Article:

Paetzold, G.H. and Specia, L. orcid.org/0000-0002-5495-3128 (2017) A survey on lexical simplification. Journal of Artificial Intelligence Research, 60. pp. 549-593. ISSN 1076-9757

https://doi.org/10.1613/jair.5526

Journal of Artificial Intelligence Research 60 (2017) 549-593 Submitted 03/17; published 11/17

A Survey on Lexical Simplification

Gustavo H. Paetzold G.H.PAETZOLD@SHEFFIELD.AC.UK
Lucia Specia L.SPECIA@SHEFFIELD.AC.UK
The University of Sheffield
Western Bank
Sheffield
United Kingdom

Abstract

Lexical Simplification is the process of replacing complex words in a given sentence with simpler alternatives of equivalent meaning. This task has wide applicability both as an assistive technology for readers with cognitive impairments or disabilities, such as Dyslexia and Aphasia, and as a pre-processing tool for other Natural Language Processing tasks, such as machine translation and summarisation. The problem is commonly framed as a pipeline of four steps: the identification of complex words, the generation of substitution candidates, the selection of those candidates that fit the context, and the ranking of the selected substitutes according to their simplicity. In this survey we review the literature for each step in this typical Lexical Simplification pipeline and provide a benchmarking of existing approaches for these steps on publicly available datasets. We also provide pointers for datasets and resources available for the task.

1. Introduction

In the context of Natural Language Processing (NLP), the task of Lexical Simplification (LS) aims to perform Text Simplification (TS) by focusing on lexical information. It can be formally described as the task of replacing words in a given sentence in order to make it simple, without applying any modifications to its syntactic structure. In our survey, the term “complex words” generally refers to individual words, even though most concepts and methods can also be applied to multiword expressions and some of the described work covers these expressions. For those cases we use the term “complex expressions”. LS consists in identifying complex words (hereafter “target words”) and finding the best candidate substitution for those target words. The best substitution needs to be simpler while keeping the sentence grammatical and preserving its meaning as much as possible. This is a very challenging task, especially because different target audiences will have distinct needs, for example, speakers of different languages will be more or less familiar with different subsets of a second language vocabulary.

As first pointed out by Chandrasekar, Doran, and Srinivas (1996), Text Simplification can be applied in a wide array of contexts, both as a reading assistance tool for end-users, and as a pre-processing step to other NLP tasks. Previous work has shown that LS plays a crucial role in TS. It has been found to be an effective way of making texts more accessible to various audiences, such as people suffering from Dyslexia (Rello, Baeza-Yates, Dempere-Marco, & Saggion, 2013b; Rello, Baeza-Yates, Bott, & Saggion, 2013a; Rello, Bautista, Baeza-Yates, Gervás, Hervás, & Saggion, 2013c), Aphasia (Carroll, Minnen, Canning, Devlin, & Tait, 1998; Devlin, 1999) and those with low literacy levels (Aluisio & Gasperin, 2010; Watanabe, Junior, Uzêda, Fortes, Pardo, & Aluísio, 2009). The task has been applied to different languages, such as English (Carroll et al., 1998; Horn, Manduca, & Kauchak, 2014; Glavaš & Štajner, 2015; Paetzold, 2015), Spanish (Rello et al., 2013b; Bott, Rello, Drndarevic, & Saggion, 2012), Swedish (Keskisärkkä, 2012) and Portuguese (Aluisio & Gasperin, 2010), and on a variety of text domains, such as news (Carroll et al., 1998) and medical (Kandula, Curtis, & Zeng-Treitler, 2010).

Research in Psycholinguistics explains why LS is such an effective way of simplifying text. The work by Hirsh and Nation (1992) and Nation (2001) show that English learners need to be familiar with 95% of a text's vocabulary in order to achieve basic comprehension, and familiar with 98%of a text's vocabulary for leisure. They observe that those who are familiar with the vocabulary of a text can often understand the entirety of its meaning even if the grammatical constructs used are confusing to them. These findings suggest that replacing words that are unknown to the reader, which is what a reliable lexical simplifier does, has great potential to effectively increase the accessibility of a text. Nonetheless, we note that for readers such as those suffering from Aphasia, who are also challenged by long sentences and passive voice, it would be important to employ syntactic operations, such as sentence splitting and passive-to-active voice transformations.

In the last two decades, multiple approaches to LS have been proposed. The earliest of them all is simplification by synonym substitution using word frequency as criterion to decide among multiple synonyms. This approach was first devised by Devlin and Tait (1998), where synonyms are extracted from WordNet (Fellbaum, 1998) and ranked according to their Kucera-Francis coefficient (Rudell, 1993). The complex word is then replaced with its synonym that has the highest Kucera-Francis coefficient value.

Other approaches to LS combine both lexical and syntactic information. Paraphrasing strategies such as the ones by Kauchak and Barzilay (2006) aim to replace complex phrasal constructions by simpler alternatives. Some approaches focus on a specific knowledge domain, or on a specific class of words. Rello et al. (2013c), for example, aim to simplify numerical expressions, while Kandula et al. (2010) and Elhadad and Sutaria (2007) focus on simplifying complex medical expressions.

Even though LS approaches differ from one another in various ways, most of them use a very similar sequence of steps to simplify sentences. In order to facilitate the understanding of the procedures involved in LS by different approaches in the literature, we use Shardlow's (2014b) definition of the task as the pipeline of steps illustrated in Figure 1. The steps are the following:

1. **Complex Word Identification**: Task of deciding which words of a given sentence may not be understood by a given target audience and hence must be simplified.

2. **Substitution Generation**: Task of finding words or expressions that could replace the target complex word.

3. **Substitution Selection**: Task of deciding which of the generated candidate substitutions can replace the complex word without compromising the sentence's grammaticality or meaning in a given context.

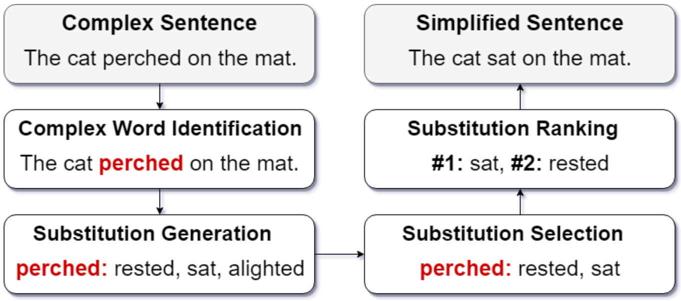

Figure 1: Lexical Simplification pipeline

4. **Substitution Ranking**: Task of ranking the remaining candidate substitutions of a given complex word by their simplicity.

Existing approaches cover one or more of these steps. Work in some of these steps can be also framed within the scope of other NLP tasks, with wider application. Substitution Generation approaches, for example, can be employed in lexical substitution (McCarthy & Navigli, 2007; Mihalcea, Sinha, & McCarthy, 2010) or paraphrasing (Deléger & Zweigenbaum, 2009; Kajiwara, Matsumoto, & Yamamoto, 2013), and also incorporated in assistive technologies that aim to perform "memory jogging" (Devlin & Unthank, 2006; Azab, Hokamp, & Mihalcea, 2015). Substitution Selection approaches can help in the creation of grammar checkers (suggestions) and correctors (Lee & Seneff, 2006), while Substitution Ranking strategies created for the purposes of LS can also be used in any other tasks that involve ranking, such as information retrieval (Burges, Shaked, Renshaw, Lazier, Deeds, Hamilton, & Hullender, 2005), medical risk evaluation (Caruana, Baluja, & Mitchell, 1996) and summarisation (Cao, Wei, Dong, Li, & Zhou, 2015).

Two recent general surveys on Text Simplification also address LS to some extent (Siddharthan, 2014; Shardlow, 2014b). However, because they cover TS at all levels, they are not able to compare and discuss in detail the wide array of strategies proposed for LS. Additionally, a significant amount of work on LS has been done after 2014. We present a more detailed and up to date survey on the many strategies used to address each step of the LS pipeline.

First, in section 2 we introduce datasets and resources that have been used in the creation and evaluation of many of the lexical simplifiers featured in this survey. In sequence, we address the contributions made for each step of the pipeline:

- In section 3 we present the lexicon-based, threshold-based, machine learning-assisted and implicit strategies used for Complex Word Identification.

- In section 4 we discuss two types of Substitution Generation approaches: those which query linguistic databases, and those which find candidates automatically from another source.

- In section 5 we address Substitution Selection strategies that exploit explicit and implicit sense labelling, as well as filtering heuristics based on POS tags and semantic similarity.

- In section 6 we present various Substitution Ranking approaches based on word frequencies, hand-crafted simplicity metrics and machine learning methods.

For each of these sections, we provide a benchmark of existing approaches using publicly available datasets and standard metrics, as well as a critical analysis of the findings. For an overview on the performance of a complete LS pipeline, in section 7 we report a full pipeline evaluation that compares various simplifiers built from combining the approaches described in sections 3 through 6. Finally, in section 8 we provide a discussion on open problems and prospects for future work.

2. Datasets and Resources

Before we delve into the LS literature, we provide a list of datasets and resources that have been employed in the creation and evaluation of the lexical simplifiers featured in this survey. We hope that this section will shed light on the design decisions made by research in previous work, as well as help foster future work on LS.

Datasets of manually annotated LS cases are very useful since they can be used for both training and evaluation. These datasets contain instances composed of a sentence, a target complex word, and a set of suitable substitutions provided and ranked by humans with respect to their simplicity. There are currently seven datasets of this kind:

- **SemEval 2012**[1] (Specia, Jauhar, & Mihalcea, 2012): 2,010 instances for English. Contains simplicity rankings produced by non-native English speakers for the datasets of the Lexical Substitution Task of SemEval 2007 (McCarthy & Navigli, 2007). This is a classic dataset for LS that has been widely used in benchmarks, and it stands out for reliably capturing the concept of simplicity as perceived by non-native speakers of English. The only evident limitation of the SemEval 2012 dataset is the fact that, while the gold simplifications provided are in their lemmatized form, many of the target complex words are inflected to some other tense.

- **LSeval**[2] (De Belder & Moens, 2012): 430 instances for English. Contains simplicity rankings produced by 46 Amazon Mechanical "turkers"[3] and 9 Ph.D students for the datasets for the Lexical Substitution Task of SemEval 2007. The intensive annotation process used in the creation of LSeval assures the gold simplifications contained in it do in fact simplify their respective target complex words. But since LSeval uses the same base data as the SemEval 2012 dataset, the gold simplifications are not necessarily inflected to the same tense as the target complex word.

- **LexMTurk**[4] (Horn et al., 2014): 500 instances for English. Contains sentences from Wikipedia with target complex words and simpler substitutions suggested by 50 English speaking turkers each. Because each instance in LexMTurk was annotated by 50 turkers, this dataset offers a very high coverage of gold simplifications, the great

[1] `https://www.cs.york.ac.uk/semeval-2012/task1`

[2] `http://people.cs.kuleuven.be/~jan.debelder/lseval.zip`

[3] `http://www.mturk.com`

[4] `http://www.cs.pomona.edu/~dkauchak/simplification/lex.mturk.14`

majority of which are correctly inflected to the same tense as that of the target word. Its only limitation is the fact that a small portion of the candidates have orthographic errors.

- **BenchLS**[5] (Paetzold & Specia, 2016a): 929 instances for English. Consists of a compilation of the LSeval and LexMTurk datasets automatically corrected for spelling and inflection errors. Since it combines two other datasets, BenchLS offers the largest range of distinct target complex words amongst the LS datasets for English. Nonetheless, it still contains a few gold simplifications with orthographic errors.

- **NNSeval**[6] (Paetzold & Specia, 2016d): 239 instances for English. Consists of a filtered version of BenchLS in which are discarded i) any instances of which the target word was not deemed complex by a non-native English speaker, and ii) any candidates that were deemed complex by a non-native English speaker. NNSeval captures the needs of non-native English speakers more accurately than the other datasets. Because of the filtering methods used, both its array of target complex words and coverage of gold simplifications are narrower than those of other datasets.

- **SNOW E4**[7] (Kajiwara & Yamamoto, 2015): 2,500 instances for Japanese. Candidates were suggested and ranked by annotators from a crowd sourcing service. To our knowledge, this is the first dataset created for Japanese LS, and it contains a larger array of contexts than any dataset for English.

- **BCCWJ Dataset**[8] (Kodaira, Kajiwara, & Komachi, 2016): 2,010 instances for Japanese. Candidates were suggested and ranked by turkers and Computer Science students. This dataset was created with the intention of addressing the limitations of SNOW E4. Unlike SNOW E4 the BCCWJ Dataset features a wide variety of contexts taken from sources that go beyond news articles, and was annotated in a way that allows for ties in simplicity rankings.

Corpora of parallel complex-to-simple documents and/or sentences have also been very frequently used in contributions. We are aware of six resources of this kind:

- **Wikipedia-Simple Wikipedia**[9] (Kauchak, 2013): 167,689 aligned sentences from 60,000 aligned articles in English.

- **PWKP**[10] (Zhu, Bernhard, & Gurevych, 2010): 108,016 aligned sentences from 65,133 aligned articles in Wikipedia and Simple Wikipedia.

- **SS Corpus**[11] (Kajiwara & Komachi, 2016): 492,993 aligned sentences extracted also from Wikipedia and Simple Wikipedia.

[5] `http://ghpaetzold.github.io/data/BenchLS.zip`
[6] `http://ghpaetzold.github.io/data/NNSeval.zip`
[7] `http://www.jnlp.org/SNOW`
[8] `https://github.com/KodairaTomonori/EvaluationDataset`
[9] `http://www.cs.pomona.edu/~dkauchak/simplification/data.v2`
[10] `https://www.ukp.tu-darmstadt.de/data/sentence-simplification`
[11] `https://github.com/tmu-nlp/sscorpus`

- **Newsela**[12] (Newsela, 2016): A continuously growing database of English and Spanish news articles simplified to various reading levels with the goal of making them more accessible to children and teenagers of varying age bands. From this corpus, Paetzold and Specia (2017) were able to extract 550,644 sentence alignments.
- **Simplext**[13] (Saggion, Štajner, Bott, Mille, Rello, & Drndarevic, 2015): 200 news texts in Spanish with manually simplified versions.
- **SIMPITIKI**[14]: 1,166 simplifications for Italian automatically extracted from Wikipedia edits.
- **PaCCSSIT**[15] (Brunato, Cimino, DellOrletta, & Venturi, 2016): 63,000 aligned sentences in Italian automatically produced from raw corpora.

There are also datasets created specifically for the task of binary CWI:

- **The CW Corpus**[16] (Shardlow, 2013b): 731 instances composed of a Wikipedia sentence and a word deemed complex by an editor. This was the first dataset of this kind to be made available, and was built through automatic methods based on Wikipedia edits.
- **SemEval 2016**[17] (Paetzold & Specia, 2016a): 90,458 words in context annotated as either simple or complex by non-native English speakers. 2,237 instances contain 20 annotations each, and the remaining 88,221 contain only one. This dataset was built with the goal of mimicking a realistic CWI scenario, where one must model individual user needs based on the needs of a group as a whole. The main limitation of this dataset lies in the fact that, because of the wide diversity of backgrounds across annotators, it is difficult to outline the needs of sub-groups, for example, native speakers of a specific language, age band or proficiency level.

