapter 5.2 contained a simple discourse constraint.

were dropped.

McDonald's (2006) model has a head start against our Discourse ILP model; it uses a large parallel corpus to learn from whereas we only have a few constraints and use fifty sentences for parameter tuning. The comparison of the two systems allows us to investigate whether discourse information is redundant when using a powerful sentence compression model. Our earlier experiments in Section 6.4 demonstrate that sentence-level constraints do not bring significant benefits for McDonald's (2006) fully supervised model.

Corpus There are three compression corpora available to us: the Ziff-Davis corpus, the spoken corpus and the written corpus. The Ziff-Davis is inappropriate for our purposes since it consists of isolated sentences only. The spoken corpus does not contain documents in the traditional sense as they are not crafted to be read easily. Coreference resolution algorithms on which the centering algorithm relies have been developed primarily for written text. Therefore we focus on the human authored written compression corpus. This comprises of 82 stories (1,629 sentences) from the British National Corpus and the LA Times Washington Post. The corpus is split into 48 documents (962 sentences) for training purposes, three for development (63 sentences) and 31 for testing (604 sentences).

Parameter Estimation Our parameters are estimated in the same manner as in Section 6.3. The language model required for our Discourse ILP system was trained on 25 million tokens from the North American News corpus. The significance score was based on 25 million tokens from the same corpus. McDonald's (2006) system was trained on the full training set (962 sentences) and the feature set was identical to his original description. A slightly modified loss function was required to encourage compression on our data set (see Section 6.3 for details).

Evaluation Method Following from Chapter 6 we perform a sentence-based evaluation on compressions using F-scores computed over grammatical relations (see Section 4.2 for details). The relational F-score evaluation provides insight into how well our systems are performing the isolated sentence compression task. It will also allow us to assess if the discourse constraints increase or reduce the quality of sentence-level compressions. Besides the intrinsic evaluation, we also wish to evaluate the compressed documents as a whole. In Section 4.3 we presented a document-level evalua-

Model	CompR	F-Score
McDonald	60.1%	36.0%*
Discourse ILP	65.4%	39.6%
Gold Standard	70.3%	—

Table 7.1: Compression results: compression rate and relation-based F-score; * sig. diff. from Discourse ILP ($p < 0.05$ using the Student t test).

tion designed to answer two questions: (1) are the document compressions readable? and (2) how much key information is preserved between the source document and its target compression? We are assuming here that the compressed document will function as a replacement for the source.

We will first briefly recap our document-level evaluation setup which uses a question-answering paradigm to measure the extent to which the compressed document can be used to find answers for questions which are derived from the source document. If the compressed document can answer the questions it implies the compression contains the core content from the source. Our evaluation items consist of six documents with between five to eight questions per document. Each question is factual-based and typically involves a who, what, where, when, how style question requiring one unambiguous answer.

Compressed documents and their accompanying questions were presented to human subjects. Three compression conditions were chosen: gold standard, Discourse ILP and McDonald's (2006) model. Each participant also rated the compressed document on a seven point scale for readability. A high score corresponds to high readability and a low score to low readability. Sixty unpaid volunteers took part in our Q&A evaluation over the Internet.

The answers provided by the participants were scored against an answer key. Each answer is marked with a score of one for a correct answer and zero of incorrect answer. In cases where two answers are required a score of 0.5 is awarded for each correct answer. The score for a compressed document is the average of its question scores. All subsequent tests and comparisons are performed on the document score.

Model	Readability	Q&A
McDonald	2.65*	54.4%*†
Discourse ILP	3.00*	67.8%*
Gold Standard	5.27†	82.2%†

Table 7.2: Human Evaluation Results: average readability ratings and average percentage of questions answered correctly. *: sig. diff. from Gold Standard; †: sig. diff. from Discourse ILP.

7.5 Results

As a sanity check, we first assessed the compressions produced by our model and McDonald (2006) on a sentence-by-sentence basis without taking the documents into account. There is no hope for generating shorter documents if the compressed sentences are either too wordy or too ungrammatical. Table 7.1 shows the compression rates (CompR) for the two systems and evaluates the quality of their output using F-score based on grammatical relations. As can be seen, the Discourse ILP compressions are slightly longer than McDonald (65.4% vs. 60.1%) but closer to the human gold standard (70.3%). This is not surprising: the Discourse ILP model takes the entire document into account, and compression decisions will be slightly more conservative. The Discourse ILP's output is significantly better than McDonald in terms of F-score, indicating that discourse-level information is generally helpful. Both systems could use further improvement as inter-annotator agreement on this data yields an F-score of 65.8%.

Let us now consider the results of our document-based evaluation. Table 7.2 shows the mean readability ratings obtained for each system and the percentage of questions answered correctly. We used an ANOVA to examine the effect of compression type (McDonald, Discourse ILP, Gold Standard). The ANOVA revealed a reliable effect on both readability and Q&A. Post-hoc Tukey tests showed that McDonald and the Discourse ILP model do not differ significantly in terms of readability. However, they are significantly less readable than the gold standard ($\alpha < 0.05$). For the Q&A task we observe that our system is significantly better than McDonald ($\alpha < 0.05$), but significantly worse than the gold standard ($\alpha < 0.05$).

These results indicate that the automatic systems lag behind the human gold standard in terms of readability. When reading entire documents, subjects are less tolerant of ungrammatical constructions. We also find out that despite relatively low readabil-

ity, the documents are overall understandable. The discourse-based model generates more informative documents — the number of questions answered correctly increases by 15% in comparison to McDonald. This is an encouraging result suggesting that there may be advantages in developing compression models that exploit contextual information.

7.6 Summary of Chapter

In this chapter we have presented a novel method for performing sentence compression on a document-level basis. Central to our approach is the use of discourse-level information which we annotate automatically. Our annotation algorithms are robust and complementary. They are inspired by two linguistic theories relating to local coherence, Centering Theory and lexical cohesion; and provide our compression model with important information for document (as opposed to sentence) compression.

Discourse related information is instilled into our model through the integer linear programming framework using a set of constraints. These constraints are designed to preserve the coherence of the source document and also provide additional evidence about which entities are important. We have shown that our model can be successfully employed to produce document compressions that preserve the core content of the source better than state-of-the-art discourse agnostic sentence compression models.

Chapter 8

Conclusions and Future Directions

This chapter summarises the main findings and contributions of this thesis and outlines future research directions.

