Dependency Tree Based Sentence Compression

Katja Filippova and **Michael Strube**
EML Research gGmbH
Schloß-Wolfsbrunnenweg 33
69118 Heidelberg, Germany
http://www.eml-research.de/nlp

Abstract

We present a novel unsupervised method for sentence compression which relies on a dependency tree representation and shortens sentences by removing subtrees. An automatic evaluation shows that our method obtains result comparable or superior to the state of the art. We demonstrate that the choice of the parser affects the performance of the system. We also apply the method to German and report the results of an evaluation with humans.

1 Introduction

Within the field of text-to-text generation, the sentence compression task can be defined as follows: given a sentence S, consisting of words $w_1 w_2 ... w_n$, what is a subset of the words of S, such that it is grammatical and preserves essential information from S? There are many applications which would benefit from a robust compression system, such as subtitle generation, compression for mobile devices with a limited screen size, or news digests. Given that to date most text and speech summarization systems are extractive, sentence compression techniques are a common way to deal with redundancy in their output.

In recent years, a number of approaches to sentence compression have been developed (Jing, 2001; Knight & Marcu, 2002; Gagnon & Da Sylva, 2005; Turner & Charniak, 2005; Clarke & Lapata, 2008, inter alia). Many explicitly rely on a language model, usually a trigram model, to produce grammatical output (Knight & Marcu, 2002; Hori & Furui, 2004; Turner & Charniak, 2005; Galley & McKeown, 2007). Testing the grammaticality of the output with a language model is justified when working with a language with rigid word order like English, and all but one approach mentioned have been applied to English data. However, compressing sentences in languages with less rigid word order needs a deeper analysis to test grammaticality. And even for languages with rigid word order the trigram model ignores the structure of the sentence and therefore may significantly distort the meaning of the source sentence. Approaches going beyond the word level either require a comprehensive lexicon (Jing, 2001), or manually devised rules (Gagnon & Da Sylva, 2005; Clarke & Lapata, 2008) to determine prunable constituents. A lexicon is not always available, whereas the hand-crafted rules may not cover all cases and are too general to be universally applicable (e.g. *PPs can be pruned*).

In this paper we present a novel unsupervised approach to sentence compression which is motivated by the belief that the grammaticality of the output can be better ensured by compressing trees. In particular, given a dependency tree, we want to prune subtrees which are neither obligatory syntactic arguments, nor contribute important information to the content of the sentence. A tree pruning approach does not generate new dependencies and is unlikely to produce a compression with a totally different meaning. Our approach is unsupervised and adaptable to other languages, the main requirement being that there are a dependency parser and a corpus available for the languages. We test our approach on English and German data sets and obtain results comparable or superior to the state of the art.

2 Related Work

Many existing compression systems use a noisy-channel approach and rely on a language model to test the grammaticality of the output (Knight & Marcu, 2002; Turner & Charniak, 2005; Galley & McKeown, 2007). Other ways to ensure grammaticality and to decide whether a constituent is obligatory or may be pruned are to utilize a subcategorization lexicon (Jing, 2001), or to define a set of generally prunable constituents. Gagnon & Da Sylva (2005) prune dependency trees by removing prepositional complements of the verb, subordinate clauses and noun appositions. Apparently, this does not guarantee grammaticality in all cases. It may also eliminate important information from the tree.

Most approaches are supervised and require training data to learn which words or constituents can be dropped from a sentence (Riezler et al., 2003; McDonald, 2006). However, it is difficult to obtain training data. Still, there are few unsupervised methods. For example, Hori & Furui (2004) introduce a scoring function which relies on such information sources as word significance score and language model. A compression of a given length which maximizes the scoring function is then found with dynamic programming. Clarke & Lapata (2008) present another unsupervised approach. They formulate the task as an optimization problem and solve it with integer linear programming. Two scores contribute to their objective function – a trigram language model score and a word significance score. Additionally, the grammaticality of the output is ensured by a handful of linguistic constraints, stating e.g. which arguments should be preserved.

In this paper we suggest an alternative to the popular language model basis for compression systems – a method which compresses dependency trees and not strings of words. We will argue that our formulation has the following advantages: firstly, the approach is unsupervised, the only requirement being that there is a sufficiently large corpus and a dependency parser available. Secondly, it requires neither a subcategorization lexicon nor hand-crafted rules to decide which arguments are obligatory. Thirdly, it finds a globally optimal compression by taking syntax and word importance into account.

3 Dependency Based Compression

Our method compresses sentences in that it removes dependency edges from the dependency tree of a sentence. The aim is to preserve dependencies which are either required for the output to be grammatical or have an important word as the dependent. The algorithm proceeds in three steps: tree transformation (Section 3.1), tree compression (Section 3.2) and tree linearization (Section 3.3).

3.1 Tree Transformation

Before a dependency tree is compressed, i.e. before some of the dependencies are removed, the tree is modified. We will demonstrate the effect of the transformations with the sentence below:

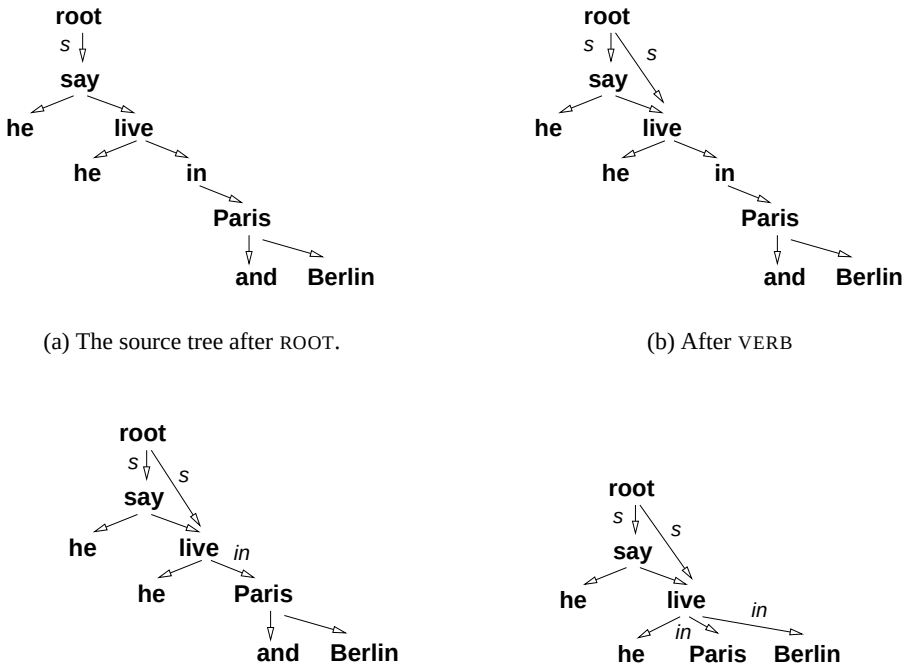
(1) He said that he lived in Paris and Berlin

The first transformation (ROOT) inserts an explicit rootnode (Fig. 1(a)). The result of the second transformation (VERB) is that every inflected verb in the tree gets an edge originating from the rootnode (Fig. 1(b)). All edges outgoing from the rootnode bear the *s* label. Apart from that we remove auxiliary edges and memorize such grammatical properties as voice, tense or negation for verbs.

The purpose of the remaining transformations is to make relations between open-class words more explicit. We want to decide on pruning an edge judging from two considerations: (i) how important for the head this argument is; (ii) how informative the dependent word is. As an example, consider a source sentence given in (2). Here, we want to decide whether one prepositional phrase (or both) can be pruned without making the resulting sentence ungrammatical.

(2) After some time, he moved to London.

It would not be very helpful to check whether an argument attached with the label *pp* is obligatory for the verb *move*. Looking at a particular preposition (*after* vs. *to*) would be more enlightening. This motivates the PREP transformation which removes prepositional nodes and places them as labels on the edge from their head to the respective noun (Fig. 1(c)). We also decompose a chain of conjoined elements (CONJ) and attach each of them to the head of the first element in the chain with the label the first



(a) The source tree after ROOT.

(b) After VERB

(c) After PREP

(d) After CONJ

Figure 1: The dependency structure of *He said that he lived in Paris and Berlin* after the transformations

element attaches to its head with (Fig. 1(d)). This way we can retain any of the conjoined elements in the compression and do not have to preserve the whole sequence of them if we are interested in only one. This last transformation is not applied to verbs.

3.2 Tree Compression

We formulate the compression task as an optimization problem which we solve using integer linear programming[1]. Given a transformed dependency tree (a graph if new edges have been added), we decide which dependency edges to remove. For each directed dependency edge from head h to word w we thus introduce a binary variable $x^l_{h,w}$ where *l* stands for the edge's label:

$$x^l_{h,w} = \begin{cases} 1 & \text{if the dependency is preserved} \\ 0 & \text{otherwise} \end{cases} \quad (1)$$

The goal is to find a subtree which gets the highest score of the objective function (2) to which both the probability of dependencies ($P(l|h)$) and the importance of dependent words ($I(w)$) contribute:

$$f(X) = \sum_x x^l_{h,w} \cdot P(l|h) \cdot I(w) \quad (2)$$

Intuitively, the conditional probabilities prevent us from removing obligatory dependencies from the tree. For example, $P(subj|work)$ is higher than $P(with|work)$, and therefore the subject will be preserved whereas the prepositional label and thus the whole PP can be pruned. This way we do not have to create an additional constraint for every obligatory argument (e.g. subject or direct object). Neither do we require a subcategorization lexicon to look up which arguments are obligatory for a certain verb. Verb arguments are preserved because the dependency edges, with which they are attached to the head, get high scores. Table 1 presents the probabilities of a number of labels given that the head is *study*. Table 2 presents the probabilities for their German counterparts.

Note that if we would not apply the PREP transformation we would not be able to distinguish be-

[1]In our implementation we use lp_solve (`http://sourceforge.net/projects/lpsolve`).

subj	dobj	in	at	after	with	to
0.16	0.13	0.05	0.04	0.01	0.01	0.01

Table 1: Probabilities of *subj, d(irect)obj, in, at, after, with, to* given the verb *study*

subj	obja	in	an	nach	mit	zu
0.88	0.74	0.44	0.42	0.09	0.02	0.01

Table 2: Probabilities of *subj, obja(ccusative), in, at, after, with, to* given the verb *studieren*

tween different prepositions and could only calculate $P(pp|studieren)$ which would not be very informative. The probabilities for English are lower than those for German because we calculate the conditional probabilities given word lemma. In English, the part of speech information cannot be induced from the lemma and thus the set of possible labels of a node is on average larger than in German.

There are many ways in which word importance, $I(w)$ can be defined. Here, we use the formula introduced by Clarke & Lapata (2008) which is a modification of the significance score of Hori et al. (2003):

$$I(w_i) = \frac{l}{N} \cdot f_i \log \frac{F_A}{F_i} \quad (3)$$

w_i is the topic word (either noun or verb), f_i is the frequency of w_i in the document, F_i is the frequency of w_i in the corpus, and F_A is the sum of frequencies of all topic words in the corpus. l is the number of clause nodes above w and N is the maximum level of embedding of the sentence w belongs to.

The objective function is subject to constraints of two kinds. The constraints of the first kind are stuctural and ensure that the preserved dependencies result in a tree. (4) ensures that each word has one head at most. (5) ensures connectivity in the tree. (6) restricts the size of the resulting tree to α words.

$$\forall w \in W, \sum_{h,l} x^l_{h,w} \leq 1 \quad (4)$$

$$\forall w \in W, \sum_{h,l} x^l_{h,w} - \frac{1}{|W|} \sum_{u,l} x^l_{w,u} \geq 0 \quad (5)$$

$$\sum_{x} x^l_{h,w} \leq \alpha \quad (6)$$

α is a function of the length of the source sentence in open-class words. The function is not linear since the degree of compression increases with the length of the sentence. The compression rate of human-generated sentences is about 70% (Clarke & Lapata, 2008)[2]. To approximate this value, we set the proportion of deleted words to be 20% for short sentences (5-9 non-stop words), this value increases up to 50% for long sentences (30+ words).

The constraints of the second type ensure the syntactic validity of the output tree and explicitly state which edges should be preserved. These constraints can be general as well as conditional. The former ensure that an edge is preserved if its source node is retained in the output. Conditional syntactic constraints state that an edge has to be preserved if (and only if) a certain other edge is preserved. We have only one syntactic constraint which states that a subordinate conjunction (*sc*) should be preserved if and only if the clause it belongs to functions as a subordinate clause (*sub*) in the output. If it is taken as the main clause, the conjunction should be dropped. In terms of edges, this can be formulated as follows (7):

$$\forall x^{sc}_{w,u}, x^{sub}_{h,w} - x^{sc}_{w,u} = 0 \quad (7)$$

Due to the constraint (4), the compressed subtree is always rooted in the node added as a result of the first transformation. A compression of a sentence to an embedded clause is not possible unless one preserves the structure above the embedded clause. Often, however, main clauses are less important than an embedded clause. For example, given the sentence *He said they have to be held in Beirut* it is the embedded clause which is informative and to which the source sentence should be compressed. The purpose of the VERB modification is to amend exactly this problem. Having an edge from the rootnode to every inflected verb allows us to compress the source sentence to any clause.

3.3 Tree Linearization

A very simple but reasonable linearization technique is to present the words of a compressed sentence in the order they are found in the source sentence. This method has been applied before and this is how we

[2]Higher rates correspond to longer compressions.

linearize the trees obtained for the English data. Unfortunately, this method cannot be directly applied to German because of the constraints on word order in this language. One of the rules of German grammar states that in the main clause the inflected part of the verb occupies the second position, the first position being occupied by exactly one constituent. Therefore, if the sentence initial position in a source sentence is occupied by a constituent which got pruned off as a result of compression, the verb becomes the first element of the sentence which results in an undesirable output. There are linearization methods developed for German which find an optimal word order for a sentence (Ringger et al., 2004; Filippova & Strube, 2007). We use our recent method to linearize compressed trees.

4 Corpora and Annotation

We apply our method to sentences from two corpora in English and German. These are presented below.

English Compression Corpus: The English data we use is a document-based compression corpus from the British National Corpus and American News Text Corpus which consists of 82 news stories[3]. We parsed the corpus with RASP (Briscoe et al., 2006) and with the Stanford PCFG parser (Klein & Manning, 2003). The output of the former is a set of dependency relations whereas the latter provides an option for converting the output into dependency format (de Marneffe et al., 2006) which we use.

TüBa-D/Z: The German corpus we use is a collection of 1,000 newspaper articles (Telljohann et al., 2003)[4]. Sentence boundaries, morphology, dependency structure and anaphoric relations are manually annotated in this corpus.

RASP has been used by Clarke & Lapata (2008) whose state of the art results we compare with ours. We use not only RASP but also the Stanford parser for several reasons. Apart from being accurate, the latter has an elaborated set of dependency relations (48 vs. 15 of RASP) which is not overly large (compared with the 106 grammatical relations of the Link Parser). This is important for our system which relies on syntactic information when making pruning decisions. A comparison between the Stanford parser and two dependency parsers, MiniPar and Link Parser, showed a decent performance of the former (de Marneffe et al., 2006). It is also of interest to see to what extent the choice of the parser influences the results.

Apart from the corpora listed above, we use the Tipster corpus to calculate conditional probabilities of syntactic labels given head lemmas as well as word significance scores. The significance score is calculated from the total number of 128 million nouns and verbs. Conditional probabilities are calculated from a much smaller portion of Tipster (about 6 million tokens). The latter number is comparable to the size of the data set we use to compute the probabilities for German. There, we use a corpus of about 4,000 articles from the German Wikipedia to calculate conditional probabilities and significance scores. The corpus is parsed with the highly accurate CDG parser (Foth & Menzel, 2006) and has the same dependency format as TüBa-D/Z (Versley, 2005).

Although all corpora are annotated with dependency relations, there are considerable differences between the annotation of the English and German data sets. The phrase to dependency structure conversion done by the Stanford parser makes the semantic head of the constituent its syntactic head. For example, in the sentence *He is right* it is the adjective *right* which is the root of the tree. Unlike that, sentences from the German corpora always have a verb as the root. To unify the formats, we write a set of rules to make the verb the root of the tree in all cases.

5 Evaluation

We evaluate the results automatically as well as with human subjects. To assess the performance of the method on the English data, we calculate the F-measure on grammatical relations. Following Riezler et al. (2003), we calculate average precision and recall as the amount of grammatical relations shared between the output of our system and the gold stan-

[3]The corpus is available from `http://homepages.inf.ed.ac.uk/s0460084/data`.

[4]The corpus is available from `http://www.sfs.uni-tuebingen.de/en_tuebadz.shtml`.

dard variant divided over the total number of relations in the output and in the human-generated compression respectively. According to Clarke & Lapata (2006), this measure reliably correlates with human judgements. The results of our evaluation as well as the state of the art results reported by Clarke & Lapata (2008) (LM+SIG+CONSTR), whose system uses language model scoring (LM), word significance score (SIG), and linguistic constraints (CONSTR), are presented in Table 3. The F-measure reported by Clarke & Lapata (2008) is calculated with RASP which their system builds upon. For our system we present the results obtained on the data parsed with RASP as well as with the Stanford parser (SP). In both cases the F-measure is found with RASP in order to allow for a fair comparison between the three systems. We recalculate the compression rate for the gold standard ignoring punctuation. On the whole corpus the compression rate turns out to be slightly higher than that reported by Clarke & Lapata (2008) (70.3%).

	F-measure	*compr.rate*
LM+SIG+CONSTR	40.5	72.0%
DEP-BASED (RASP)	40.7	49.6%
DEP-BASED (SP)	49.3	69.3%
GOLD	-	72.1%

Table 3: Average results on the English corpus

As there are no human-generated compressions for German data, we evaluate the performance of the method in terms of *grammaticality* and *importance* by means of an experiment with native speakers. In the experiment, humans are presented with a source sentence and its compression which they are asked to evaluate on two five-point scales. Higher grades are given to better sentences. Importance represents the amount of relevant information from the source sentence retained in the compression. Since our method does not generate punctuation, the judges are asked to ignore errors due to missing commas. Five participants took part in the experiment and each rated the total of 25 sentences originating from a randomly chosen newspaper article. Their ratings as well as the ratings reported by Clarke & Lapata (2008) on English corpus are presented in Table 4.

	grammar	*importance*
LM+SIG+CONSTR	3.76	3.53
DEP-BASED (DE)	3.62	3.21

Table 4: Average results for the German data

6 Discussion

The results on the English data are comparable with or superior to the state of the art. These were obtained with a single linguistic constraint (7) and without any elaborated resources which makes our system adaptable to other languages. This suggests that tree compression is a better basis for sentence compression systems than language model-oriented word deletion.

In order to explain why the choice of parser significantly influences the performance of the method, we calculate the precision P defined as the number of dependencies shared by a human-generated compression (dep_c) and the source sentence (dep_s) divided over the total number of dependencies found in the compression:

$$P = \frac{|dep_c \cap dep_s|}{|dep_c|} \quad (8)$$

The intuition is that if a parser does not reach high precision on gold standard sentences, i.e. if it does not assign similar dependency structures to a source sentence and its compression, then it is hopeless to expect it to produce good compression with our dependency-based method. However, the precision does not have to be as high as 100% because of, e.g., changes within a chain of conjoined elements or appositions. The precision of the two parsers calculated over the compression corpus is presented in Table 5.

	RASP	Stanford parser
precision	79.6%	84.3%

Table 5: Precision of the parsers

The precision of the Stanford parser is about 5% higher than that of RASP. In our opinion, this partly explains why the use of the Stanford parser increases the F-measure by 9 points. Another possible reason for this improvement is that the Stanford parser identifies three times more dependency relations than

RASP and thus allows for finer distinctions between the arguments of different types.

Another point concerns the compression rates. The compressions generated with RASP are considerably shorter than those generated with the Stanford parser. This is mainly due to the fact that the structure output by RASP is not necessarily a tree or a connected graph. In such cases only the first subtree of the sentence is taken as input and compressed.

The results on the German set are not conclusive since the number of human judges is relatively small. Still, these preliminary results are comparable to those reported for English and thus give us some evidence that the method can be adapted to languages other than English. Interestingly, the importance score depends on the grammaticality of the sentence. A grammatical sentence can convey unimportant information but it was never the case that an ungrammatical sentence got a high rating on the importance scale. Some of the human judges told us that they had difficulties assigning the importance score to ungrammatical sentences.

7 Conclusions

We presented a new compression method which compresses dependency trees and does not rely on a language model to test grammaticality. The method is unsupervised and can be easily adapted to languages other than English. It does not require a subcategorization lexicon or elaborated hand-crafted rules to decide which arguments can be pruned and finds a globally optimal compression taking syntax and word importance into account. We demonstrated that the performance of the system depends on the parser and suggested a way to estimate how well a parser is suited for the compression task. The results indicate that the dependency-based approach is an alternative to the language model-based one which is worth pursuing.

Acknowledgements: This work has been funded by the Klaus Tschira Foundation, Heidelberg, Germany. The first author has been supported by a KTF grant (09.009.2004). We would like to thank Yannick Versley who helped us convert TüBa-D/Z in the CDG format and Elke Teich and the three anonymous reviewers for their useful comments.

References

Briscoe, Edward, John Carroll & Rebecca Watson (2006). The second release of the RASP system. In *Proceedings of the COLING-ACL Interactive Presentation Session*, Sydney, Australia, 2006, pp. 77–80.

Clarke, James & Mirella Lapata (2006). Models for sentence compression: A comparison across domains, training requirements and evaluation measures. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia, 17–21 July 2006, pp. 377–385.

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

de Marneffe, Marie-Catherine, Bill MacCartney & Christopher D. Manning (2006). Generating typed dependency parses from phrase structure parses. In *Proceedings of the 5th International Conference on Language Resources and Evaluation*, Genoa, Italy, 22–28 May 2006, pp. 449–454.

Filippova, Katja & Michael Strube (2007). Generating constituent order in German clauses. In *Proceedings of the 45th Annual Meeting of the Association for Computational Linguistics*, Prague, Czech Republic, 23–30 June 2007, pp. 320–327.

Foth, Kilian & Wolfgang Menzel (2006). Hybrid parsing: Using probabilistic models as predictors for a symbolic parser. In *Proceedings of the 21st International Conference on Computational Linguistics and 44th Annual Meeting of the Association for Computational Linguistics*, Sydney, Australia, 17–21 July 2006, pp. 321–327.

Gagnon, Michel & Lyne Da Sylva (2005). Text summarization by sentence extraction and syntactic pruning. In *Proceedings of Computational Linguistics in the North East*, Gatineau, Québec, Canada, 26 August 2005.

Galley, Michel & Kathleen R. McKeown (2007). Lexicalized Markov grammars for sentence com-

pression. In *Proceedings of Human Language Technologies 2007: The Conference of the North American Chapter of the Association for Computational Linguistics*, Rochester, N.Y., 22–27 April 2007, pp. 180–187.

Hori, Chiori & Sadaoki Furui (2004). Speech summarization: An approach through word extraction and a method for evaluation. *IEEE Transactions on Information and Systems*, E87-D(1):15–25.

Hori, Chiori, Sadaoki Furui, Rob Malkin, Hua Yu & Alex Waibel (2003). A statistical approach to automatic speech summarization. *EURASIP Journal on Applied Signal Processing*, 2:128–139.

Jing, Hongyan (2001). *Cut-and-Paste Text Summarization*, (Ph.D. thesis). Computer Science Department, Columbia University, New York, N.Y.

Klein, Dan & Christopher D. Manning (2003). Accurate unlexicalized parsing. In *Proceedings of the 41st Annual Meeting of the Association for Computational Linguistics*, Sapporo, Japan, 7–12 July 2003, pp. 423–430.

Knight, Kevin & Daniel Marcu (2002). Summarization beyond sentence extraction: A probabilistic approach to sentence compression. *Artificial Intelligence*, 139(1):91–107.

McDonald, Ryan (2006). Discriminative sentence compression with soft syntactic evidence. In *Proceedings of the 11th Conference of the European Chapter of the Association for Computational Linguistics*, Trento, Italy, 3–7 April 2006, pp. 297–304.

Riezler, Stefan, Tracy H. King, Richard Crouch & Annie Zaenen (2003). Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for Lexical-Functional Grammar. In *Proceedings of the Human Language Technology Conference of the North American Chapter of the Association for Computational Linguistics*, Edmonton, Alberta, Canada, 27 May –1 June 2003, pp. 118–125.

Ringger, Eric, Michael Gamon, Robert C. Moore, David Rojas, Martine Smets & Simon Corston-Oliver (2004). Linguistically informed statistical models of constituent structure for ordering in sentence realization. In *Proceedings of the 20th International Conference on Computational Linguistics*, Geneva, Switzerland, 23–27 August 2004, pp. 673–679.

Telljohann, Heike, Erhard W. Hinrichs & Sandra Kübler (2003). *Stylebook for the Tübingen treebank of written German (TüBa-D/Z)*. Technical Report: Seminar für Sprachwissenschaft, Universität Tübingen, Tübingen, Germany.

Turner, Jenine & Eugene Charniak (2005). Supervised and unsupervised learning for sentence compression. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics*, Ann Arbor, Mich., 25–30 June 2005, pp. 290–297.

Versley, Yannick (2005). Parser evaluation across text types. In *Proceedings of the 4th Workshop on Treebanks and Linguistic Theories*, Barcelona, Spain, 9-10 December 2005.