For SG strategies that query linguistic databases, we can mention various databases that can be used:

- **WordNet**[18] (Fellbaum, 1998): One of the most widely used linguistic databases in Text Simplification literature. It contains 117,659 synsets for English.
- **Global WordNet**[19] A platform for WordNet versions of various languages. Their catalogue is currently composed of 78 distinct WordNets, including some that serve dozens of languages, such as the Open Multilingual WordNet[20].

[12] https://newsela.com/data
[13] http://www.simplext.es
[14] https://github.com/dhfbk/simpitiki
[15] http://www.italianlp.it/software-data/text-simplification
[16] http://tinyurl.com/cwcorpus
[17] http://alt.qcri.org/semeval2016/task11
[18] https://wordnet.princeton.edu
[19] http://globalwordnet.org
[20] http://compling.hss.ntu.edu.sg/omw

- **BabelNet**[21] (Navigli & Ponzetto, 2010): One of the largest multi-lingual semantic networks available. It contains 13,801,844 synsets for 271 languages.
- **Merriam**[22]: A dictionary and thesaurus for over 470,000 words in the English language.
- **PPDB**[23] (Pavlick, Rastogi, Ganitkevitch, Van Durme, & Callison-Burch, 2015): A database containing over 100 million automatically extracted paraphrases for various languages.
- **SimplePPDB**[24] (Pavlick & Callison-Burch, 2016): A subset of the PPDB database containing around 4.5 million complex-to-simple paraphrases for English.

We mention also other assorted resources that have been employed in the creation of successful simplifiers:

- **Simple Wikipedia**[25] (Kauchak, 2013): A text corpus composed of 60,000 Simple Wikipedia articles. This resource is one of the most commonly used in the creation and evaluation of LS systems.
- **SUBTLEX**[26] (Brysbaert & New, 2009): A text corpus extracted from 8,388 assorted movie subtitles. The experiments of Shardlow (2013a) reveal that word frequencies from this corpus correlate with human judgements on simplicity than many other more widely used corpora, such as Wikipedia.
- **SubIMDB**[27] (Paetzold & Specia, 2016b): A text corpus extracted from 38,102 subtitles of movies and series for family and children. The experiments of Paetzold and Specia (2016b) show that frequencies from this corpus correlate with human judgements on simplicity more strongly than frequencies from much larger corpora.
- **Bootstrapped MRC**[28] (Paetzold & Specia, 2016a): An automatically completed version of the MRC Psycholinguistic Database (Coltheart, 1981), containing psycholinguistic features for 85,942 words. Paetzold and Specia (2016a) show that their features can help in the creation of more reliable lexical simplifiers.

Finally, we list toolkits and frameworks for the creation of lexical simplifiers:

- **Simplex**[29] (Jauhar & Specia, 2012): A toolkit for the training of supervised rankers that allows one to replicate the best performing system of the English Lexical Simplification task of SemEval 2012.

[21] `http://babelnet.org`

[22] `https://www.merriam-webster.com`

[23] `http://paraphrase.org`

[24] `http://www.seas.upenn.edu/~nlp/resources/simple-ppdb.tgz`

[25] `http://www.cs.pomona.edu/~dkauchak/simplification/data.v2`

[26] `http://subtlexus.lexique.org`

[27] `http://ghpaetzold.github.io/subimdb`

[28] `http://ghpaetzold.github.io/data/BootstrappedMRC.zip`

[29] `https://github.com/sjauhar/simplex`

- **Light-LS**[30] (Glavaš & Štajner, 2015): A toolkit for the training of unsupervised lexical simplifiers that can achieve performance comparable to that of more sophisticated supervised approaches.
- **LEXenstein**[31] (Paetzold & Specia, 2015): A framework for the creation and evaluation of complete lexical simplifiers. This toolkit offers a wide range of strategies for Complex Word Identification, Substitution Generation, Selection and Ranking from contributions in the literature, including all those benchmarked in this paper.

In what follows, we present our literature survey and benchmark on each step of the typical LS pipeline.

3. Complex Word Identification

In the Complex Word Identification (CWI) step the goal is to select the words in a given sentence which should be simplified. Shardlow (2014b) illustrates an interesting example of this task: in the sentence "*The cat perched on the window*", the word "*perched*" is a clear candidate for simplification, since one could argue that this is not a very common word, especially when compared to some of its synonyms, such as "*sat*" and "*rested*".

The following sections discuss the advantages and limitations of five categories of strategies for CWI:

- Simplify everything;
- Threshold-based;
- Lexicon-based;
- Implicit Complex Word Identification; and
- Machine learning-assisted.

3.1 Simplify Everything

Early LS approaches (Devlin & Tait, 1998) did not perform Complex Word Identification. Instead, they assumed that all words in a sentence could be simplified. Although this strategy would be perfectly sensible in a scenario where the simplifier used is 100% accurate, it is not effective in most cases. This approach has lost popularity since: as demonstrated by Shardlow (2014a), a realistic simplifier without a CWI module might replace words which are already easy to understand by the target audience in question, and hence make the text even more difficult or less meaningful. In Devlin and Tait's (1998) simplification approach, all words and phrases of a sentence are targets for simplification. They report that 16.60% of the simplified sentences had their grammatical structures compromised, while 44.50% of them had their meaning modified. Paetzold (2013) also noted that, using this approach, many modifications made to the sentences were indeed performed over portions of text which did not need simplification, leading to the multiple cases of ungrammatical and/or incoherent substitutions.

[30] `https://github.com/gglavas/light-ls`

[31] `https://github.com/ghpaetzold/LEXenstein`

3.2 Threshold-Based

Threshold-based approaches aim at searching for a threshold t over a given metric of simplicity M for a word w such that if $M(w) < t$ the word w can be more confidently categorised as a complex (or simple) word.

Keskisärkkä (2012) describes a study on the effect of using a word's length as a metric for CWI. Their LS approach simplifies sentences by replacing complex words with their most frequent synonym. Results show that increasing the word length decision threshold effectively decreases the number of errors performed by their approach, i.e. simplifying only words with more than 7 letters in length proved to produce sentences with higher readability scores than simplifying all words in the sentence.

Word frequency has been a much more popular choice of metric for threshold-based approaches. Bott et al. (2012) describe an LS system for the Spanish language that discards substitutions for words in a sentence which appear in more than 1% of sentences in a large corpus. Their method outperforms a baseline approach that does not discard any candidates. Considerable improvements in both meaning preservation and simplicity have been reported. Leroy, Endicott, Kauchak, Mouradi, and Just (2013) describe a similar approach that simplifies texts in the medical domain. They choose to simplify only words with an occurrence count of less than 15.377.914 times, which is the occurrence count of the 5000th most frequent word in the Google 1T corpus (Michel, Shen, Aiden, Veres, Gray, Pickett, Hoiberg, Clancy, Norvig, Orwant, Pinker, Nowak, & Aiden, 2011). Human subjects reported a discernible reduction in the perceived reading difficulty in comparison to unsimplified sentences. Shardlow (2013a) learns the threshold that best separates complex and simple words from the CW corpus (Shardlow, 2013b). They use word frequencies from the SUBTLEX corpus (Brysbaert & New, 2009) as a metric. Their threshold-based approach offers a noticeable improvement in performance over the "simplify everything" baseline approach. The approach of Wróbel (2016), which achieved the highest F-scores in the Complex Word Identification task of SemEval 2016, also learns a threshold over Simple Wikipedia (Kauchak, 2013) frequencies through search. We present the SemEval 2016 task in more detail in section 3.5.

Threshold-based approaches are intuitive and easy to implement. However, the evaluation of a set of documents in the Spanish language by Bott et al. (2012) suggests that it is difficult to elaborate a single simplicity feature or metric that is capable of discerning between complex and simple words. Their analysis was performed over 40 manually simplified documents. When evaluating simpler synonyms of complex words, they found that less than 70% of the simpler words were actually shorter in length than their complex equivalents. This means that simplifying only words which are more than t characters in length could lead to either ignoring certain complex words, or replacing simple words.

An analysis conducted by Shardlow (2014a) provides further evidence that threshold-based approaches can be less than ideal in practice. The goal of their work is to find out the most frequent types of errors made by a baseline LS approach. Their LS approach is very similar to the one of Devlin and Tait (1998), and identifies simplifiable words by computing their Kucera-Francis coefficient: if it is lower than 5, the word is deemed complex and should be simplified. In order to find out how many mistakes are made by the system, a set of 115 sentences were simplified, and a manual evaluation performed over the output produced after each step of the LS pipeline. The findings show that a mistake was made in more than 65% of the complex word identification operations performed by the system, representing a total of 119 mistakes out of 183 operations. 99 of the 119 mistakes were caused by a simple word being identified as complex, which resulted in many simple words being unnecessarily (and incorrectly) replaced.

3.3 Lexicon-Based

To tackle the limitations of threshold-based approaches, some domain-specific LS systems have used a different approach for CWI. Their strategy consists of using a lexicon of complex words to identify simplifiable candidates: if a given word w is part of the lexicon of complex words L, then it should be simplified, or vice-versa.

Deléger and Zweigenbaum (2009) present a method for building a lexicon of complex paraphrases in the medical domain. The method consists in automatically identifying aligned paraphrases in technical medical articles from the Web which have a corresponding summary written in lay terms. The approach uses a topic segmentation tool (Hearst, 1994) to identify pairs of segments likely to have aligned paraphrases, and then selects the pairs of segments which have a word co-occurrence vector cosine similarity higher than 0.33. The lexicon is hence composed by all segments extracted from documents written in technical terms.

When available, manually constructed lexicons can also be useful. The paraphrase extraction approach of Elhadad and Sutaria (2007) uses the UMLS (Unified Medical Language System) lexicon (Bodenreider, 2004), a database of technical medical terms, to identify complex words and expressions. A different technique is used by Elhadad (2006): they consider simple all expressions from UMLS that can also be found in the Brown corpus. This filtering method explores the intuition that, if an expression that is commonly used in medical documents is found in a corpus of an unrelated domain, it is likely that such an expression is not complex. Through an experiment where college-level lay readers were asked whether or not a given medical expression was familiar, they found that abbreviations are extremely likely to be unfamiliar to readers. Based on these results, they also employed the heuristic that all abbreviations found in a sentence are considered complex to readers.

Kajiwara et al. (2013) present an automatic method for the extraction of paraphrases of complex words for children. The lexicon of simple words used is the *Basic Vocabulary to Learn*, a manually collected set of 5.404 words of the Japanese language that can help children communicate more efficiently.

Lexicon-based strategies can be very effective in practice. The FACILITA system (Watanabe et al., 2009) is a good example of that. FACILITA is a tool designed to simplify web pages, and it is part of the PorSimples project (Aluisio & Gasperin, 2010), a simplification framework for low literacy readers of the Portuguese language. The lexicon used to identify simple words in the framework is composed by words extracted from books for children, a list of very frequent words in news documents and a set of words manually judged “concrete” of Janczura, Castilho, Rocha, van Erven, and Huang (2007). The FACILITA tool proved to effectively assist low literacy readers in comprehending texts of complex nature, such as news articles.

Lexicon-based approaches however also have limitations, since manually creating large lexicons of complex or simple words can be prohibitively expensive. In addition, deciding which words should be present in the lexicon is a challenge since it is very unlikely that different individuals even within the same target audience will consider complex the same words.

3.4 Implicit Complex Word Identification

More recent approaches perform CWI not as an initial step in the process of simplification, but rather implicitly during other steps of the pipeline. They consider all words in a sentence to be targets for simplification, but during the simplification process they discard substitutions ($w_i \rightarrow w_j$) that, when applied, replace a word w_i with a more complex alternative w_j.

Biran, Brody, and Elhadad (2011) and Bott et al. (2012) define word simplicity metrics, and then discard candidate substitutions which are estimated to be more complex than the target word being simplified. Although both metrics exploit word frequencies and word length, the metric used by Bott et al. (2012) is more elaborate and was devised to account for the simplification needs of those suffering from Dyslexia, who find it difficult to comprehend words that are long and unfamiliar. A similar approach is used by the LS system of Glavaš and Štajner (2015), which only replaces a target word if it has a lower frequency than that of the candidate substitution selected.

A different strategy is employed by Horn et al. (2014), which simply adds a substitution ($w_i \rightarrow w_i$) to the set of candidate substitutions of complex word w_i. In other words, the complex word becomes a candidate substitution for itself. If their system decides that the target word is the simplest alternative to replace itself, then the target word is not simplified.

Implicit CWI is also employed by lexical simplifiers that exploit machine translation models. This category of simplifiers is trained over parallel corpora aligning complex-to-simple sentences, which may contain useful lexical simplifications. The earliest examples of such approaches employ typical phrase-based and tree-based translation models (Specia, 2010; Zhu et al., 2010), achieving promising results for LS.

Expanding on this concept, Wubben, van den Bosch, and Krahmer (2012) complement a typical phrase-based translation model by adding a re-ranking step that uses the Levenshtein distance as a metric. The work of Xu, Napoles, Pavlick, Chen, and Callison-Burch (2016) goes a step further and adapts a typical statistical translation model by crafting two new objective functions that better suit Text Simplification, and incorporating pre-produced paraphrase databases during training to compensate for the lack of large complex-to-simple parallel corpora. They show that adding these adaptations yields noticeable increases in performance, outperforming the approach of Wubben et al. (2012).

These implicit approaches avoid the problem of trying to decide which words are inherently complex enough for simplification, and instead focus on the question of whether or not simpler substitutions can be found for a given word. Although it is difficult to compare their performance to that of more traditional CWI methods, these approaches can be a very suitable alternative in cases where one can assume with reasonable confidence that the training data used captures the needs of the audience addressed.

3.5 Machine Learning-Assisted

Provided that there is data available, another possible approach is to use machine learning techniques to learn a model of word complexity. Suppose there is a number of words in context labelled as either complex or simple. In this scenario, it would be possible to extract features from these words and then train a binary classifier that attempts to discern between complex and simple words. If the labels in the data available are complexity quantifiers, one could also employ regression techniques to learn models on *how* complex a word is.

Shardlow (2013a) provides a comparison between a Support Vector Machine (SVM) classifier (Chang & Lin, 2011), a threshold-based strategy and the "simplify everything" approach. The threshold-based strategy uses the frequency of the word in the SUBTLEX corpus (Brysbaert & New, 2009) as a metric. SUBTLEX is composed of over six million sentences extracted from subtitles of assorted movies. The final threshold that separates complex from simple words was determined by 5-fold cross validation over possible frequency values. The features used by the SVM classifier are word frequency in the SUBTLEX corpus, number of films in SUBTLEX in which the word appears, length, and number of syllables, senses and synonyms.

The SVM model was trained and tested with the CW corpus (Shardlow, 2013a), which contains Wikipedia sentences with a single target complex word along with a simpler alternative. The results show that the SVM approach slightly outperforms the other strategies, but leads to the lowest recall. The "simplify everything" approach, however, obtains the highest score among the systems, which contradicts the discussion drawn in section 3.1. These results are due to the fact that the evaluation setup used is not that of a typical CWI task, in which a strategy must handle a large discrepancy between the amount of complex and simple words in a sentence. The setup they use assumes a perfect balance between complex and simple words, which makes the "simplify everything" approach more promising.

The work of Shardlow (2013a) has inspired further work, such as the Complex Word Identification task of SemEval 2016 (Paetzold & Specia, 2016a). Participants of the SemEval 2016 CWI task were asked to create systems that identify words that challenge non-native English speakers in a set of sentences. The training and test sets used are composed of $2,237$ and $88,221$ instances, respectively, where each instance contains a target word in a sentence. While the training set contains 20 binary complexity labels produced by distinct annotators for each instance, the test set contains a single annotation. 400 annotators participated in the data collection and their annotations were distributed between the training and test sets.

Among the 42 systems submitted by the 21 teams that participated, the majority use machine learning. Some use standard SVMs, Decision Trees and neural networks trained over lexical, morphological, psycholinguistic and semantic features (Kuru, 2016; Sp, Kumar, & K P, 2016; Bingel, Schluter, & Martínez Alonso, 2016). Others use more elaborate ensembles that combine various machine learning techniques and feature sets (Choubey & Pateria, 2016; Mukherjee, Patra, Das, & Bandyopadhyay, 2016; Nat, 2016). The highest performing system in the CWI task is the Performance-Oriented Soft Voting ensemble method of Paetzold and Specia (2016c), which combines different types of lexicon-based, threshold-based and machine learning approaches. It is worth pointing out, however, that this strategy was conceived by the task organizers themselves.

Some of the other machine learning approaches that performed well in the shared task are much more minimalistic. The UWB (Konkol, 2016) systems, which were among the five top performing, use only one feature: the number of documents in Wikipedia in which a word appears. The LTG (Malmasi, Dras, & Zampieri, 2016) systems, which also achieved some of the highest scores in the task, are another example. They train their model using only word length and n-gram probabilities from a language model.

3.6 Benchmarking

In what follows, we compare 17 CWI strategies that represent all of the approaches described above. We re-implemented them based on their description in the original papers so that they could be tested on the same data. Given the difference is the datasets, the results obtained in our performance comparison may differ from those reported in the original papers. The strategies tested are:

Lexicon-based (LB): We consider two lexicons: Ogden's Basic English (Ogden, 1968) and the SubIMDB corpus (Paetzold & Specia, 2016b), which is composed of subtitles of movies and series for families and children.

Threshold-based (TB): As metrics, we consider the words' length, frequency in Simple Wikipedia, and number of senses, synonyms, hypernyms and hyponyms in WordNet (Fellbaum, 1998). To find t, an exhaustive search was performed on the training set over 10,000 equally distant values in the interval between the minimum and maximum value of each metric. The performance metric maximised was the G-score (see definition in section 3.6.1). It is important to mention that, although there is no need to test 10,000 equidistant values for metrics of integer value and limited range, such as length and number of senses, our optimisation method does not compromise the performance of the threshold-based identifiers for these metrics.

Machine learning-assisted (ML): We include the systems that achieved the three highest G-scores in the CWI task of SemEval 2016, which are the soft voting method of Paetzold and Specia (2016c) (SV000gg-Soft), the random forests of Ronzano, Abura'ed, Espinosa Anke, and Saggion (2016) (TALN-WEI), and the maximum entropy classifiers of Konkol (2016) (UWB-All). We also include the systems with the top three F-scores, which are the threshold-based approach of Wróbel (2016) (PLUJAGH-F), the decision trees (LTG-1) and decision tree ensembles (LTG-2) of Malmasi et al. (2016). To complement our results, we also replicate the SVM approach of Shardlow (2013a) (which we name Shardlow).

Baselines (BA): We provide the simplify "Nothing" and simplify "Everything" baselines.

3.6.1 Datasets and Metrics

We use the dataset from the Complex Word Identification task of SemEval 2016.

For evaluation, we use Precision, Recall, Accuracy, F-score and the G-score, which is a modified version of the F-score that measures the harmonic mean between Accuracy and

Recall. Paetzold and Specia (2016a) argue that Accuracy is a better metric for CWI than Precision, since an effective simplifier must be able to do two things:

1. Avoid making unnecessary replacements of simple words as well as not making necessary replacements of complex words.
2. Make the sentence as simple as possible.

In order to help the simplifier achieve both these goals, an effective identifier should make as few mistakes as possible (1) while finding as many complex words as possible in a sentence (2). Given that Accuracy represents the amount of mistakes made by the identifier, and Recall the amount of complex words correctly captured, balancing them leads to a reliable way of assessing an system's performance. Paetzold and Specia (2016b) also show that the G-score is better at evaluating the performance of a CWI system in practice when it is used as part of a complete LS pipeline.

3.6.2 Results

The last column of Table 1 shows the results of a 10-fold bootstrap resampling statistical significance test between the CWI system in question and the highest performing approach (*SV000gg-Soft*). The • symbol represents a statistically significant difference ($p \leq 0.05$), while the ◦ symbol represents that there is no statistically significant difference (i.e. $p > 0.05$). The - (dash) symbol indicates that the system in question is the one being compared against.

The scores in Table 1 clearly show that using automatic CWI approaches to predict word complexity is a much more reliable alternative than simplifying all words in a sentence. And although resource-heavy ensembles such as *SV000gg-Soft* and *TALN-WEI* yield the highest G-scores, the much simpler threshold-based approach of *PLUJAGH-F* achieves a superior F-score and a competitive G-score, highlighting the potential cost-effectiveness of this type of strategy. For details on the performance of all systems submitted to the CWI task of SemEval 2016, we refer the reader to the work of Paetzold and Specia (2016a).

3.7 Discussion

Based on the results presented, it is quite clear that supervised approaches that use highly tuned modern machine learning techniques tend to be more effective than threshold and lexicon-based alternatives when there are labeled datasets available. Ensemble methods, such as those of Paetzold and Specia (2016c) (*SV000gg-Soft*) and Ronzano et al. (2016) (*TALN-WEI*) are particularly effective for this task. It is important to point out, however, that *SV000gg-Soft* was conceived by the authors of the dataset used in our evaluation, who consequently have more familiarity with it.

In contrast, the work of Malmasi et al. (2016) and Konkol (2016) reveals that, although resource-heavy models tend to achieve the highest performance scores in CWI, it is possible to create effective systems using resource-light approaches that do not require for heavy tuning. We do believe, however, that exploring the use of other more elaborate machine learning techniques, such as deep neural network architectures, could yield even more reliable solutions in cases where there are large amounts of data available.

	Approach	A	P	R	F	G	p
LB	Ogden's	0.248	0.056	0.947	0.105	0.393	•
LB	SubIMDB	0.913	0.217	0.332	0.262	0.487	•
TB	Length	0.332	0.057	0.852	0.107	0.478	•
TB	Hyponyms	0.384	0.065	0.906	0.121	0.539	•
TB	Synonyms	0.436	0.067	0.853	0.124	0.577	•
TB	Senses	0.436	0.068	0.861	0.125	0.579	•
TB	PLUJAGH-F	0.922	0.289	0.453	0.353	0.608	•
TB	Hypernyms	0.572	0.076	0.728	0.137	0.641	•
TB	Frequency	0.513	0.081	0.902	0.148	0.654	•
ML	Shardlow	0.636	0.045	0.332	0.079	0.437	•
ML	LTG-System1	0.933	0.300	0.321	0.310	0.478	•
ML	LTG-System2	0.889	0.220	0.541	0.312	0.672	•
ML	UWB-All	0.803	0.157	0.734	0.258	0.767	•
ML	TALN-WEI	0.812	0.164	0.736	0.268	0.772	•
ML	SV000gg-Soft	0.779	0.147	0.769	0.246	0.774	-
BA	Nothing	0.953	0.000	0.000	0.000	0.000	•
BA	Everything	0.047	0.047	1.000	0.089	0.089	•

Table 1: Complex Word Identification benchmarking results

In scenarios where there is no data available for training, one can employ lexicon-based approaches. Although they are not well suited to handle scenarios in which one must adapt to the individual simplification needs of multiple subjects, they can be a suitable alternative in cases where there are certain assurances about the vocabulary being simplified. In the medical domain, for example, one can confidently expect that a lay reader will not be familiar with context-specific technical terms and expressions.

4. Substitution Generation

Substitution Generation (SG) refers to the process of producing candidate substitutions for complex words. In the typical LS pipeline, an ideal SG strategy will be able find all words that can replace a given target complex word in all contexts in which it may appear. In the case of ambiguous words, the strategy would have to find candidate substitutions for all their possible senses. Based on this set, the Substitution Selection step (section 5) should be able to select the one that fits the context of a sentence being simplified. This strategy aims to maximise the recall of candidate substitutions for complex words, and consequently should lead to better simplifications.

The biggest challenge in SG is to avoid producing too many spurious candidates that can confuse the models employed in subsequent steps. Existing SG approaches fall under one of two categories:

- Linguistic database querying; and

- Automatic generation.

4.1 Linguistic Database Querying

When searching for candidate substitutions for complex words, using linguistic databases manually constructed by professionals is intuitively a very sensible approach: synonyms and other related words as given by humans will certainly lead to substitutions that can replace complex words. However, broad coverage versions of such resources are not available for most languages and constructing them is expensive and time consuming.

Some LS approaches address the task of creating a database of semantically related words in order to use this approach. Ong, Damay, Lojico, Lu, and Tarantan (2007) report the creation of a dictionary containing descriptions of complex medical terms. Similarly, Kandula et al. (2010) create a small database composed of paraphrases describing 150 complex medical expressions in lay terms. Both of these resources have proved useful in the creation of effective simplifiers for the medical domain.

For general-domain LS, most work use instead already existing and broader resources to search for candidate substitutions. WordNet is the most frequently used database for LS. Early LS approaches (Devlin & Tait, 1998; Carroll et al., 1998; Carroll, Minnen, Pearce, Canning, Devlin, & Tait, 1999) used synonyms extracted from WordNet as candidate substitutions of complex words. Some (Sinha, 2012; Nunes, Kawase, Siehndel, Casanova, & Dietze, 2013) used not only the synonyms listed in WordNet, but also hypernyms and hyponyms. An empirical study conducted by Drndarević and Saggion (2012) provides evidence that related words other than just synonyms, such as hypernyms, hyponyms and meronyms, can also provide strong simpler candidates for complex words. Although WordNet has proved useful for LS, Shardlow (2014a) shows that using only WordNet synonyms can limit the potential of LS, since WordNet does not cover all complex words in the English vocabulary, nor does it contain all candidates which can replace a complex word. In their experiment they evaluated the most frequent types of errors made by an LS approach similar to the one of Devlin and Tait (1998). The results showed that over 42% of the 164 errors made by their approach were caused by WordNet not having suitable simpler substitutions for complex words.

Combining multiple linguistic databases has been shown to be a more promising approach. Leroy et al. (2013) aim to simplify texts in the medical domain. As candidate substitutions, they use the relations provided by WordNet along with the ones provided by UMLS (Unified Medical Language System) and Wiktionary (Wikimedia, 2017). UMLS provides a large ontology containing semantic relations between pairs of medical terms. This resource was also used in the LS approach of Chen, Huang, Chen, and Tan (2012), which replaces complex medical expressions with simpler equivalent lay terms in order to improve the performance of statistical machine translation systems. Elhadad (2006) used the "define:" function of Google's search engine to gather multiple dictionary definitions of medical expressions.

Associating the querying of linguistic databases with an automatic approach showed promising results of De Belder and Moens (2010). They intersected the synonyms extracted from WordNet with a set of related words learned through a latent-variable language model. They found that the resulting reduced set had fewer spurious substitution candidates.

Since WordNet does not offer enough coverage especially for languages other than English, other linguistic databases have been used to find simpler candidate substitutes for complex words. The strategy of Bott et al. (2012) uses the Spanish OpenThesaurus (OpenThesaurus, 2017) to find synonyms for complex words in Spanish. The same resource is used by Baeza-Yates, Rello, and Dembowski (2015) with a more sophisticated SG strategy. They first create a list of words for each meaning registered in the Spanish OpenThesaurus. Each list is composed of all words associated with such a meaning. They then enrich such lists by querying the Google 1T corpus for the words' frequencies, as well as for all 5-grams in which the word in question is in the middle (preceded and succeeded by two tokens). They found that this generation strategy led to more suitable replacements.

SynLex (Kann, 2017) is a thesaurus for the Swedish language used by Keskisärkkä (2012) to find synonyms for complex words. The PorSimples project (Aluisio & Gasperin, 2010), which provides an LS approach for Brazilian Portuguese, uses sets of related words provided by the databases TeP 2.0 (Maziero, Pardo, Di Felippo, & Dias-da Silva, 2008) and PAPEL (Porto, 2017), which were created aiming at maximising the coverage of synonyms and antonyms available.

4.2 Automatic Substitution Generation

As discussed in the previous section, even though large linguistic databases can be of great help in gathering candidate substitutes for complex words, they are not always available or sufficient. automatic SG approaches aim to extract candidates from other, less expensive resources.

Kajiwara et al. (2013) present a straightforward strategy of this kind. It takes advantage of dictionaries that do not include synonymy relations, but still provide word descriptions. The approach first queries a dictionary for a complex word's definition and uses a tagger to produce the Part-of-Speech (POS) tag of each word in it. It then extracts as candidate substitutions all words with the same POS tag as the target word. This strategy has been shown to be effective for the Japanese language.

Replacing complex words with equivalent paraphrases is also a viable strategy. Automatic paraphrase extraction is a task that has been addressed in many ways. It is not our goal to survey paraphrase extraction methods, but rather to discuss how these methods have been employed in the task of LS. Elhadad and Sutaria (2007) extract simpler paraphrases for medical expressions using articles related to medicine aligned at document level. Their approach uses contingency tables describing the contextual differences between expressions, along with statistical modelling to determine whether or not an expression found in a technical medical article is equivalent to a simpler expression found in an aligned document written in lay terms. They compare their set of paraphrases with a gold-standard set produced by professionals in the area, achieving an F1 score of over 66%.

Deléger and Zweigenbaum (2009) also extract paraphrases from articles aligned at document level, but go a step further by performing topic segmentation to produce alignments at sentence level. Once aligned sentences have been produced, heuristics are applied in order to find nominalisations of complex expressions and paraphrases of neo-classical compounds (such as "*gastritis*").

Extracting candidate substitutions from the English Simple Wikipedia is a popular strategy. This resource contains a subset of the articles from the original English Wikipedia edited by volunteers such that more readers can understand them. Yatskar, Pang, Danescu-

Niculescu-Mizil, and Lee (2010) extract paraphrases of complex terms from Simple Wikipedia edits marked with the "simplification" label. Their method has proven to produce many useful paraphrases, such as "*stands for*" → "*is the same as*", and "*indigenous*" → "*native*". Biran et al. (2011) consider every pair of distinct words in the Wikipedia and Simple Wikipedia to be a possible simplification pair. They filter any pairs which are morphological variants of each other or that are not registered as either synonyms or hypernyms in WordNet. They manually evaluated the quality of 65 simplified sentences produced by their approach with respect to grammaticality, meaning preservation and simplicity. The results show that 77.91% of the simplified sentences were grammatically correct, while 62.79% retained the meaning of the original sentence, and 75.58% were simpler than the original. This is however a hybrid approach, since they also use WordNet to filter word pairs.

Feblowitz and Kauchak (2013) use a version of the tree transduction model of Cohn and Lapata (2009) adapted for the purposes of Text Simplification to extract lexical and syntactic simplifications from a corpus of parallel sentences taken from Wikipedia and Simple Wikipedia. Tree transduction models attempt to learn tree-to-tree rewritings from parallel data, and are often able to capture complex transformations such as word re-orderings, as well as lexical replacements. Their adapted model has been shown to capture many reliable paraphrases.

Pavlick and Nenkova (2015) introduced a simpler strategy for this task. Instead of parallel corpora, they exploit the PPDB: a large database of paraphrases extracted from bilingual parallel corpora. Their approach attempts to calculate the difference in "style" between the two sides of a paraphrase. For this purpose they employ a hand-crafted metric that measures the probability of a given phrase appearing in a set of documents that is characterised by the style being targeted. By using Wikipedia and Simple Wikipedia documents as indicators of complexity and simplicity, they are able to select paraphrases with stylistic gaps with 74% accuracy.

A similar strategy was introduced by Pavlick and Callison-Burch (2016). To identify complex-to-simple paraphrases from the PPDB they first ask annotators to classify 1,000 paraphrases into one of three classes: "bad", "simplifying" and "complicating". They then train a classifier over this data, employ it over all paraphrases in the PPDB and select all those that received the "simplifying" label. Their database, the Simple PPDB, contains billions of paraphrases and offers generation performance scores competitive to those of state-of-the-art simplifiers.

Rather than looking for paraphrases of any size, certain approaches focus on finding only single word equivalences in parallel corpora. Kauchak and Barzilay (2006), for example, use a series of syntactic and semantic filtering procedures to extract pairs of words with related meaning by comparing target texts in English produced by translation systems and equivalent reference translations produced by humans.

Horn et al. (2014) also extract complex-to-simple word correspondences from the Wikipedia-Simple Wikipedia corpus. They produce word alignments for this parallel corpus, and then discard any complex-to-simple pairs which have have different POS tags, or that have at least one word that is either a proper noun or is in a stop list. They then inflect all word pairs to all morphological variants to increase coverage. They report noticeable gains in performance over previous approaches.

In an effort to entirely avoid the need for resources such as dictionaries or parallel corpora, Glavaš and Štajner (2015) propose an unsupervised approach for the task. It uses word embedding models, which require only unannotated text corpora to be trained. In a word embeddings model, each word in the corpus vocabulary is represented by a unique vector containing a user-defined n number of real values that describe the word in a distributed semantic feature space. It has been shown that these values capture various interesting linguistic properties of words, such as gender and synonymy (Mikolov, Yih, & Zweig, 2013). Given a trained word embeddings model and a complex word, their generator extracts as candidate substitutions the 10 words whose embeddings vector has the highest cosine similarity with the vector of the complex word, except for their morphological variants. The results obtained are comparable to the ones of Horn et al. (2014). Paetzold and Specia (2016d) elaborate on this idea: instead of a typical embeddings model, they exploit a context-aware model trained over a large corpus where each word is annotated with their universal POS tag. Using this strategy, they are able to partly account for word ambiguity, leading to better performance.

Finally, it has been shown that combining embeddings with other resources, such as WordNet and parallel corpora, can lead to further performance improvements. The approach of Paetzold and Specia (2017) first extracts candidates from the parallel Newsela corpus (Newsela, 2016) (see section 2) , which contains news stories simplified to numerous reading levels. To that end, they employ the approach of Horn et al. (2014). They then perform lexicon retrofitting over the context-aware models of Paetzold and Specia (2016d) using the algorithm of Faruqui, Dodge, Jauhar, Dyer, Hovy, and Smith (2015). Retrofitting allows for pre-trained embeddings to be updated based on manually created linguistic relations with the goal of approximating words that share those relations. Some of the linguistic relations that can be used are synonymy, hypernymy and hyponymy, which are especially useful for Lexical Simplification. Using synonymy relations from WordNet, they retrofit their context-aware models and extract from them three complementary candidates for each complex word. Using this strategy, they are able to combine the inherently high precision of candidates from parallel corpora with the better recall provided by embedding models.

4.3 Benchmarking

We compare the performance of eight SG strategies that represent all varieties of generators previously described. The generators evaluated are:

Devlin (Devlin & Tait, 1998): Performs linguistic database querying. We extract synonyms of complex words from WordNet.

Biran (Biran et al., 2011): Performs automatic SG through parallel corpora. We attempt to reproduce their method by creating the cartesian product between all words in Wikipedia and all words in Simple Wikipedia. We use the parallel corpus of Kauchak (2013) to that purpose. We inflect words using the Morph Adorner module of LEXenstein (Paetzold & Specia, 2015; Burns, 2013), and filter synonyms using WordNet.

Yamamoto (Kajiwara et al., 2013): Performs automatic SG through dictionary definitions. We adapt their approach to English by using the Merriam Dictionary (Merriam-

Webster, 2017) to extract definitions of complex words, and tag them with the Stanford Tagger (Toutanvoa & Manning, 2000).

Horn (Horn et al., 2014): Performs automatic SG through sentence-aligned Wikipedia and Simple Wikipedia data. We replicate their approach using the same corpus and parameters as described by Horn et al. (2014). We tag sentences using the Stanford Tagger instead of the Berkeley Parser (Petrov & Klein, 2007) because of its higher overall performance on tagging.

Glavas (Glavaš & Štajner, 2015): Performs automatic SG with typical word embeddings. They use a 200-dimension embeddings model trained with `GloVe` (Pennington, Socher, & Manning, 2014) over a Wikipedia corpus. In order to better compare this approach with other embedding-based strategies, we train word embeddings with $1,300$-dimension vectors with the bag-of-words (CBOW) method from `word2vec`. The corpus used contains 7 billion words, combining the SubIMDB corpus (Paetzold & Specia, 2016b), UMBC webbase (UMBC, 2017), News Crawl (Callison-Burch, Koehn, Monz, & Zaidan, 2011), SUBTLEX (Brysbaert & New, 2009), Wikipedia and Simple Wikipedia (Kauchak, 2013). We select 10 candidates for each complex word in the dataset. For stemming, we use NLTK's Porter stemmer (Bird, Klein, & Loper, 2009).

SimplePPDB (Pavlick & Callison-Burch, 2016): Performs automatic SG with a filtered paraphrase database for simplification. We use the paraphrases provided by Pavlick and Callison-Burch (2016).

Paetzold-CA (Paetzold & Specia, 2016d): Performs automatic SG through context-aware word embeddings. It uses embeddings with $1,300$-dimension vectors with the bag-of-words (CBOW) method from `word2vec`, which are the same settings as in the Glavas generator. The corpus and number of candidates generated is the same as in the Glavas generator. The corpus was tagged with the Stanford Tagger.

Paetzold-NE (Paetzold & Specia, 2017): Performs Automatic Substitution Generation through sentence-aligned Newsela data and retrofitted context-aware word embeddings. We use the algorithms of Paetzold and Specia (2016e) to align the Newsela corpus (version 2016-01-29.1), which are the same settings described by Paetzold and Specia (2017). The word embeddings used were trained with $1,300$-dimension vectors and the bag-of-words (CBOW) architecture. The corpus and number of candidates generated is the same used by the Glavas generator. We tag the corpus with the Stanford Tagger.

We also include a baseline which combines the candidates produced by all other generators (All). Generators that were conceived by the authors of this paper (Paetzold-CA and Paetzold-NE) had already been implemented and were used as is. The remaining generators were re-implemented based on the original papers so that they could be tested on the same data. Because of variances in the data used and the non-deterministic nature of some methods, the performance achieved by our implementations might not be exactly the same as that of the original systems.

4.3.1 Datasets and Metrics

As our gold-standard, we use the LexMTurk dataset (Horn et al., 2014). It is composed of 500 instances, each containing a sentence, a target complex word, and 50 substitutions suggested by English speakers. The evaluation metrics are the ones from previous work (Paetzold & Specia, 2015, 2016d, 2016a):

- **Potential:** The proportion of instances for which at least one of the candidates generated is in the gold-standard.
- **Precision:** The proportion of generated substitutions that are in the gold-standard.
- **Recall:** The proportion of gold-standard substitutions that are among the generated substitutions.
- **F-score:** The harmonic average between Precision and Recall.

4.3.2 Results

Following the same convention in the benchmark of section 3, the last column of Table 2 shows the results of 10-fold bootstrap resampling statistical significance tests between the best generator and the others according to their F-score. The results in Table 2 reveal that combining parallel corpora, WordNet and retrofitted context-aware embeddings offers the highest Precision and F-scores. Combining all generators, however, allows for considerably higher Potential and Recall scores. It is also important to note also that, despite being entirely unsupervised, the *Glavas* approach performs very competitively with other more resource-heavy strategies.

Overall, however, F-scores seem to be rather low across all approaches. Inspecting the candidates produced we noticed that the low scores obtained are caused mostly by the fact that there are various ambiguous words in the dataset. Consider the target word "*released*" in the sentence "*Published by Tor Books, it was released on August 15, 1994 in hardcover, and in paperback on July 15, 1997*". The word "*released*" in this context refers to publishing, but it could also mean numerous other things in different contexts, such as someone who has been "*freed*" or "*discharged*". If we consider all synonyms in WordNet as candidates, for example, only around 20% of them would actually fit the context, even though all synonyms in WordNet could potentially replace "*released*" in the right context. Ambiguity is commonly addressed during the subsequent step of Substitution Selection, which we address in section 5.

4.4 Discussion

Considering the nature of the strategies discussed and the results of our benchmark, it is possible to conclude that generators tend to benefit from the combination of different resources available for a given language, such as demonstrated by the approach of Paetzold and Specia (2017), which exploits parallel corpora, WordNet and context-aware word embeddings.

For under-resourced languages it is still possible to create competitive candidate generators. The results obtained by the generator of Glavaš and Štajner (2015) clearly show

Generator	Potential	Precision	Recall	F-score	p
Yamamoto	0.510	0.056	0.079	0.065	•
SimplePPDB	0.492	0.045	0.154	0.070	•
Devlin	0.642	0.164	0.092	0.118	•
Biran	0.630	0.153	0.098	0.119	•
Glavas	0.766	0.151	0.122	0.135	•
Horn	0.832	0.153	0.134	0.143	•
Paetzold-CA	0.802	0.177	0.140	0.156	•
Paetzold-NE	0.876	**0.310**	0.142	**0.195**	-
All	**0.996**	0.071	**0.358**	0.119	•

Table 2: Substitution Generation benchmarking results

that one can build an approach that provides a good balance between Precision and Recall using only typical embeddings trained over raw text. If a reliable POS tagger is available, however, context-aware embeddings should also be considered (Paetzold & Specia, 2016d).

5. Substitution Selection

The goal of Substitution Selection (SS) is to determine which of the candidates available for a given complex word fit the context of the sentence being simplified. An LS approach with a "perfect" SG strategy is not guaranteed to produce sentences without errors: the Substitution Selection step must be able to effectively select the candidate substitutions that fit a given context with respect to its grammatical constructions and meaning. Because of that, this task is one of the most important in the LS pipeline, since it should prevent an LS system from performing lexical substitutions that alter the meaning or fluency of a complex sentence, hence, in some cases, rendering it incomprehensible.

The surveyed SS approaches can be divided in the following five categories:

- Select all candidates;
- Explicit sense labelling;
- Implicit sense labelling;
- Part-of-speech tag filtering; and
- Semantic similarity filtering.

5.1 Select All Candidates

Similarly to what has been observed for the task of CWI, early LS approaches do not address the task of SS, and instead choose to consider all possible substitutions of a given complex word as valid candidates for simplification (Devlin & Tait, 1998; Carroll et al., 1998).

The error analysis conducted by Shardlow (2014a) provides some insight on the impact of not performing SS on the quality of simplifications produced. 115 sentences were simplified by their approach. They found that the absence of an SS strategy led to over 29% of the simplifications made to drastically change the meaning of the original sentence. Since a single incoherent substitution can compromise the meaning of a sentence, these results imply that selecting all candidates could lead to almost a third of the simplified sentences produced by an LS approach to compromise the original meaning of the text.

5.2 Explicit Sense Labelling

LS approaches that explore explicit sense labelling attempt to model SS as a word sense disambiguation (WSD) task. This strategy uses classification methods to decide the sense label of an ambiguous target word in the sentence being simplified, and then selects as valid candidates those which have the same label. Such sense labels can be found in linguistic databases, such as WordNet.

Thomas and Anderson (2012) focus on the tasks of SG and SS. They evaluate three WSD and two lexicon reduction strategies for the creation of a reduced lexicon for a document. As sense labels, they use the synset codes from WordNet. Evaluation is done by measuring the semantic distance between the content of the reduced lexicons produced over a given document and the lexicon of its manually simplified version. Their results seem promising, but they do not conduct experiments to investigate whether or not lexicon reduction helps in the task of LS.

Nunes et al. (2013) describe a full LS system that uses explicit sense labelling. During SS, they select only synonyms found in WordNet which are linked to the meaning of a complex word. To determine a complex word's meaning, they employ the WSD approach of Navigli and Ponzetto (2010). In an experiment, they ask humans to determine whether or not a simplified sentence produced by their system is grammatical, and if it has the same meaning of the original sentence. They achieve 82% of meaning preservation, but grammaticality is compromised in 37% of the cases.

In an effort to address the lack of WSD resources for Spanish, Baeza-Yates et al. (2015) introduce a novel approach. In order to disambiguate a word to be simplified in a certain sentence, they first extract the 5-gram composed by the target word itself and two words to both its left and right sides. They then calculate the score of each of the word's sense in the Spanish OpenThesaurus by summing the frequency with which each synonym appears as the third token of the 5-gram in the Google 1T corpus. The synonyms pertaining to the sense with the highest frequency are then selected as suitable candidates. Their strategy achieved the highest scores in a performance comparison with other simplifiers for Spanish.