8.1 Main Findings

This thesis has been concerned with the task of sentence compression. We have investigated the broad spectrum of sentence compression, from the analysis of human authored and automatically gathered compressions, to evaluation techniques of compression systems and models for compression. The following is a summary of the central findings and contributions of this work:

1. We conducted a novel and detailed analysis of the sentence compression task. This involved examining manual and automatic methods for data acquisition and resulted in the creation of two new publicly available compression corpora in the domains of spoken and written text. We found that human authored compressions and those automatically obtained from the Ziff-Davis corpus are substantially different in several respects. These include: compression rate, human authored compressions are more conservative at compressing; and word span removal, humans tend to remove single words rather than large phrases.

2. We have assessed whether automatic evaluation measures can be used for the compression task. Our results show that grammatical relations-based F-score (Riezler et al. 2003) correlates reliably with human judgements. This insight allowed us to use larger test sets for comparing compression systems and also helped with system development.

3. Two judgement elicitation studies formed a major part of our manual evaluation setup. The first was a more rigorous formulation of the experimental design proposed by Knight and Marcu (2002) in which naive judges rate compression output along two dimensions: grammaticality and importance. We modified their setup to only show one compression per source sentence using a Latin square design. The second elicitation study was concerned with document-level, rather than sentence-level, evaluation and followed a question-answer paradigm. Naive judges were asked to read fully compressed documents and answer questions derived from the source material's core content. Their answers were compared with a scoring scheme designed to assess the differences between gold standard and system generated compressions. This evaluation methodology holds promise beyond sentence compression and could be used more generally to evaluate abstractive or extractive summaries.

4. We have presented a novel method for automatic sentence compression. A key aspect of our approach is the use of integer linear programming for inferring globally optimal compressions in the presence of linguistically motivated constraints. We have shown how previous formulations of sentence compression can be recast as ILPs and extended these models with local and global constraints ensuring that the compressed output is structurally and semantic well-formed. Our experiments have demonstrated the advantages of the approach. Constraint-based models consistently bring performance gains over models without constraints. These improvements are more impressive for models that require little or no supervision.

5. Finally, we extended our ILP compression model to full documents rather than isolated sentences. Important for the success of this task is the ability to annotate documents with discourse information. We thus developed two annotation algorithms inspired by linguistic theories relating to local coherence: Centering Theory and lexical cohesion. Using these annotations we instilled discourse information into our compression models through constraints. Our constraints preserve the coherence of the source document and also provide additional evidence about which entities are important. Using our question-answering evaluation we found that our discourse informed compression model successfully produces document compressions that preserve the core content of the source document better than a state-of-the-art discourse agnostic sentence compression

model.

8.2 Future Research Directions

In this thesis we have been solely focused on one instantiation of sentence compression. When we introduced the task we outlined three factors that will influence the compression's information content: (1) the user's background knowledge, (2) the user's information need, and (3) the user's compression requirements. For this thesis we have explored the most general instantiation of these factors, that is, the compression takes into account general background knowledge and relates to the main topic or topics of the document from which the sentence is drawn. Finally we have not imposed any compression related requirements such as limiting the compression rate or changing the style of language between source sentence and compression.

Obvious future research directions within sentence compression are examining how to perform compression in the face of different compression factors. The most natural extension of this is query-focused compression in which the user expresses their information desire as a query. Query-focused summarisation has been an integral part of the past few Document Understanding Conferences. The user's compression requirements could be explored through providing compressions for specific devices or purposes. For example, in television captions and subtitles the display space is physically limited. A compression system would have to adapt to the available space by compressing longer sentences much more aggressively than shorter ones. In subtitling the compressions must remain coherent in a similar way to document compression therefore, the discourse annotations are likely to help provide better compressions.

Other aspects of the compression task include investigating new objective functions and constraints for ILP-based models. As we have demonstrated the ILP framework is flexible and can allow for any linear objective function. Related to the objective function are constraints. Thus far we have only explored hard constraints (constraints which must always hold), however it would be interesting to investigate soft constraints. Soft constraints are constraints which have a cost associated with them, the cost is incurred in the objective function if the constraint is violated in the solution. An important direction for future constraint research is how to automatically discover useful constraints from compression corpora.

Within the wider task of summarisation, sentence compression holds promise. We have already demonstrated that is possible to perform document compressions by in-

corporating discourse information into our models. While these document compressions can be considered as summaries, they differ considerably from most summarisation work in that they are fairly long. However, we believe this is the first step toward understanding how compression can help summarisation. Sentence compression can be viewed as part of the summarisation process which reduces documents horizontally by squeezing the sentences. Extraction, on the other hand, squashes documents vertically by removing sentences. These two methods can be combined in a pipeline where compression is performed followed by extraction or vice versa. However, ideally these two components should interface with one another thus allowing each component to inform the other and guide the summary. This could be achieved in ILP by reformulating existing extractive summarisation models as ILPs and integrating them into our compression models. Such a formulation is an avenue for future research.

In our document compressions we only examined the effect of local coherence. A natural progression is to study the effect of global discourse structure (Daumé III and Marcu 2002) on the compression task. In general, it will be useful to assess the impact of discourse information more systematically by incorporating it into generative and discriminative modelling paradigms. Our discourse annotation algorithms provide a robust means of gathering discourse information. The simplicity of our annotations will allow discourse information to be easily incorporated into existing summarisation systems that are currently largely discourse agnostic.

Integer Linear Programming (ILP) has been the central framework adopted throughout this thesis. We believe the approach holds promise for other generation applications such as sentence level paraphrasing, headline generation and summarisation. The advantages of using an ILP framework are numerous. It allows our problems to be modelled in a well-defined mathematical manner in which the solver provides the guarantee of optimality. As we have demonstrated, ILP is a flexible framework which can model a variety of different problems with the ability to include additional constraints motivated through syntactic, semantic or domain specific knowledge.

Appendix A

Experimental Instructions

This appendix contains the instructions presented to our annotators (see Chapter 3 for details) and judges in our elicitation studies (see Chapter 4).

A.1 Annotator Sentence Compression Instructions

This experiment is concerned with sentence compression. You will be presented with a selection of sentences from a news paper article. Your task is to compress each sentence or mark it as inappropriate for compression.

Compressing a sentence involves taking a the original sentence and producing a shorter version while retaining the most important information contained within the sentence.

The compressions you will produced should be constrained such that the compressed sentence can only be composed of words found in the original sentence and the ordering of words must not change. Words can only be removed from the sentence, there is no opportunity for the addition or reordering of words.

Ideally the compressed sentence will be grammatical and retain the most important information of the original sentence. Global coherence should be taken into account when possible but not at the expense of the compression of the sentence (although this typically won't be the case).

