Sentence Compression Beyond Word Deletion

Trevor Cohn and **Mirella Lapata**
School of Informatics
University of Edinburgh
{tcohn,mlap}@inf.ed.ac.uk

Abstract

In this paper we generalise 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 a new corpus that is suited to our task and a discriminative tree-to-tree transduction model that can naturally account for structural and lexical mismatches. The model incorporates a novel grammar extraction method, uses a language model for coherent output, and can be easily tuned to a wide range of compression specific loss functions.

1 Introduction

Automatic sentence compression can be broadly described as the task of creating a grammatical summary of a single sentence with minimal information loss. It has recently attracted much attention, in part because of its relevance to applications. Examples include the generation of subtitles from spoken transcripts (Vandeghinste and Pan, 2004), the display of text on small screens such as mobile phones or PDAs (Corston-Oliver, 2001), and, notably, summarisation (Jing, 2000; Lin, 2003).

Most prior work has focused on a specific instantiation of sentence compression, namely word deletion. Given an input sentence of words, $w_1, w_2 \ldots w_n$, a 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 a dynamic program (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, 2007), and Integer Linear Programming (Clarke, 2008). 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).

Therefore, in this paper we consider sentence compression from a more general perspective and generate *abstracts* rather than *extracts*. 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 in order to generate new text, while attempting to produce a shorter resulting string. Quirk et al. (2004) 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 model is limited to lexical substitution — no reordering takes place — and is

Proceedings of the 22nd International Conference on Computational Linguistics (Coling 2008), pages 137–144
Manchester, August 2008

lacking the compression objective.

Once we move away from extractive compression we are faced with two problems. First, we must find an appropriate training set for our abstractive task. Compression corpora are not naturally available and existing paraphrase corpora do not normally contain compressions. Our second problem concerns the modeling task itself. Ideally, our learning framework should handle structural mismatches and complex rewriting operations.

In what follows, we first present a new corpus for abstractive compression which we created by having annotators compress sentences while rewriting them. Besides obtaining useful data for modeling purposes, we also demonstrate that abstractive compression is a meaningful task. 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, Eisner (2003)), a formalism that can account for structural mismatches, and is trained discriminatively. Specifically, we generalise the model of Cohn and Lapata (2007) to our abstractive task. We present a novel tree-to-tree grammar extraction method which acquires paraphrases from bilingual corpora and ensure coherent output by including a ngram language model as a feature. We also develop a number of loss functions suited to the abstractive compression task. We hope that some of the work described here might be of relevance to other generation tasks such as machine translation (Eisner, 2003), multi-document summarisation (Barzilay, 2003), and text simplification (Carroll et al., 1999).

2 Abstractive Compression Corpus

A stumbling block to studying abstractive sentence compression is the lack of widely available corpora for training and testing. Previous work has been conducted almost exclusively on Ziff-Davis, a corpus derived automatically from document abstract pairs (Knight and Marcu, 2002), or on human-authored corpora (Clarke, 2008). Unfortunately, none of these data sources are suited to our problem since they have been produced with a single rewriting operation, namely word deletion. Although there is a greater supply of paraphrasing corpora, such as the Multiple-Translation Chinese (MTC) corpus[1] 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. They contain ample rewriting operations, however they do not explicitly target information loss.

For the reasons just described, we created our own corpus. We collected 30 newspaper articles (575 sentences) from the British National Corpus (BNC) and the American News Text corpus, for which we obtained manual compressions. In order to confirm that the task was feasible, five of these documents were initially compressed by two annotators (not the authors). 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. They were encouraged to use any rewriting operations that seemed appropriate, e.g., to delete words, add new words, substitute them or reorder them.

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 default 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.[2] We also assessed whether they agreed in their rewrites by measuring BLEU (Papineni et al., 2002). The inter-annotator BLEU score was 23.79%, compared with the source agreement BLEU of only 13.22%. Both the compression rate and BLEU score indicate that the task is well-defined and the compressions valid. The remaining 25 documents were compressed by a single annotator to ensure consistency. All our experiments used the data from this annotator.[3]

Table 1 illustrates some examples from our corpus. As can be seen, some sentences contain a single rewrite operation. For instance, a 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). Here, the conjunction *high winds and snowfalls* is abbreviated to

[1] Available by the LDC, Catalog Number LDC2002T01, ISBN 1-58563-217-1.

[2] The term "compression rate" refers to the percentage of words retained in the compression.

[3] Available from `http://homepages.inf.ed.ac.uk/tcohn/paraphrase`.

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.

Table 1: Compression examples from our corpus; (a) sentences are the source, (b) sentences the target.

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*.

3 Basic Model

Our work builds on the model developed by Cohn and Lapata (2007). They formulate sentence compression as a tree-to-tree rewriting task. A synchronous tree substitution grammar (STSG, Eisner (2003)) licenses the space of all possible rewrites. Each grammar rule is assigned a weight, and these weights are learnt in discriminative training. For prediction, a specialised generation algorithm finds the best scoring compression using the grammar rules. Cohn and Lapata apply this model to extractive compression with state-of-the-art results.

This model is appealing for our task for several reasons. Firstly, the synchronous grammar provides expressive power to model consistent syntactic effects such as reordering, changes in non-terminal categories and lexical substitution. Secondly, it is discriminatively trained, which allows for the incorporation of all manner of powerful features. Thirdly, the learning framework can be tailored to the task by choosing an appropriate loss function. In the following we describe their model in more detail with emphasis on the synchronous grammar, the model structure, and the prediction and training algorithms. Section 4 presents our extensions and modifications.

Grammar The grammar defines a space of tree pairs over uncompressed and compressed sentences, which we refer to henceforth as the *source* and *target*. We use the grammar to find the set of sister target sentences for a given source sentence. Figure 1 shows a toy grammar and the set of possible target (output) trees for the given source (input) tree. Each output tree is created by applying a series of grammar rules, where each rule matches a fragment of the source and creates a fragment of the target tree. A rule in the grammar consists of a pair of elementary trees and a mapping between the variables (frontier non-terminals) in both trees. A *derivation* is a sequence of rules yielding a target tree with no remaining variables.

Grammar rules:
$\langle$S, S$\rangle$ $\rightarrow$ $\langle$NP$_{\boxed{1}}$ VBD$_{\boxed{2}}$ NP$_{\boxed{3}}$, NP$_{\boxed{1}}$ VBD$_{\boxed{2}}$ NP$_{\boxed{3}}\rangle$
$\langle$S, S$\rangle$ $\rightarrow$ $\langle$NP$_{\boxed{1}}$ VBD$_{\boxed{2}}$ NP$_{\boxed{3}}$, NP$_{\boxed{3}}$ was VBN$_{\boxed{2}}$ by NP$_{\boxed{1}}\rangle$
$\langle$NP, NP$\rangle$ $\rightarrow$ $\langle$he, him$\rangle$
$\langle$NP, NP$\rangle$ $\rightarrow$ $\langle$he, he$\rangle$
$\langle$NP, NP$\rangle$ $\rightarrow$ $\langle$he, Peter$\rangle$
$\langle$VBD, VBN$\rangle$ $\rightarrow$ $\langle$sang, sung$\rangle$
$\langle$NP, NP$\rangle$ $\rightarrow$ $\langle$a song, a song$\rangle$

Input tree:
[S [NP He$_{\text{NP}}$ [VP sang$_{\text{VBD}}$ [NP a$_{\text{DT}}$ song$_{\text{NN}}$]]]

Output trees:
[S [NP He] [VP sang [NP a song]]]
[S [NP Him] [VP sang [NP a song]]]
[S [NP Peter] [VP sang [NP a song]]]
[S [NP A song] [VP was [VP sung [PP by he]]]]
[S [NP A song] [VP was [VP sung [PP by him]]]]
[S [NP A song] [VP was [VP sung [PP by Peter]]]]

Figure 1: Example grammar and the output trees it licences for an input tree. The numbered boxes in the rules denote linked variables. Pre-terminal categories are not shown for the output trees for the sake of brevity.

Cohn and Lapata (2007) extract a STSG from a parsed, word-aligned corpus of source and target sentences. Specifically, they extract the *minimal set* of synchronous rules which can describe each tree pair. These rules are minimal in the sense that they cannot be made smaller (e.g., by replacing a subtree with a variable) while still honouring the word-alignment.

Decoding The grammar allows us to search for all sister trees for a given tree. The decoder maximises over this space:

$$\mathbf{y}^* = \arg\max_{\mathbf{y}:S(\mathbf{y})=\mathbf{x}} \Psi(\mathbf{y}) \quad (1)$$

$$\text{where} \quad \Psi(\mathbf{y}) = \sum_{r \in \mathbf{y}} \langle \phi(r, S(\mathbf{y})), \lambda \rangle \quad (2)$$

Here $\mathbf{x}$ is the source (uncompressed) tree, $\mathbf{y}$ is a derivation which produces the source tree, $\text{S}(\mathbf{y}) = \mathbf{x}$, and a target tree, $T(\mathbf{y})$,[4] and r is a grammar rule. The Ψ function scores the derivation and

[4] Equation 1 optimises over derivations rather than target trees to allow tractable inference.

is defined in (2) as a linear function over the rules used. Each rule's score is an inner product between its feature vector, $\phi(r, \mathbf{y}_S)$, and the model parameters, λ. The feature functions are set by hand, while the model parameters are learned in training.

The maximisation problem in (1) can be solved efficiently using a dynamic program. Derivations will have common sub-structures whenever they transduce the same source sub-tree into a target sub-tree. This is captured in a chart, leading to an efficient bottom-up algorithm. The asymptotic time complexity of this search is $O(SR)$ where S is the number of source nodes and R is the number of rules matching a given node.

Training The model is trained using SVM^{struct}, a large margin method for structured output problems (Joachims, 2005; Tsochantaridis et al., 2005). This training method allows the use of a configurable loss function, $\Delta(\mathbf{y}^*, \mathbf{y})$, which measures the extent to which the model's prediction, $\mathbf{y}$, differs from the reference, $\mathbf{y}^*$. Central to training is the search for a derivation which is both high scoring and has high loss compared to the gold standard.[5] This requires finding the maximiser of $H(\mathbf{y})$ in one of:

$$\begin{aligned} H_s &= (1 - \langle \Psi(\mathbf{y}^*) - \Psi(\mathbf{y}), \lambda \rangle)\Delta(\mathbf{y}^*, \mathbf{y}) \\ H_m &= \Delta(\mathbf{y}^*, \mathbf{y}) - \langle \Psi(\mathbf{y}^*) - \Psi(\mathbf{y}), \lambda \rangle \end{aligned} \quad (3)$$

where the subscripts s and m denote *slack* and *margin* rescaling, which are different formulations of the training problem (see Tsochantaridis et al. (2005) and Taskar et al. (2003) for details).

The search for the maximiser of $H(\mathbf{y})$ in (3) requires the tracking of the loss value. This can be achieved by extending the decoding algorithm such that the chart cells also store the loss parameters (e.g., for precision, the number of true and false positives (Joachims, 2005)). Consequently, this extension leads to a considerably higher time and space complexity compared to decoding. For example, with precision loss the time complexity is $O(S^3R)$ as each step must consider $O(S^2)$ possible loss parameter values.

4 Extensions

In this section we present our extensions of Cohn and Lapata's (2007) model. The latter was designed with the simpler extractive compression in mind and cannot be readily applied to our task.

[5] Spurious ambiguity in the grammar means that there are often many derivations linking the source and target. We follow Cohn and Lapata (2007) by choosing the derivation with the most rules, which should provide good generalisation.

Grammar It is relatively straightforward to extract a grammar from our corpus. This grammar will contain many rules encoding deletions and structural transformations but there will be many unobserved paraphrases, no matter how good the extraction method (recall that our corpus consists solely of 565 sentences). For this reason, we extract a grammar from our abstractive corpus in the manner of Cohn and Lapata (2007) (see Section 5 for details) and augment it with a larger grammar obtained from a parallel bilingual corpus. Crucially, our second grammar will not contain compression rules, just paraphrasing ones. We leave it to the model to learn which rules serve the compression objective.

Our paraphrase grammar extraction method uses bilingual pivoting to learn paraphrases over syntax tree fragments, i.e., STSG rules. Pivoting treats the paraphrasing problem as a two-stage translation process. Some English text is translated to a foreign language, and then translated back into English (Bannard and Callison-Burch, 2005):

$$p(e'|e) = \sum_f p(e'|f)p(f|e) \quad (4)$$

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.