However promising, explicit sense labelling has several limitations in practice. The most obvious one is the need for manually created sense/synonym databases, which are often limited to very few languages and very expensive to produce. Due to the nature of WSD, explicit sense labelling also hinders the capability of an LS approach to replace words with simpler multi-word paraphrases: while determining the sense of a single word is already challenging, determining the sense of a paraphrase is even more so, unless the phrase's meanings are already stored in the lexicon as a unit.

5.3 Implicit Sense Labelling

An intuitive way to address some of the limitations of explicit sense labelling is to use automatic methods to learn sense classes of complex words, instead of querying them from sense databases. De Belder and Moens (2010) describe, to our knowledge, the only example of LS approach that does so. They select as valid candidates only those substitutions which are grouped together by a latent variable language model trained over large corpora. Their experiments show that their LS approach effectively increases the readability of documents.

Latent variable language models differ from standard n-gram language models by automatically learning latent classes which group words that appear in similar contexts. Such classes are often interpreted as "sense classes" that have a strong correlation with synonymy groups. Such language models, however, are difficult to produce: the complexity of the algorithms used to create them is often quadratic with respect to the number of classes which it is set to learn (Brown, deSouza, Mercer, Pietra, & Lai, 1992).

5.4 Part-of-Speech Tag Filtering

Given the difficulty in determining the sense of complex words, some approaches use POS tags as surrogates for sense labels. PorSimples (Aluisio & Gasperin, 2010) and FACILITA (Watanabe et al., 2009) are simplification systems for the Brazilian Portuguese language that select as valid candidates only those substitutions which have the same POS tag as the target complex word. The main motivation behind this strategy is the absence of reliable WSD resources such as sense-based databases for the Portuguese language.

A similar concept has also been used in the context of tree transduction for simplification. Paetzold and Specia (2013) describe an LS approach which replaces words and paraphrases with simpler alternatives by automatically learning lexico-syntactic substitution rules through tree transduction. Each rule is represented by a pair of subtrees of constituency parses which represent similar content. To filter bad substitution rules, they discard all pairs in which the tags of the root nodes are different. Notice that, because they evaluate root nodes, they perform filtering based on not only POS tags, but also more abstract syntactic constructs, such as noun and verb phrases. Their results reveal that using this approach ensures grammaticality in over 83% of the cases.

POS tags have been shown to successfully filter many inappropriate substitutions for complex words which are associated with multiple grammatical classes. However, they are not sufficient for meaning disambiguation. This can be a problem when an LS system has to decide on which substitutions are valid for highly ambiguous words, such as "pitch", which, aside from being both a verb and a noun, has 23 distinct meanings in WordNet across all of its grammatical classes. The meaning preservation results of Paetzold and Specia (2013) highlight this limitation: only 56.5% of the simplified sentences have the same meaning as their original version.

5.5 Semantic Similarity Filtering

Semantic similarity filtering consists in establishing a metric of the similarity between the meaning of a complex word in context and that of a substitution candidate, and then discarding all candidates which do not have enough meaning similarity with the complex word.

In the work of Biran et al. (2011), given a set of candidate substitutions for complex words, they create 10-token window co-occurrence word vectors $C(Sent\,(t))$ and $C(c)$, where $C(Sent\,(t))$ represents the semantic content of target word t in sentence $Sent$, and $C(c)$ the semantic content of candidate c in a large corpus. Finally, they discard candidates whose cosine distance between $C(Sent\,(t))$ and $C(c)$ is lower than 0.1, a threshold achieved through experimentation.

This setup is a promising solution for capturing meaning preservation: Biran et al. (2011) report 75.86% of meaning preservation during simplification. Additionally, this approach does not rely on manually created linguistic databases, and hence can be applied to any language for which large corpora are available.

A similar approach was proposed by Paetzold and Specia (2015). Instead of a co-occurrence model, they use a word embeddings model to determine the semantic similarity between a candidate and the context of a complex word being simplified. They rank all candidates according to the average cosine distance between each of them and the content words in the sentence, and then retrieve a proportion of the most similar candidates. Their selector outperforms all other strategies evaluated.

The work of Paetzold and Specia (2016d) introduces an Unsupervised Boundary Ranking approach that is able to consider various features in deciding on the candidates that can best replace a target word. Their approach explores the Robbins-Sturgeon Hypothesis, formulated by themselves, which states that a word can only be replaced by itself. They calculate various context-aware features for each target word and candidate substitution pair, creating a binary classification setup by assigning label 1 to each target word and 0 to all other candidates. Finally, they train a linear model over this labelled data. To select candidates, they rank candidates according to how far they are from the negative portion of the data, and then select a set percentage of them. This approach was reported to achieve the highest F-scores when compared to others in the literature.

5.6 Benchmarking

We compare the performance of eight SS strategies that represent all varieties of selectors previously described. The selectors evaluated are:

No selection: Selects all candidates.

Lesk (Lesk, 1986): Performs explicit sense labelling using the Lesk algorithm. It selects the sense of the target complex word for which the words in its definitions in WordNet have the highest overlap with the words in the context of that target complex word.

Leacock (Leacock & Chodorow, 1998): Performs explicit sense labelling using the Path Similarity algorithm, which is similar to the Lesk algorithm, but differs in the sense that it also incorporates sense distances in determining a target complex word's sense.

Belder (De Belder & Moens, 2010): Performs implicit sense labelling by using automatically learned word clusters. Since we could not find exact guidelines on how to replicate their approach, we use an adaptation. We learn $2,000$ word clusters using the Brown clustering algorithm (Brown et al., 1992) over the News Crawl corpus (Callison-Burch et al., 2011) and take them as the latent variables described by De Belder and Moens (2010).

Biran (Biran et al., 2011): Performs semantic similarity filtering through word co-occurrence models. Instead of Wikipedia or Simple Wikipedia, we train the co-occurrence model over the same corpus of 7 billion words used by the Glavas generator in section 4.3 in order to increase the informativeness of the vectors. We use the same cutoff range between 0.01 and 0.1 used by Biran et al. (2011).

Aluisio (Aluisio & Gasperin, 2010): Performs POS tag filtering. We adapt their approach to English: we learn the POS tags that can be associated with each candidate by tagging each sentence in the News Crawl corpus with the Stanford Tagger, and then collecting word to POS tag counts.

Paetzold-WV (Paetzold & Specia, 2015): Performs semantic similarity filtering using word embedding models. We employ the model as in the original experiments (Paetzold & Specia, 2015), which were trained with 500 word vector dimensions using the CBOW method in `word2vec`. The training corpus is the same used by the Biran selector in order to make results comparable. We select 50% of the best candidates for each target word.

Paetzold-BR (Paetzold & Specia, 2016d): Performs semantic similarity filtering through Unsupervised Boundary Ranking. We generate candidates for the binary classification setup using the Paetzold-CA generator, described in section 4. The model used is the exact same linear model described by Paetzold and Specia (2016d).

Like in our previous benchmark, all selectors were re-implemented based on the original papers, except for those created by the authors of this paper (Paetzold-WV and Paetzold-BR), for which we already had a functional implementation available. It is important to point out that the Belder selector was not described in enough detail to be replicated exactly, but we are confident that we could reproduce the general idea in the original approach.

5.6.1 Datasets and Metrics

For evaluation, we use the same metrics and dataset used in our SG benchmarking: Potential, Precision, Recall and F-score over the LexMTurk dataset. This setup allows us to not only compare the performance of selectors, but to also check how helpful they are in comparison to not performing SS at all.

Since selectors require generated candidate substitutions to be selected, we use a combination of the candidates produced by all generators described in section 4.3 to avoid any biases towards specific generators. Using this evaluation strategy, we assess the effectiveness of selectors in practice, since their task is to select appropriate candidates from a large set extracted from many sources.

5.6.2 Results

Table 3 illustrates the results. In the last column of Table 3 are featured 10-fold bootstrap resampling statistical significance tests between the best generator and the others according to their F-score. As expected, not performing selection yields the highest Potential and Recall scores. The *Paetzold-BR* selector, on the other hand, obtains the highest F-scores, showing that it effectively balances Precision and Recall. But if one's goal is to maximise Precision, the rather minimalistic *Belder* selector is clearly the best alternative, since it only requires for word clusters trained in unsupervised fashion.

It is important to mention also that, since the candidates being filtered come from a super-set containing candidates from all SG strategies evaluated in section 4.3, it is quite challenging for the selectors to effectively discard all spurious candidates, which consequently leads to rather low F-scores across all approaches.

Selector	Potential	Precision	Recall	F-score	*p*
Leacock	0.052	0.014	0.005	0.007	•
Lesk	0.334	0.045	0.043	0.044	•
Belder	0.408	**0.257**	0.046	0.078	•
Biran	0.602	0.079	0.198	0.113	•
Paetzold-WV	0.898	0.107	0.201	0.140	•
Aluisio	0.974	0.122	0.320	0.176	•
Paetzold-BR	0.972	0.231	0.261	**0.245**	-
No Selection	**0.996**	0.071	**0.358**	0.119	•

Table 3: Substitution Selection benchmarking results

5.7 Discussion

Based on the findings presented, we can state that strategies that exploit machine learning are the most promising in SS. Classic WSD approaches, on the other hand, tend to perform poorly for this task. The selector of Paetzold and Specia (2016d) is, however, the only approach of its kind that we are aware of, which suggests that there are still many possibilities to be explored for this task when it comes to using machine learning algorithms.

Nonetheless, simpler strategies can also yield noticeable performance gains over WSD approaches. The POS tag filtering strategy of Aluisio and Gasperin (2010) is the best example of that, even though it simply discards candidates that cannot take the same grammatical form of the target complex word. Although the implicit sense labelling strategy of De Belder and Moens (2010) produces lower F-scores than selecting all candidates, it can still be a suitable alternative in scenarios where Precision is much more important than Recall.

6. Substitution Ranking

The last step in the typical LS pipeline is the decision on which of the candidate substitutions that fit the context of a complex word is the simplest. In essence, the task of Substitution Ranking (SR) consists in, given the needs of a target audience, quantifying the simplicity of candidate substitutions such that replacing a target complex word with the top candidate will yield the simplest possible output.

SR was the focus of the English Lexical Simplification task of SemEval 2012 (Specia et al., 2012). The shared task attracted many participants and led to the introduction of various novel ranking approaches. More recent approaches have also used the corpus and metrics proposed back then (Shardlow, 2013b; Horn et al., 2014).

The following sections present three categories of SR strategies:

- Frequency-based;
- Simplicity measures; and

- Machine learning-assisted.

6.1 Frequency-Based

Although simple, frequency-based SR strategies are one of the most popular choices by LS systems, and can be quite effective. Approaches in this category explore the intuition that the more frequently a word is used, the more familiar it is to readers.

The most widely used frequency-based approach is Kucera-Francis (Rudell, 1993), a metric that determines the simplicity of a word given its frequency in the Brown corpus (Devlin & Tait, 1998; Carroll et al., 1998; Shardlow, 2014a; De Belder & Moens, 2010). Although popular, especially in early work, some studies show that the Kucera-Francis coefficient is not the most sensible approach for SR. Burgess and Livesay (1998), for example, argue that the frequencies taken from the Brown corpus, composed by roughly 1 million words, can be outperformed by raw frequencies from larger corpora. Brysbaert and New (2009) showed that the origin of the corpora used to extract frequencies can greatly influence ranking results: raw frequencies extracted from a corpus composed of movie subtitles have been shown to better capture word familiarity, and consequently correlate better with word simplicity than the Kucera-Francis coefficient.

In SR, subtitles have been proven useful. The simplifier of Paetzold and Specia (2016d) uses a context-aware frequency-based SR approach. It first trains a language model over a corpus of subtitles of movies for children and families, then ranks candidates according to their 5-gram frequency, i.e. the candidate surrounded by two words to the left and right of the target word. Their strategy was found to effectively capture simplicity as perceived by non-native English speakers.

In frequency-based ranking, most work use raw frequencies from very large corpora. Ligozat, Grouin, Garcia-Fernandez, and Bernhard (2012) use frequencies extracted from the Microsoft N-gram Services platform (Microsoft, 2017), which offers access to language models of up to 5-grams for the English language. Similarly, Leroy et al. (2013) and Baeza-Yates et al. (2015) use word frequency estimates from the Google 1T Corpus (Brants, 2006), composed by over one trillion words of the English language. Kauchak (2013) discusses how combining word frequencies obtained from simplified texts, such as articles from Simple Wikipedia, and frequencies obtained from unsimplified data, can improve on the performance of frequency-based rankers. A ranker which uses interpolated data between Wikipedia and Simple Wikipedia performs 23% better at the English Lexical Simplification task of SemEval 2012 than a ranker that uses only data from Simple Wikipedia.

Frequency-based rankers for languages other than English are also a popular choice. The work of Keskisärkkä (2012) is an example for the Swedish language: it uses the Swedish Parole database (Gronostaj, 2017) as a source for word frequencies. Elhadad and Sutaria (2007), who target the simplification of medical content, rank lay expressions for technical medical terms according to their frequencies in a set of documents of the medical domain in English.

Search engines have also been often used as sources for word frequency estimates. The LS approach for the Portuguese language described by Aluisio and Gasperin (2010) ranks substitutions by their number of occurrences in pages retrieved through the Google API (Google, 2017). A similar ranking approach is presented by Nunes et al. (2013): they use the Yahoo Search Engine API (Yahoo, 2017) to query for individual candidates and rank them according to the number of pages in which they appear. This strategy can be a very practical alternative for the task in online scenarios, since it discards the need for large language models trained over billions of words, and hence allows for lightweight simplifiers to be created. However, the performance of such approaches is volatile and difficult to replicate, given the constant expansion of search engines' databases.

In practice, frequency-based approaches have been shown to outperform more sophisticated ranking approaches quite often. In the results reported by Specia et al. (2012), a baseline that ranked candidate substitutions according to their raw frequencies in the Google 1T corpus outperformed 9 out of 11 ranking approaches, and consequently placed 3rd overall. However, they might not be sufficient in some scenarios. The user studies of Rello et al. (2013b) with dyslexic readers, for example, reveals that their perception of simplicity also encompasses the word's length.