Very few of the sentences will be inappropriate for compression due to them being very short or containing no information. When you come across such a sentence you should mark it is inappropriate and not attempt to compress it.

There are no correct answers to this task. All compressions produced are considered valid provided they have been made while considering:

- The most important information in the original sentence.
- The grammaticality of the compressed sentence.

A.1.1 Interface

The interface will present you with a selection of documents to choose from. Please only select documents you have not done compressions for.

You will then be asked for your name and email address; these are used for tracking purposes and will not be passed onto any third party.

A list of sentences will be displayed with a checkbox underneath each word. Placing a tick in the box will remove the word (or punctuation) from the sentence; this will be reflected immediately in the compressed sentence box. If the sentence is not appropriate for compression, then please tick the inappropriate box.

A.1.2 Examples

Here are some examples of compressed sentences:

Example 1

- Seven states will hold presidential primaries or caucuses next Tuesday and President Bush campaigned today in one of the most important states , Georgia .
- States will hold primaries or caucuses next Tuesday and President Bush campaigned in Georgia .

Example 2

- Even though they may not like it , most women learn to tolerate being probed and examined in awkward positions .
- Women learn to tolerate being probed and examined in awkward positions .

Example 3

- The FBI also found former White House Deputy Counsel Vincent Foster ’s fingerprints on them .
- The FBI found Vincent Foster ’s fingerprints on them .

Example 4

- Sergei , who is a licensed surgeon , now practices healing of the spirit , his only instruments his hands and a bent wire that measures human energy fields for curses that cause illness and depression .
- Sergei practices healing of the spirit , his only instruments his hands and a bent wire that measures human energy fields for curses that cause illness and depression .

Example 5

- Sgt. Zuniga , when he first came on board , he had just gotten married , and , so I- I mean , I was surprised .
- Sgt. Zuniga , when he first came on board , had just gotten married , and so I was surprised .

Example 6

- Their spirit is just unbelievable - unbelievable spirit .
- Their spirit is unbelievable .

A.2 Sentence-level Evaluation Instructions

In this experiment you will be asked to judge how well a given sentence compresses the meaning of another sentence. You will see a series of sentences together with their compressed versions. Some sentence compressions will seem perfectly OK to you, but others will not. All compressed versions were generated automatically by a computer computer program.

Your task is to judge how good a compressed sentence is according to two criteria: (a) grammaticaility, and (b) importance. The grammaticality judgement is based on whether the sentence is understandable. The importance judgement relates to how well the compression preserves the most important information of the original and whether it is adequately compressed. Both judgements are rated on scales from 1 (poor) to 5 (good).

A compression with a low grammaticality score is one that is almost impossible to understand. Compressions should receive low importance scores if they miss out important information from the original sentence. Or do not remove any superfluous information from the original sentence even though it is evident that it can be omitted without drastic information loss.

A good compression is one that is readily comprehensible and retains the most important information from the original sentence. Good sentence compressions should receive a high grammatical score and importance score.

For example, if you were asked to rate the following compression:

- Nonetheless, FBI director Louis Freeh has today ordered a change - this is being reported by the New York Times - ordering new restrictions on the sharing of confidential information with the White House.

- **Nonetheless, FBI director ordered change new restrictions sharing confidential information with White House.**

This sentence would probably receive a low grammaticality score (for example, 1 or 2) as it is difficult to understand. However it should receive a high score for importance (for example, 4 or 5) as it is possible to get the gist of the original. Now, consider the following compression of the same sentence:

- Nonetheless, FBI director Louis Freeh has today ordered a change - this is being reported by the New York Times - ordering new restrictions on the sharing of confidential information with the White House.

- **FBI director Louis Freeh has today ordered a change - this is being reported by the New York Times.**

You would give the compression a higher grammaticality score (for example, 4 or 5) but a low importance score (for example, 1 or 2). The compression preserves the least important information (the fact that the New York Times is reporting). On the other hand, if you were given the following compression:

- Nonetheless, FBI director Louis Freeh has today ordered a change - this is being reported by the New York Times - ordering new restrictions on the sharing of confidential information with the White House.

- **FBI director Louis Freeh ordered new restrictions on sharing confidential information with the White House.**

You would probably give it a high number for both grammaticality and importance (for example 4 or 5). Here, the compression is meaningful (grammatical), it produces a short version of the original sentence while retaining important pieces of information (i.e., the changes that have been ordered).

You will be presented with the original sentence first. Please read the original sentence and then click on the Show Compression link. Read the compression then make your judgements. The compression will always be presented in bold.

There are no 'correct' answers, so whatever number seems appropriate to you is a valid response. While you are deciding a number for a compression, try to ask the following questions:

- Does the compressed sentence preserve the most important bits of information from the original sentence?
- Is the compressed sentence easy to understand?
- Has the compressed sentence removed information you deem not to be very important to the original sentence?
- Does the compressed sentence seem fluent?

Use high numbers if the answer to the above questions is 'yes', low numbers if it is 'no', and intermediate numbers for sentences that are understandable, yet not entirely accurate or natural compressions of the original sentence. Try to make up your mind quickly, base your judgments on your first impressions.

The experiment will take approximately 20 minutes.

A.3 Document-level Evaluation Instructions

You will be given three summaries to read. Each summary has been automatically created by a computer. Some of these summaries will be more coherent (flow better) than others. You will be asked to give a readability rating to each summary on a scale of 1 to 7 (low to high). The rating should reflect how comprehensible the summary is. If a summary does not flow naturally, the topic changes at unexpected moments, or

the text is difficult to read then you should give a low rating. Higher ratings should be given to texts that readily flow and are understandable.

Once you have read the summary and rated its readability you will be asked a series of questions. For each question you can consult the summary for the answer. Some questions may not be answerable from the summary as the information may have been omitted. In this case you should indicate 'no answer' using the check box. Please do not attempt to guess the answers, only write the answer if you can determine it from the summary.

Questions will be displayed one at a time. Once you have answered a question you cannot go back and adjust the answer (as later questions may reveal more details). Please do not use your browser's back button.

The experiment will take approximately 15 to 20 minutes.

Examples

South Korea's current account surplus for the first six months fell 58% to $2.4 billion from $5.8 billion. The drop was attributed to the sharply reduced trade surplus. During the first half, exports grew by a mere 6.8% from a year earlier to $29 billion while imports surged 19% to $27 billion. The trade surplus shrank to $2 billion from $4.5 billion.

