# Automatic Text Simplification

Zepp Uibo, Lena Shakurova, Jelmer Jansen

***Abstract*—The goal of text simplification is to reduce lexical or syntactic complexity of the input text. Text simplification systems are aimed to make texts more readable and easier to understand for a broader audience. Moreover, simplified texts might be used as an input for different NLP systems and supposedly more successfully processed.**

**One of common approaches to text simplification is lexical text simplification. It normally consists of four main steps: identification of complex words, generation of substitution candidates, filtering the candidates that fit the context and then ranking of the selected substitutes according to their simplicity. Each of this tasks can be approached in different ways.**

**In this paper we tried to implement different approaches to lexical text simplification, compared the performance and gave qualitative description.**

## I. Introduction

The goal of text simplification is to produce a simpler and more comprehensible version of the input text. This technology has been used to help reading comprehension for people with various cognitive diseases, including dyslexia, autism and aphasia [3]. Since the 2010's, neural sequence-to-sequence models have been introduced into the field of text simplification, which has greatly improved the performance of these systems [1].

Neural text simplification (NTS) is often compared to neural machine translation (NMT) within a single language, since the process consists of the same main phases. The encoding, where an abstract representation is found for the input sentence, and the decoding, where the output sentence is generated one word at a time using this representation [1]. In NTS these functions are fulfilled by two connected recurrent neural networks, usually LSTM networks [1].

In this project, we will try to build on this existing knowledge by testing an existing neural text simplification architecture [1] in combination with a new lexical simplification layer and seeing how this impacts the performance of the network.

## II. Background

### A. Neural Text Simplification Architectures

Almost all neural text simplification systems make use of recurrent neural networks (RNNs). The most common recurrent unit used is that of the Long Short Term Memory network (LSTM) [4]. Using one sequence of these units (the encoder), the words of the input sentence are converted to abstract representations one at a time, and then another LSTM layer (the decoder) is used to generate the output sentence [1]. The words are converted to their vector representation through a system of embeddings, such as Word2Vec or GloVe.

email:

### B. Evaluation Metrics

Although human evaluation of simplified text is still considered the ultimate benchmark [], it is also costly and time-consuming. However, due to the sensitive nature of language all current evaluation metrics for NTS use human evaluation in some way, usually as a reference for statistical methods to measure how appropriate the machine output is. To compare these, various automatic evaluation metrics have been developed to determine the quality of text simplification systems.

The SARI-method uses N-gram-based versions of precision and recall to measure the quality of words added and removed by the NMT system. When an added word appears in the human reference data, the SARI score is increased, while a penalty is added for each new word that does not appear in the reference data. For deletion, only precision is used to measure the simplification quality, since the deletion of words is much more likely to change the meaning of the output or make it ungrammatical [3].

The BLEU-method and the expanded iBLEU methods also use the amount of shared N-grams between the output text and input text to determine the quality of the output. However, BLUE only uses precision and not recall, as the bad translations can easily have an inflated recall in the case of multiple references []. iBLEU extends the BLEU algorithm so that it takes into account the diversity as well as the correctness of the generated output [**?**]. This is done by summing the BLEU score between the candidate sentence and the references as well as the score between the candidate sentence and the input, scaled by the parameter $\alpha$ which is usually set to 0.9. Finally, Xu et al. [3] created another extension of BLEU called FKBLEU, which takes the geometric mean between the iBLUE score and Flesich-Kincaid Index for the input and the output. Because of the incorporation of the FK-score, this measure also takes into account the readability of the generated text [3].

### C. Word Embeddings and Attention Mechanisms

Word embedding models are used by the systems to convert the words of the input text to a vector representation, which also gives the network information about the context of these words. These embeddings can be either locally trained, pre-trained on a large dataset or a combination of these two.

Attentional mechanisms were first applied to machine translation by Luang et al[2]. Later, Nisioi et al.[1] included an attentional mechanism in their neural text simplification model. Attention-based models fall into two broad categories, global and local. The idea behind global models is to use all hidden encoder states to create the context vector. This is done by comparing the target hidden state to the source hidden state. Local models instead

use computes the context vector from a window around position $p$ in the source hidden states.

### *D. Lexical Simplification*

The process of lexical simplification is usually divided into the following stages. First, the complex words are identified, which can be done by looking at their length, frequency, context and expected familiarity. Then, a set of alternatives is created either by querying a linguistic database or generating them automatically. These alternatives are then sorted and filtered, to make sure that the final substitute is a good fit for the context while also having good enough simplicity and frequency scores. When the ranking of substitutes is finally finished, the top substitute is chosen to replace the original complex word. [6].

#### D.1 Complex Word Identification

The earliest lexical simplification approaches simplified every word of the input, but this often led to ungrammatical or incoherent results. Thus, researchers developed thresholds to determine which words had to be simplified and which did not. These thresholds can be based on either the length or frequency of the words and are easy to implement, but Bott et al.[9] proved that it is hard to define a threshold that can accurately distinguish between simple and complex words [6].

Another option is to use a lexicon-based approach, where the input is compared to manually created lists of simple and complex words. This has proven to be very effective, but since the lists have to be created manually it is also vary labour-intensive. Finally, complex words can also be identified implicitly in other steps of the process (for example by discarding rules that replace a word with a more complex version) or by using trained machine learning classifiers.

The problem with 30% approach is that it will replace words even when the sentence is already simple enough.

#### D.2 Substitution Generation

Substitute words can either be queried from a database or generated automatically. The former approach is usually very reliable, but the databases are costly to create and as such only available for a few languages. Generating substitutes automatically, for example by comparing descriptions in dictionaries, has been tried for many different approaches with mixed results[6].

#### D.3 Filtering and Ranking

Many different approaches have been tried on how to select a fitting substitution for the chosen complex word. Similar to complex word identification, early approaches did not include solution rankings, but this often led to changes in meaning and so new methods were developed to test the validity of different replacements. These include sense labeling, both implicit and explicit, the use of part-of-speech tagging and semantic similarity filtering. The sense-labeling approach often manages to conserve the meaning of the original sentences, but at the cost of their grammatical correctness. PoS-tagging has been shown to improve the quality of the output, but by itself it is not enough to produce a good ranking. Semantic similarity filtering is a relatively new approach but it seems to produce the best results so far. Here, words are converted to a meaning representation and the distance between these representations is then used to create a ranking [6].

## III. Project

For this project, our idea was to combine a neural text simplification system with a rule-based lexical simplification system, and seeing if this led to an improved performance. The network we used was developed by Nisioi et al.[1] and the lexical simplification system a combination of approaches.

### *A. System Description*

#### A.1 Lexical Module

Similar to most lexical simplification systems, the lexical simplification module of our system first has to identify complex words, then generate their substitutes, before then filtering and ranking these to determine the best replacement.

For complex word identification we tried two approaches - we either chose 30% of least frequent words in the sentence (frequency is calculated on BROWN corpus) or any words that are not in the top [N] most frequent words in the corpus. We compared the effectiveness of these two methods on [training data], which gave the following result.

For our substitute generation, we use 3 different sources: the synonyms of the original word in Wordnet, top 15 most similar words in Word2vec and one-word paraphrases from paraphrase database PPDB. From the last data source, we filter out all substitutions that have the same root as the original word, and also words that have the Levenshtein distance of 1 - following the intuition that synonyms are unlikely to differ only by one symbol, while trying to avoid substitutions based only on differences in American and British English spellings.

For filtering we check whether the new word fits the context (using the bigram corpus). We do not substitute words that start with capital letter, since these are likely to be names. We also made the decision to only simplify verbs, adverbs and nouns, as the attempted simplification of function words is much more likely to change the meaning of the sentence.

For ranking we use a different method for each condition, which is summarized in table 1. The first condition only uses the bigram score, which is the averaged frequency of the bigrams composed of the candidate word and its left or right neighbor. These bigrams were taken from the Corpus of Contemporary American English [8]. The other two conditions sort the alternatives by their brown corpus frequency. The third condition also checks whether each alternative fits in the context, using PoS-tags and the bigram corpus mentioned earlier, whereas the second one

does not.

In the end, we choose the most suitable word and convert it to the same form as the original word (tense, plural/singular) using pattern library [10].

TABLE I
LEXICAL SIMPLIFICATION CONDITIONS

| | Version 1 | Version 2 | Version 3 |
|---|---|---|---|
| Word Identification | Frequency (fix) | | |
| Substitution Generation | WordNet + top 15 Word2vec + PPDB | | |
| Filtering | - | - | Bigram frequency + PoS tags |
| Ranking | Bigram score | Frequency | Frequency |

### A.2 Example functioning

Here is an example of how all three conditions process an input sentence. The sentence used here is from the Zimpliz [] repository, which we used as an inspiration for our own lexical simplification system.

**Original:** "Her face was a synthesis of perfect symmetry and unusual proportion; he could have gazed at it for hours, trying to locate the source of its fascination."

- **Condition 1:** Her face was a deduction of good balance and unexpected amount ; he could have stared at it for hours , trying to find the beginning of its fascination.
- **Condition 2:** Her face was a deduction of good balance and strange number ; he could have smiled at it for hours , trying to place the reason of its fascination.
- **Condition 3:** Her face was a deduction of good balance and strange number ; he could have smiled at it for hours , trying to place the reason of its passion.

