A New Sentence Compression Dataset and Its Use in an Abstractive Generate-and-Rank Sentence Compressor

Dimitrios Galanis[*] and **Ion Androutsopoulos**[*+]
[*]Department of Informatics, Athens University of Economics and Business, Greece
[+]Digital Curation Unit – IMIS, Research Center "Athena", Greece

Abstract

Sentence compression has attracted much interest in recent years, but most sentence compressors are extractive, i.e., they only delete words. There is a lack of appropriate datasets to train and evaluate abstractive sentence compressors, i.e., methods that apart from deleting words can also rephrase expressions. We present a new dataset that contains candidate extractive and abstractive compressions of source sentences. The candidate compressions are annotated with human judgements for grammaticality and meaning preservation. We discuss how the dataset was created, and how it can be used in generate-and-rank abstractive sentence compressors. We also report experimental results with a novel abstractive sentence compressor that uses the dataset.

1 Introduction

Sentence compression is the task of producing a shorter form of a grammatical source (input) sentence, so that the new form will still be grammatical and it will retain the most important information of the source (Jing, 2000). Sentence compression is useful in many applications, such as text summarization (Madnani et al., 2007) and subtitle generation (Corston-Oliver, 2001). Methods for sentence compression can be divided in two categories: *extractive* methods produce compressions by only removing words, whereas *abstractive* methods may additionally rephrase expressions of the source sentence. Extractive methods are generally simpler and have dominated the sentence compression literature (Jing, 2000; Knight and Marcu, 2002; McDonald, 2006; Cohn and Lapata, 2007; Clarke and Lapata, 2008; Cohn and Lapata, 2009; Nomoto, 2009; Galanis and Androutsopoulos, 2010; Yamangil and Shieber, 2010). Abstractive methods, however, can in principle produce shorter compressions that convey the same information as longer extractive ones. Furthermore, humans produce mostly abstractive compressions (Cohn and Lapata, 2008); hence, abstractive compressors may generate more natural outputs.

When evaluating extractive methods, it suffices to have a single human gold extractive compression per source sentence, because it has been shown that measuring the similarity (as F_1-measure of dependencies) between the dependency tree of the gold compression and that of a machine-generated compression correlates well with human judgements (Riezler et al., 2003; Clarke and Lapata, 2006a). With abstractive methods, however, there is a much wider range of acceptable abstractive compressions of each source sentence, to the extent that a single gold compression per source is insufficient. Indeed, to the best of our knowledge no measure to compare a machine-generated abstractive compression to a single human gold compression has been shown to correlate well with human judgements.

One might attempt to provide multiple human gold abstractive compressions per source sentence and employ measures from machine translation, for example BLEU (Papineni et al., 2002), to compare each machine-generated compression to all the corresponding gold ones. However, a large number of gold compressions would be necessary to capture all (or at least most) of the acceptable shorter rephras-

Proceedings of the UCNLG+Eval: Language Generation and Evaluation Workshop, pages 1–11,
Edinburgh, Scotland, UK, July 31, 2011.

ings of the source sentences, and it is questionable if human judges could provide (or even think of) all the acceptable rephrasings. In machine translation, n-gram-based evaluation measures like BLEU have been criticized exactly because they cannot cope sufficiently well with paraphrases (Callison-Burch et al., 2006), which play a central role in abstractive sentence compression (Zhao et al., 2009a).[1]

Although it is difficult to construct datasets for end-to-end automatic evaluation of abstractive sentence compression methods, it is possible to construct datasets to evaluate the *ranking components* of generate-and-rank abstractive sentence compressors, i.e., compressors that first generate a large set of candidate abstractive (and possibly also extractive) compressions of the source and then rank them to select the best one. In previous work (Galanis and Androutsopoulos, 2010), we presented a generate-and-rank *extractive* sentence compressor, hereafter called GA-EXTR, which achieved state-of-the art results. We aim to construct a similar *abstractive* generate-and-rank sentence compressor. As part of this endeavour, we needed a dataset to automatically test (and train) several alternative ranking components. In this paper, we introduce a dataset of this kind, which we also make publicly available.[2]

The dataset consists of pairs of source sentences and candidate extractive or abstractive compressions. The candidate compressions were generated by first using GA-EXTR and then applying existing paraphrasing rules (Zhao et al., 2009b) to the best extractive compressions of GA-EXTR. Each pair (source and candidate compression) was then scored by a human judge for grammaticality and meaning preservation. We discuss how the dataset was constructed and how we established upper and lower performance boundaries for ranking components of compressors that may use it. We also present the current version of our abstractive sentence compressor, and we discuss how its ranking component was improved by performing experiments on the dataset.

Section 2 below summarizes prior work on abstractive sentence compression. Section 3 discusses the dataset we constructed. Section 4 describes our abstractive sentence compressor. Section 5 presents our experimental results, and Section 6 concludes.

2 Prior work on abstractive compression

The first *abstractive* compression method was proposed by Cohn and Lapata (2008). It learns a set of parse tree transduction rules from a training dataset of pairs, each pair consisting of a source sentence and a single human-authored gold abstractive compression. The set of transduction rules is then augmented by applying a pivoting approach to a parallel bilingual corpus; we discuss similar pivoting mechanisms below. To compress a new sentence, a chart-based decoder and a Structured Support Vector Machine (Tsochantaridis et al., 2005) are used to select the best abstractive compression among those licensed by the rules learnt.

The dataset that Cohn and Lapata (2008) used to learn transduction rules consists of 570 pairs of source sentences and abstractive compressions. The compressions were produced by humans who were allowed to use any transformation they wished. We used a sample of 50 pairs from that dataset to confirm that humans produce mostly abstractive compressions. Indeed, 42 (84%) of the compressions were abstractive, and only 7 (14%) were simply extractive.[3] We could not use that dataset, however, for automatic evaluation purposes, since it only provides a single human gold abstract compression per source, which is insufficient as already discussed.

More recently, Zhao et al. (2009a) presented a sentence paraphrasing method that can be configured for different tasks, including a form of sentence compression. For each source sentence, Zhao et al.'s method uses a decoder to produce the best possible paraphrase, much as in phrase-based statistical machine translation (Koehn, 2009), but with phrase tables corresponding to paraphrasing rules (e.g., "X

[1] Ways to extend n-gram measures to account for paraphrases have been proposed (Zhou et al., 2006; Kauchak and Barzilay, 2006; Padó et al., 2009), but they require accurate paraphrase recognizers (Androutsopoulos and Malakasiotis, 2010), which are not yet available; or they assume that the same paraphrase generation resources (Madnani and Dorr, 2010), for example paraphrasing rules, that some abstractive sentence compressors (including ours) use always produce acceptable paraphrases, which is not the case as discussed below.

[2] The new dataset and GA-EXTR are freely available from `http://nlp.cs.aueb.gr/software.html`.

[3] Cohn and Lapata's dataset is available from `http://staffwww.dcs.shef.ac.uk/people/T.Cohn/t3/#Corpus`. One pair (2%) of our sample had a 'compression' that was identical to the input.

is the author of Y" $\approx$ "X wrote Y") obtained from parallel and comparable corpora (Zhao et al., 2008). The decoder uses a log-linear objective function, the weights of which are estimated with a minimum error rate training approach (Och, 2003). The objective function combines a language model, a paraphrase model (combining the quality scores of the paraphrasing rules that turn the source into the candidate paraphrase), and a task-specific model; in the case of sentence compression, the latter model rewards shorter candidate paraphrases.

We note that Zhao et al.'s method (2009a) is intended to produce paraphrases, even when configured to prefer shorter paraphrases, i.e., the compressions are still intended to convey the same information as the source sentences. By contrast, most sentence compression methods (both extractive and abstractive, including ours) are expected to retain only the most important information of the source sentence, in order to achieve better compression rates. Hence, Zhao et al.'s sentence compression task is not the same as the task we are concerned with, and the compressions we aim for are significantly shorter.

3 The new dataset

To construct the new dataset, we used source sentences from the 570 pairs of Cohn and Lapata (Section 2). This way a human gold abstractive compression is also available for each source sentence, though we do not currently use the gold compressions in our experiments. We actually used only 346 of the 570 source sentences of Cohn and Lapata, reserving the remaining 224 for further experiments.[4]

To obtain candidate compressions, we first applied GA-EXTR to the 346 source sentences, and we then applied the paraphrasing rules of Zhao et al. (2009b) to the resulting extractive compressions; we provide more information about GA-EXTR and the paraphrasing rules below. We decided to apply paraphrasing rules to extractive compressions, because we noticed that most of the 42 human abstractive compressions of the 50 sample pairs from Cohn and Lapata's dataset that we initially considered (Section 2) could be produced from the corresponding source sentences by first deleting words and then using shorter paraphrases, as in the following example.

source: Constraints on recruiting are constraints on safety and have to be removed.

extractive: Constraints on recruiting have to be removed.

abstractive: Recruiting constraints must be removed.

3.1 Extractive candidate compressions

GA-EXTR, which we first applied to the dataset's source sentences, generates extractive candidate compressions by pruning branches of each source's dependency tree; a Maximum Entropy classifier is used to guide the pruning. Subsequently, GA-EXTR ranks the extractive candidates using a Support Vector Regression (SVR) model, which assigns a score $F(e_{ij}|s_i)$ to each candidate extractive compression e_{ij} of a source sentence s_i by examining features of s_i and e_{ij}; consult our previous work (Galanis and Androutsopoulos, 2010) for details.[5] For each source s_i, we kept the (at most) $k_{max} = 10$ extractive candidates e_{ij} with the highest $F(e_{ij}|s_i)$ scores.

3.2 Abstractive candidate compressions

We then applied Zhao et al.'s (2009b) paraphrasing rules to each one of the extractive compressions e_{ij}. The rules are of the form *left* $\leftrightarrow$ *right*, with *left* and *right* being sequences of words and slots; the slots are part-of-speech tagged and they can be filled in with words of the corresponding categories. Examples of rules are shown below.

- get rid of *NNS*$_1$ $\leftrightarrow$ remove *NNS*$_1$
- get into *NNP*$_1$ $\leftrightarrow$ enter *NNP*$_1$
- *NNP*$_1$ was written by *NNP*$_2$ $\leftrightarrow$ *NNP*$_2$ wrote *NNP*$_1$

Roughly speaking, the rules were extracted from a parallel English-Chinese corpus, based on the assumption that two English phrases ϕ_1 and ϕ_2 that are often aligned to the same Chinese phrase ξ are

[4] The 346 sources are from 19 randomly selected articles among the 30 that Cohn and Lapata drew source sentences from.

[5] We trained GA-EXTR on approximately 1,050 pairs of source sentences and gold human extractive compressions, obtained from Edinburgh's 'written' extractive dataset; see `http://jamesclarke.net/research/resources`. The source sentences of that dataset are from 82 documents. The 1,050 pairs that we used had source sentences from 52 out of the 82 documents. We did not use source sentences from the other 30 documents, because they were used by Cohn and Lapata (2008) to build their abstractive dataset (Section 2), from which we drew source sentences for our dataset.

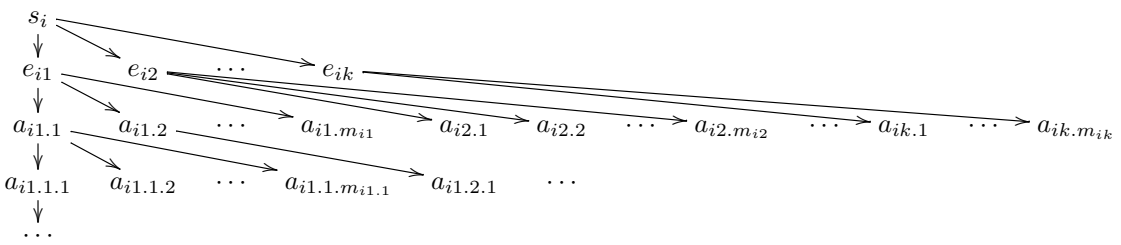

Figure 1: Generating candidate extractive (e_{ij}) and abstractive ($a_{ij...}$) compressions from a source sentence (s_i).

likely to be paraphrases and, hence, can be treated as a paraphrasing rule $\phi_1 \leftrightarrow \phi_2$. This *pivoting* was used, for example, by Bannard and Callison-Burch (2005), and it underlies several other paraphrase extraction methods (Riezler et al., 2007; Callison-Burch, 2008; Kok and Brockett, 2010). Zhao et al. (2009b) provide approximately one million rules, but we use only approximately half of them, because we use only rules that can shorten a sentence, and only in the direction that shortens the sentence.

From each extractive candidate e_{ij}, we produced abstractive candidates $a_{ij.1}, a_{ij.2}, \ldots, a_{ij.m_{ij}}$ (Figure 1) by applying a single (each time different) applicable paraphrasing rule to e_{ij}. From each of the resulting abstractive candidates $a_{ij.l}$, we produced further abstractive candidates $a_{ij.l.1}, a_{ij.l.2}, \ldots, a_{ij.l.m_{ij.l}}$ by applying again a single (each time different) rule. We repeated this process in a breadth-first manner, allowing up to at most $\textit{rule}_{max} = 5$ rule applications to an extractive candidate e_{ij}, i.e., up to depth six in Figure 1, and up to a total of $\textit{abstr}_{max} = 50$ abstractive candidates per e_{ij}. Zhao et al. (2009b) associate each paraphrasing rule with a score, intended to indicate its quality.[6] Whenever multiple paraphrasing rules could be applied, we applied the rule with the highest score first.

3.3 Human judgement annotations

For each one of the 346 sources s_i, we placed its extractive (at most $k_{max} = 10$) and abstractive (at most $\textit{abstr}_{max} = 50$) candidate compressions into a single pool (extractive and abstractive together), and we selected from the pool the (at most) 10 candidate compressions c_{ij} with the highest language model scores, computed using a 3-gram language model.[7] For each c_{ij}, we formed a pair $\langle s_i, c_{ij} \rangle$, where s_i is a source sentence and c_{ij} a candidate (extractive or abstractive) compression. This led to 3,072 $\langle s_i, c_{ij} \rangle$ pairs. Each pair was given to a human judge, who scored it for grammaticality (how grammatical c_{ij} was) and meaning preservation (to what extent c_{ij} preserved the most important information of s_i). Both scores were provided on a 1–5 scale (1 for rubbish, 5 for perfect). The dataset that we use in the following sections and that we make publicly available comprises the 3,072 pairs and their grammaticality and meaning preservation scores.

We define the GM score of an $\langle s_i, c_{ij} \rangle$ pair to be the sum of its grammaticality and meaning preservation scores. Table 1 shows the distribution of GM scores in the 3,072 pairs. Low GM scores (2–5) are less frequent than higher scores (6–10), but this is not surprising given that we selected pairs whose c_{ij} had high language model scores, that we used the k_{max} extractive compressions of each s_i that GA-EXTR considered best, and that we assigned higher preference to applying paraphrasing rules with higher scores. We note, however, that applying a paraphrasing rule does not necessarily preserve neither grammaticality nor meaning, even if the rule has a high score. Szpektor et al. (2008) point out that, for example, a rule like "X acquire Y" $\leftrightarrow$ "X buy Y" may work well in many contexts, but not in "Children acquire language quickly". Similarly, "X charged Y with" $\leftrightarrow$ "X accused Y of" should not be applied to sentences about batteries. Many (but not all) inappropriate rule applications

[6] Each rule is actually associated with three scores. We use the 'Model 1' score; see Zhao et al. (2009b) for details.

[7] We used SRILM with modified Kneser-Ney smoothing (Stolcke, 2002). We trained the language model on approximately 4.5 million sentences from the TIPSTER corpus.

	Training part			Test part		
GM score	extractive candidates	abstractive candidates	total candidates	extractive candidates	abstractive candidates	total candidates
2	13 (1.3%)	10 (1.3%)	23 (1.3%)	19 (1.9%)	2 (0.4%)	21 (1.5%)
3	26 (2.7%)	28 (3.6%)	54 (3.1%)	10 (1.0%)	0 (0%)	10 (0.7%)
4	55 (5.8%)	29 (5.1%)	94 (5.5%)	51 (5.3%)	26 (6.2%)	77 (5.5%)
5	52 (5.5%)	65 (8.5%)	117 (6.9%)	77 (8.0%)	42 (10.0%)	119 (8.6%)
6	102 (10.9%)	74 (9.7%)	176 (10.3%)	125 (13.0%)	83 (19.8%)	208 (15.1%)
7	129 (13.8%)	128 (16.8%)	257 (15.1%)	151 (15.7%)	53 (12.6%)	204 (14.8%)
8	157 (16.8%)	175 (23.0%)	332 (19.5%)	138 (14.3%)	85 (20.3%)	223 (16.1%)
9	177 (18.9%)	132 (17.3%)	309 (18.2%)	183 (19.0%)	84 (20.1%)	267 (19.3%)
10	223 (23.8%)	110 (14.4%)	333 (19.6%)	205 (21.3%)	43 (10.2%)	248 (18.0%)
total	934 (55.1%)	761 (44.9%)	1,695 (100%)	959 (69.6%)	418 (30.4%)	1,377 (100%)

Table 1: Distribution of GM scores (grammaticality plus meaning preservation) in our dataset.

lead to low language model scores, which is partly why there are more extractive than abstractive candidate compressions in the dataset; another reason is that few or no paraphrasing rules apply to some of the extractive candidates.

We use 1,695 (from 188 source sentences) of the 3,072 pairs to train different versions of our abstractive compressor's ranking component, discussed below, and 1,377 pairs (from 158 sources) as a test set.

3.4 Inter-annotator agreement

Although we used a total of 16 judges (computer science graduate students), each one of the 3,072 pairs was scored by a single judge, because a preliminary study indicated reasonably high inter-annotator agreement.[8] More specifically, before the dataset was constructed, we created 161 $\langle s_i, c_{ij} \rangle$ pairs (from 22 source sentences) in the same way, and we gave them to 3 of the 16 judges. Each pair was scored by all three judges. The average (over pairs of judges) Pearson correlation of the grammaticality, meaning preservation, and GM scores, was 0.63, 0.60, and 0.69, respectively.[9] We conjecture that the higher correlation of GM scores, compared to grammaticality and meaning preservation, is due to the fact that when a candidate compression looks bad the judges sometimes do not agree if they should reduce the grammaticality or the meaning preservation score, but the difference does not show up in the GM score (the sum). Table 2 shows the average correlation of the GM scores of the three judges on the 161 pairs, and separately for pairs that involved extractive or abstractive candidate compressions. The judges agreed more on extractive candidates, since the paraphrasing stage that is involved in the abstractive candidates makes the task more subjective.[10]

	candidate compressions	average Pearson correlation
Extractive	112	0.71
Abstractive	49	0.64
All	161	0.69

Table 2: Inter-annotator agreement on GM scores.

3.5 Performance boundaries

When presented with two pairs $\langle s_i, c_{ij} \rangle$ and $\langle s_i, c_{ij'} \rangle$ with the same s_i and equally long c_{ij} and $c_{ij'}$, an ideal ranking component should prefer the pair with the highest GM score. More generally, to consider the possibly different lengths of c_{ij} and $c_{ij'}$, we first define the compression rate $\text{CR}(c_{ij}|s_i)$ of a candidate compression c_{ij} as follows, where $|\cdot|$ is length in characters; lower values of CR are better.

$$\text{CR}(c_{ij}|s_i) = \frac{|c_{ij}|}{|s_i|}$$

The GMC_γ score of a candidate compression, which also considers the compression rate by assigning it a

[8] The judges were fluent, but not native, English speakers.

[9] The Pearson correlation ranges in $[-1, +1]$ and measures the linear relationship of two variables. A correlation of $+1$ indicates perfect positive relationship, while -1 indicates perfect negative relationship; a correlation of 0 signals no relationship.

[10] The correlation that we measured on extractive candidates (0.71) is very close to the corresponding figure (0.746) that has been reported by Clarke and Lapata (2006b).

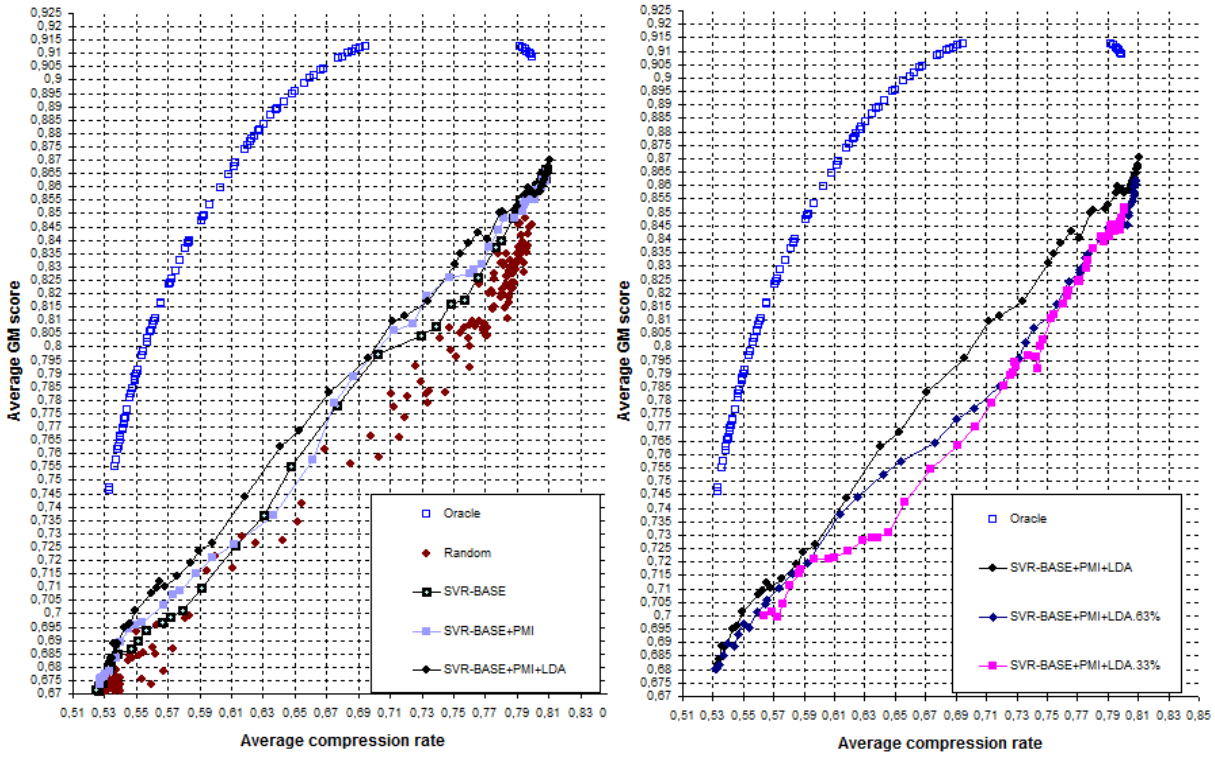

Figure 2: Results of three SVR-based ranking components on our dataset, along with performance boundaries obtained using an oracle and a random baseline. The right diagram shows how the performance of our best SVR-based ranking component is affected when using only 33% and 63% of the training examples.

weight γ, is then defined as follows.

$$\text{GMC}_\gamma(c_{ij}|s_i) = \text{GM}(c_{ij}|s_i) - \gamma \cdot \text{CR}(c_{ij}|s_i)$$

For a given γ, when presented with $\langle s_i, c_{ij} \rangle$ and $\langle s_i, c_{ij'} \rangle$, an ideal ranking component should prefer the pair with the highest GMC_γ score.

The upper curve of the left diagram of Figure 2 shows the performance of an ideal ranking component, an *oracle*, on the test part of the dataset. For every source s_i, the oracle selects the $\langle s_i, c_{ij} \rangle$ pair (among the at most 10 pairs of s_i) for which $\text{GMC}_\gamma(c_{ij}|s_i)$ is maximum; if two pairs have identical GMC_γ scores, it prefers the one with the lowest $\text{CR}(c_{ij}|s_i)$. The vertical axis shows the average $\text{GM}(c_{ij}|s_i)$ score of the selected pairs, for all the s_i sources, and the horizontal axis shows the average $\text{CR}(c_{ij}|s_i)$. Different points of the curve are obtained by using different γ values. As the selected candidates get shorter (lower compression rate), the average GM score decreases, as one would expect.[11]

The other curves of Figure 2 correspond to alternative ranking components that we tested, discussed below, which do not consult the judges' GM scores. For each s_i, these ranking components attempt to guess the GM scores of the $\langle s_i, c_{ij} \rangle$ pairs that are available for s_i, and they then rank the pairs by GMC_γ using the guessed GM scores. The lower points of the left diagram were obtained with a baseline ranking component that assigns a *random* GM score to each pair. The oracle and the baseline can be seen as establishing upper and lower performance boundaries of ranking components on our dataset.

4 Our abstractive compressor

Our abstractive sentence compressor operates in two stages. Given a source sentence s_i, extractive and

[11] The discontinuity in the oracle's curve for average compression rates above 0.7, i.e., when long compressions are only mildly penalized, is caused by the fact that many long candidate compressions have high and almost equal GM scores, but still very different compression rates; hence, a slight modification of γ leads the oracle to select candidates with the same GM scores, but very different compression rates.

abstractive candidate compressions are first generated as in Sections 3.1 and 3.2. In a second stage, a ranking component is used to select the best candidate. Below we discuss the three SVR-based ranking components that we experimented with.

4.1 Ranking candidates with an SVR

An SVR is very similar to a Support Vector Machine (Vapnik, 1998; Cristianini and Shawe-Taylor, 2000; Joachims, 2002), but it is trained on examples of the form $\langle x_l, y(x_l)\rangle$, where each $x_l \in \mathbb{R}^n$ is a vector of n features, and $y(x_l) \in \mathbb{R}$. The SVR learns a function $f : \mathbb{R}^n \rightarrow \mathbb{R}$ intended to return $f(x)$ values as close as possible to the correct $y(x)$ values.[12] In our case, each vector x_{ij} contains features providing information about an $\langle s_i, c_{ij}\rangle$ pair of a source sentence s_i and a candidate compression c_{ij}. For pairs that have been scored by human judges, the $f(x_{ij})$ returned by the SVR should ideally be $y(x_{ij}) = \text{GMC}_\gamma(c_{ij}|s_i)$; once trained, however, the SVR may be presented with x_{ij} vectors of unseen $\langle s_i, c_{ij}\rangle$ pairs.

For an unseen source s_i, our abstractive compressor first generates extractive and abstractive candidates c_{ij}, it then forms the vectors x_{ij} of all the pairs $\langle s_i, c_{ij}\rangle$, and it returns the c_{ij} for which the SVR's $f(x_{ij})$ is maximum. On a test set (like the test part of our dataset), if the $f(x_{ij})$ values the SVR returns are very close to the corresponding $y(x_{ij}) = \text{GMC}_\gamma(c_{ij}|s_i)$ scores, the ranking component will tend to select the same c_{ij} for each s_i as the oracle, i.e., it will achieve optimum performance.

4.2 Base form of our SVR ranking component

The simplest form of our SVR-based ranking component, called SVR-BASE, uses vectors x_{ij} that include the following features of $\langle s_i, c_{ij}\rangle$. Hereafter, if c_{ij} is an extractive candidate, then $e(c_{ij}) = c_{ij}$; otherwise $e(c_{ij})$ is the extractive candidate that c_{ij} was derived from by applying paraphrasing rules.[13]

- The language model score of s_i and c_{ij} (2 features), computed as in Section 3.3.
- The $F(e(c_{ij})|s_i)$ score that GA-EXTR returned.
- The compression rate $\text{CR}(e(c_{ij})|s_i)$.
- The number (possibly zero) of paraphrasing rules that were applied to $e(c_{ij})$ to produce c_{ij}.

4.3 Additional PMI-based features

For two words w_1, w_2, their PMI score is:

$$\text{PMI}(w_1, w_2) = \log \frac{P(w_1, w_2)}{P(w_1) \cdot P(w_2)}$$

where $P(w_1, w_2)$ is the probability of w_1, w_2 co-occurring; we require them to co-occur in the same sentence at a maximum distance of 10 tokens.[14] If w_1, w_2 are completely independent, then their PMI score is zero. If they always co-occur, their PMI score is maximum, equal to $-\log P(w_1) = -\log P(w_2)$.[15] We use PMI to assess if the words of a candidate compression co-occur as frequently as those of the source sentence; if not, this may indicate an inappropriate application of a paraphrasing rule (e.g., having replaced "charged Y with" by "X accused Y of" in a sentence about batteries).

More specifically, we define the $\text{PMI}(\sigma)$ score of a sentence σ to be the average $\text{PMI}(w_i, w_j)$ of every two content words w_i, w_j that co-occur in σ at a maximum distance of 10 tokens; below N is the number of such pairs.

$$\text{PMI}(\sigma) = \frac{1}{N} \cdot \sum_{i,j} \text{PMI}(w_i, w_j)$$

In our second SVR-based ranking component, SVR-PMI, we compute $\text{PMI}(s_i)$, $\text{PMI}(e)$, and $\text{PMI}(c_{ij})$, and we include them as three additional features; otherwise SVR-PMI is identical to SVR-BASE.

[12] We use LIBSVM (`http://www.csie.ntu.edu.tw/~cjlin/libsvm`) with an RBF kernel, which permits the SVR to learn non-linear functions. We also experimented with a ranking SVM, but the results were slightly inferior.

[13] All the feature values are normalized in $[0, 1]$; this also applies to the GMC_γ scores when they are used by the SVR. The $e(c_{ij})$ of each c_{ij} and the paraphrasing rules that were applied to $e(c_{ij})$ to produce c_{ij} are also included in the dataset.

[14] We used texts from TIPSTER and AQUAINT, a total of 953 million tokens, to estimate $\text{PMI}(w_1, w_2)$.

[15] A problem with PMI is that two frequent and completely dependent words receive lower scores than two other, less frequent completely dependent words (Manning and Schutze, 2000). Pecina (2005), however, found PMI to be the best collocation extraction measure; and Newman et al. (2010) found it to be the best measure of 'topical coherence' for sets of words.

4.4 Additional LDA-based features

Our third SVR-based ranking component includes features from a Latent Dirichlet Allocation (LDA) model (Blei et al., 2003). Roughly speaking, LDA models assume that each document d of $|d|$ words $w_1, \ldots, w_{|d|}$ is generated by iteratively (for $r = 1, \ldots, |d|$) selecting a topic t_r from a document-specific multinomial distribution $P(t|d)$ over K topics, and then (for each r) selecting a word w_r from a topic-specific multinomial distribution $P(w|t)$ over the vocabulary.[16] The probability, then, of encountering a word w in a document d is the following.

$$P(w|d) = \sum_t P(w|t) \cdot P(t|d) \qquad (1)$$

An LDA model can be trained on a corpus to estimate the parameters of the distributions it involves; and given a trained model, there are methods to infer the topic distribution $P(t|\hat{d})$ of a new document $\hat{d}$.[17]

In our case, we treat each source sentence as a new document $\hat{d}$, and we use an LDA model trained on a generic corpus to infer the topic distribution $P(t|\hat{d})$ of the source sentence.[18] We assume that a good candidate compression should contain words with high $P(w|\hat{d})$, computed as in Equation 1 with $P(t|d) = P(t|\hat{d})$ and using the $P(w|t)$ that was learnt during training, because words with high $P(w|\hat{d})$ are more likely to express (high $P(w|t)$) prominent topics (high $P(t|\hat{d})$) of the source.

Consequently, we can assess how good a candidate compression is by computing the average $P(w|\hat{d})$ of its words; we actually compute the average $\log P(w|\hat{d})$. More specifically, for a given source s_i and another sentence σ, we define $\text{LDA}(\sigma|s_i)$ as follows ($\hat{d} = s_i$), where $w_1, \ldots, w_{|\sigma|}$ are now the words of σ, ignoring stop-words.

$$\text{LDA}(\sigma|s_i) = \frac{1}{|\sigma|} \cdot \sum_{r=1}^{|\sigma|} \log P(w_r|s_i)$$

In our third SVR-based ranking component, SVR-PMI-LDA, the feature vector x_{ij} of each $\langle s_i, c_{ij} \rangle$ pair includes $\text{LDA}(c_{ij}|s_i)$, $\text{LDA}(e(c_{ij})|s_i)$, and $\text{LDA}(s_i|s_i)$ as additional features; otherwise, SVR-PMI-LDA is identical to SVR-PMI. The third feature allows the SVR to check how far $\text{LDA}(c_{ij}|s_i)$ and $\text{LDA}(e(c_{ij})|s_i)$ are from $\text{LDA}(s_i|s_i)$.

5 Experiments

To assess the performance of SVR-BASE, SVR-PMI, and SVR-PMI-LDA, we trained the three SVR-based ranking components on the training part of our dataset, and we evaluated them on the test part. We repeated the experiments for 81 different γ values to obtain average GM scores at different average compression rates (Section 3.5). The resulting curves of the three SVR-based ranking components are included in Figure 2 (left diagram). Overall, SVR-PMI-LDA performed better than SVR-PMI and SVR-BASE, since it achieved the best average GM scores throughout the range of average compression rates. In general, SVR-PMI also performed better than SVR-BASE, though the average GM score of SVR-BASE was sometimes higher. All three SVR-based ranking components performed better than the random baseline, but worse than the oracle; hence, there is scope for further improvements in the ranking components, which is also why we believe other researchers may wish to experiment with our dataset.

The oracle selected abstractive (as opposed to simply extractive) candidates for 20 (13%) to 30 (19%, depending on γ) of the 158 source sentences of the test part; the same applies to the SVR-based ranking components. Hence, good abstractive candidates (or at least better than the corresponding extractive ones) are present in the dataset. Humans, however, produce mostly abstractive compressions, as already discussed; the fact that the oracle (which uses human judgements) does not select abstractive candidates more frequently may be an indication that more or better abstractive candidates are needed. We plan to investigate alternative methods to produce more abstractive candidates. For example, one could translate each source to multiple pivot languages and back to the original language by using multiple commercial machine translation engines instead of, or in addition to applying paraphrasing

[16] The document-specific parameters of the first multinomial distribution are drawn from a Dirichlet distribution.

[17] We use MALLET (`http://mallet.cs.umass.edu`), with Gibbs sampling (Griffiths and Steyvers, 2004). We set $K = 800$, having first experimented with $K = 200, 400, 600, 800, 1000$.

[18] We trained the LDA model on approximately 106,000 articles from the TIPSTER and AQUAINT corpora.

source	generated
Gillette was considered a leading financial analyst on the beverage industry - one who also had an expert palate for wine tasting.	Gillette was seen as a leading financial analyst on the beverage industry - one who also had an expert palate.
Nearly 200,000 lawsuits were brought by women who said they suffered injuries ranging from minor inflammation to infertility and in some cases, death.	Lawsuits were made by women who said they suffered injuries ranging from inflammation to infertility in some cases, death.
Marcello Mastroianni, the witty, affable and darkly handsome Italian actor who sprang on international consciousness in Federico Fellini's 1960 classic "La Dolce Vita," died Wednesday at his Paris home.	Marcello Mastroianni died Wednesday at his home.
A pioneer in laparoscopy, he held over 30 patents for medical instruments used in abdominal surgery such as tubal ligations.	He held over 30 patents for the medical tools used in abdominal surgery.
LOS ANGELES - James Arnold Doolittle, a Los Angeles dance impresario who brought names such as Joffrey and Baryshnikov to local dance stages and ensured that a high-profile "Nutcracker Suite" was presented here every Christmas, has died.	James Arnold Doolittle, a Los Angeles dance impresario is dead.
After working as a cashier for a British filmmaker in Rome, he joined an amateur theatrical group at the University of Rome, where he was taking some classes.	After working as a cashier for a British filmmaker in Rome, he joined an amateur group at the University of Rome, where he was using some classes.
He was a 1953 graduate of the Johns Hopkins Medical School and after completing his residency in gynecology and surgery, traveled to Denmark where he joined the staff of the National Cancer Center there.	He was a graduate of the Johns Hopkins Medical School and traveled to Denmark where he joined a member of the National Cancer Center there.
Mastroianni, a comic but also suave and romantic leading man in some 120 motion pictures, had suffered from pancreatic cancer.	Mastroianni, a leading man in some 120 motion pictures, had subjected to cancer.

Table 3: Examples of good (upper five) and bad (lower three) compressions generated by our abstractive compressor.

rules. An approach of this kind has been proposed for sentence paraphrasing (Zhao et al., 2010).

The right diagram of Figure 2 shows how the performance of SVR-PMI-LDA is affected when using 33% or 63% of the training $\langle s_i, c_i \rangle$ pairs. As more examples are used, the performance improves, suggesting that better results could be obtained by using more training data. Finally, Table 3 shows examples of good and bad compressions the abstractive compressor produced with SVR-PMI-LDA.

6 Conclusions and future work

We presented a new dataset that can be used to train and evaluate the ranking components of generate-and-rank abstractive sentence compressors. The dataset contains pairs of source sentences and candidate extractive or abstractive compressions. The candidate compressions were obtained by first applying a state-of-the-art extractive compressor to the source sentences, and then applying existing paraphrasing rules, obtained from parallel corpora. The dataset's pairs have been scored by human judges for grammaticality and meaning preservation. We discussed how performance boundaries for ranking components that use the dataset can be established by using an oracle and a random baseline, and by considering different compression rates. We also discussed the current version of an abstractive sentence compressor that we are developing, and how the dataset was used to train and evaluate three different SVR-based ranking components of the compressor with gradually more elaborate features sets. The feature set of the best ranking component that we tested includes language model scores, the confidence and compression rate of the underlying extractive compressor, the number of paraphrasing rules that have been applied, word co-occurrence features, as well as features based on an LDA model.

In future work, we plan to improve our abstractive sentence compressor, possibly by including more features in the ranking component. We also plan to investigate alternative ways to produce candidate compressions, such as sentence paraphrasing methods that exploit multiple commercial machine translation engines to translate the source sentences to multiple pivot languages and back to the original language (Zhao et al., 2010). Using methods of this kind, it may be possible to produce a second, alternative dataset with more and possibly better abstractive candidates. We also plan to make the final version of our abstractive compressor publicly available.

Acknowledgments

This work was partly carried out during INDIGO, an FP6 IST project funded by the European Union, with additional funding from the Greek General Secre-

tariat of Research and Technology.[19]

References

I. Androutsopoulos and P. Malakasiotis. 2010. A survey of paraphrasing and textual entailment methods. *Journal of Artificial Intelligence Research*, 38:135–187.

C. Bannard and C. Callison-Burch. 2005. Paraphrasing with bilingual parallel corpora. In *Proceedings of ACL*, pages 597–604, Ann Arbor, MI.

D. Blei, A. Ng, and M. Jordan. 2003. Latent Dirichlet allocation. In *Journal of Machine Learning Research*.

C. Callison-Burch, M. Osborne, and P. Koehn. 2006. Re-evaluating the role of BLEU in machine translation research. In *Proceedings of EACL*, pages 249–256, Trento, Italy.

C. Callison-Burch. 2008. Syntactic constraints on paraphrases extracted from parallel corpora. In *Proceedings of EMNLP*, pages 196–205, Honolulu, HI.

J. Clarke and M. Lapata. 2006a. Constraint-based sentence compression: An integer programming approach. In *Proceedings of ACL-COLING*.

J. Clarke and M. Lapata. 2006b. Models for sentence compression: A comparison across domains, training requirements and evaluation measures. In *Proceedings of ACL-COLING*.

J. Clarke and M. Lapata. 2008. Global inference for sentence compression: An integer linear programming approach. *Journal of Artificial Intelligence Research*, 1(31):399–429.

T. Cohn and M. Lapata. 2007. Large margin synchronous generation and its application to sentence compression. In *Proceedings of EMNLP-CONLL*.

T. Cohn and M. Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of COLING*.

T. Cohn and M. Lapata. 2009. Sentence compression as tree to tree tranduction. *Journal of Artificial Intelligence Research*, 34:637–674.

S. Corston-Oliver. 2001. Text compaction for display on very small screens. In *Proceedings of the NAACL Workshop on Automatic Summarization*.

N. Cristianini and J. Shawe-Taylor. 2000. *An Introduction to Support Vector Machines and Other Kernel-based Learning Methods*. Cambridge University Press.

D. Galanis and I. Androutsopoulos. 2010. An extractive supervised two-stage method for sentence compression. In *Proceedings of HLT-NAACL*.

T. Griffiths and M. Steyvers. 2004. Finding scientific topics. In *Proceedings of the National Academy of Sciences*.

H. Jing. 2000. Sentence reduction for automatic text summarization. In *Proceedings of ANLP*.

T. Joachims. 2002. *Learning to Classify Text Using Support Vector Machines: Methods, Theory, Algorithms*. Kluwer.

D. Kauchak and R. Barzilay. 2006. Paraphrasing for automatic evaluation. In *Proceedings of the HLT-NAACL*, pages 455–462, New York, NY.

K. Knight and D. Marcu. 2002. Summarization beyond sentence extraction: A probalistic approach to sentence compression. *Artificial Intelligence*, 139(1).

P. Koehn. 2009. *Statistical Machine Translation*. Cambridge University Press.

S. Kok and C. Brockett. 2010. Hitting the right paraphrases in good time. In *Proceedings of HLT-NAACL*, pages 145–153, Los Angeles, CA.

N. Madnani and B.J. Dorr. 2010. Generating phrasal and sentential paraphrases: A survey of data-driven methods. *Computational Linguistics*, 36(3):341–387.

N. Madnani, D. Zajic, B. Dorr, N. F. Ayan, and J. Lin. 2007. Multiple alternative sentence compressions for automatic text summarization. In *Proceedings of DUC*.

C.D. Manning and H. Schutze. 2000. *Foundations of Statistical Natural Language Processing*. MIT Press.

R. McDonald. 2006. Discriminative sentence compression with soft syntactic constraints. In *Proceedings of EACL*.

D. Newman, J.H. Lau, K. Grieser, and T. Baldwin. 2010. Automatic evaluation of topic coherence. In *Proceedings of HLT-NAACL*.

T. Nomoto. 2009. A comparison of model free versus model intensive approaches to sentence compression. In *Proceedings of EMNLP*.

J. F. Och. 2003. Minimum error rate training in statistical machine translation. In *Proceedings of ACL*.

S. Padó, M. Galley, D. Jurafsky, and C. D. Manning. 2009. Robust machine translation evaluation with entailment features. In *Proceedings of ACL-IJCNLP*, pages 297–305, Singapore.

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

P. Pecina. 2005. An extensive empirical study of collocation extraction methods. In *Proceedings of the Student Research Workshop of ACL*.

S. Riezler, T.H. King, R. Crouch, and A. Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar. In *Proceedings of HLT-NAACL*.

[19] Consult `http://www.ics.forth.gr/indigo/`.

S. Riezler, A. Vasserman, I. Tsochantaridis, V. Mittal, and Y. Liu. 2007. Statistical machine translation for query expansion in answer retrieval. In *Proceedings of ACL*, pages 464–471, Prague, Czech Republic.

A. Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, pages 901–904.

I. Szpektor, I. Dagan, R. Bar-Haim, and J. Goldberger. 2008. Contextual preferences. In *Proceedings of ACL-HLT*, pages 683–691, Columbus, OH.

I. Tsochantaridis, T. Hofmann, T. Joachims, and Y. Altun. 2005. Support vector machine learning for independent and structured output spaces. *Machine Learning Research*, 6:1453–1484.

V. Vapnik. 1998. *Statistical Learning Theory*. John Wiley.

E. Yamangil and S. M. Shieber. 2010. Bayesian synchronous tree-substitution grammar induction and its application to sentence compression. In *Proceedings of ACL*.

S. Zhao, C. Niu, M. Zhou, T. Liu, and S. Li. 2008. Combining multiple resources to improve SMT-based paraphrasing model. In *Proceedings of ACL-HLT*, pages 1021–1029, Columbus, OH.

S. Zhao, X. Lan, T. Liu, and S. Li. 2009a. Application-driven statistical paraphrase generation. In *Proceedings of ACL*.

S. Zhao, H. Wang, T. Liu, and S. Li. 2009b. Extracting paraphrase patterns from bilingual parallel corpora. *Natural Language Engineering*, 15(4):503–526.

S. Zhao, H. Wang, X. Lan, and T. Liu. 2010. Leveraging multiple MT engines for paraphrase generation. In *Proceedings of COLING*.

L. Zhou, C.-Y. Lin, and Eduard Hovy. 2006. Re-evaluating machine translation results with paraphrase support. In *Proceedings of EMNLP*, pages 77–84.