An Abstractive Approach to Sentence Compression

TREVOR COHN, University of Sheffield
MIRELLA LAPATA, University of Edinburgh

In this article we generalize the sentence compression task. Rather than simply shorten a sentence by deleting words or constituents, as in previous work, we rewrite it using additional operations such as substitution, reordering, and insertion. We present an experimental study showing that humans can naturally create abstractive sentences using a variety of rewrite operations, not just deletion. We next create a new corpus that is suited to the abstractive compression task and formulate a discriminative tree-to-tree transduction model that can account for structural and lexical mismatches. The model incorporates a grammar extraction method, uses a language model for coherent output, and can be easily tuned to a wide range of compression-specific loss functions.

Categories and Subject Descriptors: I.I.2 [**Artificial Intelligence**]: Natural Language Processing

General Terms: Experimentation

Additional Key Words and Phrases: Language generation, language models, machine translation, sentence compression, paraphrases, transduction, synchronous grammars

ACM Reference Format:
Cohn, T. and Lapata, M. 2013. An abstractive approach to sentence compression. ACM Trans. Intell. Syst. Technol. 4, 3, Article 41 (June 2013), 35 pages.
DOI: http://dx.doi.org/10.1145/2483669.2483674

1. INTRODUCTION

Recent years have witnessed increasing interest in text rewriting. The problem of how to best reformulate natural language text applies to many applications ranging from summarization [Barzilay and McKeown 2005] to question answering [Lin and Pantel 2001] and machine translation [Callison-Burch 2007]. Text rewriting is often used as an umbrella term for different tasks. Examples include modeling paraphrase relationships between sentences or phrases [Barzilay 2003], simplifying text by identifying utterances in a document that pose reading difficulty and substituting them with simpler alternatives [Chandrasekar and Srinivas 1996], and rendering sentences shorter with minimal information loss while preserving their grammaticality [Jing 2000]. The latter sentence compression task has found use in summarization [Lin 2003; Martins and Smith 2009; Zajic et al. 2007], headline generation [Dorr et al. 2003], the display of text on small-screen devices such as PDAs [Corston-Oliver 2001], the generation of subtitles from spoken transcripts [Vandeghinste and Pan 2004], and as a reading aid for the blind [Grefenstette 1998].

The authors acknowledge the support of EPSRC (grants GR/T04540/01 and GR/T04557/01).
Authors' addresses: T. Cohn (corresponding author), Department of Computer Science, University of Sheffield, UK; email: t.cohn@dcs.shef.ac.uk; M. Lapata, School of Informatics, University of Edinburgh, UK.

DOI: http://dx.doi.org/10.1145/2483669.2483674

Most prior work has focused on a specific instantiation of sentence compression, namely word deletion. Given an input *source* sentence of words, $w_1, w_2 \ldots w_n$, a *target* compression is formed by dropping any subset of these words [Knight and Marcu 2002]. The simplification renders the task computationally feasible, allowing efficient decoding using dynamic programming [Knight and Marcu 2002; Turner and Charniak 2005; McDonald 2006]. Furthermore, constraining the problem to word deletion affords substantial modeling flexibility. Indeed, a variety of models have been successfully developed for this task ranging from instantiations of the noisy channel model [Knight and Marcu 2002; Galley and McKeown 2007; Turner and Charniak 2005], to large margin learning [McDonald 2006; Cohn and Lapata 2009], and integer linear programming [Clarke 2008; Martins and Smith 2009]. However, the simplification also renders the task somewhat artificial. There are many rewrite operations that could compress a sentence besides deletion, including reordering, substitution, and insertion. In fact, professional abstractors tend to use these operations to transform selected sentences from an article into the corresponding summary sentences [Jing 2000].

In this article we consider sentence compression from a more general perspective and generate sentence-level *abstracts* rather than *extracts*.[1] In this framework, the goal is to find a summary of the original sentence which is grammatical and conveys the most important information without necessarily using the same words in the same order. Our task is related to, but different from, paraphrase extraction [Barzilay 2003]. We must not only have access to paraphrases (i.e., rewrite rules), but also be able to combine them to generate new text, while attempting to produce a shorter resulting string. More similar is the approach of Quirk et al. [2004] who present an end-to-end paraphrasing system inspired by phrase-based machine translation that can both acquire paraphrases and use them to generate new strings. However, their approach was limited to only lexical substitution—no reordering takes place—and is lacking the compression objective. A variety of models have been proposed for sentence compression, however, they are specifically designed with word deletion in mind and are thus unable to model consistent syntactic effects such as reordering, changes in nonterminal categories, and lexical substitution.

Once we move away from extractive sentence compression we are faced with two problems. First, we must validate that abstractive sentence compression is a meaningful task. Can humans do it and if yes, what kinds of rewrite operations do they employ? For instance, they may compress sentences mostly by deletion in which case there isn't much need for an abstractive compression model. Our second problem concerns the modeling task itself. Ideally, our learning framework should handle structural mismatches and complex rewriting operations. A related issue concerns finding appropriate training data for such a model. Although some compression corpora are available (e.g., Clarke and Lapata [2008]), they only provide examples based on word deletion. And existing paraphrase corpora (such as the Multiple-Translation Chinese and Arabic corpora[2]) do not normally contain compressions.

In what follows, we first demonstrate that abstractive compression is a valid task by conducting an experimental study where participants are asked to freely compress sentences. We show that participants use a variety of rewrite operations in addition to deletion. We also find that abstractive compressions have a lower compression rate[3]

[1]Herein we refer to the general task as *abstractive* sentence compression, and the deletion-only approach as *extractive* sentence compression.

[2]Available from the LDC, catalog numbers LDC2002T01, LDC2003T17, LDC2004T07, LDC2006T04, LDC2003T18, and LDC2005T05.

[3]The term refers to the percentage of words retained from the source sentence in the compression. A low compression rate means that a large percentage of words were dropped.

in comparison to extractive compressions. Based on this experimental study, we create a new corpus for abstractive compression in order to obtain useful data for modeling purposes. We then present a tree-to-tree transducer capable of transforming an input parse tree into a compressed parse tree. Our approach is based on Synchronous Tree Substitution Grammar (STSG) [Shieber and Schabes 1990; Eisner 2003], a formalism that can account for structural mismatches, and is trained discriminatively. Specifically, we show how the model of Cohn and Lapata [2009] can be applied to our abstractive task and present a novel tree-to-tree grammar extraction method which acquires paraphrases from bilingual corpora. We also develop a number of loss functions suited to the abstractive compression task.[4]

The remainder of this article is structured as follows. Section 2 provides an overview of related work. Sections 3 and 4 detail our experimental study and corpus collection, respectively. Section 5 presents the compression model we employ in our experiments and Section 6 discusses our evaluation framework. We present our results in Section 7 and conclude the article with discussion of future work.

2. RELATED WORK

Sentence compression has been extensively studied across different modeling paradigms, most of which are based on supervised learning. Compression models are typically trained on a parallel compression corpus and decide which words and constituents to retain or delete. The retained words are then taken in order to form the compressed output. Relatively few approaches dispense with the parallel corpus and generate compressions in an unsupervised manner using either a scoring function [Hori and Furui 2004; Clarke and Lapata 2008] or compression rules that are approximated from a nonparallel corpus such as the Penn Treebank [Turner and Charniak 2005].

Most generative compression models are instantiations of the noisy channel model. The key idea here is to treat sentence compression as a translation task within the same language. To give a specific example, Knight and Marcu's [2002] seminal model consists of two components, a language model $P(y)$ whose role is to ensure that the compression output is grammatical and a channel model $P(x|y)$ capturing the probability that the source sentence x is an expansion of the target compression y. Their decoding algorithm searches for the compression y which maximizes $P(y)P(x|y)$. Their channel model is a stochastic Synchronous Context-Free Grammar (SCFG) [Aho and Ullman 1969], which when ground with a source string is equivalent to generating from a CFG. Knight and Marcu learn the grammar rules from a parallel corpus of long sentences and their corresponding compressions. The rules have weights that are estimated using maximum likelihood. Improvements upon this model include Markovization [Galley and McKeown 2007] and the addition of specialized rules to model syntactically complex expressions [Turner and Charniak 2005]. Discriminative approaches include decision-tree learning [Knight and Marcu 2002], maximum entropy [Riezler et al. 2003], support vector machines [Nguyen et al. 2004], large margin learning [McDonald 2006; Cohn and Lapata 2009], and minimum classification error learning [Hirao et al. 2009].

Despite differences in formulation, all the preceding models are restricted to word deletion and are therefore not readily applicable to the more challenging task of abstractive sentence compression. A common assumption underlying previous work is that the tree structures representing the source sentences and their target compression are *isomorphic*, that is, there exists an edge-preserving bijection between the nodes in the two trees. Although this assumption is mostly justified for deletion-based

[4]A preliminary version of this work was published in Cohn and Lapata [2008]. The current article contains a more detailed description of our approach, presents several novel experiments, and a comprehensive error analysis.

(extractive) compression, it rarely holds for abstractive sentence compression and text rewriting in general. A notable exception is Galley and McKeown [2007] who learn a sentence compression model from a corpus containing both substitutions and deletions. Their main motivation is to obtain improved SCFG estimates by exploiting larger amounts of data than previous approaches. Their model is, however, limited to word deletion. As it only has a notion of binary variables for keeping versus discarding nodes in the source tree, it cannot perform substitutions or any other rewrite operations such as reordering.

The literature is rife with methods that extract paraphrase rules [Lin and Pantel 2001; Barzilay and McKeown 2001; Barzilay and Lee 2003; Pang et al. 2003; Barzilay and Elhadad 2003; Bannard and Callison-Burch 2005; Callison-Burch 2008; Bhagat and Ravichandran 2008], that could in theory be used to reformulate a sentence in a more concise manner. However, relatively little emphasis has been placed on the rewriting task itself, that is, on algorithms that use paraphrases to generate a target sentence. A notable exception is Quirk et al. [2004] who model paraphrase generation as a monolingual machine translation problem. Similar to Knight and Marcu [2002], their approach consists of a language model and a translation model that captures the probability of a source sentence given its target paraphrase. The translation model is phrase-based[5] [Koehn et al. 2003], but their approach is limited to monotone translation in the paraphrase generation algorithm. This means that it can capture lexical substitutions but no phrase reorderings or complex structural mismatches.[6] Zhao et al. [2009] extend this approach by using multiple phrase tables. Their rationale is that monolingual corpora are in short supply in comparison to bilingual text and as a result give rise to relatively sparse phrase tables. Thus combining multiple resources into a single phrase table mitigates this problem. Although Quirk et al. [2004] aim at generating target sentences that are meaning preserving and do not delete any information from the source, Zhao et al. [2009] show that a phrase-based model can generate compressed sentences by selecting only translations where the target phrases are shorter than the source ones. More recently, Ganitkevitch et al. [2011] generalize Quirk et al.'s model to syntactic paraphrases and discuss how such a model can be adapted to sentence compression by augmenting the feature set with compression target features and by optimizing appropriately the system's training objective in a fashion similar to Zhao et al.

Our own work builds on the model developed by Cohn and Lapata [2009] and formulates abstractive compression as a tree-to-tree rewriting task. Specifically, the model uses STSG [Shieber and Schabes 1990; Eisner 2003] to capture the space of all possible rewrites. STSG is especially suited to the abstractive task as it can describe nonisomorphic tree pairs and provides expressive power to model consistent syntactic effects such as reordering, changes in nonterminal categories, and lexical substitution. The model is trained discriminatively using the large margin technique proposed by Tsochantaridis et al. [2005]. This framework is attractive in that it supports a configurable loss function that can be tailored to the task at hand. An important part of the model we present here is the synchronous grammar itself, which must be able to model paraphrases. We develop a novel tree-to-tree grammar extraction method which acquires paraphrases from bilingual corpora and show how it can be used

[5]Phrase translation tables do not only contain single-word entries, but multiword entries. These are called phrases, but this concept means nothing more than an arbitrary sequence of words, with no sophisticated linguistic motivation.

[6]It would be possible to extend their approach to allow reordering, however, it is unlikely that the basic reordering models used in phrase-based machine translation would be sufficient for modeling abstractive text compression.

to generate abstractive compressions. In contrast to previous sentence compression work, our model is not limited to word deletion and can be trained on corpora with arbitrary rewrites. We also differ from previous work on paraphrase generation in that we are able to model rewriting operations other than lexical substitutions while taking advantage of syntactic information. Furthermore, as our model is trained with a specific compression objective, it learns which rules yield valid compressions, rather than simply discarding words to produce a shorter string.

3. EXPERIMENTAL STUDY

Since abstractive sentence compression is not as well studied as extractive sentence compression, we first needed to establish whether the task is meaningful. To do this, we designed an online experiment where participants were asked to compress sentences, without being restricted to word deletion. Our instructions were taken from Clarke and Lapata's [2008] extractive compression study and modified so as to encourage annotators to use any rewrite operation that seemed suitable, including adding new words, deleting words, substituting, or reordering them as long as they: (a) preserved the most important information in the source sentence (without distorting its meaning) and (b) the compressed sentence remained grammatical. It was emphasized that their goal was to render the source sentence shorter rather than merely substitute or reorder words without reducing its length.

Participants were given several examples with rewrite operations they could apply. They were given ample flexibility in creating compressions, but were disallowed from rewriting a sentence as two or more sentences or deleting a sentence from the document. They were also informed that some sentences may be short or contain no extraneous information and thus may not be amenable to compression. When coming across such sentences, participants were asked not to perform any rewriting operations, such that the original and compressed sentence are identical. Finally, participants were instructed to ensure that the final compressed document was coherent on its own, that is, the compressions did not distort the meaning of the source document, change the order of events, or change their logical progression.

