Multi-Sentence Compression: Finding Shortest Paths in Word Graphs

Katja Filippova
Google Inc.
katjaf@google.com

Abstract

We consider the task of summarizing a cluster of related sentences with a short sentence which we call *multi-sentence compression* and present a simple approach based on shortest paths in word graphs. The advantage and the novelty of the proposed method is that it is syntax-lean and requires little more than a tokenizer and a tagger. Despite its simplicity, it is capable of generating grammatical and informative summaries as our experiments with English and Spanish data demonstrate.

1 Introduction

Sentence compression (henceforth SC) is a task where the goal is to produce a summary of a single sentence which would preserve the important part of the content and be grammatical. Starting from the early work of Jing & McKeown (2000), in the last decade SC has received considerable attention in the NLP community. Ubiquitous use of mobile devices is an obvious example of where SC could be applied–a longer text of an email, news or a Wikipedia article can be compressed sentence by sentence to fit into a limited display (Corston-Oliver, 2001). Another reason why SC is so popular is its potential utility for extractive text summarization, single or multi-document (Mani, 2001). There, a standard approach is to rank sentences by importance, cluster them by similarity, and select a sentence from the top ranked clusters. Selected sentences almost always require revision and can be reformulated succinctly as it is often only a part of the sentence which is of interest. It is this multi-document summarization scenario which motivates our work.

Given a cluster of similar, or related, sentences, we aim at summarizing the most salient theme of it in a short single sentence. We refer to this task as *multi-sentence compression*. Defined this way, it comes close to sentence fusion which was originally introduced as a text-to-text generation technique of expressing content common to most of the input sentences in a single sentence (Barzilay & McKeown, 2005). However, since then the technique has been extended so that now fusion also stands for uniting complementary content in a single concise sentence (Filippova & Strube, 2008b; Krahmer et al., 2008). Since our method is not designed for the "union" kind of fusion, we think it is more appropriate to classify it as a sentence compression technique.

Two challenges of SC as well as text summarization are (i) important content selection and (ii) its readable presentation. Most existing systems use syntactic information to generate grammatical compressions. Incidentally, syntax also provides clues to what is likely to be important–e.g., the subject and the verb of the main clause are more likely to be important than a prepositional phrase or a verb from a relative clause. Of course, syntax is not the only way to gauge word or phrase importance. In the case of sentence compression being used for text summarization, one disposes of a rich context to identify important words or phrases. For example, recurring or semantically

Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010), pages 322–330,
Beijing, August 2010

similar words are likely to be relevant, and this information has been used in earlier SC systems (Hori et al., 2003; Clarke & Lapata, 2007, inter alia). Still, syntactic parsers are assumed to be indispensable tools for both sentence compression and fusion because syntactic constraints (handcrafted or learned from the data) seem to be the only way to control the grammaticality of the output. In this paper we are going to question this well-established belief and argue that just like in some cases syntax helps to find important content (e.g., when the input is an isolated sentence), in the multi-sentence case redundancy provides a reliable way of generating grammatical sentences. In particular, the important and novel points of our work are as follows:

- We present a simple and robust word graph-based method of generating succinct compressions which requires as little as a part of speech tagger and a list of stopwords.
- To our knowledge, it is the first method which requires neither a parser, nor handcrafted rules, nor a language model to generate reasonably grammatical output.
- In an extensive evaluation with native speakers we obtain encouraging results for English as well as for Spanish.

In the following section we present our approach to sentence compression (Sec. 2); then we introduce the baseline (Sec. 3) and the data (Sec. 4). In Section 5 we report about our experiments and discuss the results. Finally, Section 6 gives an overview of related work.

2 Multi-sentence Compression

A well-known challenge for extractive multi-document summarization systems is to produce non-redundant summaries. There are two standard ways of avoiding redundancy: either one adds sentences to the summary one-by-one and each time checks whether the sentence is significantly different from what is already there (e.g., using MMR), or one clusters related sentences and selects only one from each cluster. In both cases a selected sentence may include irrelevant information, so one wishes to compress it, usually by taking syntactic and lexical factors into account. However, we think this approach is suboptimal in this case and explore a different way. Instead of compressing a single sentence, we build a *word graph* from all the words of the related sentences and compress this graph.

A word graph is a directed graph where an edge from word *A* to word *B* represents an *adjacency* relation. It also contains the *start* and *end* nodes. Word graphs have been widely used in natural language processing for building language models, paraphrasing, alignment, etc. (see Sec. 6). Compared with dependency graphs, their use for sentence generation has been left largely unexplored, presumably because it seems that almost all the grammatical information is missing from this representation. Indeed, a link between a finite verb and an article does not correspond to any grammatical relation between the two. However, the premise for our work is that redundancy should be sufficient to identify not only important words but also salient links between words. In this section we present our approach to word graph compression. We begin by explaining the graph construction process and continue with the details of two compression methods.

2.1 Word Graph Construction

Given a set of related sentences $S = \{s_1, s_2, ...s_n\}$, we build a word graph by iteratively adding sentences to it. As an illustration, consider the four sentences below and the graph in Figure 1 obtained from them. Edge weights are omitted and italicized fragments from the sentences are replaced with dots for clarity.

(1) *The wife of a former U.S. president Bill* Clinton Hillary Clinton visited China last Monday.

(2) Hillary Clinton wanted to visit China last month *but postponed her plans* till Monday last week.

(3) Hillary Clinton paid *a visit to the People Republic* of China on Monday.

(4) Last week the *Secretary of State* Ms. Clinton visited Chinese officials.

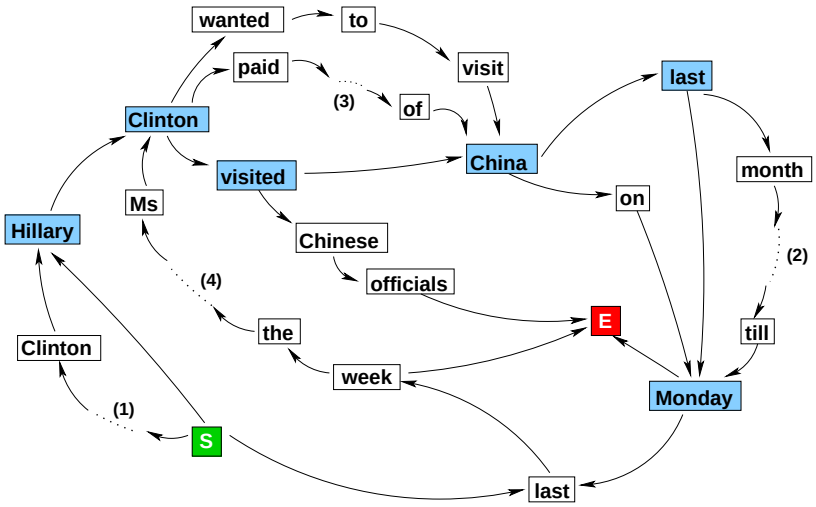

Figure 1: Word graph generated from sentences (1-4) and a possible compression path.

After the first sentence is added the graph is simply a string of word nodes (punctuation is excluded) plus the start and the end symbols (*S* and *E* in Fig. 1). A word from the following sentences is mapped onto a node in the graph provided that they have the exact same lowercased word form and the same part of speech[1] and that no word from this sentence has already been mapped onto this node. Using part of speech information reduces chances of merging verbs with nouns (e.g., *visit*) and generating ungrammatical sequences. If there is no candidate in the graph a new node is created.

Word mapping/creation is done in three steps for the following three groups of words: (1) non-stopwords[2] for which no candidate exists in the graph or for which an unambiguous mapping is possible; (2) non-stopwords for which there are either several possible candidates in the graph or which occur more than once in the sentence; (3) stopwords.

This procedure is similar to the one used by Barzilay & Lee (2003) in that we also first identify "backbone nodes" (unambiguous alignments) and then add mappings for which several possibilities exist. However, they build lattices, i.e., directed acyclic graphs, whereas our graphs may contain cycles. For the last two groups of words where mapping is ambiguous we check the immediate context (the preceding and following words in the sentence and the neighboring nodes in the graph) and select the candidate which has larger overlap in the context, or the one with a greater frequency (i.e., the one which has more words mapped onto it). For example, in Figure 1 when sentence (4) is to be added, there are two candidate nodes for *last*. The one pointing to *week* is selected as *week* is the word following *last* in (4). Stopwords are mapped only if there is some overlap in non-stopword neighbors, otherwise a new node is created.

Once all the words from the sentence are in place, we connect words adjacent in the sentence with directed edges. For newly created nodes, or nodes which were not connected before, we add an edge with a default weight of one. Edge weights between already connected nodes are increased by one. The same is done with the start and end nodes. Nodes store id's of the sentences their words come from as well as all their offset positions in those sentences.

The described alignment method is fairly simple and guarantees the following properties of the word graph: (i) every input sentence corresponds to a loopless path in the graph; (ii) words referring to the same entities or actions are likely to end up in one node; (iii) stopwords are only joined

[1] We use the OpenNLP package for tagging: `http://opennlp.sourceforge.net`.

[2] We generate a list of about 600 news-specific stopwords for English (including, e.g., *said*, *seems*) and took a publicly available list of about 180 stopwords for Spanish from `www.ranks.nl/stopwords/spanish.html`.

in one node if there is an overlap in context. The graph may generate a potentially endless amount of incomprehensible sequences connecting *start* and *end*. It is also likely to contain paths corresponding to good compressions, like the path connecting the nodes highlighted with blue in Figure 1. In the following we describe two our methods of finding the best path, that is, the best compression for the input sentences.

2.2 Shortest Path as Compression

What properties are characteristic of a good compression? It should neither be too long, nor too short. It should go through the nodes which represent important concepts but should not pass the same node several times. It should correspond to a likely word sequence. To satisfy these constraints we invert edge weights, i.e., link frequencies, and search for the shortest path (i.e., lightest in terms of the edge weights) from *start* to *end* of a predefined minimum length. This path is likely to mention salient words from the input and put together words found next to each other in many sentences. This is the first method we consider. We set a minimum path length (in words) to eight which appeared to be a reasonable threshold on a development set–paths shorter than seven words were often incomplete sentences.

Furthermore, to produce *informative* summaries which report about the main event of the sentence cluster, we filter paths which do not contain a verb node. For example, *Ozark's "Winter's Bone" at the 2010 Sundance Film Festival* might be a good title indicating what the article is about. However, it is not as informative as *"Winter's Bone" earned the grand jury prize at Sundance* which indeed conveys the gist of the event. Thus, we generate *K* shortest paths and filter all those which are shorter than eight words or do not contain a verb. The path with the minimum total weight is selected as the summary.

2.3 Improved Scoring and Reranking

The second configuration of our system employs a more sophisticated weighting function. The purpose of this function is two-fold: (i) to generate a grammatical compression, it favors strong links, i.e., links between words which appear significantly often in this order; (ii) to generate an informative compression, it promotes paths passing through salient nodes.

Strong links: Intuitively, we want the compression path to follow edges between words which are strongly associated with each other. Inverted edge frequency is not sufficient for that because it ignores the overall frequency of the nodes the edge connects. For example, edge frequency of three should count more if the edge connects two nodes with frequency of three rather than if their frequencies are much higher. Thus, we redefine edge weight as follows:

$$w(e_{i,j}) = \frac{\texttt{freq}(i) + \texttt{freq}(j)}{\texttt{freq}(e_{i,j})} \quad (1)$$

Furthermore, we also promote a connection between two nodes if there are multiple paths between them. For example, if some sentences speak of *president Barack Obama* or *president of the US Barack Obama*, and some sentences are about *president Obama*, we want to add some reward to the edge between *president* and *Obama*. However, longer paths between words are weak signals of word association. Therefore, the weight of an edge between the nodes i and j is reduced for every possible path between them but reduced proportionally to its length:

$$w'(e_{i,j}) = \frac{\texttt{freq}(i) + \texttt{freq}(j)}{\sum_{s \in S} \texttt{diff}(s,i,j)^{-1}} \quad (2)$$

where the function $\texttt{diff}(s,i,j)$ refers to the distance between the offset positions ($\texttt{pos}(s,i)$) of words i and j in sentence s and is defined as follows:

$$\texttt{diff}(s,i,j) = \begin{cases} \texttt{pos}(s,i) - \texttt{pos}(s,j) & \text{if } \texttt{pos}(s,i) < \texttt{pos}(s,j) \\ 0 & \text{otherwise} \end{cases} \quad (3)$$

Salient words: The function above only indicates how strong the association between two words is. It assigns equal weights to edges connecting words encountered in a single sentence and words encountered next to each other in every sentence. To generate a summary concerning the most salient events and entities, we force the path

to go through most frequent nodes by decreasing edge weight with respect to the frequency of the nodes it connects. Thus, we further redefine edge weight as follows:

$$w''(e_{i,j}) = \frac{w'(e_{i,j})}{\texttt{freq}(i) \times \texttt{freq}(j)} \quad (4)$$

We implement the K-shortest paths algorithm to find the fifty shortest paths from *start* to *end* using the weighting function in (4). We filter all the paths which are shorter than eight words and which do not pass a verb node. Finally, we rerank the remaining paths by normalizing the total path weight over its length. This way we obtain the path which has the lightest average edge weight.

3 Baseline

As a first baseline we are searching for the most probable string with respect to the sentence cluster. In particular, we use the Viterbi algorithm to find the sequence of words of a predefined length n which maximizes the bigram probability (MLE-based):

$$p(w_{1,n}) = p(w_1|\texttt{s})p(w_2|w_1)...p(\texttt{e}|w_n) \quad (5)$$

Similar to the shortest path implementation, we specify compression length and set it also here to eight tokens. However, the compressions obtained with this method are often unrelated to the main theme. The reason for that is that a token subsequence encountered in a single sentence is likely to get a high probability–all transition probabilities are equal to one–provided that the probability of entering this sequence is not too low. To amend this problem and to promote frequent words (i.e., words which are likely to be related to the main theme) we maximize the following baseline score which takes into account both the bigram probabilities and the token likelihood, $p(w_i)$, which is also estimated from the sentence cluster:

$$b(w_{1,n}) = p(w_1|\texttt{s})p(w_2|w_1)...p(\texttt{e}|w_n)\prod_i p(w_i) \quad (6)$$

4 Data Sources

As data for our experiments we use news articles presented in clusters on Google News[3]. The main reason for why we decided to use this service is that it is freely available and does the job of news classification and clustering with a production quality. Apart from that, it is a rich source of multilingual data.

We collected news clusters in English and Spanish, 10-30 articles each, 24 articles on average. To get sets of similar sentences we aggregated first sentences from every article in the cluster, removing duplicates. The article-initial sentence is known to provide a good summary of the article and has become a standard competitive baseline in summarization[4]. Hence, given that first sentences summarize the articles they belong to, which are in turn clustered as concerning the same event, those sentences are likely although not necessarily need to be similar.

From the total of 150 English clusters we reserved 70 for development and 80 for testing. For Spanish we collected 40 clusters, all for testing. We stripped off bylines and dates from the beginning of every sentence with a handful of regular expressions before feeding them to the baseline and our compression methods.

The data we use has two interesting properties: (i) article-initial sentences are on average longer than other sentences. In our case average sentence lengths for English and Spanish (without bylines) are 28 and 35 tokens, respectively. (ii) such sentence clusters are noisier than what one would expect in a summarization pipeline. Both properties make the task realistically hard and pose a challenge for the robustness of a compression method. If we show that reasonable compressions can be generated even from noisy clusters acquired from a publicly available news service, then we have a good reason to believe that the method will perform at least comparable on more carefully constructed clusters of shorter sentences.

[3] http://news.google.com

[4] See DUC/TAC competitions: http://www.nist.gov/tac

5 Evaluation

5.1 Experiment Design

The performance of the systems was assessed in an experiment with human raters, all native speakers. They were presented with a list of snippets of the articles from one cluster – first sentence and title linked to the original document. The raters were allowed to look up the articles if they need more background on the matter but this was not obligatory.

The first question concerned the quality of the sentence cluster. The raters were asked whether the cluster contained a single prevailing event, or whether it was too noisy and no theme stood out. Given how simple our sentence grouping procedure was, most clusters informed about more than one event. However, to answer the question positively it would be enough to identify one prevailing theme.

Below that, a summary and two further questions concerning its quality were displayed. Similar to most preceding work, we were interested in grammaticality and informativity of summaries. With respect to grammaticality, following Barzilay & McKeown (2005), we asked the raters to give one of the three possible ratings: *perfect* if the summary was a complete grammatical sentence (2 pts); *almost* if it required a minor editing, e.g., one mistake in articles or agreement (1 pt); *ungrammatical* if it was none of above (0 pts). We explicitly asked the raters to ignore lack or excess of capitalization or punctuation. Furthermore, based on the feedback from a preliminary evaluation, we provided an example in which we made clear that summaries consisting of a few phrases which cannot be reformulated as a complete sentence (e.g., *Early Monday a U.S. Navy ship.*) should not count as grammatical.

The final question, concerning informativity, had four possible options: *n/a* if the cluster is too noisy and unsummarizable in the first place; *perfect* if it conveys the gist of the main event and is more or less like the summary the person would produce himself (2 pts); *related* if it is related to the the main theme but misses something important (1 pt); *unrelated* if the summary is not related to the main theme (0 pts).

For each of the 80 sentence clusters (40 for Spanish) we generated three summaries with the three systems. Most summaries were rated by four raters, a few got only three ratings; no rater saw the same cluster twice.

5.2 Results

We report average grammaticality and informativity scores in Table 1. However, averaging system ratings over all clusters and raters is not justified in our case. It is important to remember that the score assignments (i.e., *0, 1, 2*) are arbitrary and that the score of one with respect to grammaticality (i.e., a minor mistake) is in fact closer to two than to zero. One could set the scores differently but even then, strictly speaking, it is not correct to average the scores as ratings do not define a metric space.

System	Gram	Info
Baseline	0.70 / 0.61	0.62 / 0.53
Shortest path	1.30 / 1.27	1.16 / 0.79
Shortest path++	1.44 / 1.25	1.30 / 1.25

Table 1: Average ratings for English / Spanish.

Therefore in Table 2 we present distributions over the three scores for both grammaticality and informativity together with average summary lengths in tokens. For both grammaticality and informativity, for every summary-cluster pair we did majority voting and resolved ties by assigning the lower score. For example, if a system got the ratings *1, 1, 2, 2* for a certain cluster, we counted this as *1*. We dismissed cases where the tie was between the maximum and the minimum score–this happened with some summaries which got just three scores (i.e., *0, 1, 2*) and accounted for $< 4\%$ of the cases. To obtain the informativity distribution we considered only clusters which were classified as containing a single prevailing event by at least ten raters. For English 75 out of 80 clusters qualified as such (37 out of 40 for Spanish). Similar to above, we dismissed about 3% tie cases where the ratings diverged significantly (e.g., *0, 1, 2*).

System	Gram-2	Gram-1	Gram-0	Info-2	Info-1	Info-0	Avg. Len.
Baseline (EN)	21%	15%	65%	18%	10%	73%	8
Shortest path (EN)	52%	16%	32%	36%	33%	31%	10
Shortest path++ (EN)	64%	13%	23%	52%	32%	16%	12
Baseline (ES)	12%	15%	74%	9%	19%	72%	8
Shortest path (ES)	58%	21%	21%	23%	26%	51%	10
Shortest path++ (ES)	50%	21%	29%	40%	40%	20%	12

Table 2: Distribution over possible ratings and average length for English and Spanish.

5.3 Discussion

The difference between the baseline and our shortest path systems is striking. Although more than 20% of the baseline summaries are perfectly grammatical, the gap to the improved version of shortest paths is significant, about 43%. The same holds for the percentage of informative summaries (18% vs. 52%). Both numbers are likely to be understated as we chose to resolve all ties not in our favor. 84% of the summaries generated by the improved method are related to the main theme of the cluster, and more than 60% of those (52% of the total summaries) convey the very gist of it without missing any important information. Comparing the two configurations we have proposed, improved scoring function and reranking we added on top of the shortest path method were both rewarding. Interestingly, even the straightforward approach of choosing the shortest path of a minimum length already guarantees a grammatical summary in more than half of the cases.

An interesting difference in the performance for Spanish and English is that shortest path generates more grammatical sentences than the improved version of it. However, the price for higher grammaticality scores is a huge drop in informativity: half of such summaries are not related to the main theme at all, whereas 40% of the summaries generated by the improved version got the highest rating. A possible reason for the poorer performance for Spanish is that we used a much smaller list of stopwords which did not include news-specific words like, e.g., *dijo* (*said*) which resulted in denser graphs. In the future, we would like to apply the method to more languages and experiment with longer lists of stopwords.

One may notice that the summaries produced by the baseline are shorter than those generated by the shortest paths which might look like a reason for its comparatively poor performance. However, the main source of errors for the baseline was its inability to keep track of the words already present in the summary, so it is unlikely that longer sequences would be of a much higher quality. The sentences generated by the baseline were often repetitive, e.g., *The food tax on food tax on food*. This is not an issue with the shortest path approaches as they never include loops when edge weights are strictly positive.

The reranking we added to the shortest path method is the reason for why the summaries generated by the improved version of the system are on average slightly longer than those produced by the simpler version. The average lengths for both systems are drastically shorter than the average length of the sentences served as input (10/12 vs. 28 tokens in English or 35 tokens for Spanish). This corresponds to the compression rate of 36-43% (29-34% for Spanish) which is comparatively "aggressive" as it usually varies between 50-80% in other systems.

6 Comparison with Related Work

6.1 Sentence Compression

In the last ten years a lot of research has been devoted to sentence compression. Most studies share two properties: (1) they rely on syntax, and (2) they are supervised. The degree of syntax-dependence varies between methods. Some utilize a parser to identify and later keep certain important relations but do not require a complete parse (Clarke & Lapata, 2008), or use a syntactic representation to extract features (McDonald, 2006). For other approaches correct syntac-

tic trees are crucial to obtain grammatical compressions (Galley & McKeown, 2007; Filippova & Strube, 2008a; Cohn & Lapata, 2009). Handcrafted rules (Dorr et al., 2003) as well as language models also have been utilized to generate fluent compressions (Hori et al., 2003; Clarke & Lapata, 2008).

6.2 Sentence Generation

To date the work on sentence fusion is completely dependency syntax-based. Input sentences are parsed into trees, from those trees a new dependency structure is generated, and this structure is finally converted into a sentence (Barzilay & McKeown, 2005; Filippova & Strube, 2008b; Wan et al., 2009). Parser quality is of crucial importance for such methods, and to our knowledge no attempt has been made to generate novel sentences without adhering to dependency representations. In the future, it would be of interest to compare our method with a syntax-based fusion method. Syntax-lean methods have been explored for headline generation (Banko et al., 2000; Dorr et al., 2003; Jin & Hauptmann, 2003). However, they do not aim at generating complete sentences or informative summaries but rather to indicate what the news is about.

6.3 Word Graphs and Lattices

Perhaps the work of Barzilay & Lee (2003) who align comparable sentences to generate sentence-level paraphrases seems closest to ours in that we both use word graphs for text generation. However, this is a fairly general similarity, as both the goal and the implementation are different. While we search for an optimal weighting function in noisy graphs to identify readable and informative compressions, they induce paraphrase patterns from unweighted paths in much smaller DAGs obtained from highly similar sentences. Shen et al. (2006) is another example of using word lattices to find paraphrases. Unlike Barzilay & Lee (2003), they propose to use syntax to obtain accurate alignments. Numerous examples of the utility of word lattices come from the field of finite state automata, language modeling, speech recognition, parsing and machine translation (Mohri, 1997, inter alia).

7 Conclusions

We considered the task of generating a short informative summary for a set of related sentences, called multi-sentence compression, which arises naturally in the context of multi-document text summarization. We presented a simple but robust method which proceeds by finding shortest paths in word graphs. The novelty of our work is that we demonstrated that reasonable compressions can be obtained without any syntactic information if a good weighting function is defined. This distinguishes our work from earlier research on sentence fusion and compression which relies on syntactic representations and/or language models. We provided the details of an extensive evaluation on English and Spanish data and reported high grammaticality as well as informativity scores. In the future we would like to experiment with other languages and eschew using part-of-speech information.

Acknowledgements: I am thankful to Keith Hall for the discussions on this work and the very helpful feedback on an earlier draft of this paper.

References

Banko, M., V. O. Mittal & M. J. Witbrock (2000). Headline generation based on statistical translation. In *Proc. of ACL-00*, pp. 318–325.

Barzilay, R. & L. Lee (2003). Learning to paraphrase: An unsupervized approach using multi-sequence alignment. In *Proc. of HLT-NAACL-03*, pp. 16–23.

Barzilay, R. & K. R. McKeown (2005). Sentence fusion for multidocument news summarization. *Computational Linguistics*, 31(3):297–327.

Clarke, J. & M. Lapata (2007). Modelling compression with discourse constraints. In *Proc. of EMNLP-CoNLL-07*, pp. 1–11.

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

Cohn, T. & M. Lapata (2009). Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, 34:637–674.

Corston-Oliver, S. H. (2001). Text compaction for display on very small screens. In *Proceedings of the NAACL Workshop on Automatic Summarization*, Pittsburg, PA, 3 June 2001, pp. 89–98.

Dorr, B., D. Zajic & R. Schwartz (2003). Hedge trimmer: A parse-and-trim approach to headline generation. In *Proceedings of the Text Summarization Workshop at HLT-NAACL-03*, Edmonton, Alberta, Canada, 2003, pp. 1–8.

Filippova, K. & M. Strube (2008a). Dependency tree based sentence compression. In *Proc. of INLG-08*, pp. 25–32.

Filippova, K. & M. Strube (2008b). Sentence fusion via dependency graph compression. In *Proc. of EMNLP-08*, pp. 177–185.

Galley, M. & K. R. McKeown (2007). Lexicalized Markov grammars for sentence compression. In *Proc. of NAACL-HLT-07*, pp. 180–187.

Hori, C., S. Furui, R. Malkin, H. Yu & A. Waibel (2003). A statistical approach to automatic speech summarization. *EURASIP Journal on Applied Signal Processing*, 2:128–139.

Jin, R. & A. G. Hauptmann (2003). Automatic title generation for spoken broadcast news. In *Proc. of HLT-01*, pp. 1–3.

Jing, H. & K. McKeown (2000). Cut and paste based text summarization. In *Proc. of NAACL-00*, pp. 178–185.

Krahmer, E., E. Marsi & P. van Pelt (2008). Query-based sentence fusion is better defined and leads to more preferred results than generic sentence fusion. In *Proc. of ACL-HLT-08*, pp. 193–196.

Mani, I. (2001). *Automatic Summarization.* Amsterdam, Philadelphia: John Benjamins.

McDonald, R. (2006). Discriminative sentence compression with soft syntactic evidence. In *Proc. of EACL-06*, pp. 297–304.

Mohri, M. (1997). Finite-state transducers in language and speech processing. *Computational Linguistics*, 23(2):269–311.

Shen, S., D. Radev, A. Patel & G. Erkan (2006). Adding syntax to dynamic programming for aligning comparable texts for generation of paraphrases. In *Proc. of COLING-ACL-06*, pp. 747–754.

Wan, S., M. Dras, R. Dale & C. Paris (2009). Improving grammaticality in statistical sentence generation: Introducing a dependency spanning tree algorithm with an argument satisfaction model. In *Proc. of EACL-09*, pp. 852–860.