This summary flows well and is understandable thus it should receive a high readability score such as 6 or 7.

The drop was attributed by the bank. Current account for the six months of 1989 fell. The drop was by the bank to sharply reduced trade surplus. During the first exports grew a mere from a year earlier to $27 billion. As a result, shrank to $2 billion from $4.5 billion.

In contrast this summary is difficult to read. It does not flow. For example, we first learn about a drop but we do not know what the drop occurred in. It is also difficult to read in parts. Thus the readability score should be low, around 2 or 3 as some sentences are still readable.

Finally given the question:

Which country has seen a drop in their current account surplus?

From the first example the correct answer is South Korea. However the second example does not provide an answer to this question, thus it should be marked with no answer.

Appendix B

Documents and Question-Answer Pairs

In this appendix we provide the documents and question and answer pairs derived for the document compression evaluation. These five documents, along with the document in Chapter 4 (Figures 4.5 and 4.6) form the full evaluation set for the human document compression evaluation.

AHX.5: Full Text

A British woman may have found the body of her murdered 20-year-old son after a three-year hunt.

She was in a Canadian hospital last night suffering from exhaustion. Mrs Denise Allan, 42, of Sowerby, West Yorks, led a campaign to find out what happened to her son, Charles, after he vanished while trekking across Canada.

A body was found on Saturday in Okanagan lake 200 miles east of Vancouver. It was discovered in 130 feet of water in the exact spot where two anonymous letters written to Mrs Allan had said it would be. A post mortem examination will take place in Vancouver later today to confirm identification from dental records.

Mrs Allan was taken to nearby Kelowna General Hospital after the body was found. Her husband, Stuart, 52, said yesterday he had been in daily contact with her since she flew to Canada last month on the second pilgrimage to find her son. “She is suffering from exhaustion but otherwise fine,” he said. “I spoke to her last night and she is under strict orders to have complete rest. She is spending two days isolated from the world.”

Mr Allan, a garment manufacturer who married Denise five years ago, said she was

"deeply upset". He plans to fly to Canada on Wednesday to bring her home. "It's been building up to this. Everything has pointed towards a body being found."

"If it hadn't been for her courage and fortitude in going out there and taking on the role of investigator, private detective and motivator, those files would still be closed and the police would just have an unsolved case of a missing person."

Police now considered the case a murder inquiry and were appealing for any information that would lead to the killer. Mrs Allan's son disappeared in May, 1989, after a party during his back-packing trip across North America. Nothing was heard from him after he faxed a message home giving arrangements for his mother to meet him to celebrate her 40th birthday.

She flew to Canada to retrace his steps a month later but had to return after running out of money. After two years with no news, Mrs Allan sold her beauty salon in Bradford and raised a £20,000 loan to resume the search a month ago.

After she placed an advertisement in a Canadian newspaper, an anonymous handwritten letter was delivered to her motel. It said: "We were partying with your son on May 26 and this is the last time we could establish that he was alive. Two people knocked him out but he died. His body is in Lake Okanagan by the bridge."

An underwater search was launched. Mrs Allan used her own funds to hire local divers and a submersible camera crew at a cost of £500 per day.

Then last week a second note, in the same handwriting, informed Mrs Allan that the search was on the wrong side of the bridge. The body was found a day later.

Mr Allan, who likened his wife's campaign to that of the father of murdered British woman Julie Ward in Kenya, said: "It's most important we have something positive even though it is bad news."

AHX.5: Questions and Answers

Who is Mrs Allan looking for? (her son)
What happened to Mrs Allan's son? (he disappeared)
What gave her the location of her son's body? (anonymous letters)
Where did Mrs Allan fly to once she learnt her son was missing? (Canada)
What did Mrs Allan sell to resume the search? (her beauty salon)
After what actions did she receive the letter about her son? (placed adverts in the local papers)
What did Mrs Allan do to search for her son's body in the lake? (hired divers and

camera crew)
What is Mrs Allan's current phyisical condition? (suffering from exhaustion)

A3G.15: Full text

The Treasury is refusing to fund a further phase of the city technology colleges. Plans for the creation of 20 CTCs by 1990 were announced by Kenneth Baker, the then Secretary of State for Education and Science, at the Conservative Party conference in October 1986. They were to be a new form of secondary school — "beacons of excellence" — funded mainly by industry, and would concentrate on science and technology. But the Government has been severely embarrassed by the burgeoning cost of the programme.

Mr Baker had said that industrial sponsors would pay "all or a substantial part" of the capital costs. The lack of sponsors has meant the taxpayer has had to foot more of the bill. The Department of Education and Science said yesterday that the Government had spent £19.7m on CTCs and there was a further planned expenditure over the next three years of £106.2m. So far industry had contributed £44m.

Sir Cyril Taylor, the Government's adviser on CTCs, who had earlier been successful in persuading Mr Baker to commit more government funds to the 20 schools, had been hoping to get more money for a new round of schools. But sources have confirmed that this has been ruled out by the Treasury in the current round of public expenditure talks. But yesterday, Susan Fey, of the CTC Trust, said , "We were only ever given a target of 20. We have never been to Treasury to ask for funds for more than 20. Of course there have been discussions between the Trust and civil servants but nothing has gone to Pesc (the expenditure talks)."

Although Sir Cyril had spoken in January 1988 about "hundreds" of CTCs, these would be funded by local education authorities. Jack Straw, Labour's shadow education secretary said yesterday at the Labour Party conference that the news to abandon further CTCs marked "the death of an expensive corrupt fiasco, which has already cost the taxpayer millions". "But so rotten has the policy proved that not even a 'taxpayer bail-out' could save it. Indeed even Britain's blue chip businesses boycotted the scheme despite being put under intense personal and political pressure," he added.

Mr Straw said this involved "veiled threats if they did not cough up and clear promises of honours if they did". He added: "What is so appalling is that millions of pounds which should have been invested in children 's education has been squan-

dered in pursuit of electoral advantage." He will renew calls on the public accounts committee to conduct a full investigation into "this disgraceful waste of the funds so vital to the education of our children". He is also writing to John McGregor, the Education Secretary, urging him not only to abandon the idea of additional CTCs but to hand over those in the pipeline to local authorities.

