Text Simplification as Tree Labeling

Joachim Bingel
Centre for Language Technology
University of Copenhagen
bingel@hum.ku.dk

Anders Søgaard
Centre for Language Technology
University of Copenhagen
soegaard@hum.ku.dk

Abstract

We present a new, structured approach to text simplification using conditional random fields over top-down traversals of dependency graphs that jointly predicts possible compressions and paraphrases. Our model reaches readability scores comparable to word-based compression approaches across a range of metrics and human judgements while maintaining more of the important information.

1 Introduction

Sentence-level text simplification is the problem of automatically modifying sentences so that they become easier to read, while maintaining most of the relevant information in them. This can benefit applications as pre-processing for machine translation (Bernth, 1998) and assisting technologies for readers with reduced literacy (Carroll et al., 1999; Watanabe et al., 2009; Rello et al., 2013).

Sentence-level text simplification ignores sentence splitting and reordering, and typically focuses on *compression* (deletion of words) and *paraphrasing* or *lexical substitution* (Cohn and Lapata, 2008). We include paraphrasing and lexical substitution here, while previous work in sentence simplification has often focused exclusively on deletion. Approaches that address compression and paraphrasing (or more tasks) integrally include (Zhu et al., 2010; Narayan and Gardent, 2014; Mandya et al., 2014).

Simplification beyond deletion is motivated by Pitler's (2010) observation that abstractive sentence summaries written by humans often "include paraphrases or synonyms ('said' versus 'stated') and use alternative syntactic constructions ('gave John the book' versus 'gave the book to John')." Such lexical or syntactic alternations may contribute strongly to the readability of a sentence if they replace difficult words with shorter or more familiar ones, in particular for low-literacy readers (Rello et al., 2013). Our joint approach to deletion and paraphrasing works against the limitation that abstractive simplifications "are not capable of being generated by [...] most sentence compression algorithms" (Pitler, 2010).

Furthermore, a central concern in text simplification is to ensure the grammaticality of the output, especially with low-proficiency readers as the target audience. Our approach to this problem is to remove or paraphrase entire syntactic units in the original sentence, thus avoiding to remove phrase heads without removing their arguments or modifiers. Like Filippova and Strube (2008), we rely on dependency structures rather than constituent structures, which promises more robust syntactic analysis and allows us to operate on discontinuous syntactic units.

Contributions We present a sentence simplification model which is, to the best of our knowledge, the first model that uses structured prediction over dependency trees *and* models compression and paraphrasing jointly. Our model uses Viterbi decoding rather than scoring of all candidates and outputs probabilities reflecting model confidence.

2 Data

We use the publicly available Google compression data set,[1] which consists of 10,000 English sentence triples with (1) the original sentence as present in the body of an online news article, (2) a headline based on the original sentence, and (3) a compression that is automatically derived from the original such that it only contains word forms

[1] http://storage.googleapis.com/sentencecomp/compressiondata.json

Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics, pages 337–343,
Berlin, Germany, August 7-12, 2016.

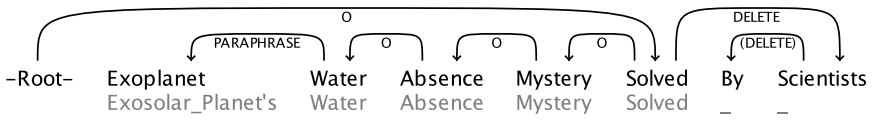

Figure 1: An example simplification tree

present in the original, preserving their order. The following sentence triple exemplifies these different versions:

(1) *In official documents released earlier this month it appears the Queen of England used the wrong name for the Republic of Ireland when writing to president Patrick Hillery.*

(2) *Queen elizabeth ii used wrong name for Republic*

(3) *The Queen of England used the wrong name for the Republic of Ireland.*

The data is pre-processed with the Stanford CoreNLP tools (Manning et al., 2014), retrieving lemmas, parts-of-speech, named entities and dependency trees. We reserve the first 200 sentences from the data set for evaluation, the next 200 for tuning parameters (including the used PPDB versions, see next paragraph), and use the remaining 9,600 sentences for training our model.

Deletion and paraphrase targets As our approach operates on dependency trees, aiming to prune or paraphrase subtrees from the dependency tree of a sentence, we identify deleted or paraphrased subtrees, marking their heads with a corresponding label. A subtree receives a Delete label if none of the words subsumed by this subtree occur in the compressed version of the sentence.

We identify paraphrased subsequences in an original sentence by looking up the subsequence string in the Paraphrase Database (PPDB) (Ganitkevitch et al., 2013) and testing if one of its possible paraphrases occurs in the headline version of the sentence in question. The Paraphrase Database 1.0 is a set of phrasal and lexical pairs that were automatically acquired from bilingual parallel corpora, and thus contain a portion of flawed paraphrase pairs. The database comes in a number of different sizes, where small editions are restricted to high-precision paraphrases with relatively high paraphrase probabilities. As the two smallest editions of PPDB only yield a very low number of paraphrase targets (less than 100 in the entire Google compression data set), we opt to employ a medium-sized version of the resource (size 'L') and find a total of 510 phrasal and lexical paraphrases in the corpus.

3 Method

We assume that text simplification is a generative process on syntactic dependency graphs with a paraphrase dictionary. A dependency graph $G = (V, A)$ is a labeled directed graph in the standard graph-theoretic sense and consists of nodes, V, and arcs, A, such that for sentence $S = w_0 w_1 \dots w_n$ and label set R, $V \subseteq \{w_0, w_1, \dots, w_n\}$, and $A \subseteq V \times R \times V$ hold, and if $(w_i, r, w_j) \in A$ then $(w_i, r', w_j) \neq A$ for all $r' \neq r$. We restrict the dependency graphs to the class of trees, i.e., for $(w_i, r, w_j) \in A$, if $(w_k, r, w_j) \in A$ then $k = i$.

The generative process traverses the tree in a top-down fashion, deleting or paraphrasing subtrees (see Figure 1). Note that elements in subtrees dominated by a deleted node are automatically deleted (analogously for paraphrases).

For each dependency tree $G = (V, A)$ in a training set of T sentences, we derive an input sequence of K-dimensional feature vectors $\mathbf{x} = x_1, \dots, x_n$ and an output sequence of $\mathbf{y} = y_1, \dots, y_n$. Our tree-to-string simplification model is a second-order linear-chain conditional random field (CRF)

$$p(\mathbf{y}|\mathbf{x}) = \frac{1}{Z(\mathbf{x})} \prod_{i=1}^{n} \exp\{\sum_{k=1}^{K} \theta_k f_k(y_t, y_{t-1}, x_t)\}$$

with $y_i = \mathsf{Delete}$ if and only if x_i represents the least upper bound in G covering a deleted span in the training data, and $y_i = \mathsf{Paraphrase}$ if and only if x_i represents the least upper bound in G covering a paraphrased span in the training data. For example, if the entire sentence is deleted, and $(w_0, r, w_i) \in A$, then $y_i = \mathsf{Delete}$ (but $y_j = \mathsf{Leave}$ for $j \neq i$).

This encoding means that theoretically we can predict to paraphrase a subtree that is dominated by a node which is in turn predicted to be deleted

(or vice versa). However, once an operation is carried out on a subtree, none of its dominated nodes are considered in the remainder of the top-down simplification process. Giving preference to operations at higher-level syntactic environments in this manner serves as a mechanism to resolve ambiguities in the decision process by taking a wider context into account.

Furthermore, predicting a node to get paraphrased at the right corner of a deleted subtree can potentially influence labeling decisions outside this subtree as a consequence of the dynamic-program Viterbi decoding. We acknowledge that this is a theoretical drawback of the presented approach, but given that we do not observe any such dependency graphs in our data, we do not expect this to be a serious problem in most cases.

Whenever our model predicts that a subtree be paraphrased, we look up the respective token sequence in PPDB and replace it with the candidate paraphrase (if available) that maximises the product of frequency and translation probability according to PPDB.

Features for CRF model We train a second-order CRF model using MarMoT (Mueller et al., 2013), an efficient higher-order CRF implementation. The model computes its observational probabilities from features based on properties of the subtree root token (incl. POS, language model probability, NE mention, word difficulty), of the internal structure of the subtree (incl. number of children, depth, length of sequence), and of the external grammatical structure (incl. dependency relation, parent POS, distance from parent, position in sentence).

4 Evaluation

Baselines In the following experiments, we compare our approach to state-of-the-art approaches to sentence compression and joint compression/paraphrasing. For the first of these two categories, we consider the LSTM system described in Filippova et al. (2015) as well as the results reported therein for the MIRA system (McDonald, 2006). As a joint approach, we consider Reluctant Trimmer (RT), a simplification system that employs synchronous dependency grammars (Mandya et al., 2014). Since the LSTM system requires great amounts of training data, which were not available to us, we cannot reproduce its output and therefore limit our comparison of human rankings to the eleven output examples provided in the paper.

		Recall	Precision	F1
Reluctant Trimmer				
tokens	Delete	54.60	20.23	29.52
	Paraphrase	01.67	66.67	03.27
	Leave	52.27	78.54	54.60
Tree Labeling				
subtrees	Delete	43.31	67.54	52.77
	Paraphrase	23.85	50.89	32.48
	Leave	94.29	84.82	89.30
tokens	Delete	49.67	77.16	60.44
	Paraphrase	21.16	51.52	30.00
	Leave	80.33	50.91	62.32

Table 1: Performance on joint deletion and paraphrasing detection for our tree labeling system (evaluating both on entire subtrees and token level) as well as for the RT baseline (tokens only). Note that RT is trained on the (Simple) English Wikipedia, not on the Google compressions, and therefore the results may not be directly comparable.

F-Scores We first evaluate our tree labeling model (TL) on its ability to predict subtree deletion and paraphrasing (i.e. whether a subtree should be paraphrased, independent of the actual replacement). The results for this evaluation setup, as well as word-level performance, are listed in Table 1 and compared to RT. Note that for deletion and paraphrasing, our model consistently has higher precision than recall, thus generating more confident simplifications and less ungrammatical output.

Automated Readability Scores Table 2 reports the compression ratio (CR, percentage of retained words) as well as automated readability scores that our model achieves on the test set and compares it to the output of the RT baseline. Our system manages to compress the original texts by more than one third, but the gold simplifications (headlines and compressions) are still considerably shorter.

Our approach improves readability as measured by the Flesch Reading Ease score[2] (Flesch,

[2]The negative value that the headlines receive for this met-

Data version	CR↓	Flesch↑	Dale-C.↓
Original	—	49.15	9.55
Headlines	**0.32***	-80.77*	17.61*
Compressions	0.40*	**70.80***	9.56
TL output	0.62*	56.25*	9.30*
RT output	0.86*	60.65*	**9.27***

Table 2: Compression ratios and automatic readability scores for the Google compression data set, compared to the system output. Readability is indicated by a high Flesh Reading Ease score and a low Dale-Chall score. * indicates differences compared to the original sentences that are significant at $p < 10^{-3}$.

System	Readability	Informativeness
MIRA	4.31	3.55
LSTM	**4.51**	3.78
TL	4.14	**4.01**
RT (11)	3.09	4.12
LSTM (11)	**4.23**	3.42
TL (11)	4.21	**4.15**

Table 3: Mean readability and informativeness ratings for the first 200 sentences in the Google data (upper) and for the 11 sample sentences listed in Filippova et al. (2015) (lower).

1948) and the Dale-Chall formula (Dale and Chall, 1948). The former score measures textual difficulty as a function of sentence length and the number of syllables per word, while the latter aims to estimate a US school grade level at which a text can be well understood, based on a vocabulary list. Both metrics deem the output of our system easier to read than the original texts, while the Dale-Chall formula also rates our system better than the gold simplifications.

Human Readability Ratings Following Filippova et al. (2015) in their evaluation setup for the sake of comparability, we ask raters to assign scores on a one-to-five Likert scale to the first 200 sentences from the Google compression data paired with the output of our system. Each pair is rated by three native or near-native speakers of English.

The raters are asked to evaluate the sentence pairs for *readability* and *informativeness*. The former, following Filippova et al. (2015), "covers the grammatical correctness, comprehensibility and fluency of the output." The latter metric pertains to the relation between the original sentence and the system output as it "measures the amount of important content preserved in the compression."

Table 3 compares the performance of our model to the figures reported in Filippova et al. (2015) for their LSTM model and McDonald's (2006) system (MIRA). For a comparison with the same judges, we repeat the evaluation with the 11 sample output compressions listed in Filippova et al. (2015)as well as the respective output from Reluctant Trimmer; see the lower part of Table 3. The results suggest that, compared to the compression-only LSTMs, our approach yields comparable performance in terms of readability, while maintaining more of the central information in the original sentences. Compared to RT, our system does considerably better in terms of readability and retains slightly more of the important information.

5 Related Work

Several approaches to sentence compression have been presented in the last decade. Knight and Marcu (2002) and Turner and Charniak (2005) apply noisy channel models, using language models to control for grammaticality. McDonald (2006) introduces a different approach, discriminatively training a scoring function, informed by syntactic features, to score all possible subtrees of a sentence. His work was inspired by Riezler et al. (2003) scoring substrings generated from LFG parses. A third approach to sentence compression is sequence labeling, which has been explored by Elming et al. (2013) using linear-chain CRFs with syntactic features, and more recently by Filippova et al. (2015) and Klerke et al. (2016) using recurrent neural networks with LSTM cells.

Most recent approaches to sentence compression make use of syntactic analysis, either by operating directly on trees (Riezler et al., 2003; Nomoto, 2007; Filippova and Strube, 2008; Cohn and Lapata, 2008; Cohn and Lapata, 2009) or by incorporating syntactic information in their model (McDonald, 2006; Clarke and Lapata, 2008). Recently, however, Filippova et al. (2015) presented an approach to sentence compression using

ric is due to an over-representation of longer words in headlines.

	Original Sentence & *Simplifications*
O	OG&E is warning customers about a prepaid debit card scam that is targeting utility customers across the county.
C	*OG&E is warning customers about a scam.*
R	*OG&E is warning customers about a debit card scam that is targeting utility customers across the country.*
T	*OG&E is warning customers **regarding** a prepaid debit card scam.*
O	The husband of murdered Melbourne woman Jill Meagher will return to Ireland later this month "to clear his head" while fighting for parole board changes.
C	*The husband of murdered woman Jill Meagher will return to Ireland.*
R	*The husband of Melbourne woman Jill Meagher will return to Ireland this month to clear his head fighting for parole board changes.*
T	*The husband of murdered Melbourne woman Jill Meagher will return to Ireland.*
O	A research project has found that taxi drivers often don't know what the speed limit is.
C	*Taxi drivers don't know the speed limit is.*
R	*A research project has found that drivers often **do not** know what the speed limit is.*
T	*A project has found taxi drivers don't know what the speed limit is.*

Table 4: Example output for original sentences (O) as generated by the Reluctant Trimmer baseline (R) and our tree labeling system (T), as well as the headline-generated Google compressions (C).

LSTMs with word embeddings, with no syntactic features. We return to working directly on trees, presenting a tree-to-string model of sentence simplification. Our model has interesting similarities to (Riezler et al., 2003), but uses Viterbi decoding rather than scoring of all candidates. Also, it follows Cohn and Lapata (2008) in going beyond most of these models, modeling compression *and* paraphrasing.

For lexical simplification, most systems typically use pre-compiled dictionaries (Devlin, 1999; Inui et al., 2003) and select the synonym candidate with the highest frequency. More recently, Baeza-Yates et al. (2015) introduced an algorithm for lexical simplification in Spanish that selects the best synonym candidate in a context-sensitive fashion.

Cohn and Lapata (2008), Woodsend and Lapata (2011) and Mandya et al. (2014) present *joint* approaches to compression and paraphrasing that are based on (quasi-) synchronous grammars, and similarly Zhu et al. (2010) take a syntax-based approach, but employ a probabilistic model of various simplification operations. Napoles et al. (2011) do not use syntactic information, but instead employ a character-based metric to compress and paraphrase.

6 Conclusion

We presented a new approach to sentence simplification that uses linear-chain conditional random fields over dependency graphs to jointly predict compression and paraphrasing of entire syntactic units. The objective of our model is to delete or paraphrase entire subtrees in dependency graphs as a strategy to avoid ungrammatical output. Our approach makes innovative use of a three-fold parallel monolingual corpus that features headlines and compressions to learn paraphrases and deletions, respectively. Human evaluation shows that our approach leads to readability figures that are comparable to previous state-of-the-art approaches to the more basic sentence compression task, and better than previous work on joint compression and paraphrasing. While our model does rely on syntactic analysis, it only needs a tiny fraction (less than 0.5%) of the training data used by Filippova et al. (2015).

Acknowledgments

This research was partially funded by the ERC Starting Grant LOWLANDS No. 313695, as well as by Trygfonden.

References

Ricardo Baeza-Yates, Luz Rello, and Julia Dembowski. 2015. Cassa: A context-aware synonym simplification algorithm. *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology*, pages 1380–1385.

Arendse Bernth. 1998. EasyEnglish: Preprocessing for MT. In *Proceedings of the Second International Workshop on Controlled Language Applications*, pages 30–41.

John Carroll, Guido Minnen, Darren Pearce, Yvonne Canning, Siobhan Devlin, and John Tait. 1999. Simplifying text for language-impaired readers. In *Proceedings of EACL*, volume 99, pages 269–270.

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

Trevor Cohn and Mirella Lapata. 2008. Sentence compression beyond word deletion. In *Proceedings of the 22nd International Conference on Computational Linguistics-Volume 1*, pages 137–144. Association for Computational Linguistics.

Trevor Cohn and Mirella Lapata. 2009. Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, pages 637–674.

Edgar Dale and Jeanne S Chall. 1948. A formula for predicting readability: Instructions. *Educational research bulletin*, pages 37–54.

Siobhan Lucy Devlin. 1999. *Simplifying natural language for aphasic readers.* Ph.D. thesis, University of Sunderland.

Jakob Elming, Anders Johannsen, Sigrid Klerke, Emanuele Lapponi, Hector Martinez Alonso, and Anders Søgaard. 2013. Down-stream effects of tree-to-dependency conversions. In *HLT-NAACL*, pages 617–626.

Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. In *Proceedings of the Fifth International Natural Language Generation Conference*, pages 25–32. Association for Computational Linguistics.

Katja Filippova, Enrique Alfonseca, Carlos A Colmenares, Lukasz Kaiser, and Oriol Vinyals. 2015. Sentence compression by deletion with lstms. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing*, pages 360–368.

Rudolph Flesch. 1948. A new readability yardstick. *Journal of applied psychology*, 32(3):221.

Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of NAACL-HLT*, pages 758–764, Atlanta, Georgia, June. Association for Computational Linguistics.

Kentaro Inui, Atsushi Fujita, Tetsuro Takahashi, Ryu Iida, and Tomoya Iwakura. 2003. Text simplification for reading assistance: a project note. In *Proceedings of the second international workshop on Paraphrasing-Volume 16*, pages 9–16. Association for Computational Linguistics.

Sigrid Klerke, Yoav Goldberg, and Anders Søgaard. 2016. Improving sentence compression by learning to predict gaze. In *Proceedings of ACL 2016 (short)*.

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

Angrosh A. Mandya, Tadashi Nomoto, and Advaith Siddharthan. 2014. Lexico-syntactic text simplification and compression with typed dependencies. In *COLING*, pages 1996–2006.

Christopher D. Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven J. Bethard, and David McClosky. 2014. The Stanford CoreNLP natural language processing toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 55–60.

Ryan T McDonald. 2006. Discriminative sentence compression with soft syntactic evidence. In *EACL*.

Thomas Mueller, Helmut Schmid, and Hinrich Schütze. 2013. Efficient higher-order CRFs for morphological tagging. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing*, pages 322–332, Seattle, Washington, USA, October. Association for Computational Linguistics.

Courtney Napoles, Chris Callison-Burch, Juri Ganitkevitch, and Benjamin Van Durme. 2011. Paraphrastic sentence compression with a character-based metric: Tightening without deletion. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*, pages 84–90. Association for Computational Linguistics.

Shashi Narayan and Claire Gardent. 2014. Hybrid simplification using deep semantics and machine translation. In *the 52nd Annual Meeting of the Association for Computational Linguistics*, pages 435–445.

Tadashi Nomoto. 2007. Discriminative sentence compression with conditional random fields. *Information Processing and Management: an International Journal*, 43(6):1571–1587.

Emily Pitler. 2010. Methods for sentence compression. Technical report, Department of Computer and Information Science, University of Pennsylvania.

Luz Rello, Ricardo Baeza-Yates, Laura Dempere-Marco, and Horacio Saggion. 2013. Frequent words improve readability and short words improve understandability for people with dyslexia. In *Human-Computer Interaction–INTERACT 2013*, pages 203–219. Springer.

Stefan Riezler, Tracy H King, Richard Crouch, and Annie Zaenen. 2003. Statistical sentence condensation using ambiguity packing and stochastic disambiguation methods for lexical-functional grammar.

In *Proceedings of the 2003 Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1*, pages 118–125. Association for Computational Linguistics.

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

Willian Massami Watanabe, Arnaldo Candido Junior, Vinícius Rodriguez Uzêda, Renata Pontin de Mattos Fortes, Thiago Alexandre Salgueiro Pardo, and Sandra Maria Aluísio. 2009. Facilita: reading assistance for low-literacy readers. In *Proceedings of the 27th ACM international conference on Design of communication*, pages 29–36. ACM.

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the conference on empirical methods in natural language processing*, pages 409–420. Association for Computational Linguistics.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd international conference on computational linguistics*, pages 1353–1361. Association for Computational Linguistics.