Natural Language Engineering **1** (1): 1–54. Printed in the United Kingdom

Identifying Signs of Syntactic Complexity for Rule-Based Sentence Simplification

RICHARD EVANS

CONSTANTIN ORĂSAN

Research Institute in Information and Language Processing,
University of Wolverhampton, UK
e-mail: {R.J.Evans, C.Orasan}@wlv.ac.uk

(*Received 11 May 2013; revised 07 August 2013*)

Abstract

This article presents a new method to automatically simplify English sentences. The approach is designed to reduce the number of compound clauses and nominally bound relative clauses in input sentences. The article provides an overview of a corpus annotated with information about various explicit signs of syntactic complexity and describes the two major components of a sentence simplification method that works by exploiting information on the signs occurring in the sentences of a text. The first component is a sign tagger which automatically classifies signs in accordance with the annotation scheme used to annotate the corpus. The second component is an iterative rule-based sentence transformation tool. Exploiting the sign tagger in conjunction with other NLP components, the sentence transformation tool automatically rewrites long sentences containing compound clauses and nominally bound relative clauses as sequences of shorter single-clause sentences. Evaluation of the different components reveals acceptable performance in rewriting sentences containing compound clauses but less accuracy when rewriting sentences containing nominally bound relative clauses. A detailed error analysis revealed that the major sources of error include inaccurate sign tagging, the relatively limited coverage of the rules used to rewrite sentences, and an inability to discriminate between various subtypes of clause coordination. Despite this, the system performed well in comparison with two baselines. This finding was reinforced by automatic estimations of the readability of system output and by surveys of readers' opinions about the accuracy, accessibility, and meaning of this output.

1 Introduction

In this article, we present an automatic method to simplify sentences on the basis of their syntactic structure with the aim of reducing the number of compound clauses and nominally bound relative clauses that they contain. Developed within the EC-funded FIRST project,[1] the method was integrated within a system

[1] `rgcl.wlv.ac.uk/projects/FIRST` (last accessed on 14th September 2018).

called *OpenBook* designed to assist carers in the conversion of texts into a more accessible form for people with autistic spectrum disorder (ASD). Many of the language technology components integrated in the system, including the sentence simplification component, are beneficial for other types of user. In this article, we refer to our automatic sentence simplification tool as OB1. It is applicable in a wide range of contexts in which sentence simplification can facilitate human and machine processing of language.

Research carried out in the FIRST project revealed that people with ASD find it difficult to read complex sentences, compound sentences, and simple sentences containing complex phrases (Martos, Freire, González, Gil, Evans, Jordanova, Cerga, Shishkova, and Orasan, 2013). Quirk, Greenbaum, Leech, and Svartvik (1985) note that the elements of clause structure in English are SUBJECT, VERB, OBJECT, COMPLEMENT, and ADVERBIAL. In this context,

- *complex sentences* are sentences in which one of the elements is realised by a subordinate clause,
- *compound sentences* are sentences of which the immediate constituents are two or more coordinate clauses, and
- *simple sentences* are independent clauses for which no element is clausal. Elements realised as phrases in simple sentences may themselves be complex and include embedded clauses of various types, including compounds (e.g. OBJECT elements realised as noun phrases with post-modifying relative clauses).

People with ASD have comparatively short working memory span (Bennetto, Pennington, and Rogers, 1996) and find it easier to process sentences less than 15 words long (Martos et al., 2013). Caplan and Waters (1999) note that the text comprehension of people with different levels of verbal working memory capacity depends on the number of propositions conveyed in the sentences that texts contain. Propositions are atomic statements that express simple factual claims (Jay, 2003). They are considered the basic units involved in the understanding and retention of text (Kintsch and Welsch, 1991).

Sentences containing compound clauses and nominally bound relative clauses[2] convey relatively large numbers of propositions and have a large *propositional density*, which is defined as the ratio of propositions to words in a sentence (DeFrancesco and Perkins, 2012). As a result, they can adversely affect the speed and accuracy of syntactic processing of a wide range of readers. Examples (1) and (2) are sentences containing a compound clause and a nominally bound relative clause, respectively.[3]

[2] In this article, we use the term nominally bound to denote relative clauses that modify a head noun and that are introduced by a relative pronoun whose interpretation is derived from that noun.

[3] The examples presented in this article are indexed using numbers in parentheses and were selected from our annotated corpus (Section 3).

(1) But [[she displays the same ingrained showmanship]; and [she plays the same straight bat to questions about his inner compulsions]].

(2) [Anaïs, [who was conceived at Christian Dior's house in Montreux]], was said to share her mother's blonde hair and blue eyes.

The method we describe in the current article automatically detects clause compounds and nominally bound relative clauses and rewrites sentences containing them as sequences of sentences containing fewer clauses. The propositional density of input sentences is thus reduced, as is the minimum verbal working memory span required for their comprehension.

Our new method for sentence simplification is designed to rewrite sentences containing compound clauses and nominally bound relative clauses without exploiting a syntactic parser, reducing the number of these constituents that they contain. An expected by-product of the rewriting process is a reduction in the propositional density of the texts being processed. In this article, for brevity, we refer to sentences containing compound clauses as *Type 1* sentences and sentences containing nominally bound relative clauses as *Type 2* sentences. For Type 1 sentences, the sentence rewriting process depends on accurate identification of compound clauses and their conjoins. For Type 2 sentences, it depends on accurate identification of bound relative clauses and the matrix elements that they modify. In this article, we present the two main components of our method. These include a ML approach to categorise various explicit signs of syntactic complexity with respect to their syntactic linking and bounding functions (sign tagging) and a rule-based method for sentence transformation which exploits sign tagging.

The work described here draws on our previous work on the development of a method for sentence simplification for use in biomedical information extraction (Evans, 2011), the development of a corpus annotated with information about the linking and bounding functions of explicit signs of syntactic complexity (Evans and Orasan, 2013), and the development of an automatic method to classify such signs (Dornescu, Evans, and Orasan, 2013). The research presented in this article extends this work in several ways, described below.

Evans (2011) presented a rule-based method for sentence simplification that is based on a shallow sentence analysis step and an iterative sentence transformation step. The main contributions of that method were a new approach to automatic sentence analysis and a method for rewriting sentences on the basis of that analysis. The analysis step includes:

1. tokenisation of input texts to enable identification of sentences, words, and a pre-specified set of potential coordinators,[4]
2. part of speech tagging, and
3. a ML method to classify potential coordinators.

[4] Comprising commas, conjunctions, and adjacent comma-conjunction pairs, these potential coordinators comprise only a subset of the signs of syntactic complexity discussed in Section 3 of the current article.

The classification scheme used in that work provides detailed information on a wide range of clausal and sub-clausal types of coordination but offers limited information about different types of subordination. Despite this, Evans's (2011) approach proved to be useful in a biomedical information extraction task and compared favourably with an approach based on full syntactic analysis using the Stanford parser.

The method for sentence simplification that we present in our current article differs from that of Evans (2011) by nature of the fact that Evans's system was designed to process text of a restricted type (clinical vignettes), containing a more restricted range of syntactic structures. For simplification of Type 1 sentences, the sentence transformation rule set used in Evans's (2011) system comprised just four rules. Lacking information about many subordinate clause boundaries, his system is unable to simplify sentences containing the types of syntactic complexity prevalent in texts of the registers of news and literature. It is incapable of simplifying Type 2 sentences. By contrast, the system that we present in our current article is able to simplify sentences containing a wider range of syntactic structures and was developed for use with texts of multiple registers. In terms of evaluation, the output produced by Evans's (2011) system was not assessed intrinsically or with respect to grammatical correctness, readability, or meaning. In the current article, we use these criteria to evaluate the output of our system and compare its performance with that of two baseline systems.

In previous work, Evans and Orasan (2013) described the development of a corpus of three registers annotated with information about the specific linking and bounding functions of various explicit signs of syntactic complexity. Dornescu et al. (2013) described the development of a sequence tagging approach to automatically classify such signs with respect to the annotation scheme presented by Evans and Orasan (2013). We summarise the research presented in these papers in Sections 3 and 4 of the current article. We provide more detailed evaluation of the sign tagger than was included in the paper by Dornescu et al. (2013) and include analysis of a confusion matrix, presentation of the 95% confidence interval of its micro-averaged F_1 score, and explore the impact of sign tagging on the task of sentence simplification. We also compare the accuracy of our sign tagger with that of a majority class baseline.

The remainder of our article is structured as follows. Section 2 provides an overview of previous work in sentence simplification. Section 3 describes the signs of syntactic complexity addressed in our research, the corpus annotated with information about those signs, and the annotation scheme used for this purpose. It also presents the findings of a corpus analysis. Section 4 presents our method for automatic sentence analysis which exploits a machine learning method to classify a range of explicit signs of syntactic complexity in accordance with our annotation scheme. Our approach to sentence transformation is described in Section 5. It takes automatically analysed sentences as input and uses an iterative rule-based method to convert multi-clause sentences into sequences of sentences containing fewer clauses (Section 5.1). Evaluation of the sentence simplification method and its two components (analysis and transformation) is presented in Section 6. Our

evaluation of the sentence transformation process includes comparisons of system output with human-produced simplifications of test sentences (Section 6.2.1), analysis of changes in the estimated readability of sentences transformed using the simplification method (Section 6.2.2), and investigation of readers' opinions on the grammaticality, accessibility, and meaning of automatically transformed sentences (Section 6.2.3). In Section 7, conclusions are drawn and possible directions for future work considered.

