Automated Text Simplification: A Survey

Article *in* ACM Computing Surveys · March 2021

DOI: 10.1145/3442695

2 authors:

Suha Althunayyan
Imam Mohammad ibn Saud Islamic University

Aqil Azmi
King Saud University

Automated text simplification: A survey

SUHA S. AL-THANYYAN and AQIL M. AZMI*, King Saud University, Saudi Arabia

Text simplification (TS) reduces the complexity of the text in order to improve its readability and understandability, while possibly retaining its original information content. Over time, TS has become an essential tool in helping those with low literacy levels, non-native learners, and those struggling with various types of reading comprehension problems. In addition, it is also used in a preprocessing stage to enhance other NLP tasks. This survey presents an extensive study of current research studies in the field of TS, as well as covering resources, corpora, and evaluation methods that have been used in those studies.

CCS Concepts: • **Computing methodologies** → **Information extraction**; **Lexical semantics**; *Natural language generation*; Machine translation.

Additional Key Words and Phrases: Text simplification, Lexical simplification, Syntactic simplification, Monolingual Machine Translation, Survey

ACM Reference Format:
Suha S. Al-Thanyyan and Aqil M. Azmi. 2021. Automated text simplification: A survey. *ACM Comput. Surv.* 39, 1, Article 1 (January 2021), 39 pages. https://doi.org/10.1145/1122445.1122456

1 INTRODUCTION

Automatic text simplification (text simplification for short, denoted TS) is the process of reducing the linguistic complexity of a text, so to improve its understandability and readability, while still maintaining its original information content and meaning. There are many reasons why such a task is needed. TS helps those with low literacy levels, second language readers, those suffering from various types of reading comprehension problems, as well as children. However, when simplifying text for a specific type of reader, text simplification should be defined to include sub-tasks that address specific characteristics of the text. These include elaborative modification; where redundancy and explicitness are used to emphasize key points, conceptual simplification; where the content is simplified along with its form, and text summarization for reducing text size by leaving out peripheral or inappropriate information [100]. The automation of this process is a difficult problem that has been explored from different angles since the late 90s. The growth of research in this area follows the rapid growth in statistical, machine learning, natural language processing (NLP), and software techniques. TS is an active research area, which goes on to show we are far from a satisfactory solution. This should not be a surprise. Healthy research on text summarization is ongoing for at least half a century.

TS commonly focuses on two tasks, which are lexical simplification and syntactic simplification. Lexical simplification attempts to identify and replace complex words with simpler synonyms.

*Corresponding author. Tel: +966 (0)11 467-6574, Fax: +966 (0)11 467-5423.

Authors' address: Suha S. Al-Thanyyan, sthunyyan@imamu.edu.sa; Aqil M. Azmi, aqil@ksu.edu.sa, Department of Computer Science, College of Computer & Information Sciences, King Saud University, Riyadh, 11543, Saudi Arabia.

0360-0300/2021/1-ART1 $15.00
https://doi.org/10.1145/1122445.1122456

Syntactic simplification tries to simplify the grammatical complexity by identifying complicated syntactic structures such as coordination, subordination, relative clauses, and passive relative clauses, which may be difficult to read or understand by certain readers.

One of the essential points in text simplification is identifying the complexity of the text, which is hard to define and differs from one user to another [62]. However, the identification of complexity will help in deciding whether the text should undergo simplification or not. It will also prove useful in evaluating the output produced by the simplification system as well as comparing different systems in terms of simplicity or complexity of the output.

Within the field of NLP, TS is related to other techniques such as paraphrase generation, text summarization, and machine translation. Many of the techniques and evaluation methods used within TS are driven from those fields. Historically, automatic text simplification started as a preprocessing step to improve other NLP tasks such as parsing [20], question generation [54], information extraction [41, 75], facts retrieving [63], and semantic role labeling [119] in that a rule-based syntactic simplification was applied. TS has also shown great enhancement in the summarization task. For example, Lal and Rüger [68] applies lexical simplification (replacing complex words with simpler synonyms) using a technique similar to the one proposed by [18] at the summary generation step, [105] uses syntactic simplification to improve sentence selection in multi-document summarization, and [106] uses sentence simplification at summary generation phase in order to produce a simpler and highly informative summary. Currently, there are several applications of TS in medical researches, including simplifying medical literature using lexical and syntactic simplifications [80], simplifying drug package leaflets through lexical simplification [96], and simplifying patent documents [15, 17, 44, 89].

Automatic simplification helps people with low literacy levels, such as children and non-native speakers [86]. In addition to that, people suffering from different kinds of reading comprehension, e.g. autism [42], aphasia [18], dyslexia [91], and deaf people [58] are known to benefit from TS.

Around 10% of the population has dyslexia [91]. Dyslexia is a neurologically-based condition that affects word-level reading accuracy, reading fluency, and spelling. According to the International Dyslexia Association (IDA), "Dyslexia is a specific learning disability that is neurobiological in origin. It is characterized by difficulties with accurate and/or fluent word recognition and by poor spelling and decoding abilities."[1] Previous studies [91, 92] found that long and less frequent words (complex words) affect negatively the text readability and understandability for people with dyslexia. Therefore, applying lexical simplification, that substitute complex words with shorter and more frequent synonyms would help people with dyslexia. On the other side, people with Autistic Spectrum Disorders (ASD) have difficulty inferring contextual information as well as understanding long sentences with complex syntactic structures [42]. These difficulties could be addressed using syntactic simplification strategies that try to simplify the complex phonemes in the text.

Saggion [93] is still the most complete available survey of TS. However, the field of TS has changed during the last few years with the emergence of the successful application of deep learning techniques. None of these were covered in [93]. Unlike other surveys (e.g., [93, 100, 103]), this survey identifies and classifies automated TS research within the period 1998–2019. It presents in detail a large scale of recent studies in machine translation (MT)-based text simplification. It also reviews the neural MT-based text simplification systems, which are not covered in [93, 100, 103]. Moreover, it presents a large set of lexical resources and parallel corpora that have been employed in the TS literature, covered by different languages.

This report is organized as follows. Section 2 lists all the lexical resources and corpora that are used for TS. Section 3, provides an overview of different text simplification evaluation methods. An

[1]Based on the 2002 definition by the IDA, https://www.idaontario.com/definition-of-dyslexia/.

extensive study of the different automatic text simplification approaches is provided in Section 4. Finally, a general conclusion is presented in Section 5.

2 AVAILABLE RESOURCES AND CORPORA

Lexical resources and corpora play an important role in the development and evaluation of simplification systems. In this section, we present a list of lexical resources and parallel corpora that have been generally employed in TS literature covering different languages.

2.1 Lexical resources

We start by covering the lexical resources. Table 1 lists the resources, and the language it is in. Most of the resources are in English but there are other languages as well, e.g. Spanish. Whenever possible we provide the link to the resource.

Table 1. List of lexical resources.

Lexical resources	Language	URL	Number entries	Generation process
SemEval-2012 [111]	English	www.cs.york.ac.uk/semeval-2012/task1	2,010 contexts	Dataset derived from the English Lexical Substitution Task of SemEval 2007 [71]. It covers 210 target words that include nouns, verbs, adverbs, and adjectives. Each word appears in 10 different contexts. The dataset contains simplicity rankings given by non-native English speakers.
LSeval [31]	English	people.cs.kuleuven.be/~jan.debelder/lseval.zip	430 sentences	Sentences are sorted by their difficulty, through 46 Mechanical Turks (MTurk), and 9 different PhD students. The dataset was originally produced from the Lexical Substitution Task of SemEval 2007. From this dataset, the authors removed the words which were listed as easy words. The list of easy words were produced by combining the Basic English words list from Simple English Wikipedia (SEW),[2] and the list of easy words from the Dale-Chall readability measure [30].
CW corpus [98]	English	tinyurl.com/cwcorpus	731 sentences	The sentences are mined from SEW edit histories, each with one annotated CW. To keep the corpus balanced, negative examples (i.e., examples of simple words only) are provided by a word picked at random from the sentence in which the CW occurs.
LexMTurk [56]	English	www.cs.pomona.edu/~dkauchak/simplification/lex.mturk.14	500 sentences	The sentences were randomly selected from the aligned corpus of English Wikipedia (EW) with SEW. For each sentence in the dataset, 50 Mechanical Turks were used to provide simpler substitutions for the target complex words.

… *Continued on next page*

[2]https://simple.wikipedia.org/wiki/Wikipedia:Basic_English_combined_wordlist.

Table 1 (continued)

Lexical resources	Language	URL	Number entries	Generation process
BenchLS [82]	English	ghpaetzold.github.io/data/BenchLS.zip	929 instances	Created by combining two datasets, namely LexMTurk and LSeval, which were automatically corrected for misspelling and inflection errors. Each instance contains a sentence, a complex word, and 7 substitutions that were ranked based on their simplicity by English speakers.
NNSeval [86]	English	ghpaetzold.github.io/data/NNSeval.zip	939 instances	Created by filtering the corrected versions of LexMTurk and LSeval datasets from all candidate synonyms, which were considered complex by a non-native speaker. Instances that contained target word—not deemed complex by any non-native speakers—were also discarded.
FLELex [48]	French	cental.uclouvain.be/cefrlex/flelex/download	777,000 words	Was obtained from available textbooks and simplified readers aimed at learners of French as a Foreign Language. It reports the words (lemmas) normalized frequencies across each level of the CEFR (Common European Framework of Reference for Languages).
SNOW E4 [60]	Japanese	www.jnlp.org/SNOW	2,500 instances	Extracted from a newswire corpus. Candidate substitutions were provided and ranked by a set of annotators using crowdsourcing service.
BCCWJ [65]	Japanese	github.com/KodairaTomonori/EvaluationDataset	2,100 instances	Candidate substitutions were provided and ranked using crowdsourcing service and by computer science students. BCCWJ dataset overcomes the limitations of the SNOW E4 dataset, in which the sentences were extracted from a balanced corpus, and tie candidates were allowed in simplicity rankings.
WaCKy [24]	German	www.informatik.tu-darmstadt.de/ukp/research_6/data/lexical_substitution/glass	2,040 words (includes 153 target words)	The words' selection was ordered according to their frequencies in a large German corpus. Candidate substitutions were provided by German native speakers using crowdsourcing service.
LexSubNC [122]	Portuguese	pageperso.lif.univ-mrs.fr/~carlos.ramisch	1,500 substitutes	The substitutions were manually validated substitutes for 180 Portuguese nominal compounds. They are classified according to one of three types: synonym, near-synonym (such as hypernyms, hyponyms, and meronyms), and paraphrase or definition.
ReSyf [8]	French	cental.uclouvain.be/resyf	121,182 synonyms	The synonyms were extracted from the lexical network JeuxDeMots [67] and then semantically disambiguated and ranked based on their reading difficulty for French learners.
PPDB-S [113]	Spanish	Available through author	5,709	Built by filtering and ordering paraphrases pairs from the paraphrases database (PPDB) [50]. The PPDB-S dataset has a small number of paraphrases that are more likely to have the same meaning (i.e. low coverage and high precision).

… *Continued on next page*

Table 1 (continued)

Lexical resources	Language	URL	Number entries	Generation process
PPDB-M [113]	Spanish	Available through author	15,524	Generated in the same way as PPDB-S, but it has higher coverage and lower precision than the PPDB-S dataset.
Synonyms from Spanish OT [113]	Spanish	Available through author	21,635	Synonyms were extracted from the Spanish Open Thesaurus (OT) by filtering out multiple senses words and ordering them either based on their frequencies in a large corpus, or on their lengths.
Synonyms from Spanish EuroWordNet [113]	Spanish	Available through author	13,970	Synonyms were extracted from the Spanish EuroWordNet [120] in the same way as OT.
CASSA [113]	Spanish	Available through author	5,640,694	Generated by extracting all unique 5-grams pairs from CASSA resource [6] where the target word not in infinitive.

2.2 Parallel corpora

A parallel corpus is one that contains a collection of complex texts in some languages and their simplification in the same language. In this section, we present a list of parallel corpora that have been used in the TS literature in table form (Table 2 lists the resources).

Table 2. List of parallel corpora.

Parallel corpora	Language	URL	# Aligned-pairs	Generation process
EW-SEW [28]	English	www.cs.pomona.edu/~dkauchak/simplification	137,000	It was generated by pairing documents and sentences from EW (English Wikipedia) with corresponding documents and sentences from SEW (Simple English Wikipedia). The data covers the main simplification operations: reordering, inserting and deleting.
Parallel Wikipedia Simplification Corpus [132]	English	www.ukp.tu-darmstadt.de/data/sentence-simplification	108,016	It was extracted from 65,133 articles in EW and SEW. The dataset is aligned using the sentence-level TF-IDF similarity measure.
SS Corpus [59]	English	github.com/tmu-nlp/sscorpus	492,993	Extracted from 126,725 article pairs obtained by aligning articles from EW and SEW by an exact match of titles.
Newsela [84]	English	newsela.com/data	10,787	Contains news articles in English simplified to different reading levels by human experts.
On-eStopEnglish [118]	English	zenodo.org/record/1219041	Up tp 3,154	Consists of 189 English texts, each in three different reading levels: elementary (ELE), intermediate (INT), and advanced (ADV). It has 1,674, 2,166, and 3,154 sentence-aligned pairs for ELE-INT, ELE-ADV, and INT-ADV respectively.

... *Continued on next page*

Table 2 (continued)

Parallel corpora	Language	URL	# Aligned-pairs	Generation process
Simplext [94]	Spanish	Available through author	200 news texts	The parallel corpus contains news from four different domains, covering national, international, culture, and society news.
SIMPITIKI	Italian	github.com/dhfbk/simpitiki	1,166	Composed of two sets of simplified pairs: (a) those extracted in a semi-automatic way from the Italian Wikipedia revision history, and (b) manually created sentence-by-sentence from documents belonging to the administrative domain.
PaCCSSIT [16]	Italian	www.italianlp.it/software-data/text-simplification	63,000	Automatically produced from a large raw corpus.
Alector [49]	French	Available on demand	79 texts and their simplified equivalent	It is extracted from authentic literary and scientific texts that were commonly used for students in French primary schools. The texts were manually simplified by experts at different linguistic levels: morpho-syntactic, lexical, and discourse levels.

3 EVALUATION METHODS

Various evaluation methods have been proposed to judge the quality of the output of text simplification systems as well as to compare the performance of different simplification systems. These methods can be regarded as either manual or automatic. Most often, the TS studies combine manual and automatic methods for output evaluation. In this section, we describe the different metrics that have been used in manual and automatic evaluation methods.

3.1 Manual evaluation

Most of the TS systems use human experts to perform a sentence-level evaluation. The assessment are usually based on three aspects: simplicity, fluency (grammaticality), and adequacy (or meaning preservation), see e.g. , [9, 52, 110, 112, 124, 125]. Simplicity measures how simple the simplified sentence is, while grammaticality measures the grammatical correctness of the simplified sentence. Meaning preservation measures how well the original meaning is preserved following simplification. Usually, these three aspects are scored on a Likert scale, with a 1-5 scale or 1-3 scale, where the higher score denotes better simplification.

The manual evaluation suffers from a few limitations. It requires native speakers with linguistic knowledge, in order to evaluate the three aspects of the output sentence. In addition, humans are inconsistent and they differ from one another. This makes the comparison between different TS systems inaccurate, especially when different humans are involved. Moreover, manual evaluation is expensive and time-consuming. These shortcomings encourage the TS researchers to explore automatic methods to evaluate the output.

3.2 Automatic evaluation

One of the automatic ways to evaluate the TS systems is through readability indices, see e.g. [101, 112, 125, 130, 132]. The readability indices are one means used to estimate how difficult a text to read [101]. It is worth noting that most of these indices are empirical in nature. Some of the common parameters used by the readability indices are *ASL* (average length of the sentence), and *ASW* (the average number of syllables in a word). *ASL* is simply the ratio of the total number of words by the number of sentences. The following are the most widely used readability indices in the English language.

The Flesch Reading Ease (*FRE*) score [132]. Here, the higher score indicates the text is easier to read. This score combines *ASL* and *ASW*, and is given by,

$$FRE = 206.835 - (1.015 \times ASL) - (84.6 \times ASW). \tag{1}$$

The Fog Index (*FOG*) score combines *ASL* and *ACW* (average number of complex words in textual fragments containing 100 words), where complex words are those with more than two syllables. Lower *FOG* score indicates text that is easier to read. *FOG*'s formula is,

$$FOG = 0.4 \cdot [ASL + 100 \times ACW]\,. \tag{2}$$

The *SMOG* grading score is similar to *FOG*. Here also, the lower score indicates that the text is easier to read. This score considers only the average number of polysyllabic words (words with 2+ syllables) in 30-sentences-long textual segments. It is calculated by,

$$SMOG = 3 + \sqrt{\text{polysyllable\_count}}. \tag{3}$$

Flesch-Kincaid Grade Level index (*FKGL*) [62], in which lower score indicates text that is easier to read. The formula combines *ASL* and *ASW*,

$$FKGL = (0.39 \times ASL) + (11.8 \times ASW) - 15.59. \tag{4}$$

In the past few years, several studies have addressed the task of text simplification as a monolingual machine translation problem. As a result, they adopted the machine translation evaluation metrics in evaluating TS systems, e.g. [76, 110, 125, 130].

Bilingual evaluation understudy (*BLEU*) [87], measures the overlapping of n-grams between a gold standard reference and system simplifications. *BLEU* penalizes heavily sentence shortening and word reordering. Let A and R respectively be the automatic and reference texts. The formula for *BLEU* is given by,

$$\begin{aligned}
BLEU &= F(A,R) \cdot \exp\left(\sum_{i=1}^{n} w_i \cdot \log\left(\frac{|A_i \bigcap R_i|}{|A_i|}\right)\right), \\
F(A,R) &= \begin{cases} 1, & \text{if } |A| > |R|, \\ \exp(1 - n/|A|), & \text{otherwise,} \end{cases} \\
w_i &= \frac{i}{\sum_{j=1}^{n} j},
\end{aligned} \tag{5}$$

where n is the size of n-gram, and A_i and R_i are the bags of i-gram for automatic and reference text.

The *NIST* (after National Institute of Standards and Technology) [38] is a metric based on *BLEU*. It is a method for evaluating the quality of the text that has been translated using machine translation. *NIST* measures the overlapping of n-grams between human reference and system simplifications. However, in NIST different n-grams obtain different weights. *NIST*'s score is calculated by,

$$\sum_{i=1}^{N} \left\{ \frac{\sum_{\text{all } w_1 \cdots w_i \text{ that co-occur}} Info(w_1 \cdots w_i)}{\sum_{\text{all } w_1 \cdots w_i \text{ in sysoutput}} (1)} \right\} \cdot \exp\left\{ \beta \log^2 \left[\min\left(\frac{L_{\text{sys}}}{\bar{L}_{\text{ref}}}, 1 \right) \right] \right\}, \tag{6}$$

where,

$$Info(w_1 \cdots w_i) = \log_2 \left(\frac{\#\text{ occurrences of } w_1 \cdots w_{i-1}}{\#\text{ occurrences of } w_1 \cdots w_i} \right), \tag{7}$$

usually $N = 5$ and, β is selected to make the brevity penalty factor = 0.5 when the number of words in the output of the system is two thirds the average of the number of words in the reference translation. The $\bar{L}_{\text{ref}}$ is the average number of words in a reference translation, averaged over all reference translations; while L_{sys} is the number of words in the translation that is being scored.

Translation Edit Rate-plus (*TERp*) [109], extends the Translation Edit Rate (*TER*) metric [108] by incorporating morphology, synonymy and phrasal substitutions. *TERp* measures the number of edits required to transform the simplified text into the original text. The higher *TERp* score denotes the less similarity between the output and the original text.

In addition to the aforementioned metrics, [127] proposed new simplification-specific metrics to evaluate the TS system's output. They argued *BLEU* is insufficient for evaluating a simplification model based on empirical study. Instead, they introduced two new metrics *FKBLEU*, and *SARI*. *FKBLEU* is a geometric mean of the *iBLEU* [115] (for paraphrase generation); and *FKGL* index of the difference between original and output sentences. Whereas *SARI* metric compares system output against both reference and input sentences, in order to measure the effectiveness of word insertion and deletion operations. These evaluation metrics have been used in [78, 114, 131].

4 SIMPLIFICATION APPROACHES

There is a considerable research body on text simplification, as evident by the great interest shown by the research community towards this topic. However, we are far from reaching a saturation stage. The automatic text simplification approach is classified into one of four main categories: lexical, syntactic, monolingual machine translation, and hybrid techniques. In this section, we focus on the recent studies in the field of automated TS besides providing the reader with sufficient historical depth of the knowledge of the different TS approaches.

4.1 Lexical approach

Lexical simplification (LS) is the technique that aims to reduce text complexity by identifying and substituting complex words with simpler, more understandable, synonyms without simplifying the syntax of the text. In addition, it may be carried out at the phrase level, where syntactic information is taken into account. Typically, this is a four-step process [100]: (1) complex word identification to identify the complex terms in a document, (2) substitutions generation to produce a list of substitutions for each one, (3) substitutions selection to refine those substitutions to keep the most appropriate synonyms for the given context, and (4) substitutions ranking to rank the remaining substitution according to their simplicity. Figure 1 illustrates the lexical simplification pipeline. Researches on LS can be divided into two approaches, rule-based and data-driven. The rule-based approach is the oldest in TS but is still used for languages where large parallel corpora do not exist in order to allow for a data-driven approach.

4.1.1 Rule-based lexical simplification. The first LS system was proposed in 1998 [18] which simplifies English newspaper texts to assist aphasic readers.[3] The system is composed of an analyzer and a simplifier. The analyzer provides syntactic analysis, while the simplifier component adjusts the output of the analyzer to aid readability for aphasic people. The analyzer consists of three subcomponents: a lexical tagger, a morphological analyzer, and a parser. After linguistically analyzing the text, the simplifier takes place. The simplifier consists of two subcomponents: a lexical simplifier and a syntactic simplifier. The lexical simplifier takes each word from the analyzed text

[3]It is the inability to comprehend language due to damage to specific brain regions.

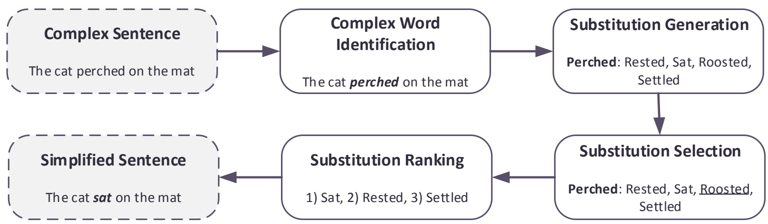


Fig. 1. The lexical simplification pipeline (reproduced) [100].

and generates a list of synonyms using WordNet [74], and extracts the frequencies of synonyms based on the Oxford Psycholinguistic Database. Then the word with the highest frequency is selected and written in the output file. We will further divide rule-based LS into two subcategories, as follows.

Identifying complex words in lexical simplification: Shardlow [99] implemented and evaluated the LS pipeline proposed in [18] in order to investigate the types of errors that are predominant in that scheme. The author followed the same procedure except for the way in which the complex words are identified, which was set to words with a Kucera-Francis frequency below five. They exposed 6 distinct categories of errors and proposed a classification scheme for them. For testing, a corpus was created as a set of 115 news articles from various topics. In total, 184 LS operations were identified out of which 164 resulted in some form of errors. The most frequent type of error was the identification of a word incorrectly as complex. This error caused by the hyphenated multi-word expressions that were not found in the Kucera-Francis frequencies or in WordNet, and so they were assigned zero frequency and no substitutions.

With the aim to develop a simplification system for Spanish, [39] presented the results of the analysis of lexical changes in a parallel corpus of original and simplified texts in this language. The corpus consists of 200 informative texts obtained from news agency Servimedia. The corpus was manually simplified by a trained human editor. Different lexical operations have been observed, which are placed in various categories applied at both word and sentence levels. The operations include lexical unit substitution, difficult terms definitions insertion, numerical expressions simplification, named entities simplification, and rewording operation. Upon examining the data, the authors found that word frequency and word length are good signs for word difficulty and important factors that should be considered when selecting a synonym to replace a complex word. However, based on their data observation, they have derived a set of simplification rules concerning reporting verbs, adjectives of nationality, named entities, and numerical expressions.

Shardlow [97] compared the performance of three solutions to complex word identification in English: "simplify everything" approach, threshold-based approach, and Support Vector Machine (SVM). The first method is the one that used in [18] which involves applying the simplification algorithm to every word. The second method is frequency thresholding in which a word whose frequency is below a threshold is considered CW (i.e. a complex word). This was learned from a corpus by examining every threshold for a training dataset using 5-fold cross-validation. The third method is using SVM to train the algorithm using only word features (e.g., word frequency, word length, syllable count). These methods are tested using the CW corpus [98]. The result shows all the methods achieve relatively high accuracy.

Context-aware lexical simplification: a major difficulty with primitive lexical substitution models is the loss of meaning and cohesion of the resultant sentence, and this is attributed to word sense ambiguity. It occurs when a word has several meanings and it is difficult to determine which one is relevant. Various word sense disambiguation techniques have been used. LexSiS (stands for, lexical simplification system for Spanish) [13] attempted to enhance the performance of LS by using context vectors. LexSiS is based on a freely-available synonym dictionary and the Web as a corpus. LexSiS finds the best candidate (a word lemma) as a replacement for every word, that is in a lexical database, in two stages. The system attempts to locate an appropriate set of synonyms for a given word, for which it attempts to find the best substitution candidate within this set. This system uses the Spanish Open Thesaurus lexical database, which lists over 21 thousand target words (lemmas) and provides a list of substitution sets (word senses) for each word.

In the first step, a word vector is extracted for each lemma in the text, which represents co-occurring lemmas in a window of size 9 words (four lemmas to the left and the same to the right). This vector is compared to all sense vectors for each of the word senses listed in the thesaurus. Then, it selects the word sense with the lowest cosine distance to the context vector. In the second step, they pick the best candidate within the selected word sense. For this, they use a simplicity measure as a function of word length and word frequency. Moreover, some thresholds are applied in order to remove candidates that are either not much simpler or do not fit into the context. There are some cases where LexSiS does not suggest a substitute. First, the cases where the target word does not exist in the LexSiS dictionary; and the cases where the target word and its substitution are the same. To evaluate LexSiS they compared it with a gold standard and two baselines: random baseline and frequency baseline (similar to [18]) using human informants. The results show that LexSiS performs better than the frequency-based method.

Ferrés et al. [46] developed an adaptable LS approach for the major Ibero-Romance languages: Portuguese, Galician, Spanish, and Catalan. The simplifier is composed of five phases: document analysis, complex word detection, word sense disambiguation, synonyms ranking, and language realization. In the document analysis stage, the linguistic features from the texts are extracted using FreeLing 3.0 system [81]. In order to identify complex words, a generic method is used that relies on the frequency thresholding method over corpus-based frequency lists. Similar to [9], the vector-space model is used to obtain the most appropriate synonyms in a given context. However, a word frequency simplicity measure is applied to rank the synonyms. The last phase is the language realization in that the right inflected form of the final selected substitution is generated through a hybrid morphological generator that combines lexicon-based generation and decision-trees based algorithm with an adapt rule-based context module. For the evaluation, seven proficient humans are asked to assess the adequacy and the simplicity of the system. The results show that the corpus-based approaches are not sufficient to deal with the difficulty of the simplification task.

Qiang et al. [90] presented an LS system that overcomes the limitations of existing LS systems. Here, the complex word substitutions are generated based on the complex word itself rather than the context of CW. The proposed approach extends the BERT (for, Bidirectional Encoder Representation form Transformer) model [35], to generate and rank the candidates for a complex word. The BERT model is trained on masked language modeling (MLM), which predicts a word based on its right and left context. Given a complex word, BERT will generate simplification candidates. Then, based on several features such as BERT prediction, language model features, frequency feature, and semantic similarity, BERT-LS ranks all the candidates and then selects the one with the highest rank as simplification candidate. Empirical results show that the BERT-LS model outperforms the baselines that rely on parallel corpus. One of the limitations of the proposed model is that it produced only a single-word candidate for CW.

4.1.2 Data-driven lexical simplification. In the data-driven approach, scientific methods and algorithms are used to extract knowledge from large datasets. It incorporates techniques from different fields, including computer science, mathematics, and statistics, to understand and analyze data [53]. During the last few years, the availability of English Wikipedia (EW) and Simple English Wikipedia (SEW) [28] encouraged the TS researchers to employ data-driven approaches in text simplification. The data-driven lexical simplification employs machine learning techniques to learn LS rules from the parallel corpus [93].

Yatskar et al. [129] used SEW edit histories to learn lexical simplifications. Since edit histories include different types of operations (e.g. spam, correct, or simplify) they identify two simplification approaches. First, a probabilistic model is used to accounts for this mixture of different edit operations. In this model, they identify which phrase w from edit history eh_i have been replaced with the aim of making edit history eh_{i+1} simpler than eh_i. Second, the Wiki editors' metadata are used to identify trusted revisions in that the extracted lexical edit pairs ($W \to w$) are likely to be a simplification. For evaluation, the top 100 substitution pairs ($W \to w$) from the proposed systems and two baselines, RANDOM and FREQUENT in addition to 100 randomly selected pairs from a human-made dictionary were presented to three native and three non-native English speakers. The results show that the proposed approach outperformed the baselines in terms of precision but worse than the dictionary.

Biran et al. [9] also used EW and SEW to learn simplification rules but without using information from SEW edit histories. The proposed system is composed of two phases: rule extraction and sentence simplification. In phase one, ordered words pairs of the form {*orignal* → *simple*} are extracted from the corpora along with a similarity score between the words, which computed using their context vectors. Moreover, in order to ensure that extracted pairs represent a complex-simple pair, they used two measures: corpus, and lexical-complexities. The corpus complexity of a word is defined as the ratio of its frequency in EW and SEW. The multiply of this value with the word length estimates its difficulty. As a result, the system discards the word pairs in which the second word's complexity higher than that of the first. In the simplification phase, the system decides which words in a sentence require simplification depending on the rules learned in the first phase and on contextual information of the input sentence. A simplification example is shown in Table 3, in which the word magnate determined as a simplification candidate. Two rules are available and based on the context the second rule is selected. The experiment results show that the proposed approach better than the frequency-based baseline [36] in terms of meaning preservation, grammaticality, and simplicity.

Table 3. An example of simplification.

Input	"In 1900, Omaha was the center of a national uproar over the kidnapping of Edward Cudahy, Jr., the son of a local meatpacking **magnate**."
Candidate rules	{magnate → king} {magnate → businessman}
Output	"In 1900, Omaha was the center of a national uproar over the kidnapping of Edward Cudahy, Jr., the son of a local meatpacking **businessman**."

Unlike [9, 129], Horn et al. [56] learned simplification rules by aligning EW with SEW using GIZA++ [79]. To select the best candidate in a given context, they learned a feature-based ranker by using SVM. They trained the model using human-labeled data that was collected using Amazon's Mechanical Turk (MTruk). Candidates are represented using several features including, candidate alignment probability, word frequency, language model, and context frequency. The proposed simplifier was compared to two existing approaches: (a) frequency-based system, and (b) the rules in [9] with the proposed ranking algorithm. Based on three metrics; precision, the percentage of phrases changed by the model, and accuracy, the proposed system achieved the best results.

Word embedding is a technique used to represent the words of the document, where each word—in the vocabulary—is represented by a real-valued vector [88]. In this representation, words with similar meaning will have a similar representation. Acknowledging that parallel corpus produces limited coverage of complex words, and the fact that "simple" words are also present in a regular text, [52] presented a simplification approach, called LIGHT-LS, that relies on word vector representations to find simpler synonyms for complex words. The proposed method employs GloVe [88], a state-of-the-art tool of distributional lexical semantics to extract vector representations for all words in the corpus. The GloVe vectors pre-trained on the merge of EW and Gigaword 5 corpus. For each word w, the top 10 words whose vectors are most closer to word w in terms of cosine similarity, are selected as simplification candidates. Then the simplification candidates rank according to several suitability-features: semantic similarity between original word and candidate, context similarity (the average semantic similarity with the context of original word), the difference of information content between the candidate and the original word, and language model likelihood. A simplification example is shown in Table 4.

To assess the performance of the system they evaluate it automatically and manually. First, the evaluation performed on the LetMturk dataset [56] with the metrics precision, percent of change, and accuracy. LIGHT-LS outperforms the other system in terms of accuracy and changed. Next, LIGHT-LS evaluated on the corpus that was produced by the SemEval lexical simplification task [83], LIGHT-LS perform better than the best system in the task [85]. Regarding the human assessment, the system obtained the lowest grade for meaning preservation than the evaluated systems. This was due to the LIGHT-LS system unable to discern synonyms from antonyms. This limitation due to representing all word's meanings by a single vector without considering lexical resources.

Table 4. A simplification example.

System	Sentence
Original sentence	"The contrast between a high level of education and a low level of political rights was particularly great in Aarau, and the city refused to send troops to defend the Bernese border."
Simplification by [9]	"The **separate** between a high level of education and a low level of political rights was particularly great in Aarau, and the city refused to send troops to defend the Bernese border."
LIGHT-LS simplification	"The contrast between a high level of education and a low level of political rights was **especially** great in Aarau, and the city **asked** to send troops to **protect** the Bernese border."

Paetzold and Specia [86] proposed a new context-aware model for word embedding to generate substitution for complex words. The model is trained over a corpus that annotated with generalized POS tags, which are nouns, verbs, adverbs, and adjectives. Given a target word's POS tag and an embedding model, the generation algorithm would extract the n candidates with the shortest cosine distance from the target word. The candidates must have the same POS tag as the target word and not be a morphological variant. To decide which of the candidates could replace the target word, they proposed a boundary ranking approach in which a decision boundary between negative and positive training examples will be learned from a binary classification setup. For the task of substitution ranking, the candidates ranked based on *n*-gram frequencies extracted from their own movie subtitles corpus (SubIMDB). The authors argue that frequencies extracted from movie subtitles are effectively captured word familiarity more than other corpora. The experiments show that the proposed approach obtained better performance than several state-of-the-art systems.

Though the embedding model in [86] is able to capture synonyms, it also allows antonyms to be among generated candidates; for instance, the word *large* will be similar to *larger* and *sizeable* but also to *small*. Thus, Paetzold and Specia [84] enhanced the embedding model by employing the lexicon retrofitting algorithm proposed by [43], in which the target word will have a closer distance to words that share a semantic relation with it as synonymy, hyponymy, and hypernymy. To retrofit the model, they created sets of universal-POS tags annotated synonym relations from WordNet. For instance, for the word *travel*, they create the set (*travel*/V, *journey*/V, *go*/V, *locomote*/V, *trip*/V). Once the synonym entries produced, a retrofitted context-aware model is trained using Word2vec toolkit [73] on a corpus of over 7 billion words. Besides the retrofitted model they also employed the Newsela corpus for substitution generation task. The Newsela corpus is composed of 1911 original news articles as well as up to five versions simplified by professionals. For all versions of a given article, they produced the sentence alignments based on TF-IDF similarity between them. Then the Stanford Tagger [117] is used to tag sentences, and Meteor [34] to produce word alignments. To generate candidates they used an approach similar to the one proposed by [56]. In order to rank candidates, they used a neural regression model that learned from the LexMturk dataset [56]. To assess the performance of the proposed LS approach, they compare it with many competitive state-of-the-art methods such as those by [9, 52, 56, 86] using two standard evaluation datasets. The results show that the proposed approach outperforms all other systems in terms of precision and F1. Table 5 summarizes the surveyed lexical simplification systems.

4.2 Syntactic approach

Syntactic simplification (SS) is the task of simplifying the complex syntactic structures in a text while preserving its information content and original meaning. There are several types of syntactic structures that may be considered as complicated such as coordination, subordination, relative clauses, and passive relative clauses. Syntactic simplification is mostly done in three stages [100]:

- Analyze the text to identify its structure and parse tree. Here, words and phrases are grouped together into 'super-tags', which represent a chunk of the underlying sentence. In order to provide a structured version of the text, we can join the 'super-tags' together with conventional grammar rules. At the analysis phase, the sentence's complexity is determined, which decides if it requires simplification. This process can be automated through matching rules, or through a binary classifier such as SVM.
- Transformation phase, where modifications are made to the parse tree according to a set of rewrite rules. These rules perform the simplification operations, e.g. sentence splitting, clause rearrangement, and clause dropping.
- Regeneration phase. Here, the text undergoes further modifications to improve cohesion, relevance, and readability.

Two main approaches have been considered in SS studies: rule-based and data-driven approaches. Most of the syntactic simplification approaches are rule-based. The performance of such systems mainly relies on linguistic expertise and accurate analyzing tools (parsers and taggers).

Table 5. Main characteristics of the surveyed lexical simplification systems. In the Evaluation column, 'M' stands for manual evaluation, and 'A' for automatic.

Approach	Year	Ref.	Language	Eval	Methodology
Rule-based	1998	[18]	English	M	Based on word frequency to measure word simplicity and word complexity
Rule-based	2012	[99]	English	M	The selection of complex words relies on Kucera-Francis frequency
Rule-based	2014	[13]	Spanish	M	Define the simplicity measure as a function of word length, and word frequency
Rule-based	2017	[46]	Portuguese, Galician, Spanish, Catalan	M	Combine lexicon-based generation and decision trees based algorithm with an adapt rule-based context module
Rule-driver	2019	[90]	English	A	Extend the BERT model to generate substitutions considering both the CW and the context of the CW
Data-driven	2010	[129]	English	M	Use SEW edit histories to learn lexical simplifications
Data-driven	2011	[9]	English	M	Rely on context similarity to extract simplification rules
Data-driven	2014	[56]	English	A	Learn simplification rules by using aligned corpus of EW with SEW and train SVM for substitutions ranking
Data-driven	2015	[52]	English	M+A	Based on word vector representations. Requires only regular corpora
Data-driven	2016	[86]	English	A	Generate substitution by joint modeling words and POS tags, and use boundary ranking for substitutions ranking
Data-driven	2017	[84]	English	A	Extract candidates by combining the Newsela corpus with a retrofitted word modeling. Use neural network for substitutions ranking

4.2.1 Rule-based approach. The first handcrafting rule-based approach to syntactic simplification is introduced in 1996 by [20]. Their motivation for text simplification was mainly to reduce sentence length as a preprocessing step for a parser. They described text simplification as a two-stage process: (a) analysis which provides a structural representation for a sentence; and (b) transformation in which sequences of rules are applied to identify the units that can be simplified. Their research focused on constructions such as relative clauses and appositives and separating out coordinated clauses. Their first approach was to handcraft simplification rules. Consider the rule: `X:NP, RELPRON Y, Z.` $\longrightarrow$ `X:NP Z. X:NP Y`. It can be read as: "if a sentence starts with a noun phrase `X`, and followed by a relative pronoun, of the form `RELPRON Y` followed by `Z`, where `Y` and `Z` are sequences of words, then the embedded clause can be simplified into two sentences, namely the sequence `X` followed by `Z`, and `X` followed by `Y`". For example, the sentence "The cyclist, who won the race, trained hard." becomes "The cyclist trained hard. The cyclist won the race." when simplifying the relative clause using the transformation rule.

In practice, the system does not work very well in all cases. It suffers from several weaknesses caused mostly by the relatively simple mechanisms used to detect phrases and attachments. Sentences that include long distances or crossed dependencies or sentences with ambiguous are not handled properly. For instance, to simplify a sentence such as: "A man from London, who won the race, trained hard, usually on Mondays." It is necessary to decide whether the relative clause attaches to “man” or “London”, and whether the clause ends at “race” or “hard”. They resolved these ambiguities by their second approach proposed in [21] using a parser. The second approach was to implement a program that learns simplification rules from a corpus of sentences and their hand-simplified forms. A Lightweight Dependency Analyzer (LDA) was used to parse the original and simplified sentences to heuristically determine the constituent structure and dependencies between constituents. Then, the resulting dependency representations were chunked into phrases. For example, the chunked LDA representation for the sentence: “Talwinder Singh, who masterminded the 1984 Kanishka crash, was killed in fierce two-hour encounter” and its simplified version is illustrated in Figure 2. The nodes in this representation contain word groups, which are linked by dependency information. Using a tree-comparison algorithm, they induced simplification rules by comparing the structures of the chunked parses of the hand-simplified and the original text. The learning algorithm worked by flattening subtrees that were the same on both sides of the rule, replacing identical strings of words with variables, and then computing tree-to-trees transformations to obtain rules in terms of these variables. These rules are generalized by changing specific words to tags. The training set for learning rules was small which involved only 65 texts. The authors did not provide any evaluations, so it is difficult to assess how well their approaches behave.

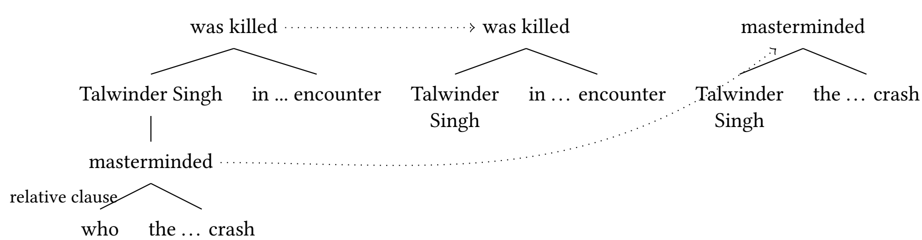


Fig. 2. Chunked LDA representation of a complex sentence and its simplified version. Reproduced from [21].

For subsequent discussion of the rule-based approach, we will divide it into four subcategories.

Preserving text cohesion in syntactic simplification: Siddharthan [101] decomposed the task of text simplification into three stages: analysis, transformation, and regeneration. The first two stages correspond to those in the two-stage theory proposed by [20]. So, this work is mainly concerned with text generation issues, e.g. sentence ordering, cue-word selection … etc. One of the things the authors addressed is preserving text cohesion, something that was not addressed by earlier rule-based approaches. However, they argued that the application of some simplification rules may badly affect the cohesion of the text. Table 6 shows one such example. The subordinate clause (c) is erroneously linked to sentence (b) instead of (a). Also, the pronoun “it” appears to refer to “an employment agency” rather than to “program trading”.

To overcome these problems a three-stage architecture is proposed. The analysis module aims to convert a text into a representation such that the transformation and regeneration modules can deal with. It performs various operations, such as text segmentation, noun chunking, third-person pronoun resolution, and clause/appositive identification and attachment. For example, the simplification in Table 7 requires knowledge that the relative clause attaches to “Cathy Tinsall” rather than “South London”, and that the relative clause does not end at the first comma, but goes all the way to the end of the sentence. To deal with appositive/clause attachment, mechanisms based on machine learning, and WordNet hierarchies, are used.

Table 6. Example showing the bad effect of blindly applying of some simplification rule on text cohesion. The original sentence containing conjunction and relative clause is simplified into three sentences (a), (b), and (c).

Original	"Mr. Anthony, who runs an employment agency, decries program trading, but he isn't sure it should be strictly regulated."
Simplified	"(a) Mr. Anthony decries program trading." "(b) Mr. Anthony runs an employment agency." "(c) But he isn't sure it should be strictly regulated."

Table 7. Another simplification example.

Original	" 'The pace of life was slower in those days,' says 51-year old Cathy Tinsall from South London, who had five children, three of them boys."
Simplified	" 'The pace of life was slower in those days,' says 51-year old Cathy Tinsall from South London. Cathy Tinsall had five children, three of them boys."

The second stage is the transformation, in which the actual syntactic simplification occurs. It consists of seven straightforward hand-crafted rules used to deal with conjunctions, relative clauses, and apposition. These rules are applied recursively on a sentence until no more simplification is possible.

The second function of the transformation module is to invoke the regeneration module when required. For instance, the cue-word "but" in the previous example is not introduced by this rule; rather, it was introduced by the regeneration module. The regeneration module as described earlier addresses issues that are essential for maintaining the cohesion and meaning of the original text. This module performs each of the following tasks: introducing cue words, deciding sentence order, generating referring expressions, selecting determiners, and preserving anaphoric links. The sentence ordering task is formulated as a constraint satisfaction problem, where constraints are introduced by rhetorical relations appearing in the original sentence which should be preserved during text regeneration. For the evaluation, three subjects were asked to judge meaning preservation and the cohesiveness of regenerated text. Also, they measured the readability of the simplified text using the Flesch formula (see Eq. 1).

Aranzabe et al. [5] proposed a text simplification system for Basque. The proposed approach is based on a linguistic study of two corpora: EPEC [2], and Consumer corpus.[4] EPEC is a reference corpus for the processing of Basque, written in Standard Basque. The corpus contains 300,000 word sample collection that has been manually tagged at different linguistic levels. The Consumer corpus is a specialized corpus containing texts written in four different languages including Basque. The objective of the linguistic study was the identification of the structures—of long sentences—that should be simplified. The simplification is a five-step process.

For the first step, the complexity of the text is evaluated using IAS, a module developed for the automatic evaluation of essays in Basque [19]. The module uses several criteria such as the clause number in a sentence, sentence types, and word types. The second step, split long sentences into clauses using Mugak, a module to detect chunks and clauses [26]. Mugak is a general purpose clause identifier that combines both, statistical and rule-based schemes. In

[4]http://corpus.consumer.es/corpus/.

the third step, a set of deletion and addition rules are applied in order to remove unnecessary morphological features such as complementizers. However, it adds grammatical elements in the split sentences. Next, a set of re-ordering rules is applied to perform the reordering needed in the new sentences. And finally, correct the spelling of the generated sentences using XUXEN, a grammatical and spelling corrector for the Basque language [3]. These rules were used for simplifying sentences that contained the following syntactic phenomena: apposition, finite relative clauses, and finite adverbial temporal clauses. Table 8 illustrates the simplification process for a sentence that has the three phenomena. The changes are underlined. There are five verbs in the original sentence, and each one of them builds a clause.

Table 8. Simplification process for a sentence in Basque that has appositions, relative, and adverbial clauses. We used Google translate for the English translation of the original (in italic) and the simplified sentence.

Original	*"Metalak igurtzitzen ditugunean nahiz eta kargen bereizketa berdin gertatu sortzen den partikulen mugimendua oso erraza da material hauetan (eroankortasun elektriko haundia dute)"*
	"When we rub metals, although charge separation happens equally, the particle movement that is generated is very easy in these materials (they have a high electrical conductivity)"
Simplified	*"Metalak igurzten ditugu. Partikulen mugimendua sortzen da. Orduan, nahiz eta kargen bereizketa berdin gertatu, partikulen mugimendua oso erraza da material hauetan. Material hauek eroankortasun handia dute."*
	"We rub metals. The particle movement is generated. Then although charge separation happens equally, the particle movement is very easy in these materials. These materials have a high electrical conductivity."

Syntactic dependencies based syntactic simplification: Siddharthan [102] presented a TS scheme based on applying transformation rules to a typed dependency representation obtained from the Stanford Parser [32]. These rules are used for simplifying sentences that contain the following syntactic phenomena: coordination (of verb phrases and full clauses), subordination, apposition, relative clauses, and passive constructions. Two approaches for generating sentences from the transformed representation are applied: a rule-based generator (gen-light), and a statistical generator (gen-heavy).

The gen-light approach uses the words and word order from the original sentence, unless when the lexical substitution rules and explicit order indication are involved in the representation. The gen-heavy approach uses an existing widely used generator RealPro [69] that makes the decisions related to word ordering and morphology. The RealPro uses as input Deep Syntactic Structure (DSyntS). For this purpose, a set of transformation procedures is used to translate the Stanford dependency types into the DSyntS notation that is required by RealPro. Since inaccurate parsing leads to error in the output, the author proposes using the n best parses to choose the best simplification out of them, according to a scoring metric. This metric deducted a point for sentences ending in subject pronouns, prepositions or conjunctions, sentences containing consecutive word repletion, prepositions followed by subject pronouns, and bad sequences of conjunctions and prepositions (e.g., "because", "but"). In addition, positive scores are given to sentences containing bigram and trigram overlap with the original sentence and when the simplification was performed on the top-ranked parse. For evaluation, the author used the extent of the simplification achieved and the precision for which the rules have been applied accurately. The results show that the gen-light approach is robust to the parsing errors, specifically in the n-best parse setting.

A text simplification system for the Spanish language was presented in [14]. For the representation of the syntactic structures, they used dependency trees, which is generated using the Mate-tools parser [10]. However, the simplification rules were developed within the Mate framework [12], a graph transducer that uses handwritten grammar. Structural simplification was carried out in two steps: first, the grammar looks for a suitable structure that could be simplified. Second, structural changes are applied. This may involve deletion, insertion, and copying of syntactic subtrees or nodes. The system was evaluated on a parallel corpus of 200 news articles compiled by the authors. Different rules of the simplification grammar were evaluated in terms of precision, recall, and frequency. These are relative clauses, simple relative clauses, complex relative clauses, gerundive constructions, quotation inversion, object coordination, verb phrase, and clause coordination. Quotation inversion was the most reliably handled operation ($P = 79\%$).

Using open source software, Ferrés et al. [45] built a text simplification system for English, namely YATS (for Yet Another Text Simplifier), that combined lexical and syntactic simplification approaches. Similar to [9] YATS used a context-vector model in the lexical simplification module in order to obtain the most appropriate substitutions for a given context. The remaining substitutions then are ranked using a word frequency simplicity measure based on the Simple English Wikipedia frequency list. Moreover, two stages of syntactic simplification are applied to simplify several types of syntactic structures, including appositive phrases, coordinated correlatives, and relative clauses. First, document analysis is applied to identify these structures. In this stage three main resources are used, which are: the GATE/ANNIE analysis pipeline to perform sentence splitting and named entity recognition, the Mate Tool decency parser [11] to add dependency labels to sentences, and a set of GATE JAPE (Java Annotation Patterns Engine) grammars for syntactic phenomena detecting and labeling. A set of dependency-tree based rules is manually developed using dependency parsed sentences from Wikipedia. These rules are used to analyze the syntactic structures appearing in the text. The second stage is sentence generation in which the syntactic dependency information and part-of-speech tags are used to produce simple structures. Human evaluation of YATS showed slightly better improvements over [104].

Event-based syntactic simplification: acknowledging that events represent the most important information in news, the authors proposed an event centered sentence simplification approach in [51]. The system is composed of two components. An event extraction component, and a sentence simplification component. The event extraction system involves supervised extraction of event anchors, and a rule-based extraction component that identifies arguments of the event anchors. The event anchors are words that convey the core meaning of the event. For argument extraction, they used dependency relation patterns, e.g. `dobj(X,Y)`. The paper focuses on extracting four types of arguments: agent, target, time, and location, arguing that these types are informationally most relevant for the event. Table 9 shows examples of anchors and the extracted arguments.

Table 9. Examples of anchors (underlined) and extracted arguments (boldfaced).

Examples	Dependency relations	Arg. type
(i) "**China** confronted Philippines"	`nsubj`(confronted, China)	Agent
(ii) "China disputes **the agreement**"	`dobj`(disputes, agreement)	Target

Once anchors and arguments have been extracted, two different simplification schemes are proposed: (a) sentence-wise simplification, which eliminates all the tokens (words and phrases) of the original sentence that are not identified by the extraction patterns; and (b) event-wise simplification, where each extracted event (i.e. event anchors or arguments) is transformed into a single sentence. Therefore, a multiple events sentence will be mapped to several sentences. In order to preserve the grammaticality of the simplified output generated by the second approach, three adjustments are considered. First, ignoring events of the reporting type. Second, ignoring events that detected with nominal anchors since these events tend to have few arguments. Third, converting gerundive events that govern the main sentence event into the past simple. Examples of simplification produced by these approaches are shown in Table 10. The experiments show that the event-wise simplification increases readability and retains the grammaticality of the text, while at the same time preserving the relevant information and discarding those which are irrelevant.

Table 10. Simplification example.

Original	"Baset al-Megrahi, the Libyan intelligence officer who was convicted in the 1988 Lockerbie bombing has died at his home in Tripoli, nearly three years after he was released from a Scottish prison."
Sentence-wise simplification	"Baset al-Megrahi was convicted in the 1988 Lockerbie bombing has died at his home after he was released from a Scottish prison."
Event-wise simplification	"Baset al-Megrahi was convicted in the 1988 Lockerbie bombing. Baset al-Megrahi has died at his home. He was released from a Scottish prison."

Štajner and Glavaš [112] proposed an automatic text simplification system that expands the work in [51]. The system comprises of two modules: the event-based simplification (EBS) module and the lexical simplification (LS) module. The EBS module is based on EVGRAPH [82], an event extraction system that extracts factual events mentioned in the text. EVGRAPH extracts four types of event arguments: agent, target, time, and location using a set of syntactically-based extraction rules. Once an event (e.g., anchors and its arguments) has been extracted, it transforms each factual event mentioned into a separate sentence in the same order as in the original sentence. For example, the sentence "President Obama argued with Putin, occasionally raising his voice." is transformed into "President Obama argued with Putin. President Obama raised his voice." We note that "President Obama" is part of both produced sentences. However, to ensure the simplified text's grammaticality, they followed the same rules as the event-wise simplification approach in [51]. The EBS module performs no lexical substitutions. This means the text generated by this algorithm could be lexically complex. Thus, they coupled the event-based simplification with an LS algorithm.

In the LS module, the authors employed Light-LS [52], which is an unsupervised LS model. Light-LS views all content words as complex, and the decision to substitute depends on the simplification candidates for each word. The authors used GloVe [88] word embeddings to find the most similar candidates for each input word and then ranked the substitution candidates. The two simplification components, EBS and LS can be applied to text in different orders which are termed EvLex, and LexEv. In EvLex, they applied the LS to the output of the event-based simplification, while in LexEv the event-based simplification is applied after the lexical substitutions.

In the end, the authors evaluated whether one order produces better simplifications than the other. The experiments showed that neither order of the modules has any significant influence on the readability of the generated texts, the grammaticality, simplicity, and the meaning preservation of the generated sentences. Table 11 shows simplification examples using this method. For evaluation, they evaluated the readability of the simplified texts using several readability metrics. In addition, a sentence-level human evaluation from three different aspects: simplicity, grammaticality, and meaning preservation. Finally, a comparison with two state-of-the-art TS systems on two different datasets: news articles, and Wikipedia. The results show that the systems produced significantly more readable and significantly simpler sentences than the other state-of-the-art TS systems while obtaining similar grammaticality and meaning preservation.

Table 11. Simplification example using [112]. Lexical changes are boldfaced, and syntactic changes are underlined.

Original	"They drove a patrol car onto the lawn in an attempt to rescue her."
LexEV + EvLex	"They drove a **police** car onto the lawn."
Original	"Johnson was rushed to hospital but died from her wounds, Goodyear said."
LexEv + EvLex	"Johnson was rushed to hospital, <u>Johnson</u> died from her **injuries**."

Multilingual syntactic simplification: MUSST [95] was inspired by the framework in [101]. It is a simplification system that supports English, Italian, and Spanish. And can be extended to support other languages. MUSST has three modules: analysis, transformation, and generation. The analysis module searches for discourse markers and relative pronouns on the output of the Stanford dependency parser [22]. Once the clauses are identified, a set of rules is applied, in the transformation stage. The objective is to simplify appositive phrases, conjoint clauses, relative clauses, and passive voice sequentially. The simplification process is implemented in a recursive manner. This, to ensure there are no more simplification rules that can be applied. Then, each simplified sentence is sent to the generation module, which is responsible for reconstructing the simplified sentences and preserving the grammar. For an example using MUSST simplification, consider the following. The original sentence, "These organizations have been checked by us and should provide you with a quality service", and its simplified version is, "These organizations have been checked by us. And these organizations should provide you with a quality service". MUSST was evaluated on sentences in the PA domain. The percentage of correct simplifications of the simplified cases was 76%, 71%, and 38% for English, Spanish, and Italian languages (respectively).

The summary of the surveyed rule-based SS systems is presented in Table 12.

4.2.2 Data-driven approach. Woodsend and Lapata [124] proposes a text simplification model based on quasi-synchronous grammar [107]. The quasi-synchronous grammar produces alignment between parse trees. Given an aligned corpus of original and simplified sentences, both of which are syntactically parsed, the quasi-synchronous grammar is used to learn simplification rules from the parsed sentences. Each simplification rule describes the transformation operations that needed to transform sources to simplified parse trees. Three types of rules are learned, which are syntactic simplification rules, lexical simplification rules, and sentence splitting rules. The splitting rule learns to separate the source sentence into the main sentence and auxiliary sentence, which can be performed in sentences containing coordinate or subordinate clauses, relative clauses, apposition, and parenthetical content. Afterward, given an input sentence, if more than one rule is matching, then, all possible simplifications will be generated. In order to select the most suitable simplification, they employ Integer Linear Programming (ILP). The IPL model selects the optimal simplification by

Table 12. Main characteristics of the surveyed rule-based syntactic simplification systems.

Year	Ref.	Language	Evaluation	Covered Syntactic Phenomena
1996	[20]	English	Manual	Relative and adverbial clauses.
1997	[21]	English	None	Relative and adverbial clauses.
2006	[101]	English	Manual	Appositions, relative, and subordinate clauses.
2011	[102]	English	Manual	Appositions, passive constructions, relative, coordinative, and subordinate clauses.
2012	[14]	Spanish	Automatic	Relative, participial, coordinative, subordinate, and adverbial clauses.
2012	[5]	Basque	None	Appositions, relative, and adverbial clauses.
2013	[51]	English	Automatic + Manual	Event-based simplification.
2016	[45]	English	Manual	Appositive phrases, adverbial clauses, coordinated clauses, coordinated correlatives, passive constructions, relative clauses, and subordinated clauses.
2017	[112]	English	Automatic + Manual	Event-based simplification.
2017	[95]	English, Italian, and Spanish	Manual	Appositions, conjoint, and relative clauses, and passive constructions.

maximizing a cost function based on grammaticality constraints such as the length of the sentence and reading ease. The model is trained using two datasets: SEW edit histories (RevILP), and EW-SEW aligned corpus (AlignILP). These models were compared to a tree-based simplification system [132], and a lexical simplification baseline which is based on automatic and human evaluations. The results showed that RevILP performed best in terms of simplicity, grammaticality, and meaning preservation. A simplification example produced by the systems is in Table 13.

Table 13. Example simplification produced by the systems AlignILP and RevILP [124].

Normal Wikipedia	"Wonder has recorded several critically acclaimed albums and hit singles, and writes and produces songs for many of his label mates and outside artists as well."
Simple Wikipedia	"He has recorded 23 albums and many hit singles, and written and produced songs for many of his label mates and other artists as well."
AlignILP	"Wonder has recorded several critically acclaimed albums and hit singles. He produces songs for many of his label mates and outside artists as well. He writes."
RevILP	"Wonder has recorded many critically acclaimed albums and hit singles. He writes. He makes songs for many of his label mates and outside artists as well."

4.3 Machine translation

Inspired by the excellent achievements of machine translation (MT) techniques, several studies address the text simplification task as a mono-lingual MT problem, where complex sentences translated to simpler ones. Recent studies in MT-based text simplification employ either statistical machine translation (SMT), or neural machine translation (NMT) approach.

4.3.1 Statistical machine translation (SMT). Specia [110] was the first work that used a standard SMT framework for the text simplification task. Here, the translation of the original sentence f (called the translation model) into a sentence e (called the language model) is modeled on the Bayes Theorem as follows:

$$\Pr(e \mid f) = \frac{\Pr(f \mid e)\Pr(e)}{\Pr(f)}, \tag{8}$$

where $\Pr(f \mid e)$ is the probability that the sentence f is the translation of sentence e, $\Pr(e)$ is the probability of the sentence e, and $\Pr(f)$ is a constant that can be disregarded. Since a sentence can have multiple translations, the proposed method aims to find the best translation $\hat{e}$ with the highest probability,

$$\hat{e} = \operatorname*{argmax}_{e \in e^*} \Pr(f \mid e)\Pr(e). \tag{9}$$

Besides these probabilities, some weights need to be estimated for these models and additional models. These weights govern the ordering of phrases in a simplified sentence and control the phrases and the length of the sentences, for example. The model was trained using Moses [66], a standard phrase-based SMT system, on a parallel corpus of original and simplified texts produced for the PorSimples project [4]. The corpus contains two levels of simplifications: natural and strong. The former is freely generated by the annotators, while the latter is generated by following certain constraints. For training the model, they used the corpus of natural simplification. For this, they randomly selected 3,383 aligned sentences for training, 500 aligned sentences for parameter tuning, and 500 aligned sentences for testing. Human and automatic simplification were compared using popular MT evaluation metrics *BLEU* [87] and *NIST* [38]. Both measure the overlapping of n-grams between human and system simplifications. The model achieved 0.6075 and 9.6244 in *BLEU*, and *NIST* respectively. In the translation task, a *BLEU* score of around 0.6 is considered a good result. In addition to these metrics, three other qualitative tests were performed. These were used to show that the automatic simplification is closer to the original than the reference simplification. The results also prove that the simplifications are likely to be correct since they score higher than the reference simplification. However, human evaluation is conducted to judge the fluency (grammatical?), adequacy (meaning preservation), and the simplicity of the simplification on a scale of 1 (worst) to 3 (best). The average scores were 2.5 for fluency and adequacy, and 2.35 for simplicity. The experiments show that using the SMT framework in TS is very promising, although it is more related to lexical operations than syntactic operations.

Zhu et al. [132] proposed a tree-based simplification model (TSM), which is based on the SMT technique. They applied four different simplification operations on the parse tree of an input sentence. These are splitting, dropping, reordering, and phrase/word substitution. The splitting operation was accomplished using two operations: segmentation, and completion. The segmentation decides—based on syntactic constituent (usually a relative pronoun)—where, and whether to split a complex sentence. Completion completes the split sentence. For example, the sentence "August was the sixth month in the ancient Roman calendar which started in 735BC." Figure 3 shows the original sentence, and the same sentence following subsequent operations (Figure 4).

In Figure 4a we see how the complex sentence (Figure 3) has been transformed into two sentences. The next operation is dropping the non-terminal nodes from the parse tree (Figure 4b), where the NNP "Roman" is dropped. This is followed by the reordering operation, in which the children of a certain node are interchanged. The last operation is the substitution. It is performed on the terminal nodes in case of word replacement, and on the non-terminal nodes in case of phrase substitution. Figure 4c shows a word substitution example where "ancient" is replaced by the word "old". As a

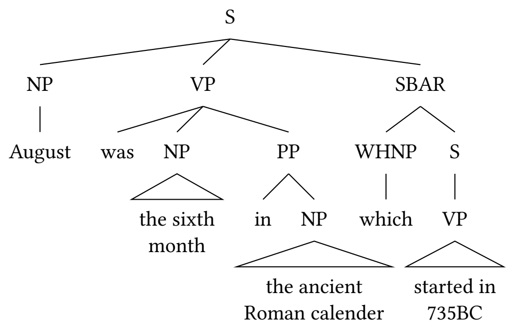


Fig. 3. The parse tree of complex sentence (reproduced) [132]. The NP, VP and PP are noun, verb and preposition phrase (respectively). The WHNP is wh-noun phrase, which is a noun containing wh-word (e.g., who, which, whose).

result of applying all the simplification operations, we get two sentences, pt1 and pt2. Thus, the simplified sentence is: "August was the sixth month in the old calendar. The old calendar started in 735BC". The TSM is a probabilistic model in which all the operations have probabilities, and the model combines all these probabilities into a direct translation model $\Pr(s \mid c)$ that translates the original complex sentence c to a simple sentence s, as follows:

$$s = \underset{s}{\operatorname{argmax}} \Pr(s \mid c) \Pr(s), \tag{10}$$

where $\Pr(s)$ is a simple sentences language model. The direct translation model $\Pr(s \mid c)$ is given by:

$$\sum_{\theta:str(\theta(s))=s} \Big(\Pr(seg \mid c) \Pr(com \mid seg) \prod_{node} \Pr(dp \mid node) \Pr(ro \mid node) \Pr(sub \mid node) \prod_{w} \Pr(sub \mid w) \Big), \tag{11}$$

where θ is the sequence of simplification operations and $str(\theta(\text{c}))$ is the leaves of the simplified tree. In the training process, the probability of each operation to the parse tree nodes is learned from a dataset that contains aligned sentence pairs from EW and SEW, the English, and the Simple English Wikipedia respectively. Guided by syntax-based statistical machine translation [128], the model is trained by maximizing $\Pr(s \mid c)$ over the training dataset using the Expectation-Maximization (EM) algorithm. Finally, a decoder is implanted using a greedy method, in order to produce the simplified sentences. Four baselines were compared to TSM: Moses [66], a sentence compression system [47], an enhanced version of a sentence compression system with a lexical component, and a sentence compression system the perform sentence splitting on the output of the previous system. The systems were evaluated using *BLEU* and *NIST* metrics, as well as different readability scores such as the Flesh Reading Ease test, and the n-gram language model perplexity (PPL). The proposed system outperformed the baselines in terms of the readability metrics, but it did not achieve better performance in terms of *BLEU* and *NIST*. A simplification example by TSM is shown in Table 14.

Coster and Kauchak [27] also adapt the standard statistical phrase-based translation system (PBMT) [66] for English text simplification. The authors observed that the phrase-based translation model requires phrases that contain one or more words. However, phrasal deletion and phrasal insertion—which commonly occur in the simplification process—were not considered. So, they modeled the deletion by relaxing this restriction, through aligning the unaligned sequence of words

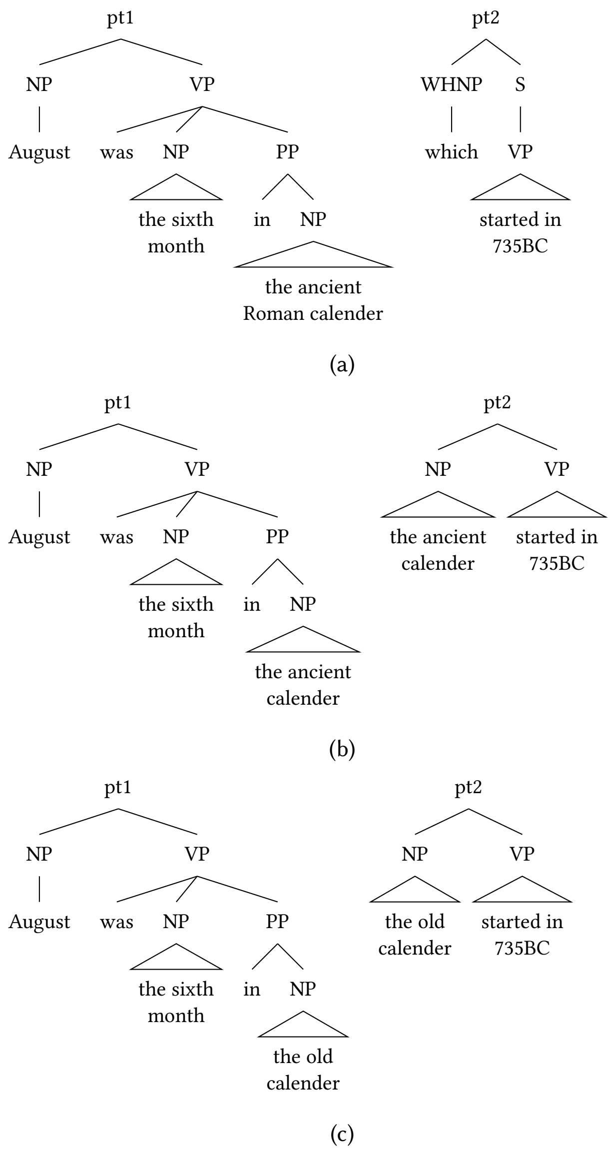


Fig. 4. The parse tree of the complex sentence in Figure 3 after, (a) segmentation and completion operation, (b) after dropping and reordering operations, and (c) after the substitution operation. Reproduced from [132].

to NULL. But, if a set of original words N align to a simple word s and there exists a word $n \in N$ where $n = s$, then for all $n_i \in N - \{n\}$ is aligned to NULL in the simplified sentence. Although this operation may allow deleting out-of-context words, they argued that these problematic cases can be avoided by the language model. The model was trained on 137,000 aligned pairs of sentences produced by aligning EW with SEW. The authors aligned sentence pairs with a similarity higher than 0.5 using a normalized TF-IDF cosine similarity function. The proposed system (Moses + Del) was compared against four different TS approaches using *BLEU*, simple string accuracy (SSA) [25],

Table 14. Tree-based simplification (TSM) example [132].

Complex	"Genetic engineering has expanded the genes available to breeders to utilize in creating desired germlines for new crops."
Simple Wikipedia	"New plants were created with genetic engineering."
TSM	"Engineering has expanded the genes available to breeders to use in making germlines for new crops."

and word-F1 (F_1 score computed over words) metrics. The competitive systems were a system of no simplification, two-sentence compression systems, and PBMT (Moses). The compression system performed worse than the baseline, which does nothing in simplification. Generally, the proposed system obtained the best scores. Table 15 presents simplification examples produced by Moses+Del system.

Table 15. Moses+Del simplification examples [27]. Changes are bold faced.

Original	"**Critical reception** for The Wild has been negative."
Simplified	"**Reviews** for the Wild has been negative."
Original	"Bauska is a town in Bauska county, in the **Zemgale** region of **southern Latvia**."
Simplified	"Bauska is a town in Bauska county, in the **Zemgale**."

Wubben et al. [125] investigated the performance of the PBMT model in text simplification, extending it to include a dissimilarity-based re-ranking method. For each input sentence, the re-ranking function chooses the n-best output sentences with lower similarity from the input sentence based on Levenshtein distance (LD) measure [77]. LD counts the minimum number of edit operations (e.g., insertion, deletion) that are required to transform the source sentence into the target sentence. Five systems were compared: the original SEW, the SMT system in [132], the text simplification system in [124], a word-substitution baseline, and the PBMT system with dissimilarity component. All the models were trained and tested using the PWKP dataset (see Section 2.2), generated by [132]. The system was evaluated using *BLEU*, and *FKGL* [62], in addition to human judgment in terms of simplicity, fluency, and adequacy.

Narayan and Gardent [76] proposed a hybrid approach to text simplification that integrates semantics and machine translation. The authors argued that the previous TS approaches, such as [124, 132], failed because the induction of rules was based only on syntactic information. However, the SMT-based simplification models also fail since they are unable to capture syntactic operations such as splitting and deletion. So, the authors introduced a sentence simplification system that combined a semantic model, that handles split and delete operations, with a machine translation model for word substitution and reordering. To represent an input sentence semantically, they used the discourse representation structure (DRS) [61]. The DSR was automatically produced by Boxer [29]. Given an input sentence c, the simplification process will be achieved through the following steps: (i) a semantic model (DRS-SM) is applied to the DRS representation of the sentence c to generate one or more simplified sentence(s) s^*; and (ii) using a phrase-based mono-lingual machine translation model and a probabilistic language model (PBMT + LM) the simplified sentence(s) s^* is further simplified to s. The DRS-SM model was trained using the PWKP dataset, and the estimation of the parameters was based on EM algorithm [33]. Table 16 shows a simplification example of the system. Three automatic metrics were used, namely *BLEU*, the LD between the generated simplification and the complex and gold simplification, and the number of generated sentences that are identical to the original and gold simplification; to compare the performance of the proposed system with the other three models [124, 125, 132]. The results indicate the superiority of the semantic-based system over the other systems. Human evaluations were also performed to judge the simplicity, fluency, and adequacy of the outputs of the systems, in which the proposed system obtained the best results.

Table 16. A simplification example produced by the semantic-based system [76].

Original	"In 1964 Peter Higgs published his second paper in Physical Review Letters describing Higgs mechanism which predicted a new massive spin-zero boson for the first time."
Simplified	"Peter Higgs wrote his paper explaining Higgs mechanism in 1964. Higgs mechanism predicted a new elementary particle."

Xu et al. [127] adapts the syntax-based statistical machine translation (SBMT) to perform sentence simplification. Arguing that the extracted rules from the parallel corpus of Normal-Simple English Wikipedia are limited in diversity and coverage, they trained the SMT model on a large scale paraphrase dataset (PPDB) [50]. This dataset was created by learning over 220 million paraphrase rules from the bilingual corpora. In the SMT framework, one crucial element is to design automatic evaluation metrics to be used as training objectives. The authors showed that using *BLEU*, as used by earlier works, for tuning was insufficient, and instead they proposed two new metrics. *FKBLEU*, which explicitly measure readability, and *SARI*, which implicitly measures it comparing against the input and references. *FKBLEU* is the geometric mean of the *iBLEU* metric [115] and *FKGL* [62]. Also, they used a large set of features for each paraphrase rule in order to promote simpler output, in addition to nine new simplification-specific features (e.g., length in words, length in characters, number of syllables). For accurate tuning and evaluation, the authors collect eight reference simplifications that were based on paraphrasing simplification via crowd-sourcing. Table 17 shows examples of simplification output of the adapted SBMT models as well as the baseline model that uses *BLEU* as a tuning metric. Automatic and human evaluations were conducted. The *SARI* metric (PPDB + SARI) achieved the highest correlation with human judgments of simplicity. However, *BLEU* metric (PPDB + BLEU) exhibited higher correlations on grammaticality and meaning preservation.

Recent lexical simplification systems, especially non-English ones, suffer from several problems. They do not have sufficient lexical coverage (supervised approaches) caused by the limitation of parallel-corpora size. However, they are not able to simplify lexical phrases with more than one-word length, and they cannot perform the word-reordering operation. To overcome these problems [113] built several new simplification-specific parallel datasets in Spanish by filtering and ordering paraphrase pairs and synonyms from four available resources. These newly built datasets are integrated into nine different combinations with an existing text simplification parallel dataset in a phrase-based statistical machine translation (PBMT) approach. Three language models (LMs) are used, one is trained on Spanish Wikipedia (SW) and the others are trained on two simpler versions of the SW that have sentences not longer than 15 and 20 tokens. As a result, 27 simplification systems are produced (nine datasets combined with three LMs) using the standard PBMT (Moses) [66] models. Manual and automatic evaluations (using *BLEU*) are conducted to assess the system's performance. The results show that the good combinations of the newly built dataset with the text simplification dataset improve the simplicity, grammaticality, and meaning preservation of the generated output comparing with the baseline PBMT models.

Table 17. Example outputs of human reference simplifications and the automatic simplification system using [127]. The bold highlights the changes from the original sentence in the Normal Wikipedia.

Normal Wikipedia	"Jeddah is the principal gateway to Mecca, Islam's holiest city, which able-bodied Muslims are required to visit at least once in their lifetime."
Simple Wikipedia	"Jeddah is the **main** gateway to Mecca, **the** holiest city **of** Islam, where able-bodied Muslims **must go** to at least once in **a** lifetime."
Human Ref #1	"Jeddah is the **main entrance** to Mecca, **the** holiest city **in** Islam, which **all healthy** Muslims **need** to visit at least once in their **life**."
Human Ref #2	"Jeddah is the **main entrance** to Mecca, Islam's holiest city, which **pure** Muslims are required to visit at least once in their lifetime."
SBMT (PPDB + BLEU)	"Jeddah is the **main door** to Mecca, Islam~~'s~~ holiest city, which **sound** Muslims are **to go to** at least ~~once~~ in **life**."
SBMT (PPDB + FKBLEU)	"Jeddah is the **main** gateway to Mecca, Islam's holiest city, which **sound** Muslims **must** visit at least once in **life**."
SBMT (PPDB + SARI)	"Jeddah is the **main** gateway to Mecca, Islam's holiest city, which **sound** Muslims **have to** visit at least once in their life."

4.3.2 Neural machine translation (NMT). NMT is a recently proposed deep learning technique that achieved great results over various difficult tasks [7, 23, 116]. It showed higher powerful capabilities than SMT systems. Wang et al. [121] presented an experimental study that investigated the performance of deep neural networks in the text simplification task. The authors build a model using the standard long short-term memory (LSTM) [55] encoder-decoder structure to see if this model is able to reverse, sort, and replace the elements of a sequence. They argued that the LSTM Encoder-Decoder model is able to learn operation rules such as sorting, reversing, and replacing from sequence pairs, which are similar to simplification rules that change sentence structure, substitute words, and remove words. The LSTM network is a type of recurrent neural network that achieves good results at learning long-range dependencies via its internal memory cells.

Two LSTM layers are used in the model for both the encoder, which represents the input sequence as a vector; and the decoder, which decodes that vector into an output sequence. The sequences of inputs are integer numbers generated randomly with a length of 25. These integers represent the words indices in the vocabulary set (V). First, they conducted experiments using three different vocabularies ($|V| = 10, 100, 1000$) to prove that the LSTM encoder-decoder is able to reverse a sequence after training on datasets of sequence pairs (X, Y), where $X = (x_1, x_2, \ldots, x_{25})$, and $Y = (x_{25}, x_{24}, \ldots, x_1)$, where $x_i \in V$. Then, they showed that the LSTM encoder-decoder can sort a sequence after training on a large set of sequence pairs (X, Y), where $Y = \text{sorted}(x_1, x_2, \ldots, x_{25})$. Next, they showed that the LSTM encoder-decoder is able to replace words in a sequence pairs (X, Y), where X as before, and $Y = (x_1 \mod n, x_2 \mod n, \ldots, x_{25} \mod n)$, for $n = 2, 20, 200$. That is, if $n = 200$ and $|V| = 1000$ then the top 20% of words in the vocabulary will be kept by the model, and these will be used to replace all the matching words in the output sequence. And finally, they combined the three functions together to check if the model is able to learn the rules between sequences. The results showed that the LSTM encoder-decoder was able to learn the operation rules: reversing, sorting, and replacement from the provided data with 90% accuracy, given large training data. It showed the model may potentially apply rules like modifying sentence structure, substituting words, and removing words for text simplification. For example, consider the sentence "Man with high intelligence.", following the simplification using LSTM Encoder-Decoder, it becomes "A very smart man."

Zhang and Lapata [130] proposed a simplification model DRESS (deep reinforcement sentence simplification). Mainly, it is based on an encoder-decoder architecture implemented by recurrent neural networks. The encoder transforms the source sentence $X = (x_1, x_2, \ldots, x_{|X|})$ into a list of continuous-space representations with a LSTM network from which the decoder uses another LSTM to generate its simplified target sequence $Y = (y_1, y_2, \ldots, y_{|Y|})$. In order to make the model meet the simplification constraints, the model was trained in a reinforcement learning framework [123], which explores the ability of simplifications while learning to optimize an expected reward function. The reward function encourages outputs that meet simplification constraints, namely simplicity, relevance, and fluency. They also proposed a lexical simplification component in order to enhance the performance. The lexical simplifications learned explicitly and integrated with the reinforcement learning-based model.

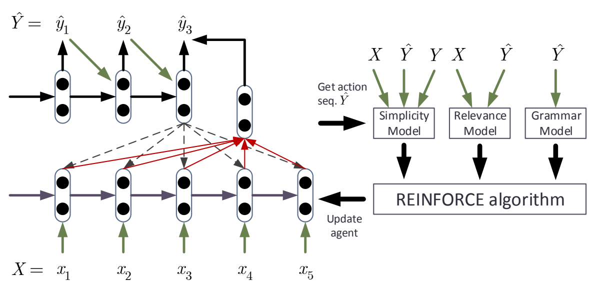


Fig. 5. DRESS model (reproduced) [130]. X is the complex sentence, Y the simplified sentence, and $\hat{Y}$ the action sequence (simplification) produced by the encoder-decoder model.

For assessing the performance of the model, they conducted experiments on three available simplification datasets, namely WikiSmall [132], WikiLarge (constructed by combining several previously created simplification corpora), and Newsela [126]. The system output was manually evaluated with respect to fluency, adequacy, and simplicity. It was also evaluated automatically using *BLEU* [87], *FKGL* [62], and *SARI* [115] metrics. Experiments show that the reinforcement-learning framework highly improves the quality of the simplified text, achieving significant improvements over competitive simplification systems. An example of the system output on the Newsela dataset is in Table 18.

Table 18. An example of the output using DRESS [130] on the Newsela dataset. Substitutions are boldfaced.

Complex	"There's just one major hitch: the primary purpose of education is to develop citizens with a wide variety of skills."
DRESS	"There's just one major hitch: the **main goal** of education is to develop **people** with **lots of** skills."
DRESS + lexical model	"There's just one major hitch: the **main goal** of education is to develop citizens with **lots of** skills."

Nisioi et al. [78] presented a neural simplification system (NTS) using a sequence of neural networks. The authors trained and built the system with a two-layer unidirectional LSTM encoder using the OpenNMT framework [64]. However, in order to increase the system's performance, they build a second model (NTS-w2v) by combining pre-trained Word2vec from Google News Corpus

[72] with locally trained embeddings. To obtain the best prediction and the best simplification at each step, they re-ranked the predictions with *BLEU* and *SARI* metrics. For training the model, the authors used the publicly available dataset [57], which is based on manual and automatic alignments between English Wikipedia (EW) and Simple English Wikipedia (SEW) and using only good matches and partial matches with a scaled threshold above 0.45. Table 19 shows an example of full, and partial matches from the EW and SEW datasets. They evaluated their system, and three different state-of-the-art TS systems using three types of human evaluation. These are correctness, number of changes including grammaticality and meaning preservation, and simplicity. The results proved the superiority of NTS models in both percentages of correct changes and simplicity score. The output examples of the proposed system and the competing systems are shown in Table 20.

Table 19. Examples of full and partial matches using the NTS system [78] on EW and SEW datasets [57].

Match	Transformation	Sentence pair
Full	Syntactic simplification, reordering of sentence constituents	"During the 13th century, gingerbread was brought to Sweden by German immigrants." and "German immigrants brought it to Sweden during the 13th century."
Partial	Adding explanation	"Humidity is the amount of water vapor in the air." and "Humidity (adjective: humid) refers to water vapor in the air, but not to liquid droplets in fog, clouds, or rain."

Table 20. Output examples of the NTS system and those of competing systems. Changes are marked in boldface.

Original	"Perry Saturn (with Terri) defeated Eddie Guerrero (with Chyna) to win the WWF European Championship (8:10); Saturn pinned Guerrero after a diving elbow drop."
NTS-w2v default	"Perry Saturn (with Terri) defeated Eddie Guerrero (with Chyna) to win the WWF European Championship (8:10); Saturn pinned Guerrero after a diving elbow drop."
NTS-w2v *SARI*	"Perry Saturn **pinned Guerrero to win the WWF European Championship**."
NTS-w2v *BLEU*	"Perry Saturn pinned Guerrero after a diving **drop**."
NTS default	"**He** (with Terri) defeated Eddie Guerrero (with Chyna) to win the WWF European Championship (8:10); Saturn pinned Guerrero after a diving elbow drop."
NTS *BLEU/SARI*	"**He** defeated Eddie Guerrero (with Chyna) to win the WWF European Championship (8:10); Saturn pinned Guerrero after a diving elbow drop."
LightLS [52]	"Perry Saturn (with Terri) defeated Eddie Guerrero (with Chyna) to win the WWF European Championship (8:10); Saturn pinned Guerrero after a **swimming shoulder fall**."
SBMT [127]	"Perry Saturn (with Terri) **beat** Eddie Guerrero (with Chyna) to win the WWF European **League** (8:10); Saturn pinned Guerrero after a diving elbow drop."
PBSMT-R [125]	"Perry Saturn with Terri **and** Eddie Guerrero, Chyna, to win the European Championship **then-wwf** (8:10); **he** pinned Guerrero after a diving elbow drop."

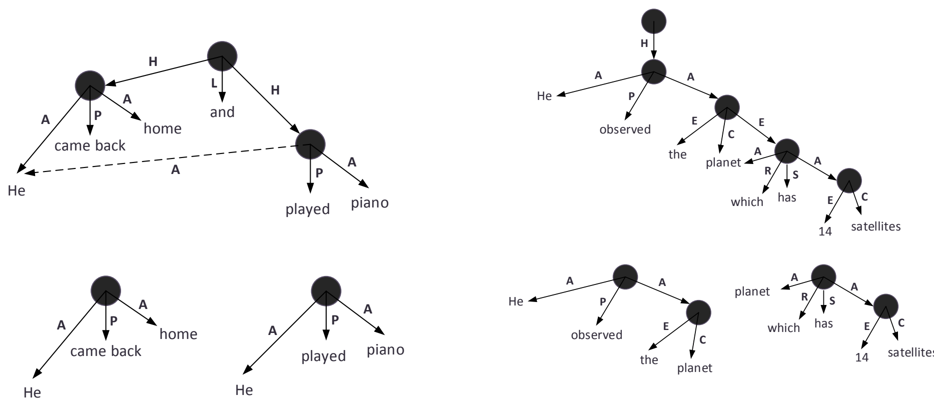
Zhang et al. [131] extends the model proposed in [78] by adding lexical constraints to the NMT model. They proposed a two-step simplification model. In the first step, the difficult words in the input sentence are identified and replaced with a simpler synonym based on a pre-constructed knowledge base. For the second step, a constrained sequence-to-sequence (Constrained Seq2Seq) model generates a simplified sentence given the simplified words (synonyms) from the first step as constraints, starting with the complex word that has the least term frequency. Moreover, to maintain the semantic meaning of the original sentence, the generation process also includes the simplified word(s) backward and forward generation using a bi-directional recurrent neural network. The model is trained and evaluated using EW and SEW. Four automatic evaluation metrics were used to evaluate the proposed model and different TS systems. These are: *FKGL* [62], *SARI* [127], *BLEU* [87], and *iBLEU* [115]. The systems' output was evaluated manually with respect to grammaticality, meaning, and simplicity. The reported results showed the superiority of both Constrained Seq2Seq (with one constraint word), and Multi-Constrained Seq2Seq (with multiple constraint words) in terms of *iBLEU*, *FKGL*, simplicity over the other MT models (e.g. Moses [66], and SBMT [127]), and lexical-based system. Table 21 simplification examples of the proposed model and the baselines.

Table 21. Output examples of the model in [131] and the baselines. Changes are marked in boldface.

EW	"Parkes became a key country location after the completion of the railway in 1893, serving as a hub for a great deal of passenger and freight transport until the 1980s."
SEW	"Parkes was an **important** transport **center** after the railway was built in 1893. **Many** passenger and freight trains stopped at Parkes up until the 1980s."
Moses [66]	"Parkes became a key country location after the completion of the railway in 1893, serving as a hub for a great deal of passenger and freight transport until the 1980s."
SBMT	"Parkes became a key country location after the completion of the railway in 1893, serving as a hub for a great deal of passenger and freight transport until the 1980s."
Lexical subs.	"Parkes became a **important** country location after the completion of the railway in 1893, serving as a **center** for **many** passenger and transport the 1980s."
Constrained Seq2Seq	"Parkes became a key country location after the completion of the railway in 1893, serving as a **center** of passenger and freight transport until the 1980s."
Multi-constr. Seq2Seq	"Parkes became **an important** country location after the completion of the railway in 1893. **It became a center** of passenger and freight transport until the 1980s."

Unlike the previous NMT-based TS systems, [114] presented a simplification system that combines semantic structure and neural machine translation. They investigated the effect of sentence splitting on the neural system's subsequent application in terms of its capability to perform other simplification operations. They used UCCA (Universal Cognitive Conceptual Annotation) [1], a cross-linguistic framework for the semantic representation of the main text semantic units. The UCCA scheme represents the text as a collection of Scenes. A Scene describes a movement, an action, or a temporally persistent state, and contains one main relation, which can be either a State or a Process. However, a Scene may contain one or many Participants. For example, the sentence "She went to the store" has one Scene whose process is "went" and has two Participants which are "She" and "to the store". There are several categories for the Scenes in a text. It may be an Elaborator (E), a Participant (A) in another Scene, a Parallel Scene (H), or a Linker (L). The non-Scene units categorize as Center (C), which denotes the head of semantic. The minimal center of a UCCA unit x defines as the leaf of a graph that is reached by starting from x and iteratively choosing the child tagged as Center. After representing the text, they applied the Direct Semantic Splitting (DSS) algorithm, in that they defined two splitting rules corresponding to Parallel Scenes and Elaborator Scenes only.

In the first rule, the Parallel Scenes are extracted from a sentence, separated into multiple sentences, and then concatenated based on their appearance order. For example, this rule will convert the sentence "He came back home and played piano" into "He came back home. He played piano". The second rule extracts Elaborator Scenes and their corresponding minimal centers from a sentence. Then, concatenate Elaborator Scenes to the original sentence and remove them, except the minimal center they elaborate. Also, pronouns such as *who, that* and *which* are removed. For instance, when applying the second rule to the sentence "He observed the planet which has 14 known satellites", it will be converted to "He observed the planet. Planet has 14 known satellites". This rule did not regenerate the articles. Figures 6a and 6b shows examples applying both rules. Afterward, the output of splitting is fed into the (NTS-w2v) model [78]. However, to increase the *SARI* score, they used the highest (h1) and fourth-ranked (h4) hypothesis at each step, which leads to two corresponding models: SENTS-h1 and SENTS-h4. All models were tested on test corpus in [127].



(a) He came back home and played piano

(b) He observed the planet which has 14 known satellites

Fig. 6. Example (reproduced) of applying (a) the first rule, and (b) the second rule in [114]. The tags are: Parallel Scene (H), Linker (L), Participant (A), Process (P), State (S), Center (C), Elaborator (E), and Relator (R).

Automatic and manual evaluations were conducted for the different variants of the proposed system along with other competitive MT-based simplification systems. They used automatic metrics, *BLEU*, *SARI*, F_{add} (the F-score of the *SARI* additional component), F_{keep} (F-score of the *SARI* keeping component), and P_{del} (the *SARI* deletion component, precision). The experimental results show that SENTS-h1 outperforms the other systems in terms of grammaticality, simplicity, and structural simplicity. Where NTS-h1 (without structural component) model obtained the highest *BLEU* and F_{keep} scores. Table 22 shows the simplification examples of the different variants of the proposed model and the HYBRID system. We summarize the main characteristics of the surveyed machine translation simplification systems in Table 23.

Acknowledging that each kind of user has specific simplification needs, Mallinson and Lapata [70] proposed a transformer-based neural encoder-decoder model along with lexical and syntactic constraints (CROSS), that allowed the users to govern the simplicity level and the simplification type. The encoder enriches with lexical constraints by attaching indicator features to every word embedding during training, which indicates whether a token should be kept or not. However, syntactic constraints are introduced to the model by annotating the complex-simplified sentences with high-level syntactic descriptions (i.e., aka templates). Once the system is trained using this templates-enriched corpus, the decoder will produce a target template and then decode the string.

During test time, the user provides a set of constraints (lexical and syntactic) that must be followed by the system. This set is then used to mark the input's syntax and to constraint the output of the decoder. The experiment results showed that a constraint-aware system produced competitive results.

Table 22. Simplification examples of the different variants of the proposed model and HYBIRD system [130].

Original	"In return, Rollo swore fealty to Charles, converted to Christianity, and undertook to defend the northern region of France against the incursions of other Viking groups."
SEW	"In return, Rollo swore fealty to Charles, converted to Christianity, and swore to defend the northern region of France against raids by other Viking groups."
HYBRID	"In return Rollo swore, and undertook to defend the region of France. Charles, converted."
NTS-h1	"In return, Rollo swore fealty to Charles, converted to Christianity, and undertook to defend the northern region of France against the raids of other Viking groups."
NTS-h4	"In return, Rollo swore fealty to Charles, converted to Christianity, and undertook to defend the northern region of France against the attacks of other Viking groups."
SENTS-h1	"Rollo swore fealty to Charles."
SENTS-h4	"Rollo swore fealty to Charles and converted to Christianity."

Table 23. Main characteristics of the surveyed machine translation (MT) systems.

Year	Ref.	Language	Approach	Evaluation
2010	[110]	Portuguese	Statistical MT (SMT)	Automatic + Manual
2010	[132]	English	SMT	Automatic
2011	[27]	English	SMT	Automatic
2012	[125]	English	SMT	Automatic + Manual
2014	[76]	English	SMT	Automatic + Manual
2016	[127]	English	SMT	Automatic + Manual
2019	[113]	Spanish	SMT	Automatic + Manual
2016	[121]	English	Neural MT (NMT)	Automatic
2017	[130]	English	NMT	Automatic + Manual
2017	[78]	English	NMT	Manual
2017	[131]	English	NMT	Automatic + Manual
2018	[114]	English	NMT	Automatic + Manual
2019	[70]	English	NMT	Automatic + Manual

4.4 Hybrid approach

The rule-based lexical simplification approach suffers from several limitations that affect its performance. Things such as the need for a large number of transformation rules in order to obtain reasonable coverage, and the fact that it is limited to word-level substitution. These problems were overcome using data-driven TS approaches, which were affected directly by the availability of parallel corpus. Besides, the data-driven syntactic simplification approaches still produce less syntactic simplification. In order to beat these limitations, Siddharthan and Mandya [104] proposed a hybrid TS system, which integrates a data-driven lexical simplification module with a hand-crafted rule-based syntactic simplification module. All this under a framework defined over synchronous dependency insertion grammars (SDIG) [37]. Using SDIG allows better modeling of lexical transformations, makes it easy to write rules, and the automated acquisition of data. Several transformation rules were used in the syntactic simplification module. These are, 26 handcrafted rules used for appositions and relative clauses, 85 rules for subordination and coordination, 11 rules to handle voice conversion from passive to active, and in addition to 14 rules to standardize the quotations into the "X said: Y" form. The lexical simplification module is trained on the EW-SEW aligned corpus that is used by [27, 124]. Experimentally, the results show the superiority of the proposed hybrid system over that of a data-driven system (QSG) [124] in terms of fluency, complicity, and meaning preservation. Table 24 presents an example of simplification produced by the systems.

Table 24. An example of simplification produced by SEW, HYBRID and QSG systems [130].

Normal Wikipedia	"Takanobu Komiyama (born October 3, 1984 in Chiba, Japan) is a Japanese football player who currently plays for the J-league team Kawasaki Frontale."
Simple Wikipedia	"Takanobu Komiyama (born 3 October 1984) is a Japanese football player. He plays for Kawasaki Frontale."
HYBRID	"Takanobu Komiyama (born October 3, 1984 in Chiba, Japan) is a Japanese football player. Takanobu Komiyama at present plays for the J-league team Kawasaki Frontale."
QSG	"His father. Komiyama is a."

5 GENERAL CONCLUSION AND DISCUSSION

This survey addressed different automatic text simplification (TS) research studies. It covered various aspects presented in the literature, such as simplification methods, corpora used, and evaluation methods. Generally, automatic text simplification approaches are classified into four classes: lexical, syntactic, monolingual machine translation, and hybrid techniques. Lexical simplification (LS) is the task of identifying and substituting complex words with simpler synonyms. Most of the LS systems work on word-level ignoring the cohesion and coherence of the text.

Syntactic simplification (SS) aims to modify the syntax of a text by removing the complex syntactic phenomena, without modifying the original meaning. There are several types of phenomena that may be considered as complicated in a text, e.g. coordination, subordination, relative clauses, and passives. In order to take the advantages of lexical and syntactic approaches, some systems integrate a data-driven LS module with a hand-crafted rule-based SS.

Recently, there has been a trend to address the TS problem as a monolingual machine translation, where the original text is translated into a simpler one. Two MT techniques have been employed in TS researches. These are statistical machine translation (SMT), and neural machine translation (NMT) approach. The phrase-based machine translation (PBMT) model is constructed using a translation model is derived from parallel data and a language model that is derived from the target language monolingual corpus. On the other hand, the NMT model differs from PBMT in that it is being trained end-to-end without needing to have language models or phrase table.

Although the majority of TS studies were geared for the English language, TS has been applied across other languages, e.g. Japanese, Dutch, Spanish, Italian, Korean, and German. Most systems do not focus on introducing new techniques for TS but instead focuses on implementing existing techniques in their own language. The main difficulty is usually in discovering appropriate resources for the language. Also, the differences between simplified text and complex text in the language must be analyzed to discover language-specific simplification rules. Typically, these are not exchangeable between languages, since each has its own grammatic structures.

Automatic text simplification is far from perfect. As thus, more studies are needed to improve upon or introduce new simplification techniques, with new reliable evaluation methods, and new corpora. Dupoux [40] showed that the recent achievements of using AI to do complex cognitive tasks, in particular by training deep neural networks on large data, have been achieved by throwing out some of the classical theories in linguistics and psychology. However, he argued that developmental psychology and in particular language acquisition can benefit from a reverse engineering approach. He then proposed a roadmap for reverse engineering infant language learning using AI. The reverse engineering process consists of constructing a scalable computational system that mimics language acquisition in infants when fed with realistic data. In the same line, we suggest reverse engineering how children learn complex linguistic structures. This may hold the key for the next breakthrough in automatic TS.

ACKNOWLEDGMENTS

The authors are grateful to the anonymous reviewers for their critical and constructive comments that greatly helped improve the presentation of this work. We also extend our appreciation to the Deputyship for Research and Innovation, "Ministry of Education" in Saudi Arabia for funding this research work through the project no. IFKSURP-90.

REFERENCES

[1] Omri Abend and Ari Rappoport. 2013. Universal conceptual cognitive annotation (UCCA). In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, Vol. 1. 228–238.

[2] Itziar Aduriz, Maxux Aranzabe, J Arriola, Aitziber Atutxa, Arantza Diaz-De-Ilarraza, Nerea Ezeiza, Koldo Gojenola, Maite Oronoz, Aitor Soroa, and Ruben Urizar. 2006. Methodology and steps towards the construction of EPEC, a corpus of written Basque tagged at morphological and syntactic levels for the automatic processing. In *Language and Computers: Corpus linguistics around the world.* Vol. 56. Brill | Rodopi, 1–15.

[3] Eneko Agirre, Inaki Alegria, Xabier Arregi, Xabier Artola, A Díaz de Ilarraza, Montse Maritxalar, Kepa Sarasola, and Miriam Urkia. 1992. XUXEN: A spelling checker/corrector for Basque based on Two-Level morphology. In *Proceedings of the 3rd Conference on Applied Natural Language Processing.* 119–125.

[4] Sandra Maria Aluísio and Caroline Gasperin. 2010. Fostering digital inclusion and accessibility: the PorSimples project for simplification of Portuguese texts. In *Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas.* 46–53.

[5] Marıa Jesús Aranzabe, Arantza Dıaz de Ilarraza, and Itziar Gonzalez-Dios. 2012. First approach to automatic text simplification in Basque. In *Proceedings of the Natural Language Processing for Improving Textual Accessibility (NLP4ITA) workshop (LREC 2012).* 1–8.

[6] Ricardo Baeza-Yates, Luz Rello, and Julia Dembowski. 2015. CASSA: A context-aware synonym simplification algorithm. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (NAACL HLT).* 1380–1385.

[7] Dzmitry Bahdanau, Kyunghyun Cho, and Yoshua Bengio. 2014. Neural machine translation by jointly learning to align and translate. *arXiv preprint arXiv:1409.0473* (2014).

[8] Mokhtar B Billami, Thomas François, and Núria Gala. 2018. ReSyf: A French lexicon with ranked synonyms. In *Proceedings of the 27th International Conference on Computational Linguistics (COLING 2018).* 2570–2581.

[9] Or Biran, Samuel Brody, and Noémie Elhadad. 2011. Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers-Volume 2.* 496–501.

[10] Bernd Bohnet. 2009. Efficient parsing of syntactic and semantic dependency structures. In *Proceedings of the 13th Conference on Computational Natural Language Learning: Shared Task.* 67–72.

[11] Bernd Bohnet. 2010. Top accuracy and fast dependency parsing is not a contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (COLING 2010).* 89–97.

[12] Bernd Bohnet, Andreas Langjahr, and Leo Wanner. 2000. A development environment for an MTT-based sentence generator. In *Proceedings of the first international conference on Natural language generation-Volume 14.* 260–263.

[13] Stefan Bott, Luz Rello, Biljana Drndarevic, and Horacio Saggion. 2012. Can Spanish be simpler? LexSIS: Lexical simplification for Spanish. *Proceedings of the 24th International Conference on Computational Linguistics (COLING 2012)* (2012), 357–374.

[14] Stefan Bott, Horacio Saggion, and Simon Mille. 2012. Text Simplification Tools for Spanish.. In *The eighth international conference on Language Resources and Evaluation (LREC-12)*. 1665–1671.
[15] Nadjet Bouayad-Agha, Gerard Casamayor, Gabriela Ferraro, Simon Mille, Vanesa Vidal, and Leo Wanner. 2009. Improving the comprehension of legal documentation: the case of patent claims. In *Proceedings of the 12th International Conference on Artificial Intelligence and Law*. 78–87.
[16] Dominique Brunato, Andrea Cimino, Felice Dell'Orletta, and Giulia Venturi. 2016. PaCCSS-IT: A parallel corpus of complex-simple sentences for automatic text simplification. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. 351–361.
[17] Alicia Burga, Joan Codina, Gabriella Ferraro, Horacio Saggion, and Leo Wanner. 2013. The challenge of syntactic dependency parsing adaptation for the patent domain. In *The 25th European Summer School in Logic, Language and Information (ESSLLI 2013) Workshop on Extrinsic Parse Improvement*.
[18] John Carroll, Guido Minnen, Yvonne Canning, Siobhan Devlin, and John Tait. 1998. Practical Simplification of English Newspaper Text to Assist Aphasic Readers. In *Proceedings of AAAI-98 Workshop on Integrating Artificial Intelligence and Assistive Technology*. 7–10.
[19] Daniel Castro-Castro, Rocío Lannes-Losada, Montse Maritxalar, Ianire Niebla, Celia Pérez-Marqués, Nancy C Álamo-Suárez, and Aurora Pons-Porrata. 2008. A multilingual application for automated essay scoring. In *Ibero-American Conference on Artificial Intelligence*. 243–251.
[20] R. Chandrasekar, Christine Doran, and B. Srinivas. 1996. Motivations and Methods for Text Simplification. In *Proceedings of the 16th Conference on Computational Linguistics – Volume 2 (COLING '96)*. 1041–1044.
[21] Raman Chandrasekar and Bangalore Srinivas. 1997. Automatic induction of rules for text simplification1. *Knowledge-Based Systems* 10, 3 (1997), 183–190.
[22] Danqi Chen and Christopher Manning. 2014. A fast and accurate dependency parser using neural networks. In *Proceedings of the 2014 Conference on Empirical Methods in NLP (EMNLP)*. 740–750.
[23] Kyunghyun Cho, Bart Van Merriënboer, Caglar Gulcehre, Dzmitry Bahdanau, Fethi Bougares, Holger Schwenk, and Yoshua Bengio. 2014. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv preprint arXiv:1406.1078* (2014).
[24] Kostadin Cholakov, Chris Biemann, Judith Eckle-Kohler, and Iryna Gurevych. 2014. Lexical Substitution Dataset for German.. In *Eighth International Conference on Language Resources and Evaluation (LREC 2014)*. 1406–1411.
[25] James Clarke and Mirella Lapata. 2006. Models for sentence compression: A comparison across domains, training requirements and evaluation measures. In *Proceedings of the 21st International Conference on Computational Linguistics and the 44th Annual Meeting of the Association for Computational Linguistics*. 377–384.
[26] Bertol Arrieta Cortajarenak. 2010. *Azaleko sintaxiaren tratamendua ikasketa automatikoko tekniken bidez: ...*. Ph.D. Dissertation. Universidad del País Vasco-Euskal Herriko Unibertsitatea.
[27] William Coster and David Kauchak. 2011. Learning to simplify sentences using Wikipedia. In *Proceedings of the workshop on monolingual text-to-text generation*. 1–9.
[28] William Coster and David Kauchak. 2011. Simple English Wikipedia: a new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies (Vol 2: Short papers)*. 665–669.
[29] James R Curran, Stephen Clark, and Johan Bos. 2007. Linguistically motivated large-scale NLP with C&C and Boxer. In *Proceedings of the 45th Annual Meeting of the ACL*. 33–36.
[30] Edgar Dale and Jeanne S Chall. 1948. A formula for predicting readability: Instructions. *Educational Research Bulletin* 27, 1 (1948), 37–54.
[31] Jan De Belder and Marie-Francine Moens. 2012. A dataset for the evaluation of lexical simplification. In *International Conference on Intelligent Text Processing and Computational Linguistics*. 426–437.
[32] Marie-Catherine De Marneffe, Bill MacCartney, Christopher D Manning, et al. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the Fifth International Conference on Language Resources and Evaluation (LREC-2006), Genoa, Italy*, Vol. 6. 449–454.
[33] Arthur P Dempster, Nan M Laird, and Donald B Rubin. 1977. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)* 39, 1 (1977), 1–38.
[34] Michael Denkowski and Alon Lavie. 2011. Meteor 1.3: Automatic metric for reliable optimization and evaluation of machine translation systems. In *Proceedings of the sixth workshop on statistical machine translation*. 85–91.
[35] Jacob Devlin, Ming-Wei Chang, Kenton Lee, and Kristina Toutanova. 2018. BERT: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805* (2018).
[36] Siobhan Devlin and Gary Unthank. 2006. Helping aphasic people process online information. In *Proceedings of the 8th international ACM SIGACCESS conference on Computers and accessibility*. 225–226.
[37] Yuan Ding and Martha Palmer. 2005. Machine translation using probabilistic synchronous dependency insertion grammars. In *Proceedings of the 43rd Annual Meeting on ACL*. 541–548.

[38] George Doddington. 2002. Automatic evaluation of machine translation quality using n-gram co-occurrence statistics. In *Proceedings of the Second International Conference on Human Language Technology Research*. 138–145.

[39] Biljana Drndarević and Horacio Saggion. 2012. Towards automatic lexical simplification in Spanish: An empirical study. In *Proceedings of the NAACL-HLT 2012 Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR 2012)*. 8–16.

[40] Emmanuel Dupoux. 2018. Cognitive science in the era of artificial intelligence: A roadmap for reverse-engineering the infant language-learner. *Cognition* 173 (2018), 43–59.

[41] R. Evans. 2011. Comparing methods for the syntactic simplification of sentences in information extraction. *Literary and Linguistic Computing* 26, 4 (2011), 371–388.

[42] Richard Evans, Constantin Orasan, and Iustin Dornescu. 2014. An evaluation of syntactic simplification rules for people with autism. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations*. 131–140.

[43] Manaal Faruqui, Jesse Dodge, Sujay K Jauhar, Chris Dyer, Eduard Hovy, and Noah A Smith. 2014. Retrofitting word vectors to semantic lexicons. *arXiv preprint arXiv:1411.4166* (2014).

[44] Gabriela Ferraro. 2012. *Towards deep content extraction from specialized discourse: the case of verbal relations in patent claims.* Ph.D. Dissertation. Universitat Pompeu Fabra, Barcelona, Spain.

[45] Daniel Ferrés, Montserrat Marimon, Horacio Saggion, and Ahmed AbuRa'ed. 2016. YATS: Yet Another Text Simplifier. In *Natural Language Processing and Information Systems (LNCS, no. 9612)*, Elisabeth Métais, Farid Meziane, Mohamad Saraee, Vijayan Sugumaran, and Sunil Vadera (Eds.). Springer International Publishing, 335–342.

[46] Daniel Ferrés, Horacio Saggion, and Xavier Gómez Guinovart. 2017. An adaptable lexical simplification architecture for major Ibero-Romance languages. In *Proceedings of the First Workshop on Building Linguistically Generalizable NLP Systems*. 40–47.

[47] Katja Filippova and Michael Strube. 2008. Dependency tree based sentence compression. In *Proceedings of the 5th International Natural Language Generation Conference*. 25–32.

[48] Thomas François, Núria Gala, Patrick Watrin, and Cédrick Fairon. 2014. FLELex: a graded lexical resource for French foreign learners. In *Proceedings of the International conference on Language Resources and Evaluation (LREC '14)*. 3766—-3773.

[49] Núria Gala, Anaïs Tack, Ludivine Javourey-Drevet, Thomas François, and Johannes C. Ziegler. 2020. Alector: A Parallel Corpus of Simplified French Texts with Alignments of Misreadings by Poor and Dyslexic Readers. In *Proceedings of The 12th Language Resources and Evaluation Conference (LREC 2020)*. European Language Resources Association, Marseille, France, 1353–1361.

[50] Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2013. PPDB: The paraphrase database. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 758–764.

[51] Goran Glavaš and Sanja Štajner. 2013. Event-centered simplification of news stories. In *Proceedings of the Student Research Workshop associated with RANLP 2013*. 71–78.

[52] Goran Glavaš and Sanja Štajner. 2015. Simplifying lexical simplification: do we need simplified corpora?. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. 63–68.

[53] Chikio Hayashi. 1998. What is data science? Fundamental concepts and a heuristic example. In *Data Science, Classification, and Related Methods. Studies in Classification, Data Analysis, and Knowledge Organization*, Chikio Hayashi, Keiji Yajima, Hans-Hermann Bock, Noboru Ohsumi, Yutaka Tanaka, and Yasumasa Baba (Eds.). Springer, Japan, 40–51.

[54] Michael Heilman and Noah A. Smith. 2010. Good Question! Statistical Ranking for Question Generation. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 609–617.

[55] Sepp Hochreiter and Jürgen Schmidhuber. 1997. Long short-term memory. *Neural computation* 9, 8 (1997), 1735–1780.

[56] Colby Horn, Cathryn Manduca, and David Kauchak. 2014. Learning a lexical simplifier using Wikipedia. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 458–463.

[57] William Hwang, Hannaneh Hajishirzi, Mari Ostendorf, and Wei Wu. 2015. Aligning sentences from standard Wikipedia to simple Wikipedia. In *Proceedings of the 2015 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*. 211–217.

[58] Kentaro Inui, Atsushi Fujita, Tetsuro Takahashi, Ryu Iida, and Tomoya Iwakura. 2003. Text simplification for reading assistance: a project note. In *Proceedings of the 2nd International Workshop on Paraphrasing – Volume 16*. 9–16.

[59] Tomoyuki Kajiwara and Mamoru Komachi. 2016. Building a monolingual parallel corpus for text simplification using sentence similarity based on alignment between word embeddings. In *Proceedings of the 26th International Conference on Computational Linguistics (COLING 2016): Technical Papers*. 1147–1158.

[60] Tomoyuki Kajiwara and Kazuhide Yamamoto. 2015. Evaluation dataset and system for Japanese lexical simplification. In *Proceedings of the ACL-IJCNLP 2015 Student Research Workshop*. 35–40.

[61] Hans Kamp. 2008. *A Theory of Truth and Semantic Representation*. John Wiley & Sons, Ltd, Chapter 8, 189–222.

[62] J Peter Kincaid, Robert P Fishburne Jr, Richard L Rogers, and Brad S Chissom. 1975. Derivation of new readability formulas (automated readability index, fog count and flesch reading ease formula) for Navy enlisted personnel. (1975).

[63] Beata Beigman Klebanov, Kevin Knight, and Daniel Marcu. 2004. Text Simplification for Information-Seeking Applications. In *OTM Confederated International Conferences "On the Move to Meaningful Internet Systems"*. 735–747.

[64] Guillaume Klein, Yoon Kim, Yuntian Deng, Jean Senellart, and Alexander M Rush. 2017. OpenNMT: Open-source toolkit for neural machine translation. *arXiv preprint arXiv:1701.02810* (2017).

[65] Tomonori Kodaira, Tomoyuki Kajiwara, and Mamoru Komachi. 2016. Controlled and balanced dataset for japanese lexical simplification. In *Proceedings of the ACL 2016 Student Research Workshop*. 1–7.

[66] Philipp Koehn, Hieu Hoang, Alexandra Birch, Chris Callison-Burch, Marcello Federico, Nicola Bertoldi, Brooke Cowan, Wade Shen, Christine Moran, Richard Zens, et al. 2007. Moses: Open source toolkit for statistical machine translation. In *Proceedings of the 45th annual meeting of the ACL on interactive poster and demonstration sessions*. 177–180.

[67] Mathieu Lafourcade. 2007. Making people play for Lexical Acquisition with the JeuxDeMots prototype. In *Proceedings of the 7th International Symposium on Natural Language Processing (SNLP '07), Pattaya, Chonburi, Thailand*.

[68] Partha Lal and Stefan Rüger. 2002. Extract-based summarization with simplification. In *Proceedings of the ACL Workshop on Text Summarization (DUC 2002)*.

[69] Benoit Lavoie and Owen Rambow. 1997. A fast and portable realizer for text generation systems. In *Proceedings of the fifth conference on Applied natural language processing*. 265–268.

[70] Jonathan Mallinson and Mirella Lapata. 2019. Controllable Sentence Simplification: Employing Syntactic and Lexical Constraints. *arXiv preprint arXiv:1910.04387* (2019).

[71] Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the 4th International Workshop on Semantic Evaluations*. 48–53.

[72] Tomas Mikolov, Kai Chen, Greg Corrado, and Jeffrey Dean. 2013. Efficient estimation of word representations in vector space. *arXiv preprint arXiv:1301.3781* (2013).

[73] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in neural information processing systems*. 3111–3119.

[74] George A Miller, Richard Beckwith, Christiane Fellbaum, Derek Gross, and Katherine J Miller. 1990. Introduction to WordNet: An on-line lexical database. *International Journal of Lexicography* 3, 4 (1990), 235–244.

[75] Makoto Miwa, Rune Sætre, Yusuke Miyao, and Jun'ichi Tsujii. 2010. Entity-Focused Sentence Simplification for Relation Extraction. In *Proceeding of 23rd International Conference on Computational Linguistics*. 788–796.

[76] Shashi Narayan and Claire Gardent. 2014. Hybrid simplification using deep semantics and machine translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. 435–445.

[77] Gonzalo Navarro. 2001. A guided tour to approximate string matching. *ACM computing surveys* 33, 1 (2001), 31–88.

[78] Sergiu Nisioi, Sanja Štajner, Simone Paolo Ponzetto, and Liviu P Dinu. 2017. Exploring neural text simplification models. In *Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*. 85–91.

[79] Franz Josef Och and Hermann Ney. 2003. A systematic comparison of various statistical alignment models. *Computational linguistics* 29, 1 (2003), 19–51.

[80] Ethel Ong, Jerwin Damay, Gerard Lojico, Kimberly Lu, and Dex Tarantan. 2007. Simplifying text in medical literature. *Journal of Research in Science, Computing and Engineering* 4, 1 (2007), 37–47.

[81] Lluís Padró and Evgeny Stanilovsky. 2012. FreeLing 3.0: Towards wider multilinguality. In *Proceedings of the 8th Language Resources and Evaluation Conference (LREC 2012)*. Istanbul, Turkey, 2473–2479.

[82] Gustavo Paetzold and Lucia Specia. 2016. Benchmarking Lexical Simplification Systems. In *Tenth International Conference on Language Resources and Evaluation (LREC 2016)*. 3074–3080.

[83] Gustavo Paetzold and Lucia Specia. 2016. Semeval 2016 task 11: Complex word identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. 560–569.

[84] Gustavo Paetzold and Lucia Specia. 2017. Lexical simplification with neural ranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics (Vol 2, Short Papers)*. 34–40.

[85] Gustavo H Paetzold and Lucia Specia. 2013. Text simplification as tree transduction. In *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*. 116–125.

[86] Gustavo H Paetzold and Lucia Specia. 2016. Unsupervised Lexical Simplification for Non-Native Speakers. In *Proceedings of the 13th AAAI Conference on Artificial Intelligence (AAAI'16)*. 3761–3767.

[87] Kishore Papineni, Salim Roukos, Todd Ward, and Wei-Jing Zhu. 2002. BLEU: A method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting on association for computational linguistics*. 311–318.

[88] Jeffrey Pennington, Richard Socher, and Christopher Manning. 2014. GloVe: Global Vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*. 1532–1543.

[89] Basel Qenam, Tae Youn Kim, Mark J Carroll, and Michael Hogarth. 2017. Text simplification using consumer health vocabulary to generate patient-centered radiology reporting: translation and evaluation. *Journal of Medical Internet Research* 19, 12 (2017).

[90] Jipeng Qiang, Yun Li, Yi Zhu, Yunhao Yuan, and Xindong Wu. 2019. A Simple BERT-Based Approach for Lexical Simplification. *arXiv preprint arXiv:1907.06226* (2019).

[91] Luz Rello, Ricardo Baeza-Yates, Laura Dempere-Marco, and Horacio Saggion. 2013. Frequent words improve readability and short words improve understandability for people with dyslexia. In *IFIP Conference on Human-Computer Interaction*, Paula Kotzé, Gary Marsden, Gitte Lindgaard, Janet Wesson, and Marco Winckler (Eds.). 203–219.

[92] Jascha Rüsseler, Stefan Probst, Sönke Johannes, and Thomas F Münte. 2003. Recognition memory for high-and low-frequency words in adult normal and dyslexic readers: an event-related brain potential study. *Journal of Clinical and Experimental Neuropsychology* 25, 6 (2003), 815–829.

[93] Horacio Saggion. 2017. Automatic text simplification. *Synthesis Lectures on Human Language Technologies* 10, 1 (2017), 1–137.

[94] Horacio Saggion, Sanja Štajner, Stefan Bott, Simon Mille, Luz Rello, and Biljana Drndarevic. 2015. Making it simplext: Implementation and evaluation of a text simplification system for Spanish. *ACM Transactions on Accessible Computing* 6, 4 (2015), 14:1–14:36.

[95] Carolina Scarton, Alessio Palmero Aprosio, Sara Tonelli, Tamara Martín Wanton, and Lucia Specia. 2017. MUSST: A Multilingual Syntactic Simplification Tool. In *Proceedings of the International Joint Conference on Natural Language Processing and workshops (IJCNLP 2017)*. 25–28.

[96] Isabel Segura-Bedmar and Paloma Martínez. 2017. Simplifying drug package leaflets written in Spanish by using word embedding. *Journal of biomedical semantics* 8, 1 (2017), 45.

[97] Matthew Shardlow. 2013. A Comparison of Techniques to Automatically Identify Complex Words. In *The 51st Annual Meeting of the ACL Proceedings of the Student Research Workshop*. 103–109.

[98] Matthew Shardlow. 2013. The CW Corpus: A new resource for evaluating the identification of complex words. In *Proceedings of 2nd Workshop on Predicting and Improving Text Readability for Target Reader Populations*. 69–77.

[99] Matthew Shardlow. 2014. Out in the Open: Finding and Categorising Errors in the Lexical Simplification Pipeline. In *The 9th International Conference on Language Resources and Evaluation (LREC 2014)*. 1583–1590.

[100] Matthew Shardlow. 2014. A Survey of Automated Text Simplification. *International Journal of Advanced Computer Science and Applications (IJACSA)* 4, 1 (2014), 58–70.

[101] Advaith Siddharthan. 2006. Syntactic simplification and text cohesion. *Research on Language and Computation* 4, 1 (2006), 77–109.

[102] Advaith Siddharthan. 2011. Text simplification using typed dependencies: A comparison of the robustness of different generation strategies. In *Proceedings of the 13th European Workshop on Natural Language Generation*. 2–11.

[103] Advaith Siddharthan. 2014. A survey of research on text simplification. *International Journal of Applied Linguistics* 165, 2 (2014), 259–298.

[104] Advaith Siddharthan and Angrosh Annayappan Mandya. 2014. Hybrid text simplification using synchronous dependency grammars with hand-written and automatically harvested rules. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics (EACL 2014)*. 722–731.

[105] Advaith Siddharthan, Ani Nenkova, and Kathleen McKeown. 2004. Syntactic Simplification for Improving Content Selection in Multi-document Summarization. In *Proceedings of the 20th International Conference on Computational Linguistics (COLING '04)*. Article 896.

[106] Sara Botelho Silveira and António Branco. 2012. Combining a double clustering approach with sentence simplification to produce highly informative multi-document summaries. In *IEEE 13th International Conference on Information Reuse and Integration (IRI)*. 482–489.

[107] David A Smith and Jason Eisner. 2006. Quasi-synchronous grammars: Alignment by soft projection of syntactic dependencies. In *Proceedings of the Workshop on Statistical Machine Translation (StatMT '06)*. 23–30.

[108] Matthew Snover, Bonnie Dorr, Richard Schwartz, Linnea Micciulla, and John Makhoul. 2006. A study of translation edit rate with targeted human annotation. In *Proceedings of Association for Machine Translation in the Americas*. 223–231.

[109] Matthew Snover, Nitin Madnani, Bonnie J Dorr, and Richard Schwartz. 2009. Fluency, adequacy, or HTER?: exploring different human judgments with a tunable MT metric. In *Proceedings of the 4th Workshop on Statistical Machine Translation*. 259–268.

[110] Lucia Specia. 2010. Translating from complex to simplified sentences. In *International Conference on Computational Processing of the Portuguese Language*. 30–39.
[111] Lucia Specia, Sujay Kumar Jauhar, and Rada Mihalcea. 2012. Semeval-2012 task 1: English lexical simplification. In *Proceedings of the First Joint Conference on Lexical and Computational Semantics (*SEM)*. 347–355.
[112] Sanja Štajner and Goran Glavaš. 2017. Leveraging event-based semantics for automated text simplification. *Expert systems with applications* 82 (2017), 383–395.
[113] Sanja Štajner, Horacio Saggion, and Simone Paolo Ponzetto. 2019. Improving lexical coverage of text simplification systems for Spanish. *Expert Systems with Applications* 118 (2019), 80–91.
[114] Elior Sulem, Omri Abend, and Ari Rappoport. 2018. Simple and Effective Text Simplification Using Semantic and Neural Methods. *arXiv preprint arXiv:1810.05104* (2018).
[115] Hong Sun and Ming Zhou. 2012. Joint learning of a dual SMT system for paraphrase generation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Vol 2: Short Papers)*. 38–42.
[116] Ilya Sutskever, Oriol Vinyals, and Quoc V Le. 2014. Sequence to sequence learning with neural networks. In *Advances in neural information processing systems*. 3104–3112.
[117] Kristina Toutanova and Christopher D Manning. 2000. Enriching the knowledge sources used in a maximum entropy part-of-speech tagger. In *Proceedings of the Joint SIGDAT conference on Empirical methods in natural language processing and very large corpora: held in conjunction with the 38th Annual Meeting of the Association for Computational Linguistics*. 63–70.
[118] Sowmya Vajjala and Ivana Lucic. 2018. OneStopEnglish corpus: A new corpus for automatic readability assessment and text simplification. In *Proceedings of the 13th Workshop on Innovative Use of NLP for Building Educational Applications*. 297–304.
[119] David Vickrey and Daphne Koller. 2008. Sentence Simplification for Semantic Role Labeling. In *Proceedings of the 46th Annual Meeting of the Association for Computational Linguistics (ACL '08)*. 344–352.
[120] Piek Vossen (Ed.). 1998. *EuroWordNet: A multilingual database with lexical semantic networks.* Kluwer Academic Publishers.
[121] Tong Wang, Ping Chen, Kevin Amaral, and Jipeng Qiang. 2016. An experimental study of LSTM encoder-decoder model for text simplification. *arXiv preprint arXiv:1609.03663* (2016).
[122] Rodrigo Wilkens, Leonardo Zilio, Silvio Ricardo Cordeiro, Felipe Paula, Carlos Ramisch, Marco Idiart, and Aline Villavicencio. 2017. LexSubNC: A Dataset of Lexical Substitution for Nominal Compounds. In *12th International Conference on Computational Semantics (IWCS 2017) — Short papers.*
[123] Ronald J Williams. 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Machine learning* 8, 3–4 (1992), 229–256.
[124] Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the Conference on Empirical Methods in NLP*. 409–420.
[125] Sander Wubben, Antal Van Den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: (Vol 1: Long papers)*. 1015–1024.
[126] Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association of Computational Linguistics* 3, 1 (2015), 283–297.
[127] Wei Xu, Courtney Napoles, Ellie Pavlick, Quanze Chen, and Chris Callison-Burch. 2016. Optimizing statistical machine translation for text simplification. *Transactions of the Association for Computational Linguistics* 4 (2016), 401–415.
[128] Kenji Yamada and Kevin Knight. 2001. A syntax-based statistical translation model. In *Proceedings of the 39th Annual Meeting on Association for Computational Linguistics*. 523–530.
[129] Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*. 365–368.
[130] Xingxing Zhang and Mirella Lapata. 2017. Sentence simplification with deep reinforcement learning. *arXiv preprint arXiv:1703.10931* (2017).
[131] Yaoyuan Zhang, Zhenxu Ye, Yansong Feng, Dongyan Zhao, and Rui Yan. 2017. A constrained sequence-to-sequence neural model for sentence simplification. *arXiv preprint arXiv:1704.02312* (2017).
[132] Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics*. 1353–1361.