6.2 Simplicity Measures

An alternative to address the limitations of frequency-based ranking strategies are metrics that incorporate multiple features to represent the simplicity of a word. The metric introduced by Biran et al. (2011), for example, considers a word's frequency and length to determine its complexity. Their metric is shown in Equation 1, where $Comp(c)$ is the corpus complexity of candidate c, and $\|c\|$, its length in characters.

$$M(c) = Comp(c) * \|c\| \tag{1}$$

the corpus complexity component is computed as illustrated in Equation 2, where $F\left(c, C\right)$ is the raw frequency of candidate c in corpus C. The "Complex" and "Simple" corpora required by the metric must contain text of complex and simple nature, respectively. To that purpose, they collect all articles available from Wikipedia (Complex) and Simple Wikipedia (Simple).

$$Comp(c) = \frac{F\left(c, \text{Complex}\right)}{F\left(c, \text{Simple}\right)} \tag{2}$$

A similar metric is used by Sinha (2012). It combines a candidate substitute's length, number of senses in WordNet, and frequency of occurrence in various corpora to determine a word's simplicity. Following the same notation used in Equation 2, the metric is computed as illustrated in Equation 3, where $S_{wn}\left(c\right)$ is the number of senses of c in WordNet.

$$\begin{aligned} M(c) = &F\left(c, \text{Simple Wiki}\right) + \\ &F\left(c, \text{Speech}\right) + \\ &F\left(c, \text{Google1T}\right) + \\ &S_{wn}\left(c\right) + \\ &\frac{1}{\|c\|} \end{aligned} \tag{3}$$

$F\left(c, \text{Simple Wiki}\right)$ and $F\left(c, \text{Google1T}\right)$ are word frequencies extracted from Simple Wikipedia (Kauchak, 2013) and the Google 1T corpus (Michel et al., 2011). The "Speech" corpus in $F\left(c, \text{Speech}\right)$ is a proprietary compilation of written dialogue content. The metric obtained the 2nd highest ranking scores in the English Lexical Simplification task of SemEval 2012.

Bott et al. (2012) describe a word simplicity measure for the Spanish language. They also consider the word's length and usage frequency in their measure, but go a step further, as described in Equation 4, to include two weighted scores, where α_1 and α_2 are adjustable weights.

$$M(c) = \alpha_1 score_{wl}\left(c\right) + \alpha_2 score_{freq}\left(c\right) \tag{4}$$

In order to estimate weights α_1 and α_2, Bott et al. (2012) resort to a heuristic search that maximises the score of the measure over a set of manually created lexical simplifications. In Equation 4, the values of $score_{wl}\left(c\right)$ and $score_{freq}\left(c\right)$ are calculated as illustrated in Equations 5 and 6, where $F\left(c, \text{Simple}\right)$ is computed over the Spanish Simplext Corpus (Bott & Saggion, 2011).

$$score_{wl}\left(c\right) = \begin{cases} \sqrt{\|c\| - 4} & if\ \|c\| \geq 5 \\ 0 & otherwise \end{cases} \tag{5}$$

$$score_{freq}\left(c\right) = \log\left(F\left(c, \text{Simple}\right)\right) \tag{6}$$

The motivation behind $score_{wl}\left(c\right)$ comes from the observation that, in their set of manually crafted lexical simplifications, the complex words in Spanish have on average four characters more than their simple counterparts.

Simplicity measures can also be more sophisticated and incorporate the relation between the candidate and the context of the complex word to be simplified. Kajiwara et al. (2013), for example, represent simplicity as the weighted sum of five metrics that consider various relations between substitution candidate and the sentence to be simplified. Their simplicity measure was designed for Japanese, and can be described as illustrated in Equation 7, where $Sense$ is the distance between the senses of candidate c and target t, $Cooc$ the co-occurrence sum of the words in sentence S and candidate c, Log the normalised co-occurrence sum between c and S, $Trigram$ the frequency sum of all trigrams surrounding c when replacing target word t in S, and Sim the distributional similarity between c and t.

$$\begin{aligned} M(S,t,c) = & \alpha_1 F_{corpus}\left(c\right) + \\ & \alpha_2 Sense\left(c,t\right) + \\ & \alpha_3 Cooc\left(c,S\right) + \\ & \alpha_4 Log\left(c,S\right) + \\ & \alpha_5 Trigram\left(c,S\right) + \\ & \alpha_6 Sim\left(c,t\right) \end{aligned} \tag{7}$$

Notice that, because this metric incorporates a variety of features that aim to capture grammaticality and meaning preservation, it could be used as a hybrid SS and SR approach. We refer the reader to the work of Kajiwara et al. (2013) for more details on how $Cooc$, Log, $Trigram$ and Sim are calculated. A manual evaluation of the output produced by this

LS approach reveals promising results, but their SR strategy has not yet been compared to others.

Glavaš and Štajner (2015) introduce a ranking strategy that attempts to automatically combine various features, avoiding the need for manual crafting. They resort to a minimalistic strategy: rank averaging. Their approach first produces various rankings resulting from the use of several features, such as n-gram frequencies and semantic similarity. It then scores each word by averaging all its rankings. Finally the words are ranked according to their score: the lower the average rank, the simpler it is. This approach was evaluated over the Lexical Simplification task of SemEval 2012 (held three years prior to its publishing), and managed to outperform all systems originally submitted to the task.

6.3 Machine Learning-Assisted

Given the effectiveness of machine learning techniques for various language processing tasks, they have also been adopted for ranking. The most successful ranking approach submitted to SemEval 2012 (Jauhar & Specia, 2012) uses an SVM ranker along with the combination of various ranking functions to order candidates by simplicity. They represent ranking functions as described in Equation 8, and employ them to determine the ranking of a candidate c. In Equation 8, r_i is a standalone ranking function that determines the rank of candidate c according to a certain metric, with respect to target word t in sentence S.

$$M(S,t,c) = \sum_{i=1}^{m} \frac{1}{r_i\,(S,t,c)} \tag{8}$$

The approach considers various ranking metrics, both context-aware and context-unaware, such as n-gram language model probabilities and psycholinguistic word properties. In order to optimise performance, they create various combinations of M ranking functions, use their resulting ranking values as features, and then train an SVM ranker.

An SVM ranker (Joachims, 2002) aims to minimise a loss function over the pairwise comparisons that characterise a set of ranked elements through the means of support vectors. This approach is also explored by Horn et al. (2014), where an SVM ranker is trained with word frequency and n-gram probability features from various corpora. Their LS approach outperformed two strong baselines in their evaluation, highlighting the potential of machine learning in SR.

Paetzold and Specia (2015) introduce a different supervised approached named Boundary Ranking. They first infer a binary classification setup from a set of ranking examples by calculating features for each candidate, and assigning label 1 to all candidates with a ranking $r \leq p$, and label 0 to the remaining. In this context, p is a hyper-parameter that can be described as a rank threshold between candidates that are and aren't simple enough. After estimating a value for p through cross-validation, then they train a linear model over the data, and rank new candidates according to their distance to the boundary between positive and negative samples. Combining this approach with n-gram frequencies from subtitles, they outperform other ranking strategies, including an SVM ranker.

In a more recent contribution, Paetzold and Specia (2017) present a supervised neural ranking model. The neural network is a multi-layer perceptron that receives a set of features for a pair of candidates as input, and produces as output the simplicity difference between them. To rank a set of candidates, they calculate the differences between all possible pairs, then simply average their respective scores. Combining this approach with a generator that exploits various resources, they achieve state-of-the-art results. The features used by them are n-gram probabilities extracted from spoken text corpora.

6.4 Benchmarking

We compare the performance of nine SR strategies that encompass all varieties of rankers previously described. The rankers evaluated are:

Devlin (Devlin & Tait, 1998): Performs frequency-based ranking, ordering candidates according to their Kucera-Francis coefficients.

Biran (Biran et al., 2011): Employs a simplicity measure. Uses the Wikipedia and Simple Wikipedia corpora as the source of complex and simple word frequencies, respectively.

Bott (Bott et al., 2012): Employs a simplicity measure. We adapt their measure for English by using the Simple Wikipedia as a source of word frequencies. The weights are estimated through 5-fold cross-validation over the set of values {-2, -1, 0, 1, 2}.

Yamamoto (Kajiwara et al., 2013): Employs a simplicity measure that extracts n-gram frequencies from the Simple Wikipedia and semantic similarity measures from WordNet. The co-occurrence model is the same as that used by the Biran selector, described in section 5.6. The weights are estimated through 5-fold cross-validation over the set of values {-2, -1, 0, 1, 2}.

Glavas (Glavaš & Štajner, 2015): Employs a form of simplicity measure through rank averaging and the same features used by Glavaš and Štajner (2015). We calculate them using the same embeddings model as in the Glavas generator in our SG benchmark.

Horn (Horn et al., 2014): Employs machine learning-assisted ranking through SVMs using the same features used by Horn et al. (2014), training the model with SVM rank and 10-fold cross-validation.

SubIMDB (Paetzold & Specia, 2016d): Employs a frequency-based approach that uses a 5-gram language model trained over SubIMDB and ranks candidates according to their 5-gram probability.

Boundary (Paetzold & Specia, 2015): Performs machine learning-assisted ranking through Boundary Ranking using a linear model learned through Stochastic Gradient Descent, $p = 3$, and the same features described by Paetzold and Specia (2015).

Neural (Paetzold & Specia, 2017): Uses the exact same features and architecture described by Paetzold and Specia (2017), with three hidden layers with eight nodes each and a model trained for 500 epochs.

We re-implemented all approaches based on the original papers, except for the ones conceived by the authors of this survey (SubIMDB, Boundary and Neural), for which we already had functioning implementations and pre-trained models. Much like in our previous benchmarks, it must be pointed out that, because of the non-deterministic nature of some approaches and differences in versions of certain resources, the results obtained in this evaluation may not reflect the performance of the original rankers exactly.

6.4.1 Datasets and Metrics

The rankers are evaluated over the datasets from the English Lexical Simplification task of SemEval 2012 (Specia et al., 2012). The training set is composed of 300 instances, and the test set, $1,710$ instances. Each instance contains a sentence, a target complex word, and candidates ranked by their simplicity. All these candidates fit the context. Their rankings were determined by three non-native speakers of English.

As an evaluation metric, we use the TRank measure introduced in the shared task. It calculates the proportion of instances for which the highest ranked candidate produced by a ranker is the same as the one in the gold-standard. This metric is considered the one that best captures the performance of a ranker in practice. Since we evaluate our re-implemented rankers on a dataset from a shared task, we also include in our results the original scores reported for the SemEval task.

6.4.2 Results

The first 12 rows in Table 4 illustrate the results for the systems originally submitted to the SemEval 2012 task, while the remaining lines refer to our re-implemented strategies. The *Neural* approach achieves the highest TRank scores, although we could not find any statistically significant difference between the two highest scoring rankers. Among unsupervised systems, the *Glavas* ranker, which employs the technique of rank averaging, has been shown the most effective solution, outperforming even supervised rankers.

Notice that since we do not have access to the ranks produced by the SemEval 2012 systems, we could not run statistical significance tests between them and the highest performing ranker in our benchmark. However, the figures differ by a large margin.

6.5 Discussion

The results from our benchmark reveal that, when there is training data available, supervised rankers are the most effective alternative. The SVM ranker *Horn*, the *Boundary* and *Neural* rankers achieved the three highest scores in our evaluation.

However, much like what was observed in the benchmarks for Substitution Generation and Selection, it is still possible to create strategies with competitive performance when no such data is available. When it comes to frequency-based approaches, it is best to employ either very large corpora, such as Google 1T, or spoken text corpora. As observed by Brysbaert and New (2009), Shardlow (2013a), Paetzold and Specia (2016a) and many others, frequencies from spoken text corpora, such as subtitles, tend to correlate more strongly with word familiarity scores than those from other larger corpora, which generally means that they are also more suitable for simplicity prediction. The *Glavas* unsupervised rank averaging strategy is the most suitable alternative in cases where there is no training data, since it offers higher performance than frequency and metric-based rankers and allows for different features to be combined.

Ranker	TRank	p
UNT-SaLSA	0.146	-
ANNLOR-lmbing	0.336	-
Baseline-Random	0.340	-
Baseline-L-Sub Gold	0.454	-
SB-mmSystem	0.477	-
EMNLPCPH-ORD2	0.530	-
EMNLPCPH-ORD1	0.539	-
ANNLOR-simple	0.564	-
UNT-SimpRankL	0.567	-
Baseline-Simple Freq.	0.585	-
UNT-SimpRank	0.585	-
UOW-SHEF-SimpLex	0.602	-
Biran	0.513	•
Bott	0.574	•
SubIMDB	0.580	•
Devlin	0.596	•
Yamamoto	0.604	•
Glavas	0.632	•
Horn	0.639	•
Boundary	0.655	○
Neural	**0.658**	-

Table 4: Substitution Ranking benchmarking results

7. Full LS Pipeline Evaluation

In this section, we compare the performance of various complete LS systems as they were presented in the papers that introduce them. We include seven simplifiers:

Devlin (Devlin & Tait, 1998): Combines the Devlin SG and SR systems.

Yamamoto (Kajiwara et al., 2013): Combines the Yamamoto SG and SR systems.

Biran (Biran et al., 2011): Combines the Biran SG, SS and SR systems.

Paetzold-UN (Paetzold & Specia, 2016d): Combines the Paetzold-CA generator, Paetzold-BR selector, and SubIMDB ranker.

Paetzold-SU (Paetzold & Specia, 2017): Combines the Paetzold-NE generator, Paetzold-BR selector, and Neural ranker.

Horn (Horn et al., 2014): Combines the Horn SG and SR systems.

Glavas (Glavaš & Štajner, 2015): Combines the Glavas SG and SR systems.

We extract the performance scores for the Biran, Horn and Glavas simplifiers from the work of Glavaš and Štajner (2015), since these simplifiers have already been subjected to an evaluation identical to the one conducted in this benchmark. The remaining simplifiers were built using the implementations featured in the benchmarks for Substitution Generation, Selection and Ranking. We choose these simplifiers because they are some of the most relevant examples of systems that do not restrict themselves to a specific domain, such as medical content, and also because they employ a wide variety of strategies for each step of the pipeline.

This benchmark does not encompass the step of Complex Word Identification because none of the aforementioned approaches offer an explicit strategy for this step, and also because there are no known datasets that allow one to evaluate CWI along with the other steps in the LS pipeline.

7.1 Dataset and Metrics

We use the LexMTurk dataset as our gold-standard. We use the same evaluation dataset across our Substitution Generation, Selection and full pipeline evaluation in order to make the results more easily interpretable and highlight the versatility of this type of resource. The metrics used are introduced by Horn et al. (2014):

- **Precision**: The ratio with which the highest ranking candidate is either the target word itself or is in the gold-standard.
- **Accuracy**: The ratio with which the highest ranking candidate is not the target word itself and is in the gold-standard.
- **Changed Proportion**: The ratio with which the highest ranking candidate is not the target word itself.

7.2 Results

Table 5 shows that the simplifiers have unique profiles. The *Horn* simplifier attained the highest Precision score, which suggests it is the least likely to replace a word with an unsuitable candidate. The *Biran* simplifier proved to be the most conservative approach: it barely ever replaced any of the complex words, leading to high Precision and very low Accuracy. In contrast, the boldest amongst the simplifiers is *Paetzold-SU*, which replaced all complex words with something else. The *Glavas* simplifier managed to attain the highest Accuracy, which suggests that it is the most proficient in promoting simplicity. It must be pointed out, however, that when it comes to Accuracy, we could not find a statistically significant difference between the *Paetzold-SU*, *Horn* and *Glavas* systems, which is in accordance with the findings of Glavaš and Štajner (2015).

8. Discussion and Conclusions

We presented a survey on Lexical Simplification which addresses the task as a series of steps: Complex Word Identification, Substitution Generation, Substitution Selection and Substitution Ranking. For each step, we provided a comprehensive literature review, as well as a benchmarking of the various existing approaches and a discussion on the results obtained. We also benchmarked various alternatives for full LS pipelines and provided a list of datasets and resources most widely used in existing LS work.

We remark that although our benchmarks provide a good overview on how each approach in the survey performs in practice, they are very challenging to run and report in a way that

System	Precision	Accuracy	Changed	p
Yamamoto	0.066	0.066	0.994	●
Biran	0.714	0.034	0.052	●
Devlin	0.368	0.366	0.994	●
Paetzold-UN	0.578	0.396	0.818	●
Horn	**0.761**	0.663	0.863	○
Paetzold-SU	0.676	0.676	**1.000**	○
Glavas	0.710	**0.682**	0.960	○

Table 5: Full pipeline benchmarking results

is undoubtedly fair to the researchers that have conceived the approaches evaluated. Since it would not be sensible to ask researchers to re-run their systems – which are often many years old – with new settings and datasets, we had to resort to replicating these systems. Although replicability is clearly a very desirable quality of any academic contribution, many of the details about the resources used and/or the training procedures employed are omitted from most papers, generally due to space constraints. It is also important to note that the authors of the survey have conceived some of the systems evaluated, and they naturally have a much firmer grasp on how to maximise these systems' performances. For these reasons, the reader should take the benchmarking results with a grain of salt and perhaps focus on what they entail in a more general sense.

Nevertheless, we believe much can be learned from our findings. We can safely state that there is much left to be explored in LS. For starters, we believe that the Text Simplification research community would greatly benefit from the results of new user studies which aim to gather more insights about specific readability and comprehensibility challenges caused by certain language impairments. These studies will not only promote the creation of more effective simplification strategies, but also result in new useful datasets and resources to be exploited in future research.

The pioneering studies of Devlin and Tait (1998), for example, have provided evidence that using frequently occurring words result in texts that are more easily understandable by those suffering from Aphasia. Rello et al. (2013b, 2013c) present two user studies that aim at understanding what types of words and numerical expressions challenge readers with Dyslexia the most. They found that replacing complex words with more frequent synonyms increases readability, but replacing them with shorter synonyms increases understandability. They also found that percentage expressions, such as 50% and 75%, cause much less confusion than equivalent fractions, such as 1/2 and 3/4. Using the information learned through these user studies, they have conceived a minimalistic word simplicity metric aimed specifically at capturing the needs of readers with Dyslexia, which was then incorporated in a Spanish lexical simplifier. Paetzold and Specia (2016b) performed studies focusing on non-native English speakers. They found that high frequency words are very rarely deemed complex by them, and that – contrary to previous belief, a word's length or number of syllables does not have any influence on its complexity.

There are also still many interesting opportunities to be explored when it comes to applying machine learning methods in LS. Neural networks, for example, have been successfully used to push the state-of-the-art in various challenging NLP tasks, such as machine translation (Bahdanau, Cho, & Bengio, 2014), sentiment analysis (Glorot, Bordes, & Bengio, 2011), semantic similarity (Yih & Qazvinian, 2012) and question answering (Iyyer, Boyd-Graber, Claudino, Socher, & Daumé III, 2014). To this day, the only example of simplifier that uses neural networks is rather simple: it employs a multi-layer perceptron for Substitution Ranking (Paetzold & Specia, 2017). There is no previous work, however, on using machine learning to create end-to-end dedicated lexical simplifiers that address the entire task in a single step. The only strategies that resemble end-to-end lexical simplifiers to some extent are data-driven sentence simplifiers, which manage to learn how to simplify some complex words from parallel data. Among them are not only complex-to-simple neural translators (Zhang, Ye, Feng, Zhao, & Yan, 2017; Nisioi, Štajner, Ponzetto, & Dinu, 2017; Zhang & Lapata, 2017), but also phrase-based (Specia, 2010; Coster & Kauchak, 2011; Wubben et al., 2012), and syntax-based strategies (Zhu et al., 2010; Bach, Gao, Vogel, & Waibel, 2011; Xu et al., 2016). But even though they do learn how to replace complex words to some extent, they offer no control over which words will be simplified in a complex sentence. One could use neural architectures, for an example, to create dedicated lexical simplifiers that treat LS as a constrained text generation task, or as a way of learning language models that are more reliable than the n-gram models used by most approaches surveyed.

When it comes to pipelined approaches, it is quite clear that candidate generators could still be greatly improved. Given the inherently challenging nature of the Substitution Selection step, simplifiers such as the ones of Horn et al. (2014) and Glavaš and Štajner (2015) have bypassed this step by creating rankers that aim to jointly account for grammaticality, meaning preservation and simplicity. Another way of bypassing this step would be by finding effective ways of joint modelling it along with Substitution Generation, which could yield generators that better account for word ambiguity.

Another clear gap still to be addressed is the scarcity of LS datasets for languages other than English. Although lexical simplifiers for any language can be manually evaluated, the existencex of datasets with annotated English simplification problems, such as LexMTurk (Horn et al., 2014), LSeval (De Belder & Moens, 2012) and BenchLS (Paetzold & Specia, 2016a), has clearly fostered the creation of new approaches for English, since they make it much easier to both train and evaluate new approaches.

For those interested in using approaches described in this survey, all of the implementations devised for our benchmarkings can be found in the LEXenstein framework[32].

References

Aluisio, S., & Gasperin, C. (2010). Fostering digital inclusion and accessibility: The porsimples project for simplification of portuguese texts. In *Proceedings of the 2010 NAACL Young Investigators Workshop on Computational Approaches to Languages of the Americas*, pp. 46–53.

Azab, M., Hokamp, C., & Mihalcea, R. (2015). Using word semantics to assist english as a second language learners. In *Proceedings of the 2015 NAACL*, pp. 116–120. Association for Computational Linguistics.

[32] http://ghpaetzold.github.io/LEXenstein

Bach, N., Gao, Q., Vogel, S., & Waibel, A. (2011). Tris: A statistical sentence simplifier with log-linear models and margin-based discriminative training. In *Proceedings of 5th IJCNLP*, pp. 474–482.

Baeza-Yates, R., Rello, L., & Dembowski, J. (2015). CASSA: A context-aware synonym simplification algorithm. In *Proceedings of the 2015 NAACL*, pp. 1380–1385.

Bahdanau, D., Cho, K., & Bengio, Y. (2014). Neural machine translation by jointly learning to align and translate. *CoRR*, *abs/1409.0473*.

Bingel, J., Schluter, N., & Martínez Alonso, H. (2016). Coastalcph at semeval-2016 task 11: The importance of designing your neural networks right. In *Proceedings of the 10th SemEval*, pp. 1028–1033.

Biran, O., Brody, S., & Elhadad, N. (2011). Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of the 49th ACL*, pp. 496–501.

Bird, S., Klein, E., & Loper, E. (2009). *Natural Language Processing with Python: Analyzing Text with the Natural Language Toolkit*. O'Reilly.

Bodenreider, O. (2004). The unified medical language system (UMLS): integrating biomedical terminology. *Nucleic acids research*, *32*(1), 267–270.

Bott, S., Rello, L., Drndarevic, B., & Saggion, H. (2012). Can spanish be simpler? lexsis: Lexical simplification for spanish. In *Proceedings of the 2012 COLING*, pp. 357–374.

Bott, S., & Saggion, H. (2011). An unsupervised alignment algorithm for text simplification corpus construction. In *Proceedings of the 2011 MTTG*, pp. 20–26.

Brants, T. (2006). Web 1t 5-gram version 1. https://catalog.ldc.upenn.edu/LDC2006T13. Accessed in: 2017-10-03.

Brown, P. F., deSouza, P. V., Mercer, R. L., Pietra, V. J. D., & Lai, J. C. (1992). Class-based n-gram models of natural language. *Computational Linguistics*, *18*, 467–479.

Brunato, D., Cimino, A., DellOrletta, F., & Venturi, G. (2016). Paccss–it: A parallel corpus of complex–simple sentences for automatic text simplification. In *Proceedings of the 2016 EMNLP*, pp. 351–361.

Brysbaert, M., & New, B. (2009). Moving beyond kučera and francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for american english. *Behavior research methods*, *41*, 977–990.

Burges, C., Shaked, T., Renshaw, E., Lazier, A., Deeds, M., Hamilton, N., & Hullender, G. (2005). Learning to rank using gradient descent. In *Proceedings of the 22nd ICML*, pp. 89–96.

Burgess, C., & Livesay, K. (1998). The effect of corpus size in predicting reaction time in a basic word recognition task: Moving on from kučera and Francis. *Behavior Research Methods, Instruments, & Computers*, *30*, 272–277.

Burns, P. R. (2013). Morphadorner v2: A java library for the morphological adornment of english language texts. http://morphadorner.northwestern.edu.

Callison-Burch, C., Koehn, P., Monz, C., & Zaidan, O. (2011). Findings of the 2011 workshop on statistical machine translation. In *Proceedings of the 6th WMT*, pp. 22–64.

Cao, Z., Wei, F., Dong, L., Li, S., & Zhou, M. (2015). Ranking with recursive neural networks and its application to multi-document summarization. In *Proceedings of the 2015 AAAI*, pp. 2153–2159.

Carroll, J., Minnen, G., Canning, Y., Devlin, S., & Tait, J. (1998). Practical simplification of english newspaper text to assist aphasic readers. In *Proceedings of 1998 AAAI Workshop on Integrating Artificial Intelligence and Assistive Technology*, pp. 7–10.

Carroll, J., Minnen, G., Pearce, D., Canning, Y., Devlin, S., & Tait, J. (1999). Simplifying text for language-impaired readers. In *Proceedings of the 9th EACL*, pp. 269–270.

Caruana, R., Baluja, S., & Mitchell, T. (1996). Using the future to "sort out" the present: Rankprop and multitask learning for medical risk evaluation. *Advances in Neural Information Processing Systems*, *8*, 959–965.

Chandrasekar, R., Doran, C., & Srinivas, B. (1996). Motivations and methods for text simplification. In *Proceedings of the 16th COLING*, pp. 1041–1044.

Chang, C.-C., & Lin, C.-J. (2011). Libsvm: a library for support vector machines. *ACM Transactions on Intelligent Systems and Technology*, *2*(3), 27.

Chen, H.-B., Huang, H.-H., Chen, H.-H., & Tan, C.-T. (2012). A simplification-translation-restoration framework for cross-domain smt applications. In *Proceedings of 2012 COLING*, pp. 545–560.

Choubey, P., & Pateria, S. (2016). Garuda & Bhasha at semeval-2016 task 11: Complex word identification using aggregated learning models. In *Proceedings of the 10th SemEval*, pp. 1006–1010.

Cohn, T., & Lapata, M. (2009). Sentence compression as tree transduction. *Journal of Artificial Intelligence Research*, *34*(1), 637–674.

Coltheart, M. (1981). The mrc psycholinguistic database. *The Quarterly Journal of Experimental Psychology*, *33*(4), 497–505.

Coster, W., & Kauchak, D. (2011). Simple english wikipedia: A new text simplification task. In *Proceedings of the 49th ACL*, pp. 665–669.

De Belder, J., & Moens, M.-F. (2010). Text simplification for children. In *Proceedings of the 2010 SIGIR Workshop on Accessible Search Systems*, pp. 19–26.

De Belder, J., & Moens, M.-F. (2012). A dataset for the evaluation of lexical simplification. In *Proceedings of the 13th CICLing 2012*, pp. 426–437.

Deléger, L., & Zweigenbaum, P. (2009). Extracting lay paraphrases of specialized expressions from monolingual comparable medical corpora. In *Proceedings of the 2nd Workshop on Building and Using Comparable Corpora: from Parallel to Non-parallel Corpora*, pp. 2–10.

Devlin, S., & Tait, J. (1998). The use of a psycholinguistic database in the simplification of text for aphasic readers. *Linguistic Databases*, *1*, 161–173.

Devlin, S., & Unthank, G. (2006). Helping aphasic people process online information. In *Proceedings of the 8th SIGACCESS*, pp. 225–226.

Devlin, S. (1999). *Simplifying Natural Language for Aphasic Readers*. Ph.D. thesis, University of Sunderland.

Drndarević, B., & Saggion, H. (2012). Towards automatic lexical simplification in spanish: an empirical study. In *Proceedings of the 1st PITR*, pp. 8–16.

Elhadad, N. (2006). Comprehending technical texts: Predicting and defining unfamiliar terms. In *Proceedings of the 2006 AMIA*, pp. 239–243.

Elhadad, N., & Sutaria, K. (2007). Mining a lexicon of technical terms and lay equivalents. In *Proceedings of the 2007 BioNLP*, pp. 49–56.

Faruqui, M., Dodge, J., Jauhar, S. K., Dyer, C., Hovy, E., & Smith, N. A. (2015). Retrofitting word vectors to semantic lexicons. In *Proceedings of the 2015 NAACL*, pp. 1606–1615.

Feblowitz, D., & Kauchak, D. (2013). Sentence simplification as tree transduction. In *Proceedings of the 2nd Workshop on Predicting and Improving Text Readability for Target Reader Populations*, pp. 1–10.

Fellbaum, C. (1998). *WordNet: An Electronic Lexical Database*. Bradford Books.

Glavaš, G., & Štajner, S. (2015). Simplifying lexical simplification: Do we need simplified corpora?. In *Proceedings of the 53rd ACL*, pp. 63–69.

Glorot, X., Bordes, A., & Bengio, Y. (2011). Deep sparse rectifier neural networks. *Aistats*, *15*(106), 275.

Google (2017). The Google JSON/Atom Custom Search API. https://developers.google.com/custom-search. Accessed in: 2017-10-03.

Gronostaj, M. T. (2017). The Swedish PAROLE Lexicon. http://spraakdata.gu.se/parole/lexikon/swedish.parole.lexikon.html. Accessed in: 2017-10-03.

Hearst, M. A. (1994). Multi-paragraph segmentation of expository text. In *Proceedings of the 32nd ACL*, pp. 9–16.

Hirsh, D., & Nation, P. (1992). What vocabulary size is needed to read unsimplified texts for pleasure?. *Reading in a Foreign Language*, *8*, 689–689.

Horn, C., Manduca, C., & Kauchak, D. (2014). Learning a lexical simplifier using wikipedia. In *Proceedings of the 52nd ACL*, pp. 458–463.

Iyyer, M., Boyd-Graber, J., Claudino, L., Socher, R., & Daumé III, H. (2014). A neural network for factoid question answering over paragraphs. In *Proceedings of the 2014 EMNLP*, pp. 633–644.

Janczura, G. A., Castilho, G. M. d., Rocha, N. O., van Erven, T. d. J. C., & Huang, T. P. (2007). Normas de concretude para 909 palavras da lingua portuguesa. *Psicologia: Teoria e Pesquisa*, *23*, 195 – 204.