2 Related Work

In this section, we provide a survey of work related to the task of sentence simplification, with an emphasis on those methods which exploit information about the syntactic structure of input sentences for the transformation process. For brevity, as it is not the focus of our method, we do not cover previous work related to the task of lexical simplification here.

Automatic sentence simplification is one aspect of text simplification, a topic that has been addressed in several lines of research since the 1990s. Numerous rule-based methods for sentence simplification have been developed (e.g. Chandrasekar, Doran, and Srinivas (1996); Siddharthan (2006); De Belder and Moens (2010)) and used to facilitate NLP tasks such as biomedical information extraction (Agarwal and Boggess, 1992; Rindflesch, Rajan, and Hunter, 2000), semantic role labelling (Vickrey and Koller, 2008), and dependency parsing (Jelínek, 2014).

Previous work has addressed the task by exploitation of a range of language resources and NLP tools, including shallow pre-processing (e.g. Siddharthan (2006)) and syntactic parsing tools (e.g. Canning (2002); Vickrey and Koller (2008); Bott, Saggion, and Figueroa (2012)), sentence-aligned parallel corpora of texts in their original form and in a manually simplified form (e.g. Coster and Kauchak (2011); Wubben, van den Bosch and Krahmer (2012); Štajner, Calixto, and Saggion (2015)), and syntactically-annotated versions of such corpora (e.g. Zhu, Bernhard, and Gurevych (2010); Feblowitz and Kauchak (2013); Siddharthan and Angrosh (2014)). In sections 2.1–2.2 we present an overview of the most relevant previous research in sentence simplification, highlighting differences between the methods used in those approaches and those used in the sentence simplification system that we present in the current article.

2.1 Rule-Based Approaches

In many of the approaches exploiting shallow pre-processing, rules are triggered by pattern-matching applied to the output of text analysis tools such as partial parsers and part-of-speech (PoS) taggers. Siddharthan (2006) describes a method in which input text is analysed using a tokeniser, chunker, and PoS tagger. In this approach, handcrafted patterns are used to identify the grammatical roles of NPs, to resolve pronominal anaphora, and to "split" complex sentences containing relative clauses and compound constituents, including clausal and sub-clausal constituents. The handcrafted patterns are expressed in terms of prefix conjunctions (e.g. *though*,

when, *if*) and infix conjunctions (e.g. *and*, *but*, *because*), and commas. The method is based on an iterative simplification method exploiting rules which include operations for sentence ordering, for ensuring anaphoric cohesion, for preserving rhetorical relations, and for generating appropriate determiners when splitting sentences that contain relative clauses. In some respects, Siddharthan's (2006) method is similar to the one we propose in this article. However, the transformation rules used in his system are based on shallow information such as part of speech and chunk patterns and no disambiguation of conjunctions or commas is performed. The rules used by our new system for sentence simplification also exploit information about the coordinating and bounding functions of various lexical and punctuational markers of syntactic complexity. Our approach integrates a ML-based classifier of these markers (Section 4) to provide a more detailed analysis of input sentences. Evans's (2011) approach, discussed in Section 1, is another example of rule-based sentence simplification.

2.1.1 Methods Exploiting Syntactic Parsing

A large number of sentence simplification methods proposed in the past exploit automatic sentence analysis using syntactic parsers. These include techniques based on handcrafted transformation rules operating over the derived syntactic structure of input sentences and extraction of the syntactic relations or dependencies between words and the syntactic roles of constituents identified in those sentences. In many cases, the syntactic transformation rules employed in these methods are implemented using synchronous grammars rather than surface-based text editing operations. They are typically used to simplify input sentences by re-ordering constituents or splitting sentences that contain compounds or complex constituents into simple sentences containing independent clauses (Angrosh and Siddharthan, 2014; Ferrés, Marimon, and Saggion, 2015; Mishra, Soni, Sharma, and Sharma, 2014; Rennes and Jönsson, 2002).

In previous work, several applications have been developed with the aim of improving text accessibility for human readers. Max (2000) described the use of a syntactic parser for sentence rewriting to facilitate the reading comprehension of people with aphasia. In the PSET project, Canning (2002) implemented a system which exploits a parser in order to rewrite compound sentences as sequences of simple sentences and to convert passive sentences into active ones. Scarton, Palmero Aprosio, Tonelli, Martin-Wanton, and Specia (2017) developed a multilingual syntactic simplification tool (MUSST) in the SIMPATICO project, which sought to improve the experience of citizens and companies in their daily interactions with public administration. The English sentence simplification tool includes components for sentence analysis, exploiting the Stanford dependency parser (de Marneffe, MacCartney, and Manning, 2006), to determine whether or not input sentences should be transformed, and to identify discourse markers and relative pronouns, which will be useful in the simplification of conjoint (compound) clauses and relative clauses. MUSST's syntactic simplification process implements the handcrafted rules proposed by Siddharthan (2004) and Siddharthan and Angrosh (2014) and applies

them to the syntactic analyses generated for input sentences by the dependency parser. These include rules to split sentences containing conjoint clauses, relative clauses, and appositive phrases and to convert passive sentences to active. After applying the simplification rules, MUSST also performs a generation step in which truecasing is applied to output sentences and discourse markers lost in the simplification process are re-inserted. When processing sentences in Public Administration texts, Scarton et al. (2017) report an accuracy of 76% for their system when simplifying English sentences, and taking into account a wider range of operations that those that are the focus of our paper. In Section 6.2, we compare the accuracy of MUSST with that of our approach, focusing only on the task of simplifying Type 1 and Type 2 sentences in texts of several different registers.

Although we focus on simplification of English sentences in this article, rule-based methods have also been proposed for the processing of other languages (e.g. Daelemans, Höthker, and Tjong Kim Sang (2004); Brouwers, Bernhard, Ligozat, and Francois (2014); Suter, Ebling, and Volk (2016)). Several researchers have also developed methods to facilitate the process of acquiring sentence simplification rules from manually simplified corpora of languages such as Brazilian Portuguese (Aluisio, Specia, Pardo, Maziero, and Fortes, 2008; Aluisio, Specia, Pardo, Maziero, Caseli, and Fortes, 2008) and Basque (Gonzalez-Dios, Aranzabe, and Diaz de Ilarraza, 2018). Seretan (2012) presented a method to semi-automatically derive syntactic simplification rules for French sentences. Her method is based on a component to automatically identify sentences that require simplification and on manual analysis of the syntactic structures of complex and simple French sentences. The outputs of these two processes are then used to formulate rules to transform sentences with complex syntactic structures into sentences with simpler syntactic structures. Due to its modular design, we expect that the method for sentence simplification that we present in the current article will be easy to port for use with other languages. However, we have not so far made this type of adaptation.

In general, the weakness of approaches exploiting syntactic parsing is that they rely on high levels of accuracy and granularity of automatic syntactic analysis. Previous research has demonstrated that the accuracy of parsers is inversely proportional to the length and complexity of the sentences being analysed (Tomita, 1985; McDonald and Nivre, 2011). Rather than exploiting full syntactic parsing, the approach to sentence simplification that we present in this article exploits a shallow and robust syntactic analysis step.

2.2 Data-Driven Approaches

More recently, the availability of resources such as Simple Wikipedia has enabled text simplification to be included in the paradigm of statistical machine translation (Yatskar, Pang, Danescu-Niculescu-Mizil, and Lee, 2010; Coster and Kauchak, 2011). In this context, translation models are learned by aligning sentences in English Wikipedia (EW) with their corresponding versions in Simple English Wikipedia (SEW). Manifesting Basic English (Ogden, 1932), the extent to which

SEW is accessible to people with reading difficulties has not yet been fully assessed. Effective SMT relies on the availability of representative pairs of texts in their original and converted forms. As a result, there are currently only a limited number of contexts in which SMT approaches are likely to be effective. Xu, Callison-Burch, and Napoles (2015) are critical of the use of SEW to support SMT-based text simplification.

2.2.1 Methods Exploiting Parallel Corpora

Despite these caveats, the availability of sentence-aligned parallel corpora of texts in their original and simplified forms provides additional opportunities to develop methods for sentence simplification. From a sentence-aligned collection of articles from EW and SEW, Coster and Kauchak (2011) derived the probabilities that various ngrams from EW occur in an edited form in SEW as a result of various transformation operations including phrase rewording, deleting, reordering, and splitting. They exploited the resultant phrase table in the development of a phrase-based statistical machine translation (PB-SMT) model to translate texts into a simplified form.

Wubben et al. (2012) applied this basic approach and also integrated a re-ranking metric to ensure that sentences generated by the model are sufficiently unlike the originals to constitute suitable transformations. This approach captures the intuition that generated sentences should be as fluent and informative as the originals, but sufficiently different from them. The models learned perform lexical substitutions, phrase deletion, and phrase reordering.

Štajner et al. (2015) exploited a sentence-aligned parallel corpus of Spanish containing texts in their original versions and two simplified versions of decreasing complexity (manifesting "light" and "heavy" simplification). They applied methods from SMT to learn the simplification model and developed a language model for use by it from a set of sentences of restricted length (fifteen words or less). This was done to promote the simplicity of sentences generated by the system.