A3G.15: Questions and Answers

Who are CTCs costing money? (tax payers OR the treasury)
Who was meant to pay for CTCs? (industry sponsors)
What is Jack Straw calling for? (an investigation)
What areas of education would CTCs concentrate on? (science and technology)
How much money has the government spent on CTCs? (£19.7 million)
How much further expenditure is planned? (£106.2 million)
How many CTCs are the government planning on building? (20)

A59.27: Full text

A Policeman was yesterday jailed for seven years for raping an 18-year-old woman in his marked patrol car while he was on duty and in uniform. Sentencing Constable Peter Anderson, 41, Mr Justice Jowitt told him he had done "great damage to the trust in police".

Anderson, married with two children, attacked the woman in a deserted allotment, after agreeing to give her and a boyfriend a lift home from a discotheque. He first dropped the man off and then drove to the allotment. He threatened her by forcing his truncheon under her chin and then raped her. She said he only refrained from inserting his truncheon into her, after she begged him not to. Afterwards he told her not to report the incident because he could have her "nicked" for soliciting. She did not report it because she did not think she would be believed.

Police investigated after an anonymous report. The victim, now 20, said she had drunk nine or 10 Pernods with blackcurrant and was merry, but knew what she was doing and saying. She said she tried to push him off, but he was too forceful. Mr Justice Jowitt told Anderson: "I accept that you were not on the prowl looking for a victim and that it was by chance that this young lady got into your car. I accept that there was no great degree of violence used by you. But you took her against her will in your car to the place where this rape happened, and one of the very disturbing and

serious features of this case is the way you abused your position as a police officer in uniform on duty." The judge added: " This girl plainly trusted herself in your company, as she was entitled to. The public expect that they can treat the police with confidence. You did great damage to that trust in the police when you behaved in this way."

Anderson, who had pleaded not guilty and claimed the woman had handed him "sex on a plate", was convicted by a 10-2 majority of raping the woman on 4 April, last year. He claimed she had instigated the intercourse by first, and without invitation, performing oral sex on him. He said he had only offered to use the truncheon as a sex aid but desisted when she shook her head.

Jean Southworth, Qc, in mitigation, said: "This was not a case of him taking away the virginity of this young woman. He has lost his pension rights and the personal affection of those dear to him and also, when a police officer goes to prison, he often carries an extra load for his misdoings."

A59.27: Questions and Answers

What crime has the policeman committed? (rape)
What has been damaged as a result of the rape? (trust in the police)
What is the policeman's defence? (She instigated the incident OR she handed sex on a plate)
Where did the incident take place? (in an allotment)
What is the main punishment the policeman received? (a jail sentence)
What benefits did the policeman lose? (pension rights)

A96.17: Full Text

A Turkish print worker alleged yesterday that a Harley Street doctor paid £2,500 for him to donate a kidney to a patient whom he believed was a fellow countryman.

Mr Ferhat Usta, a Muslim, said he realised minutes before the operation that his kidney was going to a Briton. "I suddenly got out of bed half naked. I realised I was being deceived," he told the General Medical Council's professional conduct committee.

Mr Usta, aged 34, who lives with his wife, mother and three daughters in a shack in an Istanbul shanty town, described how he came to London last year, attracted by a newspaper advertisement offering money to kidney donors. He wanted to raise £2,000 to treat one of his children who suffered from a tubercular hip infection.

Mr Usta was examined by Dr Raymond Crockett, a Harley Street physician specialising in kidney disease. Dr Crockett, Mr Michael Bewick, a leading kidney transplant surgeon, and Mr Michael Joyce, a urologist at Guy's Hospital, deny professional misconduct over their involvement in transplanting kidneys from four living Turks, all of whom were paid for the organs.

Mr Usta recalled how two brothers, described as "kidney brokers", handed him £2,500 in cash on the night before the operation in July 1988. Speaking through an interpreter, Mr Usta said: "As far as I can figure it out, one day before the operation the cheque was given by Dr Crockett, it was changed and the money given to me that night."

On Monday, the first day of the hearing, Mr Roger Henderson Qc, for the Gmc, said Dr Crockett's notes included a bill for £20,000 for a Mr B, described as a Briton living in Israel who was suffering from a disease affecting his kidneys.

Mr Usta said he had come to London under the impression that his kidney was to be donated to one of the "broker" brothers, Ata Nur Kuntar. He had said to Mr Kuntar: "You could have told me the truth from the very beginning. Because I am a very poor man you made me accept a figure like six million lire (£2,500). An Englishman, if he is going to have an operation in a hospital like that, I am sure he would have at least £5,000 in his pocket. I told him that I wanted £5,000 from him. He then accepted this and he told me he was going to pay me the other £2,500 in Turkey in Turkish money." He said he never received the extra money.

The hearing continues today.

A96.17: Questions and Answers

What organ is being donated? (kidney)
How much was Mr Usta paid? (£2,500)
Who are the kidneys going to? (Britons)
Why did Mr Usta think he was deceived? (he believed the kidney was going to a Turk or fellow countryman)
Why did Mr Usta agree to sell his kidney? (to treat his child)

AAC.10: Full Text

The Ford Motor Company faces an all-out strike next month following the 4-1 ballot rejection yesterday of a two-year pay deal by its 32,000 hourly paid workers.

They will be pressing for a settlement of more than 10 per cent in what will be the most severe test of the Government's inflation policy. The two-year deal amounted to 9.5 per cent for the first year and inflation plus 2.5 per cent for the second. Improvements in certain allowances were made, described as divisive by the unions, but the company has refused to compromise on a reduction in the shorter working week.

Ford dismissed an immediate meeting with the unions but did not rule out talks after Christmas. It said that a strike would be damaging to the company and to its staff.

Production closed down at Ford last night for the Christmas period. Plants will open again on January 2.

Staff voted 20,343 in favour of action, with 4,727 against. The electricians are holding a postal ballot with the results announced after Christmas. The unions said that they were looking for the second week in January to begin an all-out stoppage.

Mr Jimmy Airlie, secretary of the Ford union side, said: "We expected to get a favourable majority. This exceeded even our expectations." Mr Jack Adams, chairman of the union side, said that action would have to take place within a 28-day period from yesterday's anouncement or it would be ruled out of order. He thought the big strike vote was partly due to Ford's record profits last year of £673 millions.

The company is likely to be affected by a series of unofficial stoppages before any official action begins, as it was in the lead up to negotiations when Ford's final offer was rejected last month.

AAC.10: Questions and Answers

What is Ford facing? (a strike)
What caused the strike? (the rejection of a pay deal)
What are the unions pressing for? (more than 10% increase in pay)
Is Ford willing to hold talks with the unions? (they haven't ruled talks out)
What effect will the strike have? (damaging to company and staff)

Bibliography

Bangalore, Srinivas, Owen Rambow, and Steve Whittaker. 2000. Evaluation metrics for generation. In *Proceedings of the International Natural Language Generation Conference (INLG-2000)*.