Jauhar, S., & Specia, L. (2012). Uow-shef: Simplex–lexical simplicity ranking based on contextual and psycholinguistic features. In *Proceedings of the 1st SemEval*, pp. 477–481.

Joachims, T. (2002). Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM*, pp. 133–142.

Kajiwara, T., & Komachi, M. (2016). Building a monolingual parallel corpus for text simplification using sentence similarity based on alignment between word embeddings. In *Proceedings of the 26th COLING*, pp. 1147–1158.

Kajiwara, T., Matsumoto, H., & Yamamoto, K. (2013). Selecting proper lexical paraphrase for children. In *Proceedings of the 25th ROCLING*, pp. 59–73.

Kajiwara, T., & Yamamoto, K. (2015). Evaluation dataset and system for japanese lexical simplification. In *Proceedings of the 2015 ACL Student Research Workshop*, pp. 35–40.

Kandula, S., Curtis, D., & Zeng-Treitler, Q. (2010). A semantic and syntactic text simplification tool for health content. In *Proceedings of the 2010 AMIA*, pp. 366–70.

Kann, V. (2017). Folkets Synonymlexikon Synlex. http://folkets-lexikon.csc.kth.se/synlex.html. Accessed in: 2017-10-03.

Kauchak, D. (2013). Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st ACL*, pp. 1537–1546.

Kauchak, D., & Barzilay, R. (2006). Paraphrasing for automatic evaluation. In *Proceedings of the 2006 NAACL*, pp. 455–462.

Keskisärkkä, R. (2012). Automatic text simplification via synonym replacement. Master's thesis, Linköping University.

Kodaira, T., Kajiwara, T., & Komachi, M. (2016). Controlled and balanced dataset for japanese lexical simplification. In *Proceedings of the 2016 ACL Student Research Workshop*, pp. 1–7.

Konkol, M. (2016). Uwb at semeval-2016 task 11: Exploring features for complex word identification. In *Proceedings of the 10th SemEval*, pp. 1038–1041.

Kuru, O. (2016). Ai-ku at semeval-2016 task 11: Word embeddings and substring features for complex word identification. In *Proceedings of the 10th SemEval*, pp. 1042–1046.

Leacock, C., & Chodorow, M. (1998). Combining local context and wordnet similarity for word sense identification. *WordNet: An Electronic Lexical Database*, *49*(2), 265–283.

Lee, J., & Seneff, S. (2006). Automatic grammar correction for second-language learners. In *Proceedings of the 2006 Interspeech*, pp. 1978–1981.

Leroy, G., Endicott, J. E., Kauchak, D., Mouradi, O., & Just, M. (2013). User evaluation of the effects of a text simplification algorithm using term familiarity on perception, understanding, learning, and information retention. *Journal of Medical Internet Research*, *15*.

Lesk, M. (1986). Automatic sense disambiguation using machine readable dictionaries: how to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Conference on Systems Documentation*, pp. 24–26.

Ligozat, A.-L., Grouin, C., Garcia-Fernandez, A., & Bernhard, D. (2012). Annlor: A naïve notation-system for lexical outputs ranking. In *Proceedings of the 1st *SEM*, pp. 487–492.

Malmasi, S., Dras, M., & Zampieri, M. (2016). Ltg at semeval-2016 task 11: Complex word identification with classifier ensembles. In *Proceedings of the 10th SemEval*, pp. 996–1000.

Maziero, E. G., Pardo, T. A. S., Di Felippo, A., & Dias-da Silva, B. C. (2008). A base de dados lexical e a interface web do tep 2.0: Thesaurus eletronico para o portugues do brasil. In *Proceedings of the 14th Brazilian Symposium on Multimedia and the Web*, pp. 390–392.

McCarthy, D., & Navigli, R. (2007). Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th SemEval*, pp. 48–53.

Merriam-Webster (2017). The Merriam-Webster Dictionary. http://www.merriam-webster.com. Accessed in: 2017-10-03.

Michel, J.-B., Shen, Y. K., Aiden, A. P., Veres, A., Gray, M. K., Pickett, J. P., Hoiberg, D., Clancy, D., Norvig, P., Orwant, J., Pinker, S., Nowak, M. A., & Aiden, E. L. (2011). Quantitative analysis of culture using millions of digitized books. *Science*, *331*(6014), 176–182.

Microsoft (2017). Microsoft Web Language Model API. https://azure.microsoft.com/en-gb/services/cognitive-services/web-language-model. Accessed in: 2017-10-03.

Mihalcea, R., Sinha, R., & McCarthy, D. (2010). Semeval-2010 task 2: Cross-lingual lexical substitution. In *Proceedings of the 5th SemEval*, pp. 9–14.

Mikolov, T., Yih, W.-t., & Zweig, G. (2013). Linguistic regularities in continuous space word representations. In *Proceedings of 2013 NAACL*, pp. 746–751.

Mukherjee, N., Patra, B. G., Das, D., & Bandyopadhyay, S. (2016). Ju_nlp at semeval-2016 task 11: Identifying complex words in a sentence. In *Proceedings of the 10th SemEval*, pp. 986–990.

Nat, G. (2016). Sensible at semeval-2016 task 11: Neural nonsense mangled in ensemble mess. In *Proceedings of the 10th SemEval*, pp. 963–968.

Nation, I. S. P. (2001). *Learning vocabulary in another language*. Ernst Klett Sprachen.

Navigli, R., & Ponzetto, S. P. (2010). Babelnet: Building a very large multilingual semantic network. In *Proceedings of the 48th ACL*, pp. 216–225.

Newsela (2016). Newsela article corpus. https://newsela.com/data. Version: 2016-01-29.

Nisioi, S., Štajner, S., Ponzetto, S. P., & Dinu, L. P. (2017). Exploring neural text simplification models. In *Proceedings of the 55th ACL*, pp. 85–91.

Nunes, B. P., Kawase, R., Siehndel, P., Casanova, M. a., & Dietze, S. (2013). As simple as it gets - a sentence simplifier for different learning levels and contexts. In *Proceedings of the 13th ICALT*, pp. 128–132.

Ogden, C. K. (1968). *Basic English: international second language*. Harcourt, Brace & World.

Ong, E., Damay, J., Lojico, G., Lu, K., & Tarantan, D. (2007). Simplifying text in medical literature. *Journal of Research in Science, Computing and Engineering*, *4*(1), 37–47.

OpenThesaurus (2017). OpenThesaurus-es - Thesaurus in Spanish. http://openoffice-es.sourceforge.net/thesaurus. Accessed in: 2017-10-03.

Paetzold, G., & Specia, L. (2016a). Benchmarking lexical simplification systems. In *Proceedings of the 10th LREC*, pp. 23–28.

Paetzold, G., & Specia, L. (2016b). Collecting and exploring everyday language for predicting psycholinguistic properties of words. In *Proceedings of the 26th COLING*, pp. 1669–1679.

Paetzold, G., & Specia, L. (2017). Lexical simplification with neural ranking. In *Proceedings of the 15th EACL*, pp. 34–40.

Paetzold, G. H. (2013). *Um sistema de simplificação automática de textos escritos em inglês por meio de transduçao de árvores.* State University of Western Paraná.

Paetzold, G. H., & Specia, L. (2013). Text simplification as tree transduction. In *Proceedings of the 9th STIL*, pp. 116–125.

Paetzold, G. H., & Specia, L. (2016a). Semeval 2016 task 11: Complex word identification. In *Proceedings of the 10th SemEval*, pp. 560–569.

Paetzold, G. H., & Specia, L. (2016b). Understanding the lexical simplification needs of non-native speakers of english. In *Proceedings of the 26th COLING*, pp. 717–727.

Paetzold, G. H. (2015). Reliable lexical simplification for non-native speakers. In *Proceedings of the 2015 NAACL Student Research Workshop*, pp. 9–16.

Paetzold, G. H., & Specia, L. (2015). Lexenstein: A framework for lexical simplification. In *Proceedings of The 53rd ACL*, pp. 85–90.

Paetzold, G. H., & Specia, L. (2016a). Inferring psycholinguistic properties of words. In *Proceedings of the 2016 NAACL*, pp. 435–440.

Paetzold, G. H., & Specia, L. (2016b). Plumberr: An automatic error identification framework for lexical simplification. In *Proceedings of the 1st QATS*, pp. 1–9.

Paetzold, G. H., & Specia, L. (2016c). Sv000gg at semeval-2016 task 11: Heavy gauge complex word identification with system voting. In *Proceedings of the 10th SemEval*, pp. 969–974.

Paetzold, G. H., & Specia, L. (2016d). Unsupervised lexical simplification for non-native speakers. In *Proceedings of The 30th AAAI*, pp. 3761–3767.

Paetzold, G. H., & Specia, L. (2016e). Vicinity-driven paragraph and sentence alignment for comparable corpora. *CoRR*, *abs/1612.04113*.

Pavlick, E., & Callison-Burch, C. (2016). Simple ppdb: A paraphrase database for simplification. In *Proceedings of the 54th ACL*, pp. 143–148.

Pavlick, E., & Nenkova, A. (2015). Inducing lexical style properties for paraphrase and genre differentiation. In *Proceedings of the 2015 NAACL*, pp. 218–224.

Pavlick, E., Rastogi, P., Ganitkevitch, J., Van Durme, B., & Callison-Burch, C. (2015). Ppdb 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd ACL*, pp. 425–430.

Pennington, J., Socher, R., & Manning, C. D. (2014). Glove: Global vectors for word representation. In *Proceedings of the 2014 EMNLP*, pp. 1532–1543.

Petrov, S., & Klein, D. (2007). Improved inference for unlexicalized parsing. In *Proceedings of the 2007 NAACL*, pp. 404–411.

Porto (2017). PAPEL: Palavras Associadas Porto Editora - Linguateca. http://www.linguateca.pt/PAPEL. Accessed in: 2017-10-03.

Rello, L., Baeza-Yates, R., Bott, S., & Saggion, H. (2013a). Simplify or help?: text simplification strategies for people with dyslexia. In *Proceedings of the 10th W4A*, pp. 1–10.

Rello, L., Baeza-Yates, R., Dempere-Marco, L., & Saggion, H. (2013b). *Frequent Words Improve Readability and Short Words Improve Understandability for People with Dyslexia*, pp. 203–219. Springer Berlin Heidelberg, Berlin, Heidelberg.

Rello, L., Bautista, S., Baeza-Yates, R., Gervás, P., Hervás, R., & Saggion, H. (2013c). *One Half or 50%? An Eye-Tracking Study of Number Representation Readability*, pp. 229–245. Springer Berlin Heidelberg, Berlin, Heidelberg.

Ronzano, F., Abura'ed, A., Espinosa Anke, L., & Saggion, H. (2016). Taln at semeval-2016 task 11: Modelling complex words by contextual, lexical and semantic features. In *Proceedings of the 10th SemEval*, pp. 1011–1016.

Rudell, A. P. (1993). Frequency of word usage and perceived word difficulty: Ratings of kučera and francis words. *Behavior Research Methods, Instruments, & Computers*, *25*(4), 455–463.

Saggion, H., Štajner, S., Bott, S., Mille, S., Rello, L., & Drndarevic, B. (2015). Making it simplext: Implementation and evaluation of a text simplification system for spanish. *ACM Transactions on Accessible Computing*, *6*(4), 14.

Shardlow, M. (2013a). A comparison of techniques to automatically identify complex words. In *Proceedings of the 51st ACL Student Research Workshop*, pp. 103–109.

Shardlow, M. (2013b). The cw corpus: A new resource for evaluating the identification of complex words. In *Proceedings of the 2nd Workshop on Predicting and Improving Text Readability for Target Reader Populations*, pp. 69–77.

Shardlow, M. (2014a). Out in the open: Finding and categorising errors in the lexical simplification pipeline. In *Proceedings of the 9th LREC*, pp. 1583–1590.

Shardlow, M. (2014b). A survey of automated text simplification. *International Journal of Advanced Computer Science and Applications*, *4*(1), 58–70.

Siddharthan, A. (2014). A survey of research on text simplification. *ITL-International Journal of Applied Linguistics*, *165*(2), 259–298.

Sinha, R. (2012). Unt-simprank: Systems for lexical simplification ranking. In *Proceedings of the 1st SemEval*, pp. 493–496.

Sp, S., Kumar, A., & K P, S. (2016). Amritacen at semeval-2016 task 11: Complex word identification using word embedding. In *Proceedings of the 10th SemEval*, pp. 1022–1027.

Specia, L. (2010). Translating from complex to simplified sentences. In *Proceedings of the 9th PROPOR*, pp. 30–39.

Specia, L., Jauhar, S. K., & Mihalcea, R. (2012). Semeval-2012 task 1: English lexical simplification. In *Proceedings of the 1st SemEval*, pp. 347–355.

Thomas, S. R., & Anderson, S. (2012). Wordnet-based lexical simplification of a document. In *Proceedings of 2012 KONVENS*, pp. 80–88.

Toutanvoa, K., & Manning, C. (2000). Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the 2000 SIGDAT*, pp. 63–70.

UMBC (2017). UMBC WebBase Corpus. http://ebiquity.umbc.edu/resource/html/id/351. Accessed in: 2017-10-03.

Watanabe, W. M., Junior, A. C., Uzêda, V. R., Fortes, R. P. d. M., Pardo, T. A. S., & Aluísio, S. M. (2009). Facilita: reading assistance for low-literacy readers. In *Proceedings of the 27th ACM*, pp. 29–36.

Wikimedia (2017). Wiktionary, The Free Dictionary. http://en.wiktionary.org. Accessed in: 2017-10-03.

Wróbel, K. (2016). Plujagh at semeval-2016 task 11: Simple system for complex word identification. In *Proceedings of the 10th SemEval*, pp. 953–957.

Wubben, S., van den Bosch, A., & Krahmer, E. (2012). Sentence simplification by monolingual machine translation. In *Proceedings of the 50th ACL*, pp. 1015–1024.

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.

Yahoo (2017). The Yahoo BOSS Search API. https://developer.yahoo.com/boss/search. Accessed in: 2017-10-03.

Yatskar, M., Pang, B., Danescu-Niculescu-Mizil, C., & Lee, L. (2010). For the sake of simplicity: Unsupervised extraction of lexical simplifications from wikipedia. In *Proceedings of the 2010 NAACL*, pp. 365–368.

Yih, W.-T., & Qazvinian, V. (2012). Measuring word relatedness using heterogeneous vector space models. In *Proceedings of the 2012 NAACL*, pp. 616–620.

Zhang, X., & Lapata, M. (2017). Sentence simplification with deep reinforcement learning. In *Proceedings of the 2017 EMNLP*, pp. 595–605.

Zhang, Y., Ye, Z., Feng, Y., Zhao, D., & Yan, R. (2017). A constrained sequence-to-sequence neural model for sentence simplification. *CoRR*, *abs/1704.02312*.

Zhu, Z., Bernhard, D., & Gurevych, I. (2010). A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd COLING*, pp. 1353–1361.