Zhang and Lapata (2017) applied methods from neural machine translation to develop DRESS, the deep reinforcement learning sentence simplification system. Trained on parallel corpora of unmodified and manually simplified English text, their method uses recurrent neural networks to implement an encoder-decoder architecture network to transform input word sequences into a simplified form. The system was trained in a reinforcement learning framework to ensure that the generated output satisfies constraints on simplicity, fluency, and meaning. The types of transformation operation learned by the model which affect sentence structure include those performed by other systems described in this section: addition, copying, deletion, and re-ordering of words and phrases.

The approach to sentence simplification that we present in this article does not depend on the availability of parallel corpora of text in its original form and in a manually simplified form. It does not apply text editing operations of the type used

in phrase-based machine translation or neural machine translation. Our approach is iterative and rule-based rather than exploiting empirically derived phrase tables.

2.2.2 Methods Exploiting Syntactically Parsed Parallel Corpora

Several methods for sentence simplification have exploited syntactically annotated sentence-aligned parallel corpora of texts in their original and simplified forms. Zhu et al. (2010) developed an approach to sentence simplification using PB-SMT. Data from a sentence-aligned parallel corpus of EW/SEW articles was syntactically parsed. This syntactic information was exploited when computing the probabilities of transformation operations applied to sentences in EW generating the aligned sentences in SEW. The approach was able to derive these probabilities from information about syntactic structure such as constituent size and information about the occurrence in the constituent of relative pronouns. A PB-SMT approach was then used to learn syntactic transformation operations from this data. The types of transformations learned included phrase splitting, dropping and reordering, and substitution operations.

Feblowitz and Kauchak (2013) presented a method in which syntactic transformation rules are learned from a syntactically annotated parallel corpus of texts in their original and simplified forms. The rules were encoded in a synchronous tree substitution grammar (STSG) formalism, which models syntactic simplification. The authors improved the simplification model by incorporating additional syntactic information to better discriminate between input structures for which transformations should be applied and those for which they should not.

Paetzold and Specia (2013) developed a system exploiting the *Tree Transducer Toolkit* to learn syntactic transformation rules from a syntactically parsed parallel corpus of texts in their original and simplified forms. The acquired rules are applied to input sentences parsed using the Stanford constituent parser.[5] Transformations learned in this approach include lexical and syntactic simplifications. The authors developed a set of heuristic filters to prevent the system from learning spurious rules. These filters ensure that, in order to be incorporated in the model, a candidate rule must either be general enough, must split one sentence into multiple sentences, must delete information, or must apply to structures which contain connectors such as *and* and *or*.

Angrosh, Nomoto, and Siddharthan (2014) developed an approach incorporating one method for syntactic and lexical simplification and a second method for sentence compression. Lexicalised sentence transformation rules were learned from a syntactically parsed parallel corpus. These rules included both lexical and syntactic transformations. The sentence compression method employed techniques from integer linear programming and dynamic programming to select the best from among a large set of candidate node deletions to be applied to the syntactically analysed input sentences.

[5] Available at `https://nlp.stanford.edu/software/srparser.html` (last accessed on 11th September 2018).

Siddharthan and Angrosh (2014) present a method exploiting techniques from PB-SMT to learn a synchronous grammar applying transformations to parse trees for text simplification. In their method, the synchronous grammar is semi-automatically acquired from a syntactically parsed sentence-aligned parallel corpus of articles from EW and SEW. These transformations include lexicalised rules for insertion, deletion, and re-ordering of syntactic constituents together with morphological transformation of verb forms to enable conversion of sentences from passive voice to active. A substantial part of the grammar consists of handcrafted rules to enable transformations that are more difficult to learn, such as the conversion of passive sentences to active, the splitting of complex sentences and compounds, and the standardisation of quotations. The authors applied a simple text generation component to ensure that sentences produced by the system are ordered in a way that matches that of the original text.

Narayan and Gardent (2014) present a method for sentence simplification in which a phrase-based SMT model learned from a parallel corpus of sentence-aligned EW and SEW articles is improved through the integration of deep semantic information. This is derived from *Boxer* (Bos, 2008), a tool which provides information on the discourse representation structure of input sentences. Semantic information from this parser is used to improve the splitting of complex sentences by ensuring preservation of multi-word units (entities and concepts) in the generated sentences and by avoiding the deletion of the obligatory arguments of verbs.

The field of text summarisation also includes approaches that exploit sentence rewriting. For example, Cohn and Lapata (2009) present a syntactic tree-to-tree transduction method to filter non-essential information from syntactically parsed sentences. This compression process often reduces the syntactic complexity of those sentences. An advantage of their method over the one that we present in this article is that it can identify elements for deletion in the absence of explicit signs of syntactic complexity. However, as with all methods exploiting full syntactic parsing, the approach is computationally expensive, with relatively long run times. One recent approach to sentence compression was presented by Klerke, Goldberg, and Søgaard (2016). Their method exploits LSTM learning in a joint-learning task to integrate information from CCG supertagging and eye tracking data for sentence simplification. The model is used to compress sentences by identifying non-essential words and phrases for deletion. The methods proposed by Cohn and Lapata (2009) and Klerke et al. (2016) both work by applying deletion operations. As with all such methods, they run the risk of omitting relevant information in their output.

In addition to the exploitation of handcrafted rules, several systems based on a syntactic analysis of input texts include a post-processing module to improve the quality of the sentences that they generate. Bott et al. (2012) integrated a probabilistic component into their system to assess the suitability of applying transformation operations to input sentences. This approach to syntactic simplification was integrated into the *Simplex* text simplification system, designed to convert texts into a more accessible form for people with Down's syndrome (Saggion, Štajner, Bott, Mille, Rello, and Drndarevic, 2015). Vickrey and Koller (2008) included a machine learning method in their sentence simplification tool

to decide on the set of transformations to apply when processing input sentences. In addition to a syntactic dependency analysis, Siddharthan (2011) integrated a generator in his system to sequence the application of handcrafted transformation rules and to ensure agreement between constituents in automatically generated sentences. Brouwers et al. (2014) developed an approach in which all possible transformations in their grammar are applied to input sentences and a method using integer linear programming and four assessment criteria is used to select the best of these.

The methods described in this section rely on two resources that are not exploited by the approach to sentence simplification that we present in this article: syntactic parsers and large syntactically analysed corpora. The former are ill-suited to the sentence simplification task, as noted in Section 2.1.1, while the latter are relatively scarce resources that are not available for texts of all registers. These methods involve the learning of sentence transformation rules from syntactically parsed parallel corpora. The rules used by our sentence simplification system are not derived in this way. They are based on a shallow and robust analysis step in which explicit signs of syntactic complexity are classified with respect to their specific syntactic coordinating and subordinating functions. In our context, implicit syntactic structure is not tagged directly, but is inferred from the classification of explicit signs.

2.2.3 Methods Exploiting Deep Parsing and Semantic Analysis

Several methods for sentence simplification exploit deep parsing and automatic methods for semantic analysis. Jonnalagadda, Tari, Hakenberg, Baral, and Gonzalez (2009) presented a method for syntactic simplification which includes a pre-processing step in which spurious phrases are deleted, the names of certain entities (genes) are normalised, and noun phrases are replaced. After pre-processing, input sentences containing multiple clauses are split into independent clauses using information about linkage relations identified by the Link Grammar parser[6] and about the distribution of commas in the sentences.

Miwa, Sætre, Miyao, and Tsujii (2010) applied a method based on deep parsing to pre-process sentences, removing unnecessary information to expedite a relation extraction task. They developed handcrafted rules to identify entity mentions, relative clauses, and copulas in sentences and to exploit the syntactic analysis to delete those parts of the sentence not mentioning entities. Transformation operations performed in this approach include the replacement of compound phrases by the final conjoin in the phrase that refers to an entity of interest and the deletion of matrix NPs with appositions referring to an entity of interest.

Sheremetyeva (2014) presents an approach to sentence simplification in patent claims. Her method exploits a variety of advanced pre-processing steps including

[6] Available at `https://www.abisource.com/projects/link-grammar` (last accessed on 11th September 2018).

supertagging to identify semantic information about the words, terminology, and predicates in the text. Phrases are identified using a chunker based on phrase structure grammar rules and relations between predicates and their arguments are identified using an approach based on domain permutation graphs. These tools are used to identify the full set of predicate-argument dependencies in input sentences and to generate new simple sentences on the basis of each of them. Our approach, which is used to process texts of multiple registers and domains, performs no deep semantic analysis of this kind. We suspect that the approach presented by Sheremetyeva (2014), involving extraction of predicate-argument dependencies and sentence generation, would be difficult to adapt for accurate simplification of texts that are not patent claims.

Sections 2.2.2 and 2.2.3 have included descriptions of several sentence compression methods. One disadvantage of such methods is that they are "destructive" in the sense that information is deleted rather than preserved as a result of compression. Although some information loss is inevitable in text simplification, our method is designed to minimise it. Currently, information conveyed by conjunctions and other signs of syntactic complexity is lost in our approach but information conveyed by other function words and content words is preserved.