Barzilay, R. and M. Elhadad. 1997. Using lexical chains for text summarization. In *Proceedings of the Intelligent Scalable Text Summarization Workshop (ISTS), ACL-97*.

Barzilay, Regina and Mirella Lapata. 2006. Aggregation via set partitioning for natural language generation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*. New York, NY, pages 359–366.

Barzilay, Regina, Kathleen R. McKeown, and Michael Elhadad. 1999. Information fusion in the context of multi-document summarization. In *Proceedings of the 37th conference on Association for Computational Linguistics*. Association for Computational Linguistics, pages 550–557.

Bramsen, Philip, Pawan Deshpande, Yoong Keok Lee, and Regina Barzilay. 2006. Inducing temporal graphs. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Sydney, Australia, pages 189–198.

Briscoe, E. J. and J. Carroll. 2002. Robust accurate statistical annotation of general text. In *Proceedings of the Third International Conference on Language Resources and Evaluation (LREC 2002)*. pages 1499–1504.

Brown, Peter F., Vincent J. Della Pietra, Stephen A. Della Pietra, and Robert L. Mercer. 1993. The mathematics of statistical machine translation: parameter estimation. *Computational Linguistics* 19(2):263–311.

Charniak, Eugene. 2000. A maximum-entropy-inspired parser. In *Proceedings of the 1st North American Chapter of the Association for Computational Linguistics*. Seattle, WA, pages 132–139.

Charniak, Eugene. 2001. Immediate-head parsing for language models. In *Meeting of the Association for Computational Linguistics*. pages 116–123.

Church, Kenneth Ward. 1988. A stochastic parts program and noun phrase parser for unrestricted text. In *Proceedings of the second conference on Applied natural language processing*. Association for Computational Linguistics, Morristown, NJ, USA, pages 136–143.

Clarke, James and Mirella Lapata. 2006a. Constraint-based sentence compression: An integer programming approach. In *Proceedings of the COLING/ACL 2006 Main Conference Poster Sessions*. Association for Computational Linguistics, Sydney, Australia, pages 144–151.

Clarke, James and Mirella Lapata. 2006b. Models for sentence compression: A comparison across domains, training requirements and evaluation measures. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Sydney, Australia, pages 377–384.

Clarke, James and Mirella Lapata. 2007. Modelling compression with discourse constraints. In *Proceedings of the 2007 Joint Conference on Empirical Methods in Natural Language Processing and Computational Natural Language Learning (EMNLP-CoNLL)*. Prague, pages 1–11.

Clarke, James and Mirella Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research (JAIR)* 31:399–429.

Clarkson, Philip and Ronald Rosenfeld. 1997. Statistical language modeling using the CMU–cambridge toolkit. In *Proc. Eurospeech '97*. Rhodes, Greece, pages 2707–2710.

Collins, Michael. 1997. Three generative, lexicalised models for statistical parsing. In *Proceedings of the 35th annual meeting on Association for Computational Linguistics*. Association for Computational Linguistics, Morristown, NJ, USA, pages 16–23.

Collins, Michael. 2000. Discriminative reranking for natural language parsing. In *ICML '00: Proceedings of the Seventeenth International Conference on Machine Learning*. San Francisco, CA, USA, pages 175–182.

Cormen, T. H., C. E. Leiserson, R. L. Rivest, and C. Stein. 2000. *Introduction to Algorithms, 2nd edition*. MIT Press, McGraw-Hill Book Company.

Corston-Oliver, Simon. 2001. Text Compaction for Display on Very Small Screens. In Jade Goldstein and Chin-Yew Lin, editors, *Proceedings of the Workshop on Automatic Summarization at the 2nd Meeting of the North American Chapter of the Association for Computational Linguistics*. pages 89–98.

Crammer, Koby and Yoram Singer. 2003. Ultraconservative online algorithms for multiclass problems. *Journal of Machine Learning Research* 3:951–991.

Dantzig, George B. 1963. *Linear Programming and Extensions*. Princeton University Press, Princeton, N.J.

Daumé III, Hal and Daniel Marcu. 2002. A noisy-channel model for document compression. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics (ACL – 2002)*. Philadelphia, PA, pages 449 – 456.

Denis, Pascal and Jason Baldridge. 2007. Joint determination of anaphoricity and coreference resolution using integer programming. In *Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics; Proceedings of the Main Conference*. Rochester, NY, pages 236–243.

Dras, Mark. 1999. *Tree Adjoining Grammar and the Reluctant Paraphrasing of Text*. Ph.D. thesis, Macquarie University.

Eisner, Jason M. 1996. Three new probabilistic models for dependency parsing: an exploration. In *Proceedings of the 16th conference on Computational Linguistics*. Association for Computational Linguistics, Morristown, NJ, USA, pages 340–345.

Endres-Niggemeyer, Brigitte. 1998. *Summarising Information*. Springer, Berlin.

Frank, Anette. 1999. From parallel grammar development towards machine translation. In *Proceedings of the MT Summit VII. MT in the Great Translation Era*. Kent Ridge Digital Labs, Singapore, pages 134–142.

Galley, Michel and Kathleen McKeown. 2003. Improving word sense disambiguation in lexical chaining. In *Proceedings of 18th International Joint Conference on Artificial Intelligence (IJCAI–03)*. pages 1486–1488.

Galley, Michel and Kathleen McKeown. 2007. Lexicalized markov grammars for sentence compression. In *In Proceedings of the North American Chapter of the Association for Computational Linguistics (NAACL-HLT–2007)*. Rochester, NY.

Germann, Ulrich, Michael Jahr, Kevin Knight, Daniel Marcu, and Kenji Yamada. 2004. Fast and optimal decoding for machine translation. *Artificial Intelligence* 154(1-2):127–143.

Gomory, R. E. 1960. Solving linear programming problems in integers. In R. Bellman and M. Hall, editors, *Combinatorial analysis, Proceedings of Symposia in Applied Mathematics*. Providence, RI, volume 10.

Grefenstette, Gregory. 1998. Producing Intelligent Telegraphic Text Reduction to Provide an Audio Scanning Service for the Blind. In Eduard Hovy and Dragomir R. Radev, editors, *Proceedings of the AAAI Symposium on Intelligent Text Summarization*. Stanford, CA, pages 111–117.

Grishman, Ralph, Catherine Macleod, and Adam Meyers. 1994. Comlex syntax: building a computational lexicon. In *Proceedings of the 15th conference on Computational linguistics*. Association for Computational Linguistics, Morristown, NJ, USA, pages 268–272.

Grosz, Barbara J., Scott Weinstein, and Aravind K. Joshi. 1995. Centering: a framework for modeling the local coherence of discourse. *Computational Linguistics* 21(2):203–225.

Halliday, M. A. K. 1985. *Spoken and Written Language*. Oxford University Press.

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

Hooker, John N. 2002. Logic, optimization, and constraint programming. *INFORMS Journal on Computing* 14(4):295–321.

Hori, Chiori and Sadaoki Furui. 2003. A new approach to automatic speech summarization. *IEEE Transactions on Multimedia* 5(3):368–378.

Hori, Chiori and Sadaoki Furui. 2004. Speech summarization: an approach through word extraction and a method for evaluation. *IEICE Transactions on Information and Systems* E87-D(1):15–25.

Hori, Chiori, Sadaoki Furui, Rob Malkin, Hua Yu, and Alex Waibel. 2003. A statistical approach for automatic speech summarization. *EURASIP Journal on Applied Signal Processing* pages 128–139.

Hori, Chiori, Tsutomu Hirao, and Hideki Isozaki. 2004. Evaluation measures considering sentence concatenation for automatic summarization by sentence or word extraction. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*. Association for Computational Linguistics, Barcelona, Spain, pages 82–88.

Jelinek, Frederick. 1997. *Statistical methods for speech recognition*. MIT Press, Cambridge, MA, USA.

Jing, Hongyan. 2000. Sentence reduction for automatic text summarization. In *Proceedings of the 6th Applied Natural Language Processing Conference*. Seattle,WA, pages 310–315.

Jing, Hongyan and Kathleen McKeown. 1998. Combining multiple, large-scale resources in a reusable lexicon for natural language generation. In *Proceedings of the 17th international conference on Computational linguistics*. Association for Computational Linguistics, Morristown, NJ, USA, pages 607–613.

Jing, Hongyan and Kathleen McKeown. 1999. The decomposition of human-written summary sentences. In *Research and Development in Information Retrieval*. pages 129–136.

Johnson, Mark. 1998. Pcfg models of linguistic tree representations. *Computational Linguistics* 24(4):613–632.

Knight, Kevin and Daniel Marcu. 2002. Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artif. Intell.* 139(1):91–107.

Lafferty, John, Andrew McCallum, and Fernando Pereira. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*. Morgan Kaufmann, San Francisco, CA, pages 282–289.

Land, A. H. and A. G. Doig. 1960. An automatic method for solving discrete programming problems. *Econometrica* 28:497–520.

Lapata, Mirella. 2006. Automatic evaluation of information ordering. *Computational Linguistics* 32(4):471–484.

Levin, Beth. 1993. *English Verb Classes and Alternations: a preliminary investigation*. University of Chicago Press, Chicago and London.

Lin, Chin-Yew. 2003. Improving summarization performance by sentence compression — a pilot study. In *Proceedings of the 6th International Workshop on Information Retrieval with Asian Languages*. Sapporo, Japan, pages 1–8.

Lin, Dekang. 2001. LaTaT: Language and text analysis tools. In *Proceedings of the 1st Human Language Technology Conference*. San Francisco, CA, pages 222–227.

Lustig, Irvin J. and Jean-François Puget. 2001. Program does not equal program: Constraint programming and its relationship to mathematical programming. *Interfaces* 31(6):29–53.

Mani, Inderjeet. 2001. *Automatic Summarization*. John Benjamins Publishing Company, Amsterdam/Philadephia.

Mani, Inderjeet, Thérèse Firmin, David House, Gary Klein, Beth Sundheim, and Lynette Hirschman. 2002a. The TIPSTER SUMMAC Text Summarization Evaluation. In *Natural Language Engineering*. volume 8, pages 43–68.

Mani, Inderjeet, Barbara Gates, and Eric Bloedorn. 1999. Improving summaries by revising them. In *Proceedings of the 37th annual meeting of the Association for Computational Linguistics on Computational Linguistics*. Association for Computational Linguistics, Morristown, NJ, USA, pages 558–565.

Mani, Inderjeet, Gary Klein, David House, Lynette Hirschman, Therese Firmin, and Beth Sundheim. 2002b. SUMMAC: A text summarization evaluation. *Natural Language Engineering* 8(1):43–68.

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

Marciniak, Tomasz and Michael Strube. 2005. Beyond the pipeline: Discrete optimization in NLP. In *Proceedings of the Ninth Conference on Computational Natural Language Learning*, Ann Arbor, MI, 29-30 June, 2005. pages 136–143.

Marcu, Daniel. 2000. *The Theory and Practice of Discourse Parsing and Summarization*. MIT Press, Cambridge, MA, USA.

McCord, Michael C. 1989. Slot grammar: A system for simpler construction of practical natural language grammars. In *Natural Language and Logic*. pages 118–145.

McDonald, Ryan. 2006. Discriminative sentence compression with soft syntactic constraints. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*. Trento, Italy.

McDonald, Ryan. 2007. A study of global inference algorithms in multi-document summarization. In *European Conference on Information Retieval*.

McDonald, Ryan, Koby Crammer, and Fernando Pereira. 2005a. Flexible text segmentation with structured multilabel classification. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Vancouver, British Columbia, Canada, pages 987–994.

McDonald, Ryan, Koby Crammer, and Fernando Pereira. 2005b. Online large-margin training of dependency parsers. In *43rd Annual Meeting of the Association for Computational Linguistics*. Ann Arbor, MI, USA, pages 91–98.

McDonald, Ryan, Fernando Pereira, Kiril Ribarov, and Jan Hajic. 2005c. Non-projective dependency parsing using spanning tree algorithms. In *Proceedings of Human Language Technology Conference and Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Vancouver, British Columbia, Canada, pages 523–530.

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

Miltsakaki, Eleni and Karen Kukich. 2000. The role of centering theory's rough-shift in the teaching and evaluation of writing skills. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics (ACL–2000)*. pages 408–415.

Morris, A., G. Kasper, and D. Adams. 1992. The effects and limitations of automated text condensing on reading comprehension performance. *Information Systems Research* 3(1):17–35.

Morris, Jane and Graeme Hirst. 1991. Lexical cohesion computed by thesaural relations as an indicator of the structure of text. *Computational Linguistics* 17(1):21–48.

Nemhauser, George L. and Laurence A. Wolsey. 1988. *Integer and Combinatorial Optimization*. Wiley-Interscience series in discrete mathematicals and opitmization. Wiley, New York, NY, USA.

Nguyen, Minh Le, Susumu Horiguchi, Akira Shimazu, and Bao Tu Ho. 2004a. Example-based sentence reduction using the hidden markov model. *ACM Transactions on Asian Language Information Processing (TALIP)* 3(2):146–158.

Nguyen, Minh Le, Akira Shimazu, Susumu Horiguchi, Tu Bao Ho, and Masaru Fukushi. 2004b. Probabilistic sentence reduction using support vector machines. In *Proceedings of the 20th COLING*. Geneva, Switzerland, pages 743–749.

Poesio, Massimo, Rosemary Stevenson, Barbara Di Eugenio, and Janet Hitzeman. 2004. Centering: a parametric theory and its instantiations. *Computational Linguistics* 30(3):309–363.

Press, William H., Saul A. Teukolsky, William T. Vetterling, and Brian P. Flannery. 1992. *Numerical Recipes in C: The Art of Scientific Computing*. Cambridge University Press, New York, NY, USA.

Punyakanok, Vasin, Dan Roth, Wen-tau Yih, and Dav Zimak. 2004. Semantic role labeling via integer linear programming inference. In *Proceedings of the International Conference on Computational Linguistics (COLING)*. Geneva, Switzerland, pages 1346–1352.

Quinlan, J. R. 1993. *C4.5 – Programs for Machine Learning*. The Morgan Kaufmann series in machine learning. Morgan Kaufman Publishers.

Riedel, Sebastian and James Clarke. 2006. Incremental integer linear programming for non-projective dependency parsing. In *Proceedings of the 2006 Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, Sydney, Australia, pages 129–137.

Riezler, Stefan, Tracy King, Ronald Kaplan, Richard Crouch, John T. Maxwell III, and Mark Johnson. 2002. Parsing the wall street journal using a lexical-functional grammar and discriminative estimation techniques. In *Proceedings of the 40th Annual Conference of the Association for Computational Linguistics (ACL-02)*. Philadelphia, PA, pages 271–278.

Riezler, Stefan, Tracy H. King, Richard Crouch, and Annie Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Human Language Technology Conference and the 3rd Meeting of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'03)*. pages 118–125.

Roark, Brian. 2001. Probabilistic top-down parsing and language modeling. *Computational Linguistics* 27(2):249–276.

Rosenblatt, Frank. 1988. The perceptron: a probabilistic model for information storage and organization in the brain. *Neurocomputing: foundations of research* pages 89–114.

Roth, D. and W. Yih. 2005. Integer linear programming inference for conditional random fields. In *Proceedings of the International Conference on Machine Learning (ICML)*. pages 737–744.

Roth, Dan and Wen-tau Yih. 2004. A linear programming formulation for global inference in natural language tasks. In Hwee Tou Ng and Ellen Riloff, editors, *Proceedings of the Annual Conference on Computational Natural Language Learning (CoNLL)*. Association for Computational Linguistics, pages 1–8.

Salton, Gerald, editor. 1988. *Automatic text processing*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA.

Sang, Erik F. Tjong Kim and Fien De Meulder. 2003. Introduction to the conll-2003 shared task: language-independent named entity recognition. In *Proceedings of the seventh conference on Natural language learning at HLT-NAACL 2003*. Association for Computational Linguistics, Morristown, NJ, USA, pages 142–147.

Sarawagi, Sunita and William W. Cohen. 2004. Semi-markov conditional random fields for information extraction. In *Proceedings of Neural Information Processing Systems*.

Shen, Libin, Anoop Sarkar, and Franz Josef Och. 2004. Discriminative reranking for machine translation. In Daniel Marcu Susan Dumais and Salim Roukos, editors, *HLT-NAACL 2004: Main Proceedings*. Boston, MA, USA, pages 177–184.

Sparck-Jones, Karen. 1998. Automatic summarising: factors and directions. In *Advances in Automatic Text Summarization*, MIT Press, pages 1–14.

Sparck-Jones, Karen, Julia R. Galliers, and J. R. Galliers. 1996. *Evaluating Natural Language Processing Systems: An Analysis and Review*. Springer-Verlag New York, Inc., Secaucus, NJ, USA.

Tetreault, Joel R. 2001. A corpus-based evaluation of centering and pronoun resolution. *Computational Linguistics* 27(4):507–520.

Teufel, Simone and Marc Moens. 2002. Summarizing scientific articles: experiments with relevance and rhetorical status. *Comput. Linguist.* 28(4):409–445.

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

Vandeghinste, Vincent and Yi Pan. 2004. Sentence compression for automated subtitling: A hybrid approach. In Stan Szpakowicz Marie-Francine Moens, editor, *Text Summarization Branches Out: Proceedings of the ACL-04 Workshop*. Association for Computational Linguistics, Barcelona, Spain, pages 89–95.

Vanderbei, Robert J. 2001. *Linear Programming: Foundations and Extensions*. Kluwer Academic Publishers, Boston, 2nd edition.

Vapnik, Vladimir N. 1998. *Statistical Learning Theory*. Wiley-Interscience.

Walker, Marilyn, Arivind Joshi, and Ellen Prince. 1998. Centering in naturally occurring discourse: An overview. In *Centering Theory in Discourse*, Oxford University Press, Oxford, pages 1–28.

Weiss, Sholom M. and Casimir A. Kulikowski. 1991. *Computer systems that learn: classification and prediction methods from statistics, neural nets, machine learning, and expert systems*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

Williams, H. Paul. 1999. *Model Building in Mathematical Programming.* Wiley, 4th edition.

Williams, H. Paul and John M. Wilson. 1998. Connections between integer linear programming and constraint logic programming-an overview and introduction to the cluster of articles. *INFORMS Journal on Computing* 10(3):261–264.

Winston, Wayne L. and Munirpallam Venkataramanan. 2003. *Introduction to Mathematical Programming - Applications and Algorithms.* Duxbury, 4th edition.

Witbrock, Michael J. and Vibhu O. Mittal. 1999. Ultra-summarization: A statistical approach to generating highly condensed non-extractive summaries (poster abstract). In *Research and Development in Information Retrieval*. pages 315–316.

Zelenko, Dmitry, Chinatsu Aone, and Anthony Richardella. 2003. Kernel methods for relation extraction. *J. Mach. Learn. Res.* 3:1083–1106.

Zhang, K. and D. Shasha. 1989. Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal on Computing* 18(6):1245–1262.