As can be seen above, the replacements are quite similar across the conditions. However, "source" is replaced with "beginning" in the bigram-condition and with "reason" in the frequency conditions. In instances like this, the bigram condition seems to produce more contextually accurate results than the frequency-based ones.

### A.3 Neural module

For the neural part of the system, we used the same architecture as Nisioi et al.[1] with some minor modifications. This architecture consists of an encoder and decoder LSTM units, as well as an attention module and it can be combined with word embeddings. The shape of this model can be seen in figure 1.

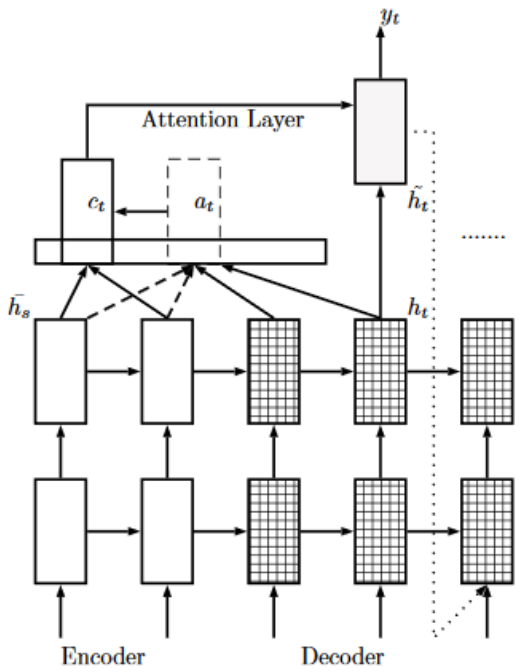


Fig. 1. Neural module architecture.

### *B. Experimental design*

In the experiment, we compare the performance of our neural module on the original input text with that on the simplified text. Thus, we have four conditions in total, namely the original input text plus the three different versions of our lexical simplification module. For comparison we use the 359 sentences from the testing part of the text simplification dataset released by Xu et al. [3], which contains 8 simplifications for each sentence created by different Amazon Turkers. More exactly, we use the evaluation script released by Sinisioi et al. [1] and their custom data format for calculating both SARI and BLEU scores on our simplified set.

### B.1 Expectation

Personally we expect the condition with bigram-based ranking to achieve the best score, since many of the most frequent replacements for complex words do not fit in the context of the original sentence. The condition with the frequency score and context should come next, with the condition with only frequency performing worse than both, since it has no contextual information at all.

## IV. RESULTS

The results are visible in table 2.

TABLE II
LEXICAL SIMPLIFICATION CONDITIONS

| | SARI | BLEU | Version 1 |
|---|---|---|---|
| Version 2 | | | Version 3 |

## V. Summary

## VI. Conclusion

What we found out... Future research...

## References


[1] Nisioi, S., tajner, S., Ponzetto, S. P., & Dinu, L. P. (2017). Exploring neural text simplification models. In "Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics" (Volume 2: Short Papers) (Vol. 2, pp. 85-91).

[2] Luong, M. T., Pham, H., & Manning, C. D. (2015). Effective approaches to attention-based neural machine translation. "arXiv preprint arXiv:1508.04025."

[3] Xu, W., Napoles, C., Pavlick, E., Chen, Q., & Callison-Burch, C. (2016). Optimizing statistical machine translation for text simplification. "Transactions of the Association for Computational Linguistics", 4, 401-415.
ibleu Sun, H., & Zhou, M. (2012, July). Joint learning of a dual SMT system for paraphrase generation. In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers-Volume 2 (pp. 38-42). Association for Computational Linguistics.

[4] Zhang, X., & Lapata, M. (2017). Sentence simplification with deep reinforcement learning. "arXiv preprint arXiv:1703.10931".

[5] Paetzold, G., & Specia, L. (2015). Lexenstein: A framework for lexical simplification. Proceedings of ACL-IJCNLP 2015 System Demonstrations, 85-90.

[6] Paetzold, G. H., & Specia, L. (2017). A survey on lexical simplification. Journal of Artificial Intelligence Research, 60, 549-593.

[7] Williams, S., Siddharthan, A., Nenkova, A. (2013). Proceedings of the Second Workshop on Predicting and Improving Text Readability for Target Reader Populations. In Proceedings of the Second Workshop on Predicting and Improving Text Readability for Target Reader Populations.

[8] Davies, Mark. (2008-) The Corpus of Contemporary American English (COCA): 560 million words, 1990-present. Available online at https://corpus.byu.edu/coca/.

[9] Bott, S., Rello, L., Drndarevic, B., Saggion, H. (2012). Can spanish be simpler? lexsis: Lexical simplification for spanish. Proceedings of COLING 2012, 357-374.

[10] Pattern. (2010, November 26). Retrieved June 25, 2018, from https://www.clips.uantwerpen.be/pattern

[11]

# Appendix