To conclude this review of related work, we noted that many of the previous rule-based approaches to sentence simplification are based on a relatively coarse analysis of syntactic structure and are often designed for use in a specific application area, such as domain-specific information extraction. In our case, we sought to develop an approach incorporating an analysis step detailed enough to support simplification of a variety of syntactically complex structures. Approaches based on the full syntactic analysis of input sentences have the potential to perform a larger variety of more precise transformation operations but they may be time consuming to run and unreliable when processing the types of sentence most in need of simplification. Methods for sentence simplification based on statistical machine translation are efficient to run but depend on the availability of large collections of sentence-aligned parallel corpora. This type of data is expensive to produce, especially in the case of systems designed to exploit syntactically parsed data. In general, methods for simplification based on sentence compression are unsuitable for our purpose because we seek to improve the accessibility of information in input sentences rather than deleting it. For these reasons, we developed a method for sentence simplification of English (Section 5) which is designed to be meaning-preserving and which integrates a new component for syntactic analysis that is not based on syntactic parsing but is based on the automatic classification of various explicit textual signs of syntactic complexity (Section 4). Development of this classifier is based on the analysis described in the next section.

3 Empirical Analysis of Syntactic Complexity in English

In this section, we consider syntactic phenomena that can hinder the comprehensibility of English texts. Caplan and Waters (1999) observe that

propositionally dense sentences are relatively difficult to understand. Syntactic structures such as compound clauses and complex NPs contain multiple clauses and can be propositionally dense. Intuitively, automatic syntactic parsing could be used to detect the occurrence of these types of structures in input sentences. However, the reliability of syntactic parsing depends on the length and complexity of the sentence being processed, with long complex sentences being processed less accurately and considerably more slowly than short simple ones (Tomita, 1985; McDonald and Nivre, 2011). In light of this drawback, we instead focus on various shallow markers which can indicate the presence of syntactically complex structures in English sentences. We present a scheme to encode information on the specific linking and bounding functions of these signs of syntactic complexity, including information on the sentence structures that they signal. We annotated a corpus in accordance with this scheme and present an analysis of the corpus.

Compound and nominally bound relative clauses contribute to sentence complexity. The conjoins of compound clauses are linked by paratactic syntactic relations, while nominally bound relative clauses are linked to the noun phrases that they modify by hypotactic syntactic relations (Quirk et al., 1985). Our analysis of English texts indicated that hypotactic and paratactic relations are often indicated by the occurrence of punctuation marks, conjunctions, complementisers, wh-words, and multi-token units consisting of punctuation marks followed by one of the other types of indicating word. In this paper, we refer to these lexical/punctuational markers as *signs of syntactic complexity*, or *signs*.

Numerous syntactically annotated resources have been developed for English. The Penn Treebank (Marcus, Santorini, and Marcinkiewicz, 1993) is widely exploited for the development of syntactic processors in NLP. However, this resource does not encode detailed information on the coordinating and bounding functions of punctuation marks and other signs of syntactic complexity. Maier, Kübler, Hinrichs, and Kriwanek (2012) described work in which a new annotation layer was added to the Penn Treebank dataset, encoding information about the coordinating function of commas and semicolons. However, their scheme only discriminates between coordinators and non-coordinators without identifying specific subtypes of coordination. For this reason, researchers exploiting the resource developed by Maier et al. (2012) for the purpose of sentence simplification would need to develop additional methods to further disambiguate commas and semicolons. They would need additional sources for information on the bounding and linking functions of lexical signs. Van Delden and Gomez (2002) developed a dataset to support automatic identification of the syntactic roles of commas. That annotation only encodes information about commas and does not support analysis of other signs (conjunctions, complementisers, *etc.*).

3.1 Corpus and Annotation Scheme

Due to the noted poor performance of syntactic parsers when analysing complex sentences, we sought to develop an approach to sentence simplification that does not exploit a syntactic parser. Our intuition was that information on the specific

linking and bounding functions of signs of syntactic complexity could form the basis of a shallow rule-based approach to sentence simplification. This intuition guided our design of the annotation scheme.

Rather than directly annotating the syntactic structure of each sentence, which is a time consuming and onerous process, we annotated a relatively small set of signs of syntactic complexity with information that will enable us to make inferences about the syntactic structure of sentences. The annotation provides information on the spans of subordinated syntactic constituents (their left and right boundaries). It also provides information on the syntactic categories and relative size of both subordinate constituents and the conjoins of compound constituents. It should be noted that under this annotation scheme, conjoins and subordinate constituents are not explicitly annotated. Only the signs themselves are tagged. Full details of the annotation scheme are presented in Evans and Orasan (2013).

We took this approach to support development of a sentence simplification method that can easily be ported to other languages in the context of a multilingual project. We therefore opted for the simple, two-step approach presented in this article which, with the exception of the language-specific rewriting rules, can be applied to other languages by performing the relatively simple annotation task described in this section.

Texts of three registers (*news*, *health*, and *literature*), were collected from the METER corpus (Gaizauskas, Foster, Wilks, Arundel, Clough, and Piao, 2001), and the collections available at *patient.co.uk* and Project Gutenberg (*gutenberg.org*), respectively, to form the corpus. These texts were annotated with information about the syntactic functions of various signs of syntactic complexity. The characteristics of the texts are summarised in Table 1. The columns *Docs* and *Sents* display the total number of documents and sentences in the corpus. The next two columns provide information on the numbers of words and the average length of sentences in each collection. The final column provides information on the numbers of signs of syntactic complexity in each collection (*Total*) and the average number of signs per sentence (*Per Sent*).

Table 1: *Characteristics of the annotated corpus.*

			Words		**Signs**	
Register (Source)	**Docs**	**Sents**	**Total**	**Per Sent**	**Total**	**Per Sent**
Health (*patient.co.uk*)	783	175 037	1 969 753	11.25	180 623	1.032
Literature (Gutenberg)	24	4 608	95 739	20.78	11 235	2.440
News (METER)	825	14 854	307 734	20.71	29 676	1.997

Inspection of Table 1 reveals that sentences in texts of the health register are approximately half as long as those found in texts of the other two registers. In line with intuition, sentences in these texts contain just over half as many signs

of syntactic complexity as sentences in texts of the news register. From this table, we may infer that sentences in texts of the literary register are more syntactically complex than those of news or health (2.440 signs per sentence vs. 1.997 and 1.032, respectively). We manually annotated information about 10 756 signs in texts of the register of health, 11 204 in the register of literature, and 12 718 in the register of news.[7] In the corpus, two main classes of signs were observed: *coordinators* and *subordination boundaries*.

3.1.1 Coordinators

Coordinators include conjunctions ([*and*], [*but*], and [*or*]), punctuation marks ([,] and [;]), and multi-token units consisting of a punctuation mark followed by a lexical sign ([, *and*], [, *but*], [, *or*], [; *and*], [; *but*], and [; *or*]). We restricted the annotation to a relatively unambiguous subset of coordinators to facilitate both the manual annotation task and the automatic tagging process described in Section 4. Conjunctions have an exclusively coordinating function while the functions of signs that include punctuation marks are more ambiguous, serving as coordinators in some contexts and as subordination boundaries in others.

The tags used in our annotation scheme are acronyms indicating the type of constituents coordinated by each sign:

- The first part of the acronym indicates the coordinating function (C).
- The second part indicates the syntactic projection level (Chomsky, 1970) of the linked constituents: lexical (L), intermediate (I), maximal (M), and extended (E).
- The third part of the acronym indicates the syntactic category of the compound: verbal (V), nominal (N), adjectival (A), adverbial (Adv), prepositional (P), or quantificational (Q). Sentences (3)–(6) are examples of coordination involving constituents of different projection levels and different grammatical categories.

(3) A drifter who killed three friends in horrific axe [**CLN** and] knife attacks was given three life sentences yesterday.

(4) Addressing the six men [**CIN** and] six women, Judge Mellor said: 'You may have guessed that this is the GM trial.

(5) BA immediately announced that it would appeal against the ruling, which it branded "wrong in fact [**CMP** and] in law".

(6) After months of hype, Star Wars: The Phantom Menace had its royal premiere last night - [**CEV** and] my fears came true.

This approach follows from the treatment of constituent structure in syntactic theories such as *Government and Binding*. In our context, it provides a

[7] The annotated corpus and annotation guidelines are available at `http://rgcl.wlv.ac.uk/demos/SignTaggerWebDemo/` (last accessed on 14th September 2018).

way to discriminate between signs that coordinate or bound different types of constituents that share the same syntactic category (morphemic, lexical, intermediate, phrasal, or clausal).[8]

- The fourth part of the acronym is optional and takes a numerical value. It is used to distinguish between coordination of different subtypes of NPs and VPs. Each number corresponds to coordination of conjoins that are:

  1. complete, such as tag CMV1 in (7).
  2. incomplete, with ellipsis of the head of the second conjoin, such as tag CMV2 in (8).
  3. incomplete, with ellipsis of the complement of the head of the first conjoin, such as tag CMV3 in (9).
  4. incomplete, with ellipsis of the head of the first conjoin, such as tag CMN4 in (10).

  Thus, we tag signs linking two adjectives as CLA whereas signs linking noun phrases, verb phrases, and clauses are tagged CMN1, CMV1, and CEV, respectively.[9]

(7) McKay had been on the payroll five years[**CMV1** , but] feared he would go on a 'last-in, first out' basis, said Mr Sloan.