Instead of using strings (Bannard and Callison-Burch, 2005), we use elementary trees on the English side, resulting in a monolingual STSG. We obtain the elementary trees and foreign strings using the GKHM algorithm (Galley et al., 2004). This takes as input a bilingual word-aligned corpus with trees on one side, and finds the minimal set of tree fragments and their corresponding strings which is consistent with the word alignment. This process is illustrated in Figure 2 where the aligned pair on the left gives rise to the rules shown on the right. Note that the English rules and foreign strings shown include variable indices where they have been generalised. We estimate $p(f|e)$ and $p(e'|f)$ from the set of tree-to-string rules and then then pivot each tree fragment to produce STSG rules. Figure 3 illustrates the process for the [VP does not VP] fragment.

Modeling and Decoding Our grammar is much larger and noisier than a grammar extracted solely for deletion-based compression. So, in order to encourage coherence and inform lexical se-

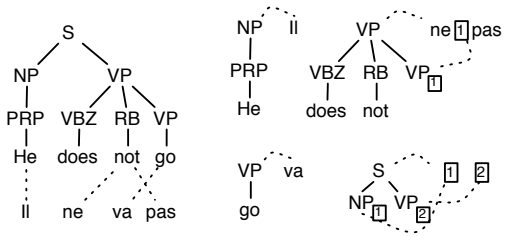

Figure 2: Tree-to-string grammar extraction using the GHKM algorithm, showing the aligned sentence pair and the resulting rules as tree fragments and their matching strings. The boxed numbers denote variables.

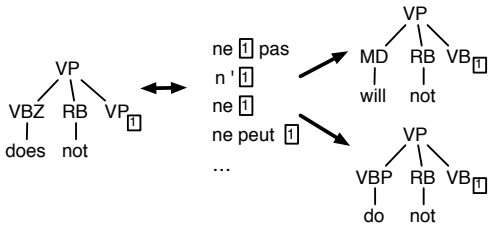

Figure 3: Pivoting the [VP does not VP] fragment.

lection we incorporate a ngram language model (LM) as a feature. This requires adapting the scoring function, Ψ, in (2) to allow features over target ngrams:

$$\Psi(\mathbf{y}) = \sum_{r \in \mathbf{y}} \langle \phi(r, S(\mathbf{y})), \lambda \rangle + \sum_{m \in T(\mathbf{y})} \langle \psi(m, S(\mathbf{y})), \lambda \rangle \quad (5)$$

where m are the ngrams and ψ is a new feature function over these ngrams (we use only one ngram feature: the trigram log-probability). Sadly, the scoring function in (5) renders the chart-based search used for training and decoding intractable. In order to provide sufficient context to the chart-based algorithm, we must also store in each chart cell the $n-1$ target tokens at the left and right edges of its yield. This is equivalent to using as our grammar the intersection between the original grammar and the ngram LM (Chiang, 2007), and increases the decoding complexity to an infeasible $O(SRL^{2(n-1)V})$ where L is the size of the lexicon. We adopt a popular approach in syntax-inspired machine translation to address this problem (Chiang, 2007). The idea is to use a beam-search over the intersection grammar coupled with the cube-pruning heuristic. The beam limits the number of items in a given chart cell to a fixed constant, regardless of the number of possible LM contexts and non-terminal categories. Cube-pruning further limits the number of items considered for inclusion in the beam, reducing the time complexity to a more manageable $O(SRBV)$ where B is the beam size. We refer the interested reader to Chiang (2007) for details.

Training The extensions to the model in (5) also necessitate changes in the training procedure. Recall that training the basic model of Cohn and Lapata (2007) requires finding the maximiser of $H(\mathbf{y})$ in (3). Their model uses a chart-based algorithm for this purpose. As in decoding we also use a beam search for training, thereby avoiding the exponential time complexity of exact search. The beam search requires an estimate of the quality for incomplete derivations. We use the margin rescaling objective, H_m in (3), and approximate the loss using the current (incomplete) loss parameter values in each chart cell. We use a wide beam of 200 unique items or 500 items in total to reduce the impact of the approximation.

Our loss functions are tailored to the task and draw inspiration from metrics developed for extractive compression but also for summarisation and machine translation. They are based on the Hamming distance over unordered bags of items. This measures the number of predicted items that did not appear in the reference, along with a penalty for short output:

$$\Delta_{hamming}(\mathbf{y}^*, \mathbf{y}) = f + \max(l - (t+f), 0) \quad (6)$$

where t and f are the number of true and false positives, respectively, when comparing the predicted target, $\mathbf{y}$, with the reference, $\mathbf{y}^*$, and l is the length of the reference. The second term penalises short output, as predicting very little or nothing would otherwise be unpenalised. We have three Hamming loss functions over: 1) tokens, 2) ngrams ($n \leq 3$), or 3) CFG productions. These losses all operate on unordered bags and therefore might reward erroneous predictions. For example, a permutation of the reference tokens has zero token-loss. The CFG and ngram losses have overlapping items which encode a partial order, and therefore are less affected.

In addition, we developed a fourth loss function to measure the edit distance between the model's prediction and the reference, both as bags-of-tokens. This measures the number of insertions and deletions. In contrast to the previous loss functions, this requires the true positive counts to be clipped to the number of occurrences of each type in the reference. The edit distance is given by:

$$\Delta_{edit}(\mathbf{y}^*, \mathbf{y}) = p + q - 2\sum_i \min(p_i, q_i) \quad (7)$$

where p and q denote the number of target tokens in the predicted and reference derivation, respectively, and p_i and q_i are the counts for type i.

$\langle$ADJP,NP$\rangle \rightarrow \langle$subject [PP to NP$_{\boxed{1}}$], part [PP of NP$_{\boxed{1}}$]$\rangle$ (T)

$\langle$ADVP,RB$\rangle \rightarrow \langle$as well, also$\rangle$ (T)

$\langle$ADJP,JJ$\rangle \rightarrow \langle$too little, insufficient$\rangle$ (P)

$\langle$S,S$\rangle \rightarrow \langle$S$_{\boxed{1}}$ and S$_{\boxed{2}}$, S$_{\boxed{2}}$ and S$_{\boxed{1}}\rangle$ (P)

$\langle$NP,NP$\rangle \rightarrow \langle$DT$_{\boxed{1}}$ NN$_{\boxed{2}}$, DT$_{\boxed{1}}$ NN$_{\boxed{2}}\rangle$ (S)

$\langle$NP,NP$\rangle \rightarrow \langle$DT$_{\boxed{1}}$ NN$_{\boxed{2}}$, NN$_{\boxed{2}}\rangle$ (S)

Table 2: Sample grammar rules extracted from the training set (T), pivoted set (P) or generated from the source (S).

5 Experimental Design

In this section we present our experimental setup for assessing the performance of our model. We give details on the corpora and grammars we used, model parameters and features,[6] the baseline used for comparison with our approach, and explain how our system output was evaluated.

Grammar Extraction Our grammar used rules extracted directly from our compression corpus (the training partition, 480 sentences) and a bilingual corpus (see Table 2 for examples). The former corpus was word-aligned using the Berkeley aligner (Liang et al., 2006) initialised with a lexicon of word identity mappings, and parsed with Bikel's (2002) parser. From this we extracted grammar rules following the technique described in Cohn and Lapata (2007). For the pivot grammar we use the French-English Europarl v2 which contains approximately 688K sentences. Again, the corpus was aligned using the Berkeley aligner and the English side was parsed with Bikel's parser. We extracted tree-to-string rules using our implementation of the GHKM method. To ameliorate the effects of poor alignments on the grammar, we removed singleton rules before pivoting.

In addition to the two grammars described, we scanned the source trees in the compression corpus and included STSG rules to copy each CFG production or delete up to two of its children. This is illustrated in Table 2 where the last two rules are derived from the CFG production NP→DT NN in the source tree. All trees are rooted with a distinguished TOP non-terminal which allows the explicit modelling of sentence spanning sub-trees. These grammars each had 44,199 (pivot), 7,813 (train) and 22,555 (copy) rules. We took their union, resulting in 58,281 unique rules and 13,619 unique source elementary trees.

Model Parameters Our model was trained on 480 sentences, 36 sentences were used for development and 59 for testing. We used a variety of syntax-based, lexical and compression-specific features (196,419 in total). These are summarised in Table 3. We also use a trigram language model trained on the BNC (100 million words) using the SRI Language Modeling toolkit (Stolcke, 2002), with modified Kneser-Ney smoothing.

For every rule:
origin of rule
for each origin, o: $\log p_o(s,t), \log p_o(s\|t), \log p_o(t\|s)$
$s_R, t_R, s_R \vee t_R$
$s, t, s \vee t, s = t$
both s and t are pre-terminals and $s = t$ or $s \neq t$
number of terminals/variables/dropped variables
ordering of variables as numbers/non-terminals
non-terminal sequence of vars identical after reordering
pre-terminal or terminal sequences are identical
number/identity of common/inserted/dropped terminals
source is shorter/longer than target
target is a compression of the source using deletes
For every ngram :
$\log p(w_i \| \mathbf{w}_{i-(n-1)}^{i-1})$

Table 3: The feature set. Rules were drawn from the training set, bilingual pivoting and directly from the source trees. s and t are the source and target elementary trees in a rule, the subscript $_R$ references the root non-terminal, **w** are the terminals in the target tree.

An important parameter in our modeling framework is the choice of loss function. We evaluated the loss functions presented in Section 4 on the development set. We ran our system for each of the four loss functions and asked two human judges to rate the output on a scale of 1 to 5. The Hamming loss over tokens performed best with a mean rating of 3.18, closely followed by the edit distance (3.17). We chose the former over the latter as it is less coarsely approximated during search.

Baseline There are no existing models that can be readily trained on our abstractive compression data. Instead, we use Cohn and Lapata's (2007) extractive model as a baseline. The latter was trained on an extractive compression corpus drawn from the BNC (Clarke, 2008) and tuned to provide a similar compression rate to our system. Note that their model is a strong baseline: it performed significantly better than competitive approaches (McDonald, 2006) across a variety of compression corpora.

Evaluation Methodology 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

[6] The software and corpus can be downloaded from `http://homepages.inf.ed.ac.uk/tcohn/paraphrase`.

Models	Grammaticality	Importance	CompR
Extract	3.10*	2.43*	82.5
Abstract	3.38*	2.85*†	79.2
Gold	4.51	4.02	58.4

Table 4: Mean ratings on compression output elicited by humans; *: significantly different from the gold standard; †: significantly different from the baseline.

sentences were compressed automatically by our model and the baseline. We also included gold standard compressions. Our materials thus consisted of 90 (30×3) source-target sentences. We collected ratings from 22 unpaid volunteers, all self reported native English speakers. Both studies were conducted over the Internet using a custom built web interface.

6 Results

Our results are summarised in Table 4, where we show the mean ratings for our system (Abstract), the baseline (Extract), and the gold standard. 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 ($p < 0.01$)).

We next examined in more detail between-system differences. Post-hoc Tukey tests revealed that our abstractive model received significantly higher ratings than the baseline in terms of importance ($\alpha < 0.01$). We conjecture that this is due to the synchronous grammar we employ which is larger and more expressive than the baseline. In the extractive 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, e.g., by allowing substitutions. As far as grammaticality is concerned, our abstractive model is numerically better than the extractive baseline but the difference is not statistically significant. Note that our model has to work a lot harder than the baseline to preserve grammaticality since we allow 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.

Finally, both the abstractive and extractive 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 4. 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.

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.

Table 5: Compression examples including human and system output (O: original sentence, E: Extractive model, A: Abstractive model, G: gold standard)

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%), 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. In most cases (63%) the compressions produced by our system did not distort the meaning of the original. Humans are clearly better at this, 96.5% of their compressions were meaning preserving.

We illustrate example output of our system in Table 5. For comparison we also present the gold standard compressions and baseline output. 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. Finally, our last sentence illustrates a counter-intuitive substitution, the pronoun *her* is rewritten as *him*. This is because they share the French translation *lui* and thus pivoting learns to replace the less common word (in legal corpora) *her* with *him*. This problem could be addressed by pivoting over multiple bitexts with different foreign languages.

Possible extensions and improvements to the current model are many and varied. Firstly, as hinted at above, the model would benefit from extensive feature engineering, including source conditioned features and ngram features besides the LM. A richer grammar would also boost performance. This could be found by pivoting over more bitexts in many foreign languages or making use of existing or paraphrase corpora. Finally, we plan to apply the model to other paraphrasing tasks including fully abstractive document summarisation (Daumé III and Marcu, 2002).

Acknowledgements

The authors acknowledge the support of EPSRC (grants GR/T04540/01 and GR/T04557/01). Special thanks to Phil Blunsom, James Clarke and Miles Osborne for their insightful suggestions.

References

C. Bannard, C. Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of the 43rd ACL*, 255–262, Ann Arbor, MI.

R. Barzilay. 2003. *Information Fusion for Multi-Document Summarization: Praphrasing and Generation*. Ph.D. thesis, Columbia University.

D. Bikel. 2002. Design of a multi-lingual, parallel-processing statistical parsing engine. In *Proceedings of the HLT*, 24–27, San Diego, CA.

J. Carroll, G. Minnen, D. Pearce, Y. Canning, S. Devlin, J. Tait. 1999. Simplifying text for language impaired readers. In *Proceedings of the 9th EACL*, 269–270, Bergen, Norway.

D. Chiang. 2007. Hierarchical phrase-based translation. *Computational Linguistics*, 33(2):201–228.

J. Clarke. 2008. *Global Inference for Sentence Compression: An Integer Linear Programming Approach*. Ph.D. thesis, University of Edinburgh.

T. Cohn, M. Lapata. 2007. Large margin synchronous generation and its application to sentence compression. In *Proceedings of the EMNLP/CoNLL*, 73–82, Prague, Czech Republic.

S. Corston-Oliver. 2001. Text Compaction for Display on Very Small Screens. In *Proceedings of the NAACL Workshop on Automatic Summarization*, 89–98, Pittsburgh, PA.

H. Daumé III, D. Marcu. 2002. A noisy-channel model for document compression. In *Proceedings of the 40th ACL*, 449–456, Philadelphia, PA.

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

M. Galley, K. McKeown. 2007. Lexicalized Markov grammars for sentence compression. In *Proceedings of the NAACL/HLT*, 180–187, Rochester, NY.

M. Galley, M. Hopkins, K. Knight, D. Marcu. 2004. What's in a translation rule? In *Proceedings of the HLT/NAACL*, 273–280, Boston, MA.

H. Jing. 2000. Sentence reduction for automatic text summarization. In *Proceedings of the ANLP*, 310–315, Seattle, WA.

T. Joachims. 2005. A support vector method for multivariate performance measures. In *Proceedings of the 22nd ICML*, 377–384, Bonn, Germany.

K. Knight, D. Marcu. 2002. Summarization beyond sentence extraction: a probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

P. Liang, B. Taskar, D. Klein. 2006. Alignment by agreement. In *Proceedings of the HLT/NAACL*, 104–111, New York, NY.

C.-Y. Lin. 2003. Improving summarization performance by sentence compression — a pilot study. In *Proceedings of the 6th International Workshop on Information Retrieval with Asian Languages*, 1–8, Sapporo, Japan.

R. McDonald. 2006. Discriminative sentence compression with soft syntactic constraints. In *Proceedings of the 11th EACL*, 297–304, Trento, Italy.

K. Papineni, S. Roukos, T. Ward, W.-J. Zhu. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th ACL*, 311–318, Philadelphia, PA.

C. Quirk, C. Brockett, W. Dolan. 2004. Monolingual machine translation for paraphrase generation. In *Proceedings of the EMNLP*, 142–149, Barcelona, Spain.

A. Stolcke. 2002. SRILM – an extensible language modeling toolkit. In *Proceedings of the ICSLP*, Denver, CO.

B. Taskar, C. Guestrin, D. Koller. 2003. Max margin Markov networks. In *Proceedings of NIPS 16*.

I. Tsochantaridis, T. Joachims, T. Hofmann, Y. Altun. 2005. Large margin methods for structured and interdependent output variables. *Journal of Machine Learning Research*, 6:1453–1484.

J. Turner, E. Charniak. 2005. Supervised and unsupervised learning for sentence compression. In *Proceedings of 43rd ACL*, 290–297, Ann Arbor, MI.

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