We randomly selected five documents from a news corpus created by Clarke and Lapata [2008]. The corpus contains 82 newspaper articles (1,433 sentences) from the British National Corpus (BNC) and the American News Text corpus.[7] Each source sentence is associated with a human-authored target compression, created using word deletion (i.e., extractive compression). Although in our experiment participants saw the uncompressed documents only, we also made use of the accompanying compressions in analyzing whether the abstractive sentences differed substantially from extractive ones. The study was conducted over the Internet using WebExp [Keller et al. 2009], an interactive software package for administering Web-based psychological experiments. Documents were randomly assigned to subjects and each subject compressed one document. The experiment was completed by 15 volunteers, all native speakers of English. Examples of the compressions our subjects produced are given in Table I. For comparison, we also show the extractive compressions available with our corpus.

As can be seen, the abstractive compressions are less wordy than their extractive counterparts. The examples illustrate several rewrite operations such as lexical substitution (high winds is paraphrased as *bad weather conditions*, *hampered* as *are preventing*, *but* as *despite*, *dashed hopes of* as *prevented*, *400 lb of dynamite* as *explosives*) and reordering (in the third sentence the order of main and subordinate clause is reversed).We examined more formally the differences between the source sentences and their targets by computing Translation Edit Rate (TER) [Snover et al. 2006], a measure commonly

[7]The corpus can be downloaded from http://jamesclarke.net/research/resources.

Table I. Abstractive Compressions Produced by Naive Subjects

S	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.
T_D	Snow, winds and bitter disagreement hampered attempts to tame Mount Etna, which is threatening the Sicilian town of Zafferana with millions of tons of lava.
T_A	Bad weather conditions are preventing attempts to halt Mount Etna from swamping the town of Zafferana.
S	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.
T_D	The wall of molten lava has come to a halt 150 yards from the first home, but officials said that its flow appeared to have picked up speed further up the slope.
T_A	Although the molten lava has come to a halt, experts believe that it has picked up speed.
S	A crust appears to have formed over the volcanic rubble, but red-hot lava began creeping over it yesterday and into a private orchard.
T_D	A crust formed, but red-hot lava began creeping over it yesterday and into a private orchard.
T_A	Lava has begun to pour into a private orchard, despite a crust having already formed.
S	Bad weather dashed hopes of attempts to halt the flow during what was seen as a natural lull in the lava's momentum.
T_D	Bad weather dashed attempts to halt the flow during a lull in the lava's momentum.
T_A	The weather prevented attempts to stop the lava flow.
S	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.
T_D	Experts say even if the eruption stopped, the sheer pressure of lava piled up for miles would bring debris down on to the town.
T_A	Even if the eruption stopped, the town could be destroyed anyway.
S	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.
T_D	The volcano is pouring out one million tons of debris a day, at 15 ft per second, from a fissure that opened in mid-December.
T_A	Since December the volcano is estimated to be pouring out 1 million ton of debris a day.
S	The Italian army yesterday detonated nearly 400lb of dynamite 3,500 feet up Mount Etna's slopes.
T_D	The army yesterday detonated 400 lb of dynamite 3,500 feet up Mount Etna.
T_A	The army have used explosives.

S is the source sentence, T_D the extractive target, and T_A the abstractive target.

used to automatically evaluate the quality of machine translation output. TER is defined as the minimum number of edits required to change the system output so that it exactly matches a reference translation

$$\text{TER} = \frac{\text{Ins} + \text{Del} + \text{Sub} + \text{Shft}}{n_r}, \tag{1}$$

where n_r is the length of the reference sentence. The number of possible edits include insertions (Ins), deletions (Del), substitutions (Sub), and shifts (Shft). TER is similar to word error rate, the only difference being that it allows shifts. A shift moves a contiguous sequence to a different location within the the same system output and is counted as a single edit. When multiple references are available, the edits from the closest reference (i.e., the reference with the least number of edits) are divided by the average reference length. The perfect TER score is 0, however, note that it can be higher than 1 due to insertions. We use Snover et al.'s implementation of TER[8] to find approximately the sequence of edit operations with the minimum error rate.

In our setting, we have a source (long) sentence and several target compressions provided by our participants. We compute pairwise TER scores between source and target

[8]http://www.cs.umd.edu/∼snover/tercom/.

Table II. Comparative Statistics over the Five Different Texts in the New Compression Corpus

(a) TER scores for abstractive (TER_A) and extractive sentence compression (TER_D) with corresponding compression rates (CompR_A and CompR_D).

	TER_D	CompR_D	TER_A	CompR_A
Text 1	0.30	74.8	0.54	75.4
Text 2	0.30	72.8	0.69	62.6
Text 3	0.27	74.1	0.61	67.8
Text 4	0.38	83.0	0.38	75.0
Text 5	0.64	73.2	0.64	66.7
All	0.38	75.6	0.57	69.5

(b) Number of edits by type — insertions, deletions, substitutions, shifts – needed to convert source sentences into abstracts.

	Ins	Del	Sub	Shft
Text 1	0.3	7.0	17.4	2.2
Text 2	0.9	30.1	9.8	1.3
Text 3	0.2	30.7	7.3	1.0
Text 4	1.1	26.0	11.3	1.6
Text 5	0.4	36.7	11.4	1.2
ALL	0.8	26.1	11.5	1.5

for each participant and report the mean. Table II(a) shows TER scores per document and overall. For comparison, we also calculate TER scores for the extractive compressions provided with our corpus, and report the compression rates for abstractive and extractive compressions.

Perhaps unsurprisingly, we see that the abstractive compressions yield higher TER scores compared to extractive compressions. This means that the participants choose to employ additional edit operations, such as shifts, substitutions, and insertions. Moreover, the compression rate for the abstractive sentences is lower, indicating that these operations yield shorter output. Table II(b) tabulates the number of insertions, substitutions, deletions, and shifts needed (on average) to convert the longer sentences into abstracts. The comparatively high numbers of deletions are consistent with the overall compression aim of rendering the source sentence shorter. However, participants resort to further rewrite operations when given instructions that are not deletion specific. These additional operations are mostly substitutions, indicating that subjects use paraphrases to abbreviate the source sentence, followed by shifts and insertions. The majority of substitutions involve substituting a longer expression with a shorter one. The latter is on average 2.4 words shorter compared to the original expression. It is also worth noting that target sentences exhibit at least one shift on average (see the last column in Table II(b)).[9] Therefore, the assumption that the order of words in the compression remains unchanged, as is typically the case with extractive approaches, may be too restrictive to model the full range of compression phenomena.

4. CORPUS COLLECTION

The experimental study just described demonstrates that nonexpert participants can produce abstractive compressions while using rewrite operations that are not confined to word deletion. Our results suggest that an ideal compression model ought to handle lexical substitution and insertion and word reordering. Creating such a model is challenging; it must not only rewrite the source sentence (employing some form of paraphrases) but do so in a way that produces a shorter string that is both meaningful and grammatical. Technical difficulties aside, an additional stumbling block concerns the lack of widely available corpora for model training and testing. Previous work has been conducted almost exclusively on Ziff-Davis, a corpus derived automatically from document-abstract pairs [Marcu 1999; Knight and Marcu 2002], or on human-authored corpora [Clarke 2008]. These data sources are not suited to our problem as they do not

[9]Note that TER tends to underestimate the number of shifts, so it is likely that there is more reordering than reported in Table II(b). TER will only use the shift operator to facilitate exactly matching tokens; when there is paraphrasing TER often misclassifies shifts as substitutions, insertions, or deletions. This incurs a lower edit cost than performing both a shift and then substituting each of the paraphrased tokens.

contain rewrite operations other than word deletion. Galley and McKeown [2007] obtain a larger version of the Ziff-Davis corpus by gathering sentence pairs containing substitutions[10] in addition to deletions. Unfortunately, this version is not publicly available and, besides, was limited to only two types of rewrite operations, whereas our aim is to model a broader spectrum of rewrites including insertions and reordering.

Although there is a greater supply of paraphrasing corpora, such as the Multiple-Translation Chinese (MTC) corpus[11] and the Microsoft Research (MSR) Paraphrase Corpus [Quirk et al. 2004], they are also not ideal, since they have not been created with compression in mind. An obvious avenue would be to align sentential paraphrases differing in length under the assumption that longer sentences are the source and the shorter ones their target compression. Initial experiments with this approach revealed two difficulties. First, the automatic word alignments were noisy, presumably because the sentences varied considerably in terms of vocabulary and syntactic structure. Second, target sentences were often inappropriate compressions; they either compressed the source too much or changed its meaning drastically. This is somewhat expected given the erroneous alignments and the fact that the paraphrases did not explicitly target information loss.

Our own experimental study (see Section 3) yielded some useful data, however, it is relatively small scale (five documents, 110 sentences) and potentially a nonrepresentative sample of the complexity and range of the task. For these reasons, we created a larger abstractive compression corpus. We collected 30 newspaper articles (625 sentences) from the British National Corpus (BNC) and the American News Text corpus, for which we obtained manual compressions. Five of these documents were compressed by two annotators (not the authors) in order to measure inter-annotator agreement. The annotators were given instructions that explained the task and defined sentence compression with the aid of examples. They were asked to paraphrase while preserving the most important information and ensuring the compressed sentences remained grammatical and meaning preserving. They were encouraged to use any rewriting operations that seemed appropriate, for example, to delete words, add new words, substitute them, or reorder them. Annotation proceeded on a document-by-document basis, and annotators were specifically instructed to ensure that the resulting (compressed) document was coherent on its own. The full set of instructions given to the annotators is listed in Appendix A.

Assessing inter-annotator agreement is notoriously difficult for paraphrasing tasks [Barzilay 2003] since there can be many valid outputs for a given input. Also our task is doubly subjective in deciding which information to remove from the sentence and how to rewrite it. In lieu of an agreement measure that is well suited to the task and takes both decisions into account, we assessed them separately. We first examined whether the annotators compressed at a similar level. The compression rate was 56% for one annotator and 54% for the other. We also assessed whether they agreed in their rewrites by measuring TER [Snover et al. 2006] and BLEU [Papineni et al. 2002]. The inter-annotator TER score was 0.728, whereas annotator agreement with the source yielded a worse TER score of 0.939. As far as BLEU is concerned, inter-annotator agreement was 23.79% (the precision of unigrams, bigrams, trigrams, and fourgrams was 58.8%, 35.8%, 23.1%, and 15.1%, respectively), whereas agreement with the source was only 12.22%. In other words, the annotators agreed much more with one another than with the source. These results taken together with the comparable compression rate indicate that the annotators agreed in what to compress even though they did not

[10]Specifically, they gathered sentence pairs with up to six substitutions using minimum edit distance matching.
[11]LDC, Catalog Number LDC2002T01, ISBN 1-58563-217-1.

Table III. Compression Examples from Our Corpus

1a.	The future of the nation is in your hands.
1b.	The nation's future is in your hands.
2a.	As he entered a polling booth in Katutura, he said.
2b.	Entering a polling booth in Katutura, he said.
3a.	Mr Usta was examined by Dr Raymond Crockett, a Harley Street physician specialising in kidney disease.
3b.	Dr Raymond Crockett, a Harley Street physician, examined Mr Usta.
4a.	High winds and snowfalls have, however, grounded at a lower level the powerful US Navy Sea Stallion helicopters used to transport the slabs.
4b.	Bad weather, however, has grounded the helicopters transporting the slabs.
5a.	To experts in international law and relations, the US action demonstrates a breach by a major power of international conventions.
5b.	Experts say the US are in breach of international conventions.

Sentences marked (a) are the input and (b) are the human-authored output compressions.

always employ the same rewrites. The remaining 25 documents were compressed by a single annotator to ensure consistency. All our experiments used the data from this annotator.[12]

Table III illustrates some examples from our corpus. A fully compressed document can be found in Appendix B. As can be seen, some sentences contain a single rewrite operation. For instance, a Prepositional Phrase (PP) is paraphrased with a genitive (see (1)), a subordinate clause with a present participle (see (2)), a passive sentence with an active one (see (3)). However, in most cases many rewrite decisions take place all at once. Consider sentence (4) where the conjunction *high winds* and *snowfalls* is abbreviated to *bad weather* and the infinitive clause *to transport* to the present participle *transporting*. Note that the prenominal modifiers *US Navy Sea Stallion* and the verb *used* have been removed. In sentence (5), the verb *say* is added and the NP *a breach by a major power of international conventions* is paraphrased by the sentence *the US are in breach of international conventions*.

5. ABSTRACTIVE SENTENCE COMPRESSION AS TREE TRANSDUCTION

In order to model the problem of abstractive compression we resort to statistical machine learning techniques, the aim being to develop an algorithm which can automatically generate an abstractive compression for a given test sentence. As demonstrated in the preceding examples, abstractive compression is a complex linguistic process. Like most natural language processing tasks, to replicate human performance would require deep knowledge of syntax, semantics, pragmatics, and world knowledge. Instead of modeling the full complexity of the problem, we opt for a shallow approach using only syntax. This makes the model considerably simpler to formulate while also limiting its expressive power such that it can be learned directly from data.

5.1. Modeling Framework

Our work builds on the model developed by Cohn and Lapata [2009] who formulate sentence compression as a tree-to-tree rewriting task using a weighted synchronous grammar. The model learns from a parallel corpus of input (uncompressed) and output (compressed) pairs $(\mathbf{x}_1, \mathbf{y}_1) \dots (\mathbf{x}_N, \mathbf{y}_N)$ to predict a target labeled tree $\mathbf{y}$ from a source labeled tree $\mathbf{x}$. The dependency between $\mathbf{x}$ and $\mathbf{y}$ is captured as a weighted STSG. The grammar encodes various tree-based transformations including deletions, structural transformations, and lexical substitutions (paraphrasing). Our model associates a score

[12]http://www.dcs.shef.ac.uk/~tcohn/t3/#Corpus.

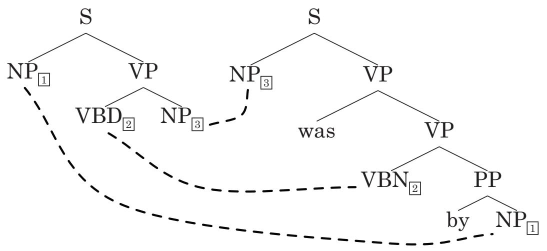

Fig. 1. Illustration of the production which converts from active tense to passive tense. The dotted lines and boxed indices denote $\sim$, the alignments between frontier nonterminals in the two elementary trees. Note the tense of the verb must change between past tense (VBD) and past participle (VBN).

with each complete derivation as a linear function of its rules and the n-grams in the compressed output string. In this framework decoding amounts to finding the best target tree licensed by the grammar given a source tree.

In Section 5.2, we define STSGs and the means by which we extract a grammar from a parallel corpus. We introduce our scoring function in Section 5.3, explain our training algorithm in Section 5.5, and discuss our decoding procedure in Section 5.4.

5.2. Synchronous Grammar

Synchronous grammars generate pairs of related strings, much as standard grammars generate single strings. This is achieved by the recursive application of rewrite rules, where each rule is applied synchronously to both strings. Synchronous grammars can be treated as string transducers by reasoning over the space of possible sister strings for a given string. In this work we use a synchronous grammar to define a tree transducer which operates over input and output trees rather than strings.

Our model is based on synchronous tree-substitution grammar [Shieber and Schabes 1990; Eisner 2003], which uses as rewrite rules pairs of arbitrarily large tree fragments. STSG is a simple grammar formalism, and consequently has efficient inference algorithms while still being complex enough to model a rich suite of tree edit operations. More formally, an STSG is a 7-tuple, $G = (\mathcal{N}_I, \mathcal{N}_O, \Omega_I, \Omega_O, P, R_I, R_O)$ where $\mathcal{N}$ are the nonterminals and Ω are the terminals, with the subscripts I and O indicating input and output, respectively, P are the productions, and $R_I \in \mathcal{N}_I$ and $R_O \in \mathcal{N}_O$ are the distinguished root symbols. Each production is a rewrite rule for two aligned nonterminals $X \in \mathcal{N}_I$ and $Y \in \mathcal{N}_O$ in the input and output

$$\langle X, Y \rangle \rightarrow \langle \alpha, \gamma, \sim \rangle,$$

where α and γ are *elementary trees* rooted with the symbols X and Y, respectively. Nonterminal leaves of the elementary trees are referred to as *frontier nodes* or *variables*. These are the points of recursion in the transductive process. A one-to-one alignment between the frontier nodes in α and γ is specified by $\sim$. This is illustrated in Figure 1 which shows an example production.

Our model uses an STSG as a tree transducer, where the input tree is given and the output tree is generated. The generative process starts with the given input tree and an output tree consisting of only the root symbol R_O, which is aligned to the root of the input tree. Next, each frontier nonterminal in the output tree is rewritten using a production in the grammar which also consumes a fragment of the input tree rooted at the aligned node. This process continues recursively and terminates when there are no remaining frontier nonterminals. At this point we have a complete output tree and the input tree has been completely consumed. The sequence of productions is referred to

$$\begin{aligned}
\langle \text{S, S}\rangle &\rightarrow \langle \text{NP}_{\boxed{1}} \text{ [VP VBD}_{\boxed{2}} \text{ NP}_{\boxed{3}}\text{], NP}_{\boxed{1}} \text{ [VP VBD}_{\boxed{2}} \text{ NP}_{\boxed{3}}\text{]}\rangle \\
\langle \text{S, S}\rangle &\rightarrow \langle \text{NP}_{\boxed{1}} \text{ [VP VBD}_{\boxed{2}} \text{ NP}_{\boxed{3}}\text{], NP}_{\boxed{3}} \text{ [VP was [VP VBN}_{\boxed{2}} \text{ [PP by NP}_{\boxed{1}}\text{]]]}\rangle \\
\langle \text{NP, NP}\rangle &\rightarrow \langle \text{he, him}\rangle \\
\langle \text{NP, NP}\rangle &\rightarrow \langle \text{he, he}\rangle \\
\langle \text{NP, NP}\rangle &\rightarrow \langle \text{he, Peter}\rangle \\
\langle \text{VBD, VBN}\rangle &\rightarrow \langle \text{sang, sung}\rangle \\
\langle \text{NP, NP}\rangle &\rightarrow \langle \text{a song, a song}\rangle
\end{aligned}$$

[S [NP He] [VP [VBD sang] [NP a song]]
[S [NP Him] [VP [VBD sang] [NP a song]]]
[S [NP Peter] [VP [VBD sang] [NP a song]]]
[S [NP A song] [VP was [VP [VBN sung] [PP by [NP he]]]]
[S [NP A song] [VP was [VP [VBN sung] [PP by [NP him]]]]]
[S [NP A song] [VP was [VP [VBN sung] [PP by [NP Peter]]]]]

Fig. 2. Example grammar (top) and the space of output trees (bottom) licensed when given the input tree [S [NP He] [VP [VBD sang] [NP a song]]]]. Numbered boxes in the rules denote linked variables.

as a *derivation*, and the output string is the *yield* of the output tree, given by reading the terminals from the tree in a left-to-right manner. Figure 2 shows an example STSG and the set of output trees licensed for a given input tree. We refer the reader to Eisner [2003] and Cohn and Lapata [2009] for a more detailed exposition of STSG.

Our discussion of STSG has so far focused on the details of the grammar formalism without explaining how such a grammar can be obtained. Creating an STSG by hand is one option, although this would require considerable manual effort and is unlikely to generalize well to new datasets. Instead we define a data-driven procedure for extracting a grammar automatically which is simple and robust, and can make use of two different types of parallel corpora. The first type is parallel compression data, consisting of pairs of input sentences and their target compressions, and the second is parallel bilingual text, consisting of sentences and their translations in a foreign language. Compression corpora do not occur naturally (they have to be either created manually or automatically, e.g., by matching sentences found in an abstract and its corresponding document) and as a result the grammar extracted from such data is likely to have low coverage, especially with regard to paraphrases. We view bilingual corpora as a complementary data source. By being more readily available and overall larger, they can potentially yield a larger number of paraphrase rules. In the following we present two algorithms for extracting a synchronous grammar from these data sources, starting with the algorithm for parallel compression data.

5.2.1. Direct Grammar Extraction. The algorithm takes a parsed parallel compression corpus from which it extracts a set of elementary trees, which are then aggregated to form the grammar. We adopt the grammar extraction technique of Cohn and Lapata [2009] which we now briefly outline. The algorithm proceeds in three stages: word alignment, constituent alignment, and rule extraction. Word alignment seeks to automatically find which input and output words correspond to one another in each sentence pair. For this we use models designed for word-based translation between different languages [Brown et al. 1993; Och and Ney 2004], where in our setting the input and output are treated as the two languages. The second step is constituent alignment, which uses a heuristic process to identify alignments between nodes in the parse trees by projecting via the word alignment (inspired by finite state transducer induction techniques in phrase-based translation [Och and Ney 2004; Koehn et al. 2003]). Finally, the rule extraction

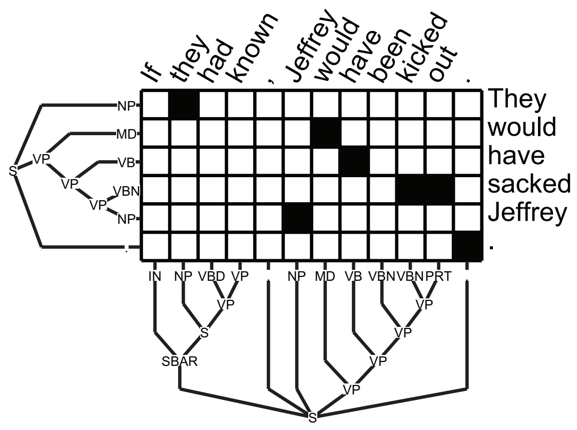

Fig. 3. Example sentence, its compression, and word alignment. Parse trees are also shown for the input and output sentences on the horizontal and vertical axes, respectively. Unary productions (e.g., NP → NN) have been omitted for clarity. The word alignment is displayed as a binary matrix where black cells denote an alignment between the pair of words on the given row and column, and white cells denote no alignment.

step identifies aligned tree fragments which collectively form the elementary trees in the grammar.

Let us illustrate the algorithm by way of an example (for a formal exposition we refer the interested reader to Cohn and Lapata [2009]). Figure 3 shows an example of the parse trees and word alignment for the input sentence "*If they had known, Jeffrey would have been kicked out*" and its target compression "*They would have sacked Jeffrey*." In the example, all the output words are aligned to at least one input word, denoting 1-1 alignments, for example, *they–They*, or multiword alignments, for example, *kicked out–sacked*. A number of the input words are not aligned to anything, for example, the empty column for "*If*" which denotes word deletion.

The second stage is to identify pairs of constituents in the two trees whose yields are aligned to one another in the word alignment (fully or partially, which allows for words to be deleted or inserted). This is illustrated in Figure 4 which shows the derived constituent alignment for the earlier example. Note that word alignments are also included as constituent alignments, while new alignments between higher-level constituents are also included, for example, [NP They] aligned with [NP If they had known]. Some constituents in one sentence do not align to constituents in the other sentence, for example, [VP sacked Jeffrey], and are therefore absent from the constituent alignment.

The next step is to generalize the aligned subtree pairs by replacing aligned child subtrees with variable nodes. For example, in Figure 4 when we consider the pair of aligned subtrees [SBAR If they had known] and [NP they], we could extract the rule

$$\langle \text{SBAR,NP} \rangle \rightarrow \langle \text{[SBAR [IN If] [S [NP they] [VP [VBD had] [VP known]]]], [NP They]} \rangle. \quad (2)$$

However, this rule is very specific and consequently will not be very useful in a transduction model. In order for it to be applied, we must observe the full SBAR subtree, which is unlikely to occur in another sentence. A more robust approach is to generalize the rule so as to match many more source trees, and thereby allow transduction of previously unseen structures. In the example, the node pairs labeled (S, NP) and (NP, NP) can be generalized as these nodes are aligned constituents. In addition, the nodes IN, VP, VBD, and VP in the source are unaligned, and therefore can be generalized using ϵ-alignment to signify deletion. Performing as many generalizations as possible

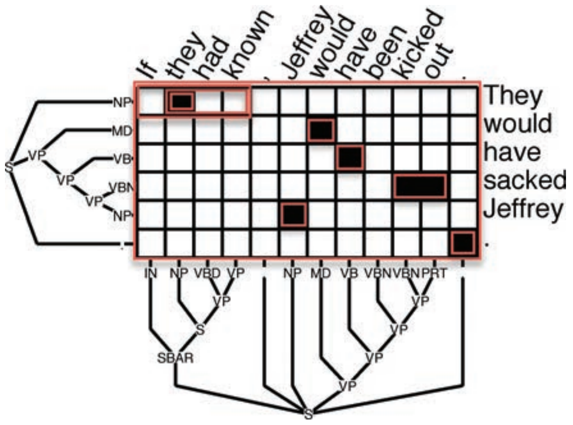

Fig. 4. Constituent-level alignment for the example in Figure 3. Each red rectangle denotes an alignment between constituents with matching spans on the horizontal and vertical axes.

for the preceding example, we would produce the rule

$$\langle \text{SBAR,NP} \rangle \rightarrow \langle [\text{SBAR IN}_{\boxed{\epsilon}}\ \text{S}_{\boxed{1}}], \text{NP}_{\boxed{1}} \rangle, \tag{3}$$

which encodes that we can transform a subordinate clause (SBAR) into a noun phrase (NP) by way of deleting its preposition child (IN) and then transforming its declarative clause (S) child into an NP. There are many other possible rules which can be extracted by applying different legal combinations of the generalizations. In this work we extract maximally generalized rules, thus avoiding a combinatorial explosion in the number of grammar rules. Figure 5 shows the full set of maximally general rules derived from the running example. These rules describe structural transforms (including reordering), deletion, and paraphrasing. Note that in order to delete an input constituent, we require that its entire subtree be covered using epsilon-aligned rules, thus necessitating the bottom five rules in the figure which explicitly encode lexical deletion.

5.2.2. Pivoted Grammar Extraction. The algorithm presented before extracts synchronous grammar rules directly from a parallel abstractive compression corpus. Overall this results in a high-quality grammar with rules encoding the syntactic and paraphrase transformations used by humans when compressing text. However, this grammar alone is insufficient for describing the full range of compression phenomena, due largely to the small size of the parallel corpus from which the rules are derived. There will be many unobserved paraphrases, no matter how good the extraction method. One way to achieve a higher level of robustness would be to use a corpus many orders of magnitude larger. However, this kind of data is not readily available and would be expensive to create. For this reason we develop an alternative technique for deriving rules from bilingual parallel corpora, which are in plentiful supply. Our approach extracts a second, much larger, grammar, which is used to augment the original directly extracted grammar. Crucially, the second grammar will not contain explicit compression rules, just paraphrasing ones. We leave it to the model to learn which rules serve the compression objective.

The paraphrase grammar extraction method uses bilingual pivoting to learn paraphrases over syntax tree fragments. These paraphrase pairs of tree fragments are treated as rules and added to our synchronous grammar. The central assumption underlying the pivoting technique is that strings (elementary trees in our case) are paraphrases if they share the same translation(s) in a foreign language [Bannard and Callison-Burch 2005]. Practically, this is equivalent to treating the paraphrasing

$$\begin{aligned}
\langle \text{S, S}\rangle &\rightarrow \langle [\text{S SBAR}_{\boxed{1}}\ ,_{\boxed{\epsilon}}\ \text{NP}_{\boxed{2}}\ [\text{VP MD}_{\boxed{3}}\ [\text{VP VB}_{\boxed{4}}\ \text{VP}_{\boxed{5}}]]\ ._{\boxed{6}}],\\
&\qquad [\text{S NP}_{\boxed{1}}\ [\text{VP MD}_{\boxed{3}}\ [\text{VP VB}_{\boxed{4}}\ [\text{VP VBN}_{\boxed{5}}\ \text{NP}_{\boxed{2}}]]]\ ._{\boxed{6}}]\rangle\\
\langle \text{SBAR, NP}\rangle &\rightarrow \langle [\text{SBAR IN}_{\boxed{\epsilon}}\ \text{S}_{\boxed{1}}], \text{NP}_{\boxed{1}}\rangle\\
\langle \text{S, NP}\rangle &\rightarrow \langle [\text{S NP}_{\boxed{1}}\ \text{VP}_{\boxed{\epsilon}}], \text{NP}_{\boxed{1}}\rangle\\
\langle \text{VP, VBN}\rangle &\rightarrow \langle [\text{S VBN}_{\boxed{1}}\ \text{VP}_{\boxed{\epsilon}}], \text{VBN}_{\boxed{1}}\rangle\\
\langle \text{NP, NP}\rangle &\rightarrow \langle [\text{NP they}], [\text{NP They}]\rangle\\
\langle \text{NP, NP}\rangle &\rightarrow \langle [\text{NP Jeffrey}], [\text{NP Jeffrey}]\rangle\\
\langle \text{MD, MD}\rangle &\rightarrow \langle [\text{MD would}], [\text{MD would}]\rangle\\
\langle \text{VB, VB}\rangle &\rightarrow \langle [\text{VB have}], [\text{VB have}]\rangle\\
\langle \text{VP, VBN}\rangle &\rightarrow \langle [\text{VP [VBN kicked] [PRT out]}], [\text{VBN sacked}]\rangle\\
\langle ., .\rangle &\rightarrow \langle [.\ .], [.\ .]\rangle\\
\langle \text{IN}, \epsilon\rangle &\rightarrow \langle [\text{IN If}], \epsilon\rangle\\
\langle \text{VBD}, \epsilon\rangle &\rightarrow \langle [\text{VBD had}], \epsilon\rangle\\
\langle \text{VP}, \epsilon\rangle &\rightarrow \langle [\text{VP known}], \epsilon\rangle\\
\langle ,, \epsilon\rangle &\rightarrow \langle [,\ ,], \epsilon\rangle\\
\langle \text{VBN}, \epsilon\rangle &\rightarrow \langle [\text{VBN been}], \epsilon\rangle
\end{aligned}$$

Fig. 5. Minimal synchronous grammar rules extracted from the example in Figure 3.

problem as a two-stage translation process, which works by translating some English text into a foreign language, and then translating it back into English. The original and doubly translated text are then considered to be paraphrases, and the foreign language is said to be the pivot. The process is formulated as a statistical model of paraphrasing e into e' as

$$p(e'|e) = \sum_f p(e'|f)p(f|e), \tag{4}$$

where $p(f|e)$ is the probability of translating an English string e into a foreign string f and $p(e'|f)$ the probability of translating the same foreign string into some other English string e'. We thus obtain English-English translation probabilities $p(e'|e)$ by marginalizing out the foreign text.

In contrast to previous work that used only strings for paraphrasing [Bannard and Callison-Burch 2005], we instead use elementary trees in English. These elementary trees are translated into foreign language strings, which are then retranslated into English elementary trees. This results in pairs of elementary trees which encode syntactic paraphrases, that is, an STSG. To translate between elementary trees and foreign strings we use a bilingual grammar extraction algorithm designed for syntax-based machine translation [Galley et al. 2004]. This algorithm is similar to our direct grammar extraction algorithm presented earlier, except that it works over pairs of trees and strings rather than pairs of trees. As input it uses a bilingual word-aligned parallel corpus with parse trees in one language and tokenized sentences in the other, from which it extracts a set of rules. Each rule is a pair of an elementary tree and its corresponding translation string fragment. Figure 7 illustrates the output rules produced when applied to the sentence pair in Figure 6. Note that the English rules and foreign strings shown include variable indices where they have been generalized. We refer the reader to Galley et al. [2004] for a more detailed exposition of the bilingual grammar extraction algorithm.

The translation rules are extracted from a bilingual corpus and their counts are used to compute conditional frequencies estimates for $p(e|f)$ and $p(f|e)$ in (4) where e are

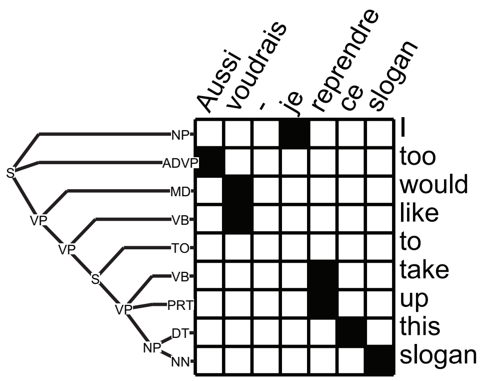

Fig. 6. Sample bilingual sentence pair in French and English. The English parse tree shown on the left has been simplified by removing unary productions.

$$\begin{aligned}
\text{NP} &\rightarrow \langle [\text{NP I}], \text{je}\rangle\\
\text{ADVP} &\rightarrow \langle [\text{ADVP too}], \text{aussi}\rangle\\
\text{DT} &\rightarrow \langle [\text{DT this}], \text{ce}\rangle\\
\text{NN} &\rightarrow \langle [\text{NN slogan}], \text{slogan}\rangle\\
\text{NP} &\rightarrow \langle [\text{NN DT}_{\boxed{1}}\ \text{NN}_{\boxed{2}}], \boxed{1}\ \boxed{2}\rangle\\
\text{VP} &\rightarrow \langle [\text{VP [VB take] [PRT up] NP}_{\boxed{1}}], \text{reprendre } \boxed{1}\rangle\\
\text{S} &\rightarrow \langle [\text{S [TO to] VP}_{\boxed{1}}], \boxed{1}\rangle\\
\text{S} &\rightarrow \langle [\text{S NP}_{\boxed{1}}\ \text{ADVP}_{\boxed{2}}\ \text{[VP [MD would] [VP [VB like] S}_{\boxed{3}}]]], \boxed{2}\ \text{voudrais - } \boxed{1}\ \boxed{3}\rangle
\end{aligned}$$

Fig. 7. Translation rules extracted from the sentence pair in Figure 6 using the method of Galley et al. [2004].

English elementary trees and f are foreign strings. Finally, we apply (4) by marginalizing over all foreign strings to find the weighted set of elementary tree paraphrases, thus forming our paraphrase grammar. To allow matching of different reordering patterns, we first normalize rules such that the variable markers are sorted in increasing order in the foreign string, for example, $\langle[\text{NP JJ}_{\boxed{1}}\ \text{NN}_{\boxed{2}}]\rangle \rightarrow \langle\boxed{2}\ \text{va}\ \boxed{1}\rangle$ is normalized to $\langle[\text{NP JJ}_{\boxed{2}}\ \text{NN}_{\boxed{1}}]\rangle \rightarrow \langle\boxed{1}\ \text{va}\ \boxed{2}\rangle$. This means that compatible foreign strings are rendered string identical.

Figure 8 illustrates the process for the [take up NP] fragment, showing four of its translations in French and their corresponding translations back into English. In our experiments we marginalize over a number of different foreign languages, not just a single language as shown in the example. Overall the pivoting method results in a large grammar covering a broad range of paraphrases and their syntactic contexts, which is used to supplement the higher precision but lower coverage directly extracted grammar.

5.2.3. Copy and Delete Rules. Aggregating the synchronous grammar rules from direct extraction and pivoting results in a large grammar, however, it still does not have perfect coverage on unseen trees to be processed at test time. These trees may contain unknown words or unseen CFG productions, and therefore no derivations are possible under the transduction grammar. For this reason we add new copy and delete rules to the grammar, which allow test source trees to be fully covered. Copy rules copy a CFG production verbatim into the target, which trivially allows the transducer to

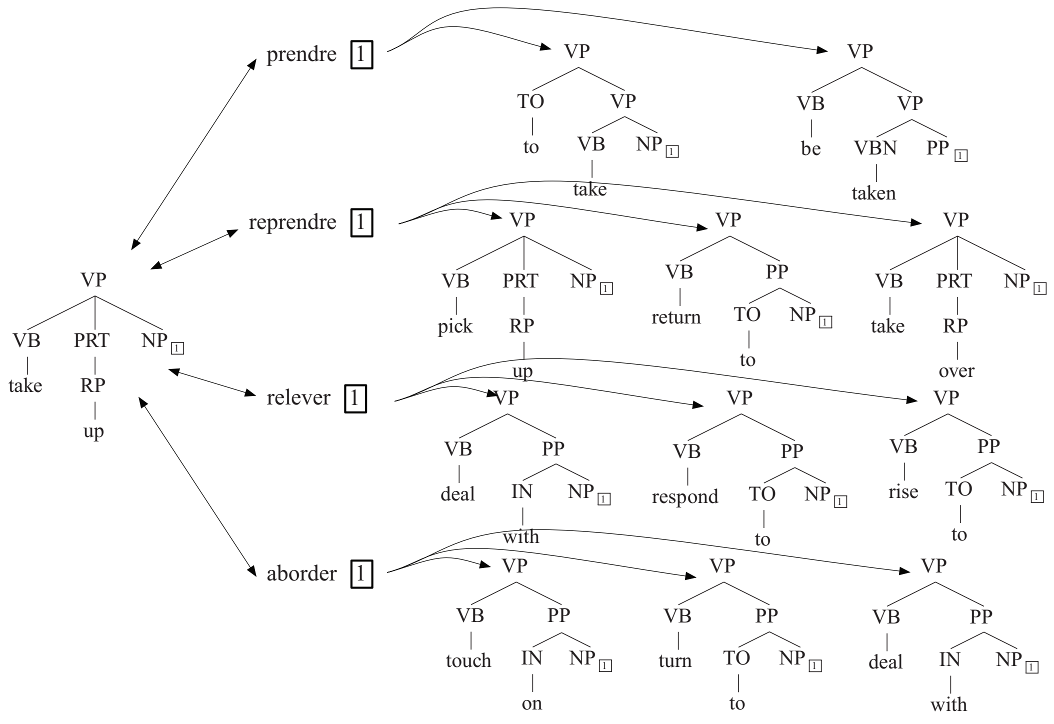
prendre 1
reprendre 1
relever 1
aborder 1
VP VB PRT NP 1 take RP up
VP TO VP to VB NP 1 take
VP VB VP be VBN PP 1 taken
VP VB PRT NP 1 pick RP up
VP VB PP return TO NP 1 to
VP VB PRT NP 1 take RP over
VP VB PP deal IN NP 1 with
VP VB PP respond TO NP 1 to
VP VB PP rise TO NP 1 to
VP VB PP touch IN NP 1 on
VP VB PP turn TO NP 1 to
VP VB PP deal IN NP 1 with

Fig. 8. Illustration of the pivoting process for finding paraphrase tree fragments for the fragment ‘take up NP’. The process works by translating the fragment into a string in a foreign language (French here) and then translating the string into an English fragment. Pairs of the resulting fragments are included as synchronous rules in the grammar. Notice that category transformations can occur, as seen in the top-right fragment which has a prepositional phrase (PP) child.

cover the source. However, this constrains the model to retain all unknown words and productions, which limits its ability to compress the data. For greater flexibility, we also add rules to delete all or part of the CFG production. Partial deletion rules are created to delete each contiguous subsequence of the child nodes of the production.

5.3. Linear Model

A synchronous grammar defines a transducer capable of mapping a source tree into many possible target trees, however, it is of little use without a weighting towards grammatical trees which yield fluent compressed target sentences. Following Cohn and Lapata [2009], we use a linear model which assigns a score to each derivation[13]

$$\text{score}(\mathbf{d}; \mathbf{w}) = \langle \Psi(\mathbf{d}), \mathbf{w} \rangle, \tag{5}$$

where $\mathbf{d}$ is a derivation consisting of a sequence of STSG rules which uniquely specifies the source, $\mathbf{x} = \text{source}(\mathbf{d})$, and target trees, $\mathbf{y} = \text{target}(\mathbf{d})$; $\mathbf{w}$ are the model parameters; Ψ is a vector-valued feature function; and the operator $\langle \cdot, \cdot \rangle$ is the inner product. The parameters, $\mathbf{w}$, are learned during training, described in Section 5.5.

[13]The model applies to derivations rather than target trees or strings for the reason of tractability. In STSGs many derivations can produce the same target tree, and properly accounting for this would incur a significant increase in time and space complexity.

The feature function, Ψ, returns a vector of feature values for a derivation

$$\Psi(\mathbf{d}) = \sum_{r \in \mathbf{d}} \phi(r, \text{source}(\mathbf{d})) + \sum_{m \in \text{ kgrams}(\mathbf{d})} \psi(m, \text{source}(\mathbf{d})), \tag{6}$$

where r are the rules of a derivation, kgrams($\mathbf{d}$) are contiguous sequences of words[14] up to length k, and ϕ is a feature function returning a vector of feature values for each rule. This is a very general feature representation, allowing features to weight not just the rules in the grammar but also over the word sequence in the output string. The later is used to include as a single feature the log probability of the output sentence under a trigram language model. This feature is critical for generating fluent output, as also evidenced by the ubiquity of language models in models of statistical machine translation [Koehn et al. 2003], another task in which output fluency is paramount.

In addition to the language model, we extract features for each rule, $\langle X,Y \rangle \rightarrow \langle \alpha, \gamma, \sim \rangle$, to encode the rule's syntax, lexical items, and compression operations. These features were developed specifically for the abstractive compression task and are instantiated according to the templates detailed next. Most of the templates give rise to binary indicator features, except for the count and frequency features. The indicator features perform a boolean test, returning value 1 when the test succeeds and 0 otherwise. Our templates resulted in 196,419 features on our compression corpus.

Origin. This is the source of the rule, which is either: (a) directly extracted from training, (b) extracted via bilingual pivoting, and/or (c) explicitly created as a copy or delete rule. These features allow the model to learn a preference for the different knowledge sources.

Frequency. This is the log count of the rule, $\log c(\langle X, Y \rangle \rightarrow \langle \alpha, \gamma, \sim \rangle)$ and its component parts $\log c(\alpha)$ and $\log c(\gamma)$. These features are real-valued rather than binary indicator features. They allow the model to represent the forward and backward conditional probabilities, which have both proven critical features in machine translation systems. These three features are replicated to allow separate treatment of directly extracted rules and pivoted rules.

Default. This is a default feature with value 1, which counts the number of rules used in a derivation.

Length. This is the number of terminals in γ, to allow better modeling of the output length and balance the language model log probabilities.

Variable counts. These are the number of variables in γ and the difference in the number of variables between α and γ, allowing a bias over different granularity of segmentation of the target and the amount of deletion.

Rule match. These are the input and output tree fragments, α and γ, and both trees as a pair. This allows the model to learn a specific weight for each rule and specific to each input or output elementary tree. We also include an indicator feature to test whether $\alpha = \gamma$.

Root categories. These are the nonterminal categories X, Y, and the pair (X, Y). This allows the model to learn a preference for different segmentations of the source tree and target trees, and to discourage the nonterminal category to be changed.

Preterminal compatibility. If both X and X are preterminals (special nonterminals that permit only one terminal child), this feature tests whether their root categories and/or terminals are identical. This allows the model to learn a preference

[14]These are called n-grams in the rest of the article, but here we use k for clarity of notation, as n is already used to refer to the sentence length in words.

for paraphrases which maintain the same part-of-speech, separate to the aforesaid feature which also operates over higher-level nodes in the trees.

Yield match. If the yield terminals of α and γ match, similarly for the preterminal sequence. These allow rules that change the structure of the tree but preserve the words or part-of-speech tags to be rewarded.

Compression. This tests whether the yield of γ is a subsequence of the yield of α, that is, if it could form part of and extractive compression.

Yield difference. This is the number of common, dropped, and inserted words when comparing the two yields, and a lexicalized variant which includes the word identity. These allow the model to learn a bias towards including or excluding specific words, and a general bias for the different edit operations.

5.4. Decoding

Decoding aims to find the best target tree for a given source tree. The synchronous grammar defines a space of possible target trees for the source tree. For all but the simplest grammars, there are an exponential number of possible trees, and the decoding algorithm aims to find efficiently from these options the tree with the highest score.

Decoding finds the maximizing derivation,[15] $\mathbf{d}^*$, of

$$\mathbf{d}^* = \operatorname*{arg\,max}_{\mathbf{d}:\,\mathrm{source}(\mathbf{d})=\mathbf{x}} \mathrm{score}(\mathbf{d};\mathbf{w}), \tag{7}$$

where $\mathbf{x}$ is the (given) source tree, source($\mathbf{d}$) extracts the source tree from the derivation $\mathbf{d}$, and score($\cdot$) is defined in (5). The maximization is performed over the space of derivations for the given source tree using an approximate beam search algorithm [Cohn and Lapata 2009], which we now summarize.

For simplicity of exposition, we first present an exact dynamic programming algorithm for finding the best scoring tree under a model with only rule features, but no string n-gram features, before presenting its extension to approximate search with these additional features.[16] The algorithm builds a chart in a bottom-up fashion, by performing postorder traversal over the source tree, at each node v computing the best target compression trees for the subtree rooted at v. Each entry is referred to as the "chart cell" for v. The grammar licenses change in nonterminal symbols (e.g., a PP becoming an NP), and therefore we store the best target tree for each different root nonterminal category. The algorithm terminates at the root node, at which point we have recovered the best target tree, namely the chart entry with nonterminal symbol R_O. The central part of the algorithm is to compute the value of the chart cell for each node v. This is done by finding grammar rules whose source sides match the subtree rooted at v, and from these recording the target side of the rule with the best score (for each different root nonterminal). Scoring fully lexicalized rules is straightforward, simply requiring the computation of an inner product of the rule's features and the model weight vector. The score for grammar rules with frontier nonterminals is defined recursively, by adding to the rule's score the score for the best target trees for each frontier node in the source tree. These values will have been computed earlier in the traversal of the source tree, and can thus simply be looked up in the chart. All of the different transductions for node v are pooled together, and the best scoring tree for each nonterminal category is taken and stored in the chart.

[15]As mentioned before, for tractability of computation we deal with derivations in place of target trees. The target tree is recovered using $\mathbf{y}^* = \mathrm{target}(\mathbf{d}^*)$.

[16]The decoding algorithm presented here is similar to algorithms used for decoding in statistical machine translation for tree-to-string models [Huang et al. 2006; Liu et al. 2006].

Although efficient, this chart-based algorithm does not support n-gram features, which cannot be evaluated locally to each node in the source tree. The target trees for sibling nodes in the source need to be known in order to evaluate the n-grams overlapping between neighboring target trees. For this reason we adapt the algorithm to defer the evaluation of border n-grams until there is sufficient context. This necessitates storing a list of possible options in each chart cell, rather than only the best tree for a given nonterminal (we have no way of knowing if a seemingly low-scoring tree will have high-scoring n-grams until these are evaluated later on). To store the exhaustive set of all trees is intractable, necessitating approximation.[17] We base our approach on those developed for grammar-based machine translation [Chiang 2007], which solves a similar maximization problem. Specifically, we use a beam search by pruning the entries for a cell to a fixed constant number. Low-scoring entries are pruned from the search, where an entry's score is defined as its local score plus an approximation of its nonlocal score (a unigram estimate of the language model log probability). Pruning occasionally eliminates the globally best solution, however, we have found empirically that search error is quite modest. Further, we also use the cube pruning heuristic [Chiang 2007] to further limit the number of items considered for inclusion in the beam, resulting in an asymptotic time complexity linear in the size of the source tree, grammar, and beam. We refer the interested reader to Cohn and Lapata [2009] for further details.

5.5. Training

In the preceding, we assumed that the model parameters, $\mathbf{w}$, were given. Now we turn to the problem of estimating $\mathbf{w}$ from data, using a supervised learning setting where examples of sentences and their compressions are given. The challenge for the learning algorithm is to find parameters which model the training data accurately and generalize well to unseen data. This is framed as a maximum margin optimization problem using SVM^{struct} [Joachims 2005; Tsochantaridis et al. 2005]. Here we present a summary of the algorithm and refer the interested reader to Cohn and Lapata [2009] for a more detailed exposition.

The training algorithm finds the approximate minimizer of

$$\min_{\mathbf{w},\xi} \frac{1}{2}||\mathbf{w}||^2 + \frac{\mathcal{C}}{N}\sum_{i=1}^{N}\xi_i, \;\; \xi_i \geq 0 \tag{8}$$

$$\forall i, \forall \mathbf{d} : \text{source}(\mathbf{d}) = \mathbf{x}_i \wedge \mathbf{d} \neq \mathbf{d}_i : \langle \mathbf{w}, \Psi(\mathbf{d}_i) - \Psi(\mathbf{d})\rangle \geq \Delta(\mathbf{d}, \mathbf{d}_i) - \xi_i,$$

where i indexes the training examples, $(\mathbf{x}_i, \mathbf{y}_i)$, N is the number of training examples, and $\mathbf{d}_i$ is a derivation linking $\mathbf{x}_i$ and $\mathbf{y}_i$ (we use a heuristic to select one of the many possible derivations, opting for the one using the most STSG productions). The constraints ensure that for each training example the true derivation scores more highly under the model than alternative derivations (subject to the slack variables, ξ_i discussed later). The magnitude of the difference must exceed the loss, $\Delta(\mathbf{d}, \mathbf{d}_i)$, which quantifies the difference between the prediction and the truth. Alternative derivations which differ only slightly from the true value need only a small margin of separation, while wildly incorrect derivations require a much larger margin. In this work the loss function computes the total number of words in the predicted compressions which are

[17]We can improve matters by recognizing that only $2(k-1)$ tokens cannot be processed locally: $k-1$ on the extreme left and right ends of the yield, where k is the Markov order of the language model. However, even if we account for this the time and space complexity is still exponential in k.

not in the reference, subject to a length penalty

$$\Delta(\mathbf{d}, \mathbf{d}_i) = \sum_{w \in \text{yield}(\mathbf{d})} [\![w \notin \text{yield}(\mathbf{d}_i)]\!] + \max\left(|\text{yield}(\mathbf{d}_i)| - |\text{yield}(\mathbf{d})|, 0\right),$$

where yield($\mathbf{d}$) returns the yield of the compressed tree (token sequence) specified by $\mathbf{d}$ and $[\![\cdot]\!]$ returns 1 if the condition is true and 0 otherwise. If the predicted string is shorter than the reference, then each omitted word is treated as being incorrect. This penalty serves to discourage overly short output, for example, predicting nothing would otherwise be considered perfect. Note that the loss function is asymmetric, $\Delta(\mathbf{d}, \mathbf{d}_i) \neq \Delta(\mathbf{d}_i, \mathbf{d})$, due to the treatment of word tokens versus word types in yield of the first and second arguments, respectively, and the length penalty which is asymmetric by design. Cohn and Lapata [2009] show that this loss function (referred to as Hamming loss) is more effective than more elaborate variants such as edit distance and F1, which is most likely a consequence of this loss function permitting more accurate approximation.

Slack variables, ξ_i, are introduced for each training example to allow for nonseparable input, where it is impossible (or undesirable) to achieve zero training error. These allow constraints to be violated with a penalty term in the objective function. Finally, $\mathcal{C}$ is a constant that controls the trade-off between training error minimization and margin maximization. This constant is chosen using manual tuning for optimal performance on a development set.

The optimization problem in (8) is approximated using an algorithm proposed by Tsochantaridis et al. [2005]. This algorithm finds a small set of constraints from the full-sized optimization problem that ensures a sufficiently accurate solution. Specifically, it constructs a nested sequence of successively tighter relaxations of the original problem using a (polynomial time) cutting plane algorithm. For each training instance, the algorithm keeps track of the selected constraints defining the current relaxation. Iterating through the training examples, it proceeds by finding the output that most radically violates a constraint. In our case, the optimization relies on finding the derivation which is both high scoring and has high loss compared to the gold standard. This requires finding the maximizer of

$$H(\mathbf{d}) = \Delta(\mathbf{d}_i, \mathbf{d}) - \langle \mathbf{w}, \Psi(\mathbf{d}_i) - \Psi(\mathbf{d}) \rangle. \quad (9)$$

The search for the maximizer of $H(\mathbf{d})$ in (9) can be performed by the decoding algorithm presented in Section 5.4 with some extensions. Firstly, by expanding (9) to $H(\mathbf{d}) = \Delta(\mathbf{d}_i, \mathbf{d}) - \langle \Psi(\mathbf{d}_i), \mathbf{w} \rangle + \langle \Psi(\mathbf{d}), \mathbf{w} \rangle$ we can see that the second term is constant with respect to $\mathbf{d}$, and thus does not influence the search. The decoding algorithm maximizes the last term, so all that remains is to include the loss function into the search process. This amounts to augmenting each chart entry to also store a tuple (TP, FP) representing the number of terminals in the target tree which are also in the reference (True Positives, TP) and those that are not in the reference (False Positives, FP). By design, this decomposes with the derivation such that the values of TP and FP from child chart cells can simply be added together as part of calculating the tuple value for parent chart cells. Finally the loss of the maximizing tree can be computed from the chart entry for the root node, $\Delta = \text{FP} + \max(|\text{yield}(\mathbf{d}_i)| - (\text{TP} + \text{FP}), 0)$.

6. EXPERIMENTAL SET-UP

In this section we present our experimental setup for assessing the performance of our model.[18] We give details on the corpora and grammars we used, model parameters and

[18]The software can be downloaded from http://staffwww.dcs.shef.ac.uk/people/T.Cohn/t3.

features, the systems employed for comparison with our approach, and explain how model output was evaluated.

6.1. Model Selection

The framework presented in Section 5 allows great flexibility in modeling abstractive compression. Depending on the grammar extraction strategy, choice of features, and loss function, different classes of models can be derived. Before presenting our results on the test set, we discuss how these modeling choices were instantiated in our experiments and motivate the reasons for their selection.

The STSG lies at the core of our model. It therefore makes sense to experiment with a variety of grammars in order to assess how the type and number of rules affect model performance. We compared a grammar using rules obtained from our abstractive compression corpus (using the 503 sentence training partition; a further 20 and 110 sentences are reserved for development and testing, respectively), the extractive compression corpus (1510 training sentences),[19] and the union of both corpora. The corpora were word-aligned using the Berkeley aligner [Liang et al. 2006] initialized with a lexicon of word identity mappings, and parsed with Bikel's [2002] parser. We extracted grammar rules following the technique described in Section 5.2.

As mentioned earlier, the rules obtained from compression corpora will exemplify many structural transformations but will have relatively few paraphrases. To give our model the ability to perform a wider range of rewrite operations such as substitutions we also complemented the aforesaid grammars with rules extracted from bitexts. Specifically, we obtained a pivot grammar by harvesting rules from the French-English, Czech-English, German-English, Spanish-English, and Hungarian-English Europarl version 2.[20] These language pairs were selected so as to represent a range of language families (Slavic, Romance, Germanic, and Finno-Ugric) exhibiting variation in word order and more generally syntactic structure as well as morphology. When using only one pivot language, problems can arise for terms with very general translations (e.g., gender or case information being lost), resulting in large sets of poor-quality paraphrases for the terms. Using many different languages as pivots ameliorates this problem because a term with overly general translations in one language often has better translations in the other languages; effectively this smoothes out the effect of pivoting errors. The parallel corpora contained approximately 688K sentences for each language pair. Again, we obtained alignments using the Berkeley aligner and parsed the English side with Bikel's parser. We extracted minimal tree-to-string rules using our implementation of Galley et al. [2004]. To ameliorate the effects of poor alignments on the grammar, we removed rules appearing less than twenty times and used only the five best translations for each source fragment when pivoting. The final paraphrase rules were further pruned to exclude those with conditional probability less than the maximum of 10^{-3} and 1% of the highest probability paraphrase for each source.

An important question concerns the size of the pivot grammar and its bearing on the quality of paraphrase rules. Unfortunately, it is not feasible to conduct a detailed study on the trade-off between grammar size and compression quality; to do so convincingly would require repeated human evaluations as the compression task has no widely accepted automatic evaluation metric. Nonetheless, experiments with grammars obtained from a single language pair (e.g., fr-en) as opposed to multiple pairs (see the row "all merged" in Table IV) revealed that good paraphrases can be obtained from less data (see also Cohn and Lapata [2008]). However, as with research in machine translation,

[19]The abstractive compression corpus was created for a subset of the input sentences used in the extractive compression corpus. The development and test input sentences are identical.
[20]http://www.statmt.org/europarl/.

Table IV. Composition of Pivoted Paraphrase Rules

Language pair	Clone	Same yield	Delete	Other edit	Total
cz-en	13720	6409	1217	14993	36339
de-en	14951	4709	555	18958	39173
es-en	16761	6418	516	23091	46786
fr-en	15861	5316	414	20781	42372
hu-en	16102	7101	792	15196	39191
all merged	37483	20486	3230	70812	132011

The total count of rules is shown, which is divided into mutually exclusive categories for rules which make no syntactic or lexical changes (Clone), otherwise make no lexical changes (Same yield), delete some lexical items (Delete), or include other edits such as reordering, paraphrase, or insertion (Other edit). The statistics are shown for each parallel corpus and after these have been merged. Duplicate rules were removed in the merging process.

better coverage can be obtained by using large parallel corpora. We also experimented with less strict rule pruning parameters, but found that these did not significantly alter the models' performance. This was despite their resulting in significantly larger grammars, and correspondingly slower inference.

The statistics of the pivoting rules are shown in Table IV. This shows the majority of rules extracted encode edits other than deletion, such as substitution, reordering, insertion, or some combination. Surprisingly few rules perform purely lexical deletion—a total of 3,230 from a grammar of 132,011 rules—although if we consider rules which combine deletion with other edits, the total number rises to 6,356. We attribute this to our choice of corpora: translations of parliamentary proceedings need to remain faithful to the original, and accordingly there are few instances where information is dropped or added during translation. The deletion rules predominantly performed minor syntactic and stylistic changes, such as dropping the title from a proper name. We considered including the deletion rules in a purely extractive compression system, however, in informal evaluations we noticed little difference between the model predictions when trained with or without the pivoted deletion rules. For this reason, we did not include the pivoted rules in our extractive compression system in the following experiments.

In addition to these grammar rules, we also scanned the source trees in the compression corpus to supplement the grammar with further rules to ensure complete coverage, that is, ensuring that a derivation exists for each tree. We created rules to either copy each CFG production, delete it entirely, or selectively delete any subsequence of its children. This is illustrated in Table V where the rules flagged with a C are a selection of those derived from the CFG production NP $\rightarrow$ DT JJ NN. All trees are rooted with a distinguished TOP nonterminal which allows the explicit modeling of sentence spanning subtrees. The grammars each had 7,870 (directly extracted from abstractive training), 16,424 (directly extracted from extractive training), 132,011 (bilingual pivoting), and 24,118 rules (coverage copy/delete rules).

We trained different compression models using the extractive or abstractive rules and their union. In addition, we trained a model with both the pivot and union rules (Extract+Abstract+Pivot).[21] All grammars included the coverage rules, ensuring that predictions could be made for all test input trees. The total size of each of these grammars used in each system are reported in Table VI; note that these sizes are less than

[21]We also experimented with other grammar combinations such as Extract+Pivot and Abstract+Pivot; however, these models did not outperform the Abstract+Extract+Pivot combination and we omit them for the sake of brevity.

Table V. Sample Grammar Rules Showing the Source(s) from Which They Were Extracted

⟨NP,NP⟩	→	⟨[NP $\text{DT}_{\boxed{1}}$ $\text{JJ}_{\boxed{2}}$ $\text{NN}_{\boxed{3}}$], [NP $\text{DT}_{\boxed{1}}$ $\text{JJ}_{\boxed{2}}$ $\text{NN}_{\boxed{3}}$]⟩	E, A, C, P
⟨NP,NP⟩	→	⟨[NP $\text{DT}_{\boxed{1}}$ $\text{JJ}_{\boxed{\epsilon}}$ $\text{NN}_{\boxed{2}}$], [NP $\text{DT}_{\boxed{1}}$ $\text{NN}_{\boxed{2}}$]⟩	E, A, C
⟨NP,ϵ⟩	→	⟨[NP $\text{DT}_{\boxed{\epsilon}}$ $\text{JJ}_{\boxed{\epsilon}}$ $\text{NN}_{\boxed{\epsilon}}$], ϵ⟩	E, A, C
⟨NP,NN⟩	→	⟨[NP $\text{DT}_{\boxed{\epsilon}}$ $\text{JJ}_{\boxed{\epsilon}}$ $\text{NN}_{\boxed{1}}$], $\text{NN}_{\boxed{1}}$⟩	E, A
⟨NP,NP⟩	→	⟨[NP $\text{DT}_{\boxed{\epsilon}}$ $\text{JJ}_{\boxed{\epsilon}}$ $\text{NN}_{\boxed{1}}$], [NP $\text{NN}_{\boxed{1}}$]⟩	C
⟨NP,NP⟩	→	⟨[NP $\text{DT}_{\boxed{1}}$ $\text{JJ}_{\boxed{2}}$ $\text{NN}_{\boxed{3}}$], [NP $\text{DT}_{\boxed{1}}$ $\text{JJ}_{\boxed{2}}$ $\text{NNS}_{\boxed{3}}$]⟩	P
⟨NP,NP⟩	→	⟨[NP $\text{DT}_{\boxed{1}}$ $\text{JJ}_{\boxed{2}}$ $\text{NN}_{\boxed{3}}$], [NP $\text{DT}_{\boxed{1}}$ $\text{NNP}_{\boxed{2}}$ $\text{NNP}_{\boxed{3}}$]⟩	P
⟨NP,NP⟩	→	⟨[NP $\text{DT}_{\boxed{1}}$ $\text{JJ}_{\boxed{2}}$ $\text{NN}_{\boxed{3}}$], [NP $\text{NP}_{\boxed{1}}$ $\text{CC}_{\boxed{3}}$ $\text{NP}_{\boxed{2}}$]⟩	P
⟨ADJP,PP⟩	→	⟨[ADJP [JJ due] [PP [TO to] $\text{NP}_{\boxed{1}}$]], [PP [RB because] [IN of] $\text{NP}_{\boxed{1}}$]⟩	P
⟨ADJP,JJ⟩	→	⟨[ADJP [RB very] [JJ good]], [JJ outstanding]⟩	P
⟨JJ,RB⟩	→	⟨[JJ first], [RB initially]⟩	A, P
⟨JJ,NNP⟩	→	⟨[JJ first], [NNP prime]⟩	P
⟨JJ,JJ⟩	→	⟨[JJ first], [JJ initial]⟩	P
⟨S,S⟩	→	⟨[S $\text{S}_{\boxed{1}}$ [CC and] $\text{S}_{\boxed{2}}$], [S $\text{S}_{\boxed{2}}$ [CC and] $\text{S}_{\boxed{1}}$]⟩	P

E = extractive compression corpus, A = abstractive compression corpus, C = coverage of test sentences (copy or delete rules), P = pivoting using multilingual corpora.

the sum of their component grammar sizes because of duplicate rules. As well as using different grammars, each of these models was trained on the corresponding dataset: the extractive compression training set, abstractive compression training set, or the union of both. The models' performance was evaluated on development set comprising 22 sentences taken from the abstractive compression corpus. 110 sentences were reserved for testing and used in the experiments reported in Section 7. We used the features described in Section 5.5, the Hamming loss function over tokens, and a trigram language model trained on the BNC (100 million words) using the SRI Language Modeling toolkit [Stolcke 2002], with modified Kneser-Ney smoothing.

We next asked two human judges to rate on a scale of 1 to 5 the system's compressions when optimized for different grammar rules. To get an idea of the quality of the output we also included human-authored abstractive reference compressions (Reference). Sentences given high numbers were both grammatical and preserved the most important information contained in the source sentence (without drastically altering its meaning). The mean ratings are shown in Table VI. As can be seen, the extractive compression system is rated higher, which is not surprising. By being more restrictive—it only performs deletions—this model has less scope for error and thus produces more grammatical output. Training our model only on the abstractive compression corpus obtains inferior performance, due partly to the smaller training set and also to the task being considerably more difficult. Moreover, while the abstractive compression system has access to some paraphrase rules, it has only few such rules (compared to the vast number of possible paraphrases) and little evidence of when they should be applied. The union of the extract and abstract rules improves over using the abstract rules alone. However, this model still has little knowledge of paraphrasing. Enhancing the union rules with pivot rules harvested from multilingual data improves the system output considerably. Finally, note that all model variants fall short of the human output which receives an average rating of 4.79 (in comparison, the extractive compression model has a rating of 2.84 and the best abstractive compression system a rating of 2.79). The differences between the various grammars are illustrated in Table VI. While the grammar extracted from the abstractive compression corpus contains many deletion rules, it has comparatively few paraphrases or reordering patterns, which is not surprising due to the small size of the training set. In contrast, the pivoted grammar contains a richer variety of paraphrases, covering many different words and phrases,

Table VI. Mean Ratings on System Output (development set) for Models Trained with Various Different Grammars

Grammar	Rating	Rules
Extract	2.84	51,011
Abstract	2.63	26,778
Extract+Abstract	2.68	54,213
Extract+Abstract+Pivot	2.79	180,142
Reference	4.79	—

Also shown are the number of rules in each grammar.

and therefore licenses a much broader set of abstractive compressions for input trees at test time.

Evaluation. Sentence compression output is commonly evaluated by eliciting human judgments. Following Knight and Marcu [2002], we asked participants to rate the grammaticality of the target compressions and how well they preserved the most important information from the source. In both cases they used a five-point rating scale where a high number indicates better performance.

We randomly selected 30 sentences from the test portion of our corpus. These sentences were compressed automatically by two configurations of our model: one trained on the union of extractive and abstractive compression rules (Extract+Abstract) and another one trained on the union and pivot rules (Extract+Abstract+Pivot). We also compared the output of these systems against a purely extractive one [Cohn and Lapata 2009]. Note that the latter model is a state-of-the-art extractive compression system; it performed significantly better than competitive extractive approaches [McDonald 2006] across a variety of corpora. All three systems were tuned so as to provide a similar compression rate.[22] We also asked participants to rate the gold-standard abstractive compressions. Our materials thus consisted of 120 (30×4) source-target sentences. We collected ratings from 27 unpaid volunteers, all self-reported native English speakers. The study was conducted over the Internet using the WebExp software package [Keller et al. 2009]. The experimental instructions are given in Appendix C.

7. RESULTS

Our results are summarized in Table VII, where we show grammaticality and importance mean ratings for the extractive compression system (Extract) and two versions of our abstractive compression system (Extract+Abstract and Extract+Abstract+Pivot). 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 (significant over both subjects and items at $p < 0.01$).

We next examined in more detail between-system differences. Post hoc Tukey tests revealed that the grammaticality ratings obtained for Extract and Extract+Abstract+Pivot are not significantly different, indicating that both systems produce comparable output. The Extract+Abstract system is significantly worse than Extract and Abstract+Extract+Pivot ($\alpha < 0.05$), again with regard to grammaticality. This is not entirely surprising, as the model attempts to use paraphrase rules

[22]It would be preferable to compare the output of the systems at their natural compression rates, however, this is difficult to do objectively. This is because the evaluation metrics (automatic and manual) are strongly biased towards longer outputs. It is much more difficult to compress well at lower compression rates while remaining grammatical and not discarding key information. To eliminate this bias we compare compression outputs with similar average compression rates.

Table VII. Mean Ratings on Compression Output Elicited by Humans

Models	Grammaticality	Importance	CompR	Words/Sent
Extract	3.67*	3.20*	78.7	21.7
Extract+Abstract	2.96*†	3.30*	80.0	22.4
Extract+Abstract+Pivot	3.65*	3.60*†	78.5	21.9
Reference	4.69	4.18	58.0	15.4

*: significantly different from the gold standard; †: significantly different from Extract. Also shown are the compression rate (macroaveraged) and the average number of words per sentence (microaveraged); the input had 27.8 words per sentence.

but since it has seen only a few of them, their application is mostly infelicitous, resulting in awkward sentences that our participants rate unfavorably. Compared to a purely extractive compression system, an abstractive compression model has to work a lot harder to preserve grammaticality, since it allows for arbitrary rewrites which may lead to agreement or tense mismatches and selectional preference violations. The scope for errors is greatly reduced when performing solely deletions.

As far as importance is concerned, our abstractive compression models receive higher ratings than the extractive compression system and the difference is statistically significant for Extract+Abstract+Pivot ($\alpha < 0.01$). We conjecture that this is due to the synchronous grammar we use which is larger and more expressive than the one employed by the extractive compression system. In the latter case, a word sequence is either deleted or retained. We may, however, want to retain the meaning of the sequence while rendering the sentence shorter, and this is precisely what our model can achieve, for example, by allowing substitutions. Finally, both the abstractive and extractive compression outputs are perceived as significantly worse than the gold standard, both in terms of grammaticality and importance ($\alpha < 0.01$). This is not surprising: human-authored compressions are more fluent and tend to omit genuinely superfluous information. This is also mirrored in the compression rates shown in Table VII. When compressing, humans employ not only linguistic but also world knowledge which is not accessible to our model. Although the system can be forced to match the human compression rate, the grammaticality and information content both suffer. More sophisticated features could allow the system to narrow this gap.

We also measured how well our participants agreed in their ratings. We employed leave-one-out resampling [Weiss and Kulikowski 1991], by correlating the data obtained from each participant with the ratings obtained from all other participants. We used Spearman's ρ, a nonparametric correlation coefficient, to avoid making any assumptions about the distribution of the ratings. The average inter-subject agreement on grammaticality was $\rho = 0.75$ and on importance $\rho = 0.72$.[23] We believe that this level of agreement is reasonably good indicating that participants can reliably judge the output of our systems on the dimensions of grammaticality and importance. The fact that participants agree with regard to importance is interesting given that the notion can be subjective and was defined rather loosely in our experimental instructions (see Appendix C).

We next examined the output of our system in more detail by recording the number of substitutions, deletions, and insertions it performed on the test data. Deletions accounted for 67% of rewrite operations, substitutions for 27%, and insertions for 6%. Interestingly, we observe a similar ratio in the human compressions. Here, deletions are also the most common rewrite operation (69%) followed by substitutions (24%),

[23]Note that Spearman's rho tends to yield lower coefficients compared to parametric alternatives such as Pearson's r.

Table VIII. Compression Examples Including Human and System Output

O:	Kurtz came from Missouri, and at the age of 14, hitch-hiked to Los Angeles seeking top diving coaches.
E:	Kurtz came from Missouri, and at 14, hitch-hiked to Los Angeles seeking top diving coaches.
A:	Kurtz hitch-hiked to Los Angeles seeking top diving coaches.
G:	Kurtz came from Missouri, and at 14, hitch-hiked to Los Angeles seeking diving coaches.
O:	The scheme was intended for people of poor or moderate means.
E:	The scheme was intended for people of poor means.
A:	The scheme was planned for poor people.
G:	The scheme was intended for the poor.
O:	He died last Thursday at his home from complications following a fall, said his wife author Margo Kurtz.
E:	He died last at his home from complications following a fall, said wife, author Margo Kurtz.
A:	His wife author Margo Kurtz died from complications after a decline.
G:	He died from complications following a fall.
O:	But a month ago, she returned to Britain, taking the children with her.
E:	She returned to Britain, taking the children.
A:	But she took the children with him.
G:	But she returned to Britain with the children.
O:	Firstly, the Swapo-democrat emblem is placed just above the Swapo emblem on the ballot paper, meaning that it will be seen first.
E:	The Swapo-democrat emblem is placed just above the Swapo emblem meaning that it will be seen first.
A:	Initially, the Swapo-democrat emblem is above the Swapo emblem on the ballot paper.
G:	Firstly, the Swapo-democrat emblem, placed above the Swapo emblem, will be seen first.
O:	That was conceded in an interview yesterday by Dr Kenneth Abrahams, a member of the National Front and a former Swapo man who was thrown into jail by Mr Nujoma.
E:	That was conceded in an interview by Dr Kenneth Abrahams, a member of the national front and a former Swapo man who was thrown into jail by Nr Nujoma.
A:	That was conceded in an interview of Dr Kenneth Abrahams, a National Front member who was thrown into jail by Mr Nujoma.
G:	Dr Kenneth Abrahams of the National Front conceded that.

(O: original sentence, E: extractive compression model, A: abstractive compression model, G: gold standard).

and insertions (7%). The ability to perform substitutions and insertions increases the compression potential of our system, but can also result in drastic meaning changes. We therefore inspected the compressions produced by the automatic systems in Table VII and the gold standard and recorded whether they preserved the meaning of the source. In most cases (69%) the compressions produced by the extractive compression system retained the meaning of the source. The abstractive compression systems performed better, with Extract+Abstract preserving the meaning of the source 82% of the time, and Extract+Abstract+Pivot 85%. Humans are clearly better at this, as 96.5% of their compressions were meaning preserving.

We illustrate example output of our Extract+Abstract+Pivot system in Table VIII. For comparison we also present the gold-standard compressions and output of the extractive compression system. In the first sentence the system rendered *Kurtz* the subject of *hitch-hiked*. At the same time it deleted the verb and its adjunct from the first conjunct (*came from Missouri*) as well as the temporal modifier *at the age of 14* from the second conjunct. The second sentence shows some paraphrasing: the verb *intended* is substituted with *planned* and *poor* is now modifying *people* rather than *means*. In the third example, our system applies multiple rewrites. It deletes *last Thursday at his home*, moves *wife author Margo Kurtz* to the subject position, and substitutes *fall* with *decline*.

Unfortunately, the compressed sentence expresses a rather different meaning from the original. It is not Margo Kurtz who died but her husband. The fourth sentence illustrates a fair degree of compression and paraphrasing: the infinitival clause *meaning that it will be seen first* is dropped, the verbal clause *is placed* is substituted with *is*, and the adverbial *firstly* with *initially*. Finally, our last example is not as compressed as the human gold standard (the original sentence has 32 tokens and is reduced to 10). However, it demonstrates an interesting rewrite: the prepositional phrase *a member of the National Front* is paraphrased with the compound noun *a National Front member*.

8. DISCUSSION

In this article we have presented an end-to-end text rewriting system that simultaneously compresses and paraphrases sentences. We have shown that abstractive sentence compression is a meaningful task which humans can perform with relative ease while employing several rewrite operations in addition to deletion. Importantly, the greater flexibility of the abstractive compression task permits better compression rates compared to word deletion, and thus holds promise for a variety of applications that must produce shortened textual output.

We have proposed a discriminative tree-to-tree transduction model for the abstractive compression that can account for structural and lexical mismatches. The model incorporates a synchronous tree substitution grammar which encodes a large space of paraphrasing rules and is extracted from bilingual corpora. Experimental evaluation shows that our approach yields shorter target sentences that are grammatical and (mostly) preserve the meaning of the longer source sentences while using rewrite rules. Although we have applied this modeling framework to the compression task, we argue that it can be easily ported to other rewriting applications such as text simplification [Chandrasekar and Srinivas 1996] and even fully abstractive document summarization [Daumé III and Marcu 2002]. The abstractive compression task itself could also serve as a testbed for paraphrase induction systems whose rewrite rules are often evaluated out-of-context.

Possible extensions and improvements to the current model are many and varied. Firstly, as hinted at earlier, the model would benefit from extensive feature engineering, including source conditioned features and n-gram features besides the language model. For example, the model parameters from Galley and McKeown [2007] (e.g., conditioning deletions on syntactic contexts of variable length, treating head-modifier relations independently, lexicalization of the synchronous productions) could be easily included as features in our approach. Importantly, our model can incorporate all kinds of nonindependent features while tailoring the optimization objective more directly to the task at hand. A richer grammar would also boost performance. This could be obtained from existing paraphrase resources such as the Multiple-Translation Arabic and Chinese corpora. Wikipedia is another valuable resource for text rewriting. For example, we could learn rewrite rules from Wikipedia's revision histories [Yamangil and Nelken 2008].

The approach presented here applies to individual sentences rather than entire documents. Although extracting a document-level synchronous grammar is computationally expensive [Daumé III and Marcu 2002] potentially leading to prohibitively many rules, it is possible to render the model more document aware by incorporating discourse-specific features [Clarke and Lapata 2010]. For example, decisions on whether to remove or retain a word (or phrase) could be informed by its discourse properties (e.g., whether it introduces a new topic, or whether it is semantically related to the previous sentence). An obvious extension would be to interface our compression model with sentence extraction, for example, by adopting a two-stage architecture where the sentences are first extracted and then compressed or the other way round [Lin 2003].

Finally, an interesting direction for future work is the development of loss functions that are more suited to the abstractive compression task. The loss function employed in our experiments was based on the Hamming distance over unordered bags of tokens. Ideally, we would like a loss that guides the model towards shorter output that is meaning preserving. Unfortunately, this is not so easy to measure while maintaining a shallow approach. We could compute meaning equivalence by resorting to WordNet [Fellbaum 1998] or taking advantage of recent advances in recognizing textual entailment [Padó et al. 2009] and compositional vector-based models [Mitchell and Lapata 2010].

APPENDIX

A. ANNOTATION INSTRUCTIONS

This annotation task is concerned with sentence compression. You will be presented with a selection of newspaper articles. Your task is to read each sentence in the article and compress it so as to produce a shorter version. The compressed sentence should be grammatical and retain the most important information of the original sentence without changing its meaning.

In producing compressions, you are free to delete words, add new words, substitute them, or reorder them. While doing the task you will find that word deletion is the most frequent compression operation. You should use substitution, insertion, and reordering operations only if they render the original sentence shorter. In other words, we are only interested in rewriting operations that reduce the original sentence. Simply paraphrasing the original without reducing its length will not yield appropriate compressions.

There are several rewrite operations you may wish to apply in order to render the original sentence shorter. For instance, you may delete appositions or parentheticals, relative clauses, or you may rewrite a passive verb as an active one. In other cases you may choose to substitute two or more words with a shorter word or phrase that conveys a similar meaning. You can find a list of examples illustrating a variety of rewrite operations here.[24] We recommend that you study these examples before embarking on the annotation.

Although there is a certain degree of flexibility in creating compressions, you should not rewrite a sentence as two or more sentences. In other words you should produce one sentence (possibly with a main and subordinate clauses) but not multisentence output (e.g., a discourse). You are also not allowed to delete any sentence from the original document.

A small number of sentences will be very short or will contain no information and thus will not be amenable to compression. When you come across such a sentence you should not perform any rewriting operations. The original and compressed sentence are identical in this case.

The annotation will proceed on a document-by-document basis. In compressing individual sentences you should ensure that the resulting (compressed) document is coherent. This will be relatively easy to enforce and in most cases will come naturally with your compressions.

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;
—rewrite operations that reduce the length of the original sentence;
—rewrite operations that do not distort the meaning of the original sentence.

[24]We omit the list of examples from the sake of brevity; most of these were taken from Dras [1999].

The interface will present you with a selection of documents to choose from. Once you are done with your annotation, please hit the “submit” button and your compressions will be automatically saved. It is also possible to view and revise your compressions; simply go to the Document Selector, choose the document you wish to change, and hit the reload button. Once you load a document, 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.

Each sentence will be followed by a Compressed Sentence box for writing down its corresponding compression. If the sentence cannot be compressed, then please copy the original sentence verbatim in the Compressed Sentence box. The interface is illustrated next.

Source Sentence:
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.

Compressed Sentence:
Sergei practices healing of the spirit with his hands and a wire measuring energy fields causing illness and depression.

Before starting the annotation task make sure to study some examples of compressed sentences. In addition to the individual example sentences mentioned before, we have also provided you with a fully compressed document. Please read it to get an idea of how your compressions should look. Finally, if you have any questions or comments regarding this experiment please contact us.

B. ANNOTATION EXAMPLE

The following table illustrates a source document from our corpus and its compressed version as created by our annotator.

Source	Target
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 town of Zafferana is threatened by Mount Etna, still untamed due to snow, winds and disagreement
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.	The lava has stopped 150 yards from the town, but its flow is accelerating 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.	Lava began creeping over the formed crust and into an orchard.
Bad weather dashed hopes of attempts to halt the flow during what was seen as a natural lull in the lava’s momentum.	Bad weather stopped attempts to hold the lava’s flow.

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.	Even if the eruption stopped today, the lava's pressure would bring debris on the town anyway.
Some estimate the volcano is pouring out one million tons of debris a day, at a rate of 15 ft per second, from a fissure that opened in mid-December.	One million tons of debris a day is pouring out of a fissure that opened in mid-December.
The Italian army yesterday detonated nearly 400 lb of dynamite 3,500 feet up Mount Etna's slopes.	The army yesterday detonated 400 lb of dynamite on the 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.	The explosives were detonated experimentally above a dam 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.	They closed off the third of five underground magma conduits.
But the teams later discovered that the conduit was dry, suggesting that the lava had already found a new course.	The conduit was dry, suggesting the lava had found a new course.
Rumours have been circulating that experts are bitterly divided over what to do.	Rumour has it experts are 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 6,750 ft above sea level.	In another experiment, concrete slabs are to be dumped from a 6,750 ft height.
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.	The slabs may block the main lava conduit.
High winds and snowfalls have, however, grounded at a lower level the powerful Us Navy Sea Stallion helicopters used to transport the slabs.	Bad weather, however, has grounded the helicopters transporting the slabs.
Prof Letterio Villari, a noted vulcanologist, said yesterday he had "absolutely no faith whatsoever" in the plan.	Vulcanologist Prof Letterio Villari has no faith 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.	Saving Zafferana from the lava would be a fluke, he said.

C. EXPERIMENTAL 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 program.

Your task is to judge how well a compressed sentence paraphrases the original sentence. You will judge each compression according to two criteria: (a) grammaticality, and (b) importance. The grammaticality judgment is based on whether the sentence is understandable. The importance judgment relates to how well the compression preserves the most important information of the original (without distorting its meaning) and whether it is adequately compressed. Both judgments 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, change its meaning, or do not remove any superfluous information even though it is evident that it can be omitted without drastic information loss. A good compression is one that is readily comprehensible, retains the most important information from the original sentence, and is meaning preserving. Good sentence compressions should receive a high grammaticality and importance scores. For example, if you were asked to rate the following compression indicated in boldface:

> 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 boss 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. It should receive a low score for importance (e.g., 1 or 2) as it is not 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 initiated a change - as 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). Now suppose that you were given the following 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.

> **Nonetheless, FBI director Louis Freeh has 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.**

Here, the sentence is not compressed very much at all. In fact it is identical to the original except for the word *today*. Although this sentence would receive a high score in terms of grammaticality, it should have a low importance score (probably 1) since it has not removed any extraneous information. 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 introduced new restrictions on information sharing 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 and its corresponding compression which will always be presented in bold. Read the compression then make your judgments. There are no "correct" answers, so whatever numbers seem appropriate to you are 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 very important to the original sentence?
—Does the compressed sentence seem fluent?
—Has the compressed sentence preserved the meaning of the original sentence?

Use high numbers if the answer to the preceding 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 and base your judgments on your first impressions. The experiment will take approximately 20 minutes.

ACKNOWLEDGMENTS

We are grateful to the three anonymous referees whose feedback helped to improve the present article. Special thanks to Phil Blunsom, James Clarke and Miles Osborne for their insightful suggestions.

REFERENCES

AHO, A. V. AND ULLMAN, J. D. 1969. Syntax directed translations and the pushdown assembler. *J. Comput. Syst. Sci. 3*, 37–56.

BANNARD, C. AND CALLISON-BURCH, C. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 597–604.

BARZILAY, R. 2003. Information fusion for multi-document summarization: Paraphrasing and generation. Ph.D. thesis, Columbia University, New York.

BARZILAY, R. AND ELHADAD, N. 2003. Sentence alignment for monolingual comparable corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. M. Collins and M. Steedman, Eds., Association for Computational Linguistics, 25–32.

Barzilay, R. and Lee, L. 2003. Learning to paraphrase: An unsupervised approach using multiple sequence alignment. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics.* 16–23.

Barzilay, R. and Mckeown, K. 2001. Extracting paraphrases from a parallel corpus. In *Proceedings of the 39th Annual Meeting of the Association for Computational Linguistics*. N. Reithinger and G. Satta, Eds., Association for Computational Linguistics, 50–57.

Barzilay, R. and Mckeown, K. R. 2005. Sentence fusion for multidocument news summarization. *Comput. Linguist. 31*, 3, 297–327.

Bhagat, R. and Ravichandran, D. 2008. Large scale acquisition of paraphrases for learning surface patterns. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics with the Human Language Technology Conference*. J. D. Moore, S. Teufel, J. Allan, and S. Furui, Eds., Association for Computational Linguistics, 674–682.

Bikel, D. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of the 2nd International Conference on Human Language Technology Research (HLT'02)*. Morgan Kaufmann Publishers, San Francisco, 24–27.

Brown, P. F., Pietra, S. A. D., Pietra, V. J. D., and Mercer, R. L. 1993. Mathematics of statistical machine translation: Parameter estimation. *Comput. Linguist. 19*, 2, 263–311.

Callison-Burch, C. 2007. Paraphrasing and translation. Ph.D. thesis, University of Edinburgh, U.K.

Callison-Burch, C. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. M. Lapata and H. T. Ng, Eds., Association for Computational Linguistics, 196–205.

Chandrasekar, R. and Srinivas, C. D. B. 1996. Motivations and methods for text simplification. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING'96).* 1041–1044.

Chiang, D. 2007. Hierarchical phrase-based translation. *Comput. Linguist. 33*, 2, 201–228.

Clarke, J. 2008. Global inference for sentence compression: An integer linear programming approach. Ph.D. thesis, University of Edinburgh.

Clarke, J. and Lapata, M. 2008. Global inference for sentence compression: An integer linear programming approach. *J. Artif. Intell. Res. 31*, 273–381.

Clarke, J. and Lapata, M. 2010. Discourse constraints for document compression. *Comput. Linguist. 36*, 3, 411–441.

Cohn, T. and Lapata, M. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics (COLING'08)*. D. Scott and H. Uszkoreit, Eds., 137–144.

Cohn, T. and Lapata, M. 2009. Sentence compression as tree transduction. *J. Artif. Intell. Res. 34*, 637–674.

Corston-Oliver, S. 2001. Text compaction for display on very small screens. In *Proceedings of the NAACL Workshop on Automatic Summarization*. J. Goldstein and C.-Y. Lin, Eds., Association for Computational Linguistics, 89–98.

Daume III, H. and Marcu, D. 2002. A noisy-channel model for document compression. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. E. Charniak and D. Lin, Eds., Association for Computational Linguistics, 449–456.

Dorr, B., Zajic, D., and Schwartz, R. 2003. Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the HLT-NAACL Text Summarization Workshop*. D. Radev and S. Teufel, Eds., Association for Computational Linguistics, 1–8.

Dras, M. 1999. Tree adjoining grammar and the reluctant paraphrasing of text. Ph.D. thesis, Macquarie University, Australia.

Eisner, J. 2003. Learning non-isomorphic tree mappings for machine translation. In *Proceedings of the ACL Interactive Poster/Demonstration Sessions*. Association for Computational Linguistics, 205–208.

Fellbaum, C., Ed. 1998. *WordNet: An Electronic Database*. MIT Press, Cambridge, MA.

Galley, M., Hopkins, M., Knight, K., and Marcu, D. 2004. What's in a translation rule? In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics (HLT-NAACL'04).* Association for Computational Linguistics, 273–280.

Galley, M. and Mckeown, K. 2007. Lexicalized markov grammars for sentence compression. In *Proceedings of the NAACL-HLT Conference of the North American Chapter of the Association for Computational Linguistics, Human Language Technologies*. C. Sidner, T. Schultz, M. Stone, and C. Zhai, Eds., Association for Computational Linguistics, 180–187.

Ganitkevitch, J., Callison-Burch, C., Napoles, C., and Van Durme, B. 2011. Learning sentential paraphrases from bilingual parallel corpora for text-to-text generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 1168–1179.

GREFENSTETTE, G. 1998. Producing intelligent telegraphic text reduction to provide an audio scanning service for the blind. In *Proceedings of the AAAI Symposium on Intelligent Text Summarization*. E. Hovy and D. R. Radev, Eds., The AAAI Press, 111–117.

HABASH, N. AND LAVIE, A., EDS. 2006. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA'06).*

HEARST, M. AND OSTENDORF, M., EDS. 2003. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics.

HIRAO, T., SUZUKI, J., AND ISOZAKI, H. 2009. A syntax-free approach to japanese sentence compression. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Association for Computational Linguistics. 826–833.

HORI, C. AND FURUI, S. 2004. Speech summarization: an approach through word extraction and a method for evaluation. *IEICE Trans. Inf. Syst. E87-D*, 1, 15–25.

HUANG, L., KNIGHT, K., AND JOSHI, A. 2006. Statistical syntax-directed translation with extended domain of locality. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA'06).* 66–73.

JING, H. 2000. Sentence reduction for automatic text summarization. In *Proceedings of the 6th Applied Natural Language Processing Conference*. S. Nirenburg, Ed., Association for Computational Linguistics, PA, 310–315.

JOACHIMS, T. 2005. A support vector method for multivariate performance measures. In *Proceedings of the 22nd International Conference on Machine Learning*. L. D. Raedt and S. Wrobel, Eds., ACM Press, New York, 377–384.

KELLER, F., GUNASEKHARAN, S., MAYO, N., AND CORLEY, M. 2009. Timing accuracy of web experiments: A case study using the WebExp software package. *Behav. Res. Methods 41*, 1, 1–12.

KNIGHT, K. AND MARCU, D. 2002. Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artif. Intell. 139*, 1, 91–107.

KNIGHT, K., NG, H. T., AND OFLAZER, K., EDS. 2005. *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics.

KOEHN, P., OCH, F. J., AND MARCU, D. 2003. Statistical phrase-based translation. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics. 48–54.

LIANG, P., TASKAR, B., AND KLEIN, D. 2006. Alignment by agreement. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL'06)*. R. C. Moore, J. Bilmes, J. Chu-Carroll, and M. Sanderson, Eds., Association for Computational Linguistics, 104–111.

LIN, C.-Y. 2003. Improving summarization performance by sentence compression — A pilot study. In *Proceedings of the 6th International Workshop on Information Retrieval with Asian Languages*. J. Adachi and K.-F. Wong, Eds., Association for Computational Linguistics, 1–8.

LIN, D. AND PANTEL, P. 2001. Discovery of inference rules for question answering. *Natural Lang. Engin. 7*, 4, 342–360.

LIU, Y., LIU, Q., AND LIN, S. 2006. Tree-to-string alignment template for statistical machine translation. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*. O. Kwong, Ed., Association for Computational Linguistics, 609–616.

MARCU, D. 1999. The automatic construction of large-scale corpora for summarization research. In *Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval (SIGIR'99).* ACM Press, New York, 137–144.

MARTINS, A. F. T. AND SMITH, N. A. 2009. Summarization with a joint model for sentence extraction and compression. In *Proceedings of the Workshop on Integer Linear Programming for Natural Language Processing*. Association for Computational Linguistics, 1–9.

MCDONALD, R. 2006. Discriminative sentence compression with soft syntactic constraints. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*. D. McCarthy and S. Wintner, Eds., Association for Computational Linguistics, 297–304.

MITCHELL, J. AND LAPATA, M. 2010. Composition in distributional models of semantics. *Cogn. Sci. 34*, 8, 1388–1429.

NGUYEN, M. L., SHIMAZU, A., HORIGUCHI, S., HO, T. B., AND FUKUSHI, M. 2004. Probabilistic sentence reduction using support vector machines. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING'04).* 743–749.

Och, F. J. and Ney, H. 2004. The alignment template approach to statistical machine translation. *Comput. Linguist. 30*, 4, 417–449.

Pado, S., Cer, D., Galley, M., Jurafsky, D., and Manning, C. D. 2009. Measuring machine translation quality as semantic equivalence: A metric based on entailment features. *Mach. Transl. 23*, 2–3, 181–193.

Pang, B., Knight, K., and Marcu, D. 2003. Syntax-based alignment of multiple translations: Extracting paraphrases and generating new sentences. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics. 181–188.

Papineni, K., Roukos, S., Ward, T., and Zhu, W.-J. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th Annual Meeting of the Association for Computational Linguistics*. E. Charniak and D. Lin, Eds., Association for Computational Linguistics, PA, 311–318.

Quirk, C., Brockett, C., and Dolan, W. 2004. Monolingual machine translation for paraphrase generation. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*. Association for Computational Linguistics, 142–149.

Riezler, S., King, T. H., Crouch, R., and Zaenen, A. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics.* Association for Computational Linguistics. 118–125.

Shieber, S. and Schabes, Y. 1990. Synchronous tree-adjoining grammars. In *Proceedings of the 13th International Conference on Computational Linguistics (COLING'90).* Vol. 3. 253–258.

Snover, M., Dorr, B., Schwartz, R., Micciulla, L., and Makhoul, J. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of the 7th Conference of the Association for Machine Translation in the Americas (AMTA'06).* 223–231.

Stolcke, A. 2002. SRILM – An extensible language modeling toolkit. In *Proceedings of the 7th International Conference on Spoken Language Processing*. J. H. L. Hansen and B. Pellom, Eds., Casul Prod. Ltd., Denver, CO.

Su, K.-Y., Su, J., Wiebe, J., and Li, H., Eds. 2009. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Association for Computational Linguistics.

Tsochantaridis, I., Joachims, T., Hofmann, T., and Altun, Y. 2005. Large margin methods for structured and interdependent output variables. *J. Mach. Learn. Res. 6*, 1453–1484.

Turner, J. and Charniak, E. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, 290–297.

Vandeghinste, V. and Pan, Y. 2004. Sentence compression for automated subtitling: A hybrid approach. In *Proceedings of the ACL Workshop on Text Summarization*. Association for Computational Linguistics, 89–95.

Weiss, S. M. and Kulikowski, C. A. 1991. *Computer Systems that Learn: Classification and Prediction Methods from Statistics, Neural Nets, Machine Learning, and Expert Systems*. Morgan Kaufmann, San Fransisco, CA.

Yamangil, E. and Nelken, R. 2008. Mining wikipedia revision histories for improving sentence compression. In *Proceedings of the ACL-HLT Short Papers.* Association for Computational Linguistics, 137–140.

Zajic, D. M., Dorr, B. J., Lin, J., and Schwartz, R. 2007. Multi-candidate reduction: Sentence compression as a tool for document summarization tasks. *Inf. Process. Manag. 43*, 1549–1570.

Zhao, S., Lan, X., Liu, T., and Li, S. 2009. Application-driven statistical paraphrase generation. In *Proceedings of the Joint Conference of the 47th Annual Meeting of the ACL and the 4th International Joint Conference on Natural Language Processing of the AFNLP*. Association for Computational Linguistics, 834–842.

Received February 2011; revised July 2011; accepted November 2011