(8) But Justice Kay added: "It is a sorry state of affairs when Mr Blunkett {has to explain away}$_i$ his own letters as mistaken and unclear [**CMV2** and] ϕ_i a statement by the Prime Minister as an incorrect representation of policy, taken out of context".[10]

(9) A spokeswoman told The Express: "There is not ϕ_i [**CMV3** and] cannot be {a deal}$_i$, the BBC owns the rights to the name Blackadder and no deal can be signed without our consent."[11]

(10) 'This case is not about whether GM crops are a good ϕ_i [**CMN4** or] a bad {thing}$_i$,' he said.

We include example sentences containing one of the three most commonly occurring types of coordinator in examples (11)–(13). Although some of these examples contain multiple signs of syntactic complexity, for clarity, we only highlight those

[8] In generative approaches to syntax, phrases are considered to be the maximal projections of lexical items that are their heads and clauses are considered to be extended projections of the tenses of main verbs. These constituents are termed *projections* of words and verb tenses because their structure is determined by them.

[9] In this article, we provide examples only of the most relevant classes. Examples illustrating every class of sign are available in the annotation guidelines (see Footnote 5).

[10] In these examples, the antecedent of the elision, ϕ_i, is marked using braces.

[11] This could be considered an example of the coordination of head verbs in the sentence, but the fact that these words are modified by negation implies that this coordination is at the phrasal rather than lexical level. Despite its rarity (2 or 3 cases in more than 30 000 signs), we included this class in our scheme to ensure that it could capture a more complete range of grammatical distinctions.

of the relevant class. Our automatic classifier (Section 4) tags all the signs in each input sentence.

(11) The following weekend he was off sick [**CEV** and] no pancakes were contaminated.

(12) When the bailiffs ignored Mrs Amor's requests to "sod off", she stormed upstairs [**CMV1** and] returned with a double-barrelled shotgun.

(13) The victims were Steven Parker, 21[**CMN1** ,] Paul Thompson, 22[**CMN1** , and] Panayi Kouroushi, 30, all from Groby, Leics[**CMN1** ;] Steven Curtis, 28, of Newton Linford, Leics[**CMN1** ; and] Jeremy Goodall, 30, of Leicester Forest East.

3.1.2 Subordination Boundaries

Subordination boundaries include complementisers ([*that*]), wh-words ([*what*], [*when*], [*where*], [*which*], [*while*], and [*who*]), punctuation marks ([,], [;], [:]), and multi-token units consisting of a punctuation mark followed by any of the lexical signs (e.g. [*, which*] or [*; and*]). As in the case of coordinators, we restricted the annotation to a relatively unambiguous subset of explicit subordination boundaries. In terms of their function, complementisers and wh-words are used exclusively to bound subordinate constituents while signs which include punctuation marks (e.g. [,] and [*, and*]) are ambiguous, as noted earlier. For subordination boundaries, the tags used in the annotation scheme are acronyms indicating the type of constituents bounded by these signs. The first part of the acronym may indicate the left (SS) or right (ES) boundary of a subordinated constituent.[12] The second part of the acronym indicates the syntactic projection level of the bounded constituent. These include maximal and extended projections.[13] The third part of the acronym indicates the syntactic category of the bounded constituent. These may be verbal (V), nominal (N), adjectival (A), adverbial (Adv), or prepositional (P). There are also tags for subordinated constituents such as interjections (SSMI/ESMI), direct quotes (SSCM/ESCM), tag questions (STQ), and constituents of ambiguous syntactic category (SSMX/ESMX).

We provide example sentences containing one of the three most commonly occurring types of *left* boundaries of subordinate constituent in (14)–(16). As in (11)–(13), although some of these examples contain multiple signs of syntactic complexity, for clarity, we only highlight those of the relevant class.

(14) "Yolanda was the sort of person [**SSEV** who] always needed to have a

[12] Start of a subordinated constituent (SS) or end of a subordinated constituent (ES).

[13] Grammarians typically consider subordination to be a link between clauses, sometimes with elided elements (Quirk et al., 1985). In this research, we label clauses with elided elements in accordance with their extant elements. For example, subordination of an -ed participle clause is considered subordination of a verb phrase under our approach. When extant elements cannot be categorised as a single subclausal unit such as a noun phrase, then the subordinated constituent is considered clausal.

boyfriend and would pick up when she had one, but if not she would nosedive."

(15) Noel Gallagher became a father yesterday after his wife[**SSMN** ,] Meg Mathews, gave birth to a girl.

(16) They left the tenant a note on a cornflakes box, which read[**SSCM** :] 'We've missed you this time.'

Examples (17)–(19) are sentences containing one of the three most commonly occurring types of *right* boundary of subordinate constituent. As before, in these examples we only highlight signs of syntactic complexity of the relevant class.

(17) Yolanda, who was due to sit exams in English, History and Art[**ESEV** ,] was "disorganised" and had been under-achieving at school, Ms Hamblett-Jahn said.

(18) 'Within the confines of the nursery[**ESMP** ,] her treatment of them was markedly different.'

(19) The rival chef, Oliver Peyton[**ESMN** ,] had claimed Mr White's Titanic restaurant was a replica of his own Atlantic Bar and Grill - which is housed in the same West End hotel.

In the next section, we provide some analysis of the annotated corpus that we developed.

3.2 Annotated Data

Two linguists annotated signs of syntactic complexity in the corpus with information on their linking and bounding functions. A subset of each register (1000 signs of each) was annotated by both linguists.[14] The values of Kappa obtained for the annotations were 0.80 for signs annotated in texts of the news register, 0.74 for those in the health register, and 0.76 for those in the literary register. These levels imply a minimum of "substantial agreement" between annotators (Cohen, 1960). Evans and Orasan (2013) provide information on the most frequent types of disagreement.

In Table 1, we presented statistics about the number of signs annotated for each register. Table 2 displays the frequency distribution of the twenty most common signs and the twelve most common tags assigned in the three text registers. Space restrictions prevent display of the full distribution of 35 annotated signs and 38 assigned tags. The row *Total* provides information on the total number of signs of each class in the corpus.

Sentence (20) is an example illustrating annotation in accordance with this scheme.

[14] With regard to annotation rate, after a period of approximately one month away from the task, Evans was able to annotate 100 signs in under 25 minutes (four signs per minute).

Table 2: Frequency distribution of the twenty most frequent signs and twelve most frequent tags in the data set

Sign/ Tag	**SSEV**	**CEV**	**CMV1**	**CMN1**	**CLN**	**SSMN**	**ESEV**	**SSCM**	**ESMP**	**ESMN**	**ESCM**	**SSMP**
, that	89	0	0	0	0	0	2	0	6	2	0	0
while	124	0	0	0	0	0	0	0	0	0	0	0
where	190	0	0	0	0	0	0	0	0	0	0	0
; and	0	187	2	7	0	0	0	0	0	0	0	0
, which	194	0	0	0	0	0	0	0	0	1	0	0
, or	7	63	65	86	18	8	1	0	0	0	0	0
, who	353	0	0	0	0	1	0	0	1	2	0	0
what	406	0	0	0	0	0	0	0	0	0	0	0
but	0	237	137	7	0	0	0	0	0	0	0	0
;	57	190	6	42	7	3	4	133	2	1	2	4
, but	3	374	98	11	0	0	0	0	2	1	1	2
which	561	0	0	0	0	0	0	0	0	0	0	0
when	640	0	0	0	0	0	0	0	0	0	0	0
who	665	0	0	0	0	0	0	0	0	0	0	0
:	233	130	0	0	0	10	0	702	1	0	2	0
or	0	46	101	353	372	0	0	0	0	0	0	0
, and	53	1 005	823	315	53	5	6	0	2	8	1	2
that	2 487	0	0	0	0	0	0	0	0	0	0	1
and	0	712	1 543	1 253	873	0	0	0	0	0	0	0
,	967	235	219	745	559	986	1 375	531	1 187	495	1 040	828
Total	**7 357**	**3 298**	**2 995**	**2 820**	**1 882**	**1 627**	**1 588**	**1 366**	**1 275**	**1 162**	**1 046**	**909**

(20) Before he was hired by Addaction[**ESMAdv** ,] he had headed a team of armed robbers from the Meadows estate [**CEV** and] he had already been convicted of more than 30 offences involving burglary[**CMN1** ,] theft[**CMN1** ,] firearms[**CMN1** ,] unlawful sexual intercourse [**CMN1** and] possession of drugs.

It contains multiple signs of syntactic complexity. The annotation provides information on the syntactic function of each. The first sign is the right boundary of the subordinate adverbial phrase *Before he was hired by Addaction*, the second sign links two clauses in coordination, while the subsequent signs link noun phrase arguments of the verb *involving* in coordination. As noted earlier, the purpose of our annotation scheme is not to mark syntactic constituents themselves, as is the case in most types of syntactic annotation. In our scheme, only the signs are labeled, not the constituents whose presence they signal.

The annotated data indicates no upper limit on the number of signs of syntactic complexity that a sentence may contain. The largest number of signs found in a sentence of this corpus is 30.[15] Almost two thirds (63.64%) of the sentences containing more than 20 signs are of the literary register. In our corpus, 46.55% of sentences contained at least one sign of syntactic complexity.

The signs [,], [*and*], and [*that*] were the most common signs of syntactic complexity across all three registers. The signs [*;*] and [*, and*] were relatively frequent in literary text while the conjunction [*or*] was most frequent in the register of public health information. When considering all signs and not just the twenty most frequent, most were left boundaries of subordinate clauses (SSEV), coordinators of verb phrases (CMV1), and coordinators of noun phrases (CMN1).

In this article we present a method designed to rewrite sentences that contain compound clauses and bound relative clauses, which contain signs of class CEV and/or SSEV. The rewriting process depends on accurate identification of several additional tags. Tabulating absolute and cumulative frequencies of signs and tags reveals a skewed distribution. This fact, illustrated in Table 2, provides an indication of the challenge posed in developing automatic methods to accurately assign tags that occur infrequently in the annotated data to signs of syntactic complexity. At present, due to limited resources, we consider the processing of additional signs of syntactic complexity (*e.g.* conjuncts, adjuncts, conditional/adversative/comparative conjunctions, conditionals, *etc.*) to be a topic for future research. The automatic identification and categorisation of such signs may bring benefits for other syntactic processing systems. In the next section, we present our automatic method for classifying signs of syntactic complexity.

4 Sign Tagging: An Automatic Approach to Sentence Analysis

State of the art PoS taggers provide little information on the syntactic functions of conjunctions, complementisers, *wh*-words, and punctuation marks, and are of

[15] Sentences containing more than 25 signs are present in each of the three registers.

limited use for automatic sentence simplification. Detailed information can be derived from syntactic parsers but this information may be inaccurate for long complex sentences. Further, exploitation of such information may be non-trivial, requiring detailed processing of the syntactic tree structure.

Van Delden and Gomez (2002) developed a tool to provide information on the linking and bounding functions of commas. Their method operates in two phases. In the first, 38 finite state automata are applied to PoS-tagged data to derive an initial tagging of commas. In the second, information from a tag co-occurrence matrix derived from hand annotated training data is used to improve the initial tagging. The system achieves accuracy of 91-95% in identifying the syntactic functions of commas in a collection of encyclopaedia and news articles. The inability of the method to process other signs limits its usefulness in automatic sentence simplification.

The method proposed by Evans (2011), and described in Section 1, relied on memory-based learning to classify potential coordinators (including commas and a limited number of conjunctions) in clinical texts. The method had an overall accuracy of 83.2% in assigning one of 30 class labels to 7 types of potential coordinator. His approach was developed for use in a restricted, relatively homogeneous domain (patient notes), demonstrating only a limited range of syntactic constructions. As a result, the approach is inadequate for tagging signs in the texts considered in our current research.

The absence of automatic tools to identify the full set of syntactic functions of the full set of signs motivated us to develop a new sign tagger, exploiting our annotated corpus (Section 3.2). This sign tagger, together with details of its development and implementation, are presented in the paper by Dornescu et al. (2013). In this section, we provide an overview of the method and provide examples to clarify certain aspects of the implementation. Our initial motivation for developing the sign tagger was to perform a shallow syntactic annotation of input texts for use in various NLP applications such as information extraction, where a wider range of syntactic transformation operations is seen to be beneficial. The method proposed is based on a ML algorithm which is able to classify each sign according to the classes annotated in our corpus. The algorithm relies mainly on the context of the sign to determine its class. We conducted experiments to optimise the performance of our sign tagger by evaluating it with alternate settings of four parameters: algorithm type, tagging mode, features used to represent instances in the training data, and the selection of training data. The evaluation was carried out over the annotated part of the corpus presented in Section 3 and is expressed using the evaluation metrics typically used in NLP: precision, recall, f-measure and accuracy. In all the experiments, 10-fold cross validation was employed.[16]

With regard to algorithm type, we found that sequence-based CRF tagging models (Lafferty, McCallum, and Pereira, 2001; Sutton and McCallum, 2011) provided better performance in the automatic tagging of signs than methods in

[16] An online demo of our sign tagger is also available at `rgcl.wlv.ac.uk/demos/SignTaggerWebDemo`.

which each sign is tagged independently of other signs in the same sentence. Our approach thus contrasts with that of Evans (2011), in which signs are classified independently. In our approach, texts are treated as sets of token sequences, with each sequence corresponding to a sentence in the text. A prediction is made of the tag of every token in the text, not just the subset of tokens that are signs of syntactic complexity.[17] The tags to be assigned are treated as variables that depend both on other observed variables and on the probabilities of the potential tags of other tokens occurring in the same sentence.

When applying the CRF tagger, two tagging modes were evaluated. In the first (*simple*), signs of syntactic complexity in the training data were tagged with the classes specified in Section 3, while non-signs were tagged *NA* to indicate that they are not signs of syntactic complexity. 90% of the tokens being tagged in this setting are non-signs and we were concerned that the derived tagging models would prioritise accurate tagging of non-signs at the expense of the task we are really interested in, the tagging of signs. In this article, evaluation scores are reported in the context only of sign tagging, not token tagging. In the *simple* tagging mode, the model operates at acceptable levels of accuracy when sign tagging ($0.7846 < acc < 0.8323$). In the second tagging mode (*BIO*), signs of syntactic complexity in the training data were tagged with the class labels specified in Section 3, while non-signs were tagged with a class label matching that of the closest preceding sign. Table 3 displays a sample of the annotations used in each of the two tagging modes. The sign tagger has slightly better accuracy when operating in the *BIO* tagging mode ($0.7991 < acc < 0.8383$).[18]

Two types of representation of training instances were tested. In the first (*core*), tokens were represented by evaluating three sets of feature templates:

1. Unigrams consisting of:
   - the orthographic form of the token being tagged,
   - the orthographic form and the part of speech, in combination, of the token being tagged,
2. Bigrams consisting of:
   - the parts of speech of the token being tagged and the following token,
3. Trigrams consisting of:
   - the parts of speech of the preceding token, the token being tagged, and the following token,
   - the parts of speech of the token being tagged, and the following two tokens.

The CRF++ package (Kudo, 2005) was used to derive the sequence tagging model. Tokens in the training data were represented using a set of feature templates which encode an evaluation of the external observed variables. We built the *core*

[17] In this context, signs comprising a punctuation mark followed by a word are treated as single tokens.

[18] The difference is marginal, but the *simple* tagging mode achieves superior performance to the *BIO* mode when applied to texts of the health register ($F_1 = 0.8358$ vs. 0.8300).

Table 3: *Training sample for the Simple and BIO tagging modes*

Token	**PoS**	**Simple**	**BIO**
There	EX	NA	NA
are	VBP	NA	NA
a	DT	NA	NA
couple	NN	NA	NA
of	IN	NA	NA
scenes	NNS	NA	NA
that	WDT	SSEV	SSEV
involve	VBP	NA	SSEV
sex	NN	NA	SSEV
in	IN	NA	SSEV
that	DT	SPECIAL	SPECIAL
show	NN	NA	SPECIAL
but	CC	CEV	CEV
they	PRP	NA	CEV
focus	VBP	NA	CEV
on	IN	NA	CEV
the	DT	NA	CEV
faces	NNS	NA	CEV
.	.	NA	CEV

feature set by first evaluating a baseline sequence tagging model, derived using CRF++, in which tokens were represented by a single feature template specifying the orthographic form of the token being tagged. Models in which tokens were represented by a candidate feature template in isolation were then derived and evaluated. Those with superior performance to the baseline were included in the *core* feature set. This *core* set was supplemented with unigram feature templates evaluating the features proposed by Evans (2011) to create an *extended* feature set. In evaluations exploiting the CRF model to tag signs in texts of the news register, use of the *extended* feature set was found to be more accurate than use of the *core* feature set (*acc* of 0.8058 vs. 0.7846).

We were also interested in variation in the performance of the sign tagger as a result of a mismatch between the register of the text being tagged and the register of the text from which training data was derived. In every case, there was a considerable reduction in accuracy when training data of one register was used to tag signs in text of a different register. We conducted a comparative evaluation of sequence taggers exploiting training data of a register matching that of the testing data with taggers exploiting training data derived by combining instances belonging to all three registers (ensuring complementarity with test instances). This experiment showed that training a single tagging model on the entire multi-register

dataset yields slightly better performance ($acc = 0.8250$) than models trained on data derived from texts matching the register of the input ($acc = 0.8196$).

The sign tagger presented in this article uses a CRF sequence tagging model, running in the *BIO* tagging mode, using the *extended* feature set to represent instances, and exploiting training data derived from texts of all three registers. A detailed evaluation of the sign tagger and its influence on the sentence simplification task is presented in Section 6.1.

5 Sentence Transformation

Our new method for sentence simplification combines data-driven and rule-based approaches. In the first stage, input sentences are tokenised and part-of-speech tagged using the TTT2 language processing package (Grover, Matheson, Mikheev, and Moens, 2000).[19] After this, signs of syntactic complexity are identified and classified using the machine learning approach described in Section 4. One of the strengths of our method is that it only requires these two shallow and reliable pre-processing steps.

Our system is an improved version of Evans's (2011) method, exploiting a better method for sentence analysis (Section 4) and a larger set of sentence transformation rules (Section 5.2). In response to our specific user requirements (Martos et al., 2013), the new method is designed only to simplify sentences containing compound clauses (Type 1) and nominally bound relative clauses (Type 2). This is in contrast to Evans's (2011) approach which simplified sentences containing clausal, phrasal, and lexical compounds for the purpose of information extraction.

5.1 The Algorithm

We observed in our corpus that there is no upper limit on the number of signs of syntactic complexity that a sentence may contain. For this reason, we applied an iterative approach to sentence rewriting. A single rewriting operation is applied in each iteration according to the class labels of the signs occurring in the sentence. Each application of a rewriting operation converts an input sentence containing signs of syntactic complexity into two sentences, each containing fewer signs. These rewriting operations apply exhaustively until the system is unable to detect any bound relative clauses or compound clauses in the derived sentences.

Our sentence simplification method exploits Algorithm 1. Two iterative processes are used to rewrite the original sentence and each of the sentences generated in the working set. The first process applies rules to rewrite Type 1 sentences (containing compound clauses). It ends when no compound clauses can be detected in any of the sentences in the working set. The second process applies rules to rewrite Type

[19] The experiments described in this paper relied on TTT2 but the current version is the implementation of the Brill tagger (Brill, 1994) distributed with GATE and used in the ANNIE application (Hepple, 2000). In our estimation, PoS tagging errors do no have a great influence on the accuracy of our sentence simplification method.

Input: Sentence s_0, containing at least one sign of syntactic complexity of class c, where $c \in \{\text{CEV, SSEV}\}$.

Output: The set of sentences A derived from s_0, that have reduced propositional density.

```
1  The empty stack W;
2  O ← ∅;
3  push(s_0, W);
4  while isEmpty(W) is false do
5  |  pop(s_i, W);
6  |  if s_i contains a sign of syntactic complexity of class c (specified in Input)
   |   then
7  |  |  s_i1, s_i2 ← rewrite_c(s_i);
8  |  |  push(s_i1, W);
9  |  |  push(s_i2, W);
10 |  else
11 |  |  O ← O ∪ {s_i}
12 |  end
13 end
```

Algorithm 1: Sentence rewriting algorithm

2 sentences (containing bound relative clauses). In a similar fashion, this process ends when no bound relative clauses can be detected in any of the sentences in the working set.[20]

Application of the rules used in these processes (line 7 of Algorithm 1) is triggered by detection of tagged words and signs in the input sentence. Signs of class CEV indicate the occurrence of at least one compound clause in the input sentence. Signs of class SSEV following nouns in the sentence indicate the occurrence of at least one nominally bound relative clause. Clause coordinators in input sentences are detected from right to left in this algorithm, so that the rightmost conjoins of compound clauses are split first. By contrast, the left boundaries of subordinate clauses are detected from left to right, so that least deeply embedded relative clauses are split first. We have not evaluated the impact of the direction of matching on the quality of output produced by our method but the transformation rules discussed in the next section were manually developed with these facts in mind. Detection of other types of signs has a role to play in the automatic rewriting process as it can be used by our rules to identify clause boundaries.

5.2 Transformation Rules

Rules applied in the first process convert Type 1 sentences into two new sentences, each of which contains one fewer sign of class CEV than its antecedent. In total, 28

[20] The sentences of input documents are processed one at a time, rather than all being enqueued in a single batch. The stack only holds the sentence being processed and its intermediate derivations.

rules of this type were developed. Rules applied in the second process convert Type 2 sentences into two new sentences, each containing fewer signs of class SSEV than its antecedent. 125 rules of this type were developed.

The rule sets associated with each sign tag (SSEV and CEV) were developed incrementally by using the sentence simplification method to process the annotated corpus described in Section 3.1. The texts described earlier, annotated with information about signs of syntactic complexity, were used for this purpose. Each rule set was initialised as an empty set. When processing a sentence which both contains at least one sign of the relevant class and does not match any existing sentence rewriting pattern, the sentence was printed and the program stopped. We then manually formulated a new pattern to match the compound clause (CEV) or complex phrase (SSEV) in the sentence together with an associated transformation operation and added the resulting rule to the relevant rule set. This process continued until we perceived that the addition of new rules to process previously unseen sentences was introducing errors in the processing of sentences that had previously been processed successfully. In this approach to development of the rule sets, the focus was on the capacity of the rules to correctly match the different elements of sentences containing compound clauses and complex constituents. After inclusion in the rule set, transformation operations were edited manually on inspection of the resulting output sentences generated. During development, formal evaluation of the sentence rewriting rules and the combined operation of the rule sets was not performed due to the absence at that time of gold standard evaluation data. The texts used for development of the rules were not included in the gold standard used to evaluate our system.

Table 4 and Table 5 each display the three most frequently triggered rules used to rewrite Type 1 and Type 2 sentences, respectively. In these examples, the triggering patterns are expressed in terms of elements defined in Table 6. The * operator is used to indicate non-greedy matching. Sentence rewriting was facilitated by accurate identification of signs linking clauses (CEV), noun phrases (CMN1), and adjective phrases (CMA) in coordination and signs serving as the left or right boundaries of bound clauses, including relative (SSEV/ESEV), nominal/appositive (SSMN/ESMN), and adjective (SSMA/ESMA) clauses.

The transformation operations applied to Type 1 sentences generate pairs of sentences in which the sentence containing the first conjoin precedes the sentence containing the second. In the case of Type 2 sentences, the reduced sentence containing the matrix NP[21] precedes the sentence linking the matrix NP to the predication of the relative clause. The use of stack operations means that the simplification occurs in a depth-first manner. In a sentence containing two clause conjoins, each of which contains one complex NP, the output is ordered so that the sentence containing the reduced first conjoin is followed first by the sentence linking the matrix NP of that conjoin to the predication of its bound relative clause, then by the sentence containing the reduced second conjoin, and finally by the sentence

[21] This sentence is reduced because the transformation operations delete the nominally bound relative clause.

linking the matrix NP in the second conjoin to the predication of its bound relative clause. In this way, sentences containing formerly complex NPs are immediately followed by the sentences that provide more information about those NPs.

Table 4: *Example rules used to rewrite Type 1 sentences ($rewrite_{CEV}(s_i)$)*

Rule	**Pattern**	**Original sentence**	**Rewritten form**
CEV-24	$[A \_ B]$ ↓ $[A.]$ $[B.]$	Kattab of Eccles, Greater Manchester, was required to use diluted chloroform water in the remedy[, *but*] the pharmacy only kept concentrated chloroform, which is twenty times stronger.	Kattab, of Eccles, Greater Manchester, was required to use diluted chloroform water in the remedy. The pharmacy only kept concentrated chloroform, which is twenty times stronger.
CEV-12	$[A\ that\ B \_ C]$ ↓ $[A\ that\ B.]$ $[A\ that\ C.]$	"He was trying to intimate that mum was poorly [*and*] we should have expected that she might die at any time."	"He was trying to intimate that mum was poorly." "He was trying to intimate that we should have expected that she might die at any time."
CEV-27	$[A\ v_{EV}\ B\ "C \_ D]$ ↓ $[A\ v_{EV}\ B\ "C.]$ $[A\ v_{EV}\ B\ "D.]$	He said to me, 'You're dodgy[,] you're bad news[,] you know you're bad news.'	He said to me, 'You're dodgy.' He said to me, 'you're bad news.' He said to me, 'you know you're bad news.'

Although the patterns used in the rule sets only explicitly refer to a small number of class labels, it is necessary to discriminate between them accurately. For example, when simplifying a sentence such as (21),

(21) Helen[$_{SSCCV}$, who] has attended the Carol Godby Theatre Workshop in Bury[$_{SSMN}$,] Greater Manchester[$_{ESMN}$,] since she was five[$_{ESCCV}$,] has also appeared in several television commercials.

it is necessary to discriminate between the two final commas to accurately identify the span of the nominally bound relative clause.

6 Evaluation

In this section, we present our evaluation of the sentence analysis and sentence transformation methods that we developed. This includes an assessment of the suitability of the sign tagger for the sentence transformation task (Section 6.1.1) and evaluation of the sentence transformation method by comparison of its output

Table 5: *Example rules used to rewrite Type 2 sentences ($rewrite_{SSEV}(s_i)$)*

Rule	Pattern	Original sentence	Rewritten form
SSEV-1	[*A* w_n* _ *B* s_{ESEV} *C*] ↓ [*A* w_n *C*.] [w_n *B*.]	Drummond[, *who*] had pleaded not guilty, was jailed for three months concurrently on each of six charges of wilfully killing, taking and mistreating badgers.	Drummond was jailed for three months concurrently on each of six charges of wilfully killing, taking and mistreating badgers. Drummond had pleaded not guilty.
SSEV-43	[A *a/an* w_n* w_n _ w_{NNP} w_{VBD} *C*] ↓ [*A a/an* w_n* w_n.] [*It was the* w_n* w_n w_{NNP} w_{VBD} *C*]	In February last year police raided a council house [*which*] Francis rented in St Ann's.	In February last year police raided a council house. It was the council house Francis rented in St Ann's.
SSEV-61	[*A* w_{IN} w_{DT} w_n* _ w_V *B*] ↓ [*A* w_{IN} w_{DT} w_n*.] [*That* w_n* w_V *B*]	One's heart goes out to the parents of the boy [*who*] died so tragically and so young.	One's heart goes out to the parents of the boy. That boy died so tragically and so young.

Table 6: *Elements used in sentence rewriting patterns*

Element	Denotation
_	The detected sign of class *c*
Upper case letters (A-D)	Sequences of zero or more characters matched in a non-greedy fashion
w_{POST}	Word of PoS POST, from the Penn Treebank tagset (Marcus et al., 1993)
w_n	Nominal word
w_v	Verbal word, including –ed verbs tagged as adjectives
s_{TAG}	Sign of syntactic complexity with tag TAG
v_{EV}	Clause complement verb (e.g. *accept*, *deny*, *mean*, *retort*, *said*, *etc.*)
word	Word *word*

with human simplified text (Section 6.2.1), by reference to automatic estimations of the readability of its output (Section 6.2.2), and by reference to readers' opinions on the grammaticality, accessibility, and meaning of its output (Section 6.2.3).

6.1 Evaluation of the Sign Tagger

Table 7 shows the results of testing the performance of our sign tagger in texts of all three registers, using ten-fold cross-validation. *Register* is the register of the

text data being tagged. Columns P, R, and F_1 display the precision, recall, and F_1-score statistics obtained by the tagger. *Signs* is the total number of signs of syntactic complexity in the test data, *Corr* is the number of signs tagged correctly while *Incorr* is the number tagged incorrectly. Accuracy (*Acc*) is the ratio of *Corr* to the sum of *Corr* and *Incorr*. Column *Bsln* displays the accuracy of a baseline classifier which tags signs with the class labels most frequently observed for signs of each type in the annotated corpus presented in Section 3.

Table 7: *Evaluation results of the sign tagger for text of three registers*

Register	**P**	**R**	F_1	**Signs**	**Corr**	**Incorr**	**Acc**	**Bsln**
Health	0.841	0.824	0.830	10 796	8900	1896	0.824	0.422
Literature	0.860	0.838	0.847	11 204	9392	1812	0.838	0.387
News	0.816	0.799	0.805	12 718	10 163	2555	0.799	0.393

Our analysis focuses on performance statistics obtained when tagging signs in texts of the news register but our findings also apply to the sign tagging of texts of other registers (health and literature). Table 8 shows the *per tag* performance of the sign tagger. The column *Support* displays the number of signs assigned each tag by human annotators in the training data. The column *Cumulative Frequency* displays the percentage of training instances assigned the current tag and the tags preceding it in the table. To illustrate, 42.81% of the training data consists of signs tagged SSEV, CMV1, and CMN1. Statistics are displayed for the 14 most frequently occurring tags in the test data. The lower part of the table displays statistics micro-averaged over the fourteen most frequently occurring tags (*Top 14*), the 26 tags occurring least frequently (*Bottom 26*), and all the tags (*All*) in the test data. Inspection of the micro-averaged statistics reveals that the predictions have a good balance between precision and recall. There is more variance when looking at performance over specific tags or signs. For example, sign tagging is accurate for some tags (*e.g.* SSEV, SSCM, SSMA and ESCM), with $F_1 > 0.9$. Most of these tags mark the left boundaries of subordinate clauses. Other tags, despite occurring with comparable frequency, are more difficult to assign (*e.g.* CMN1, ESEV, ESMP, ESMN, and ESMA) and the tagger is substantially less accurate in tagging them ($F_1 < 0.7$). Signs tagged by this latter set serve as the right boundaries of subordinate clauses, suggesting that identification of the ends of constituents is more difficult than identification of their beginnings. This is especially true of the right boundaries of multiply-embedded clauses, where one sign serves as the right boundary of several constituents. As we note in Section 6.1.1, this influences the accuracy of the sentence transformation process. The human annotators were required to tag these signs as the right boundaries or coordinators of the most superordinate left-bounded constituent. Signs are labelled with only one tag.

Table 9 is a confusion matrix that includes statistics on the eight most frequent

Table 8: *Evaluation of the sign tagger over individual tags in the register of news*

Tag	**P**	**R**	F_1	**Support**	**Cumulative Frequency (%)**
SSEV	0.96422	0.92977	0.94668	3275	25.75
CMV1	0.86180	0.80828	0.83418	1111	34.48
CMN1	0.73812	0.66006	0.69691	1059	42.81
SSMN	0.88650	0.83842	0.86179	885	49.94
CEV	0.80708	0.77949	0.79305	907	56.90
SSCM	0.96587	0.97586	0.97084	580	61.51
ESEV	0.63830	0.56314	0.59837	586	66.07
SSMA	0.93032	0.95736	0.94365	516	70.12
ESMP	0.58577	0.56112	0.57318	499	74.05
SSMP	0.84691	0.81667	0.83152	420	77.70
CLN	0.75352	0.69181	0.72135	464	81.00
ESMN	0.59719	0.61005	0.60355	418	84.29
SSMV	0.84179	0.81034	0.82577	348	87.03
ESCM	0.92073	0.93789	0.92923	322	89.56
Micro-average:					
Top 14	0.8504	0.8133	0.8315	11390	89.56
Bottom 26	0.4926	0.6769	0.5702	1328	10.44
All	0.7991	0.7991	0.7991	12718	100.0

types of confusion made by our tagger in news texts. The tags listed in column 1 are those assigned by human annotators, ranked by frequency of confusion, while tags listed in the column headers are those assigned by our tagger.[22] The *Errors* column displays the total number of errors of each type made by the sign tagger when processing the test data.

Two types of confusion are of direct relevance to the sentence simplification process. First, the sign tagger confuses signs coordinating noun phrases (CMN1) with signs coordinating clauses (CEV). This results in the system applying transformation rules in sentences to which they should not apply. These errors may occur because the constituents adjacent to clause coordinators are often noun phrases in the object position of the first clause and the subject position of the second. Second, the sign tagger frequently confuses the relative pronoun *that* as a determiner or anaphor (SPEC) rather than as the left boundary of a subordinate clause (SSEV). As a result, some sentences containing complex noun phrases will

[22] In Table 9, K/R abbreviates *key* (gold standard) and *response* (sign tagger), respectively.

not be transformed by the system. The sign tagger also frequently mistakes the right boundaries of subordinate clauses for the right boundaries of other phrases, most notably prepositional phrases and noun phrases. As a result, the sentence simplification method is likely to make errors when identifying the right boundaries of relative clauses, and simplifying sentences that contain complex noun phrases. As with the confusions when tagging NP coordinators as clause coordinators, there are surface similarities in the contexts of use of the signs involved in these latter types of confusion.

Table 9: *Confusion matrix of the sign tagger for texts of the news register*

K/R	**CEV**	**CMN1**	**CMV1**	**ESEV**	**ESMN**	**ESMP**	**SPEC**	**SSEV**	**Errors**
CMN1	0.352	-	0.179	0.029	0.045	0.029	0.047	0.012	351
ESEV	0.193	0.008	0.074	-	0.232	0.243	0.047	0.193	296
ESMP	0.023	0.027	0.018	0.265	0.173	-	0.031	0.134	221
ESMN	0.017	0.008	0.037	0.167	-	0.234	0.071	0.023	208
SSEV	0.057	0.016	0.006	0.110	0.064	0.084	0.323	-	202
SSMN	0.028	0.191	0.012	0.053	0.050	0.184	0.055	0.094	199
CMV1	0.102	0.113	-	0.053	0.023	0.004	0.126	0.000	192
CEV	-	0.152	0.167	0.090	0.014	0.021	0.118	0.064	176
CLN	0.028	0.245	0.025	0.004	0.000	0.004	0.008	0.000	131

We presented *per sign* evaluation of the tagger in Dornescu et al. (2013), though the results are omitted here for brevity. When testing the system on texts of the *news* register, excellent performance was achieved when tagging the complementiser [*that*] and *wh*-words such as [*who*], [*when*] or [*which*] ($F_1 > 0.95$). Due to the skewed distribution of signs, more than 83% of tagging errors were linked to the two most frequent signs [,] and [*and*] ($F_1 = 0.75$).

For manual annotation of the signs of syntactic complexity, Evans and Orasan (2013) report agreement between human annotators of $\kappa = 0.7667$. When evaluating our sign tagger in each of the ten folds, mean $\kappa = 0.7533$. In both cases, the confidence interval ($\alpha = 0.05$) was narrow (0.0011 and 0.0014, respectively). In light of this, and given the similarity of the sign tagger to human annotators in terms of classification accuracy, we would not expect that the availability of additional training data would evoke significantly better performance from the sign tagger. We also doubt the proposition that performance would be dramatically improved through the use of more recent ML approaches such as deep learning or the use of more recent pre-processing tools.

With a level of accuracy similar to that of human annotators, we believe that the output of the sign tagger will be useful in the analysis step of our method for sentence simplification. In Section 6.1.1, we consider performance of the tagger in assigning the correct class labels to signs that are explicit elements in our sentence transformation rules.

6.1.1 Suitability for Use in Sentence Simplification

The rules comprising function $rewrite_{CEV}$ (Section 5.2) depend on accurate detection of two classes of signs:

1. clause coordinators (CEV) and
2. left boundaries of subordinate constituents ({SSEV, SSCM, SSMA, SSMAdv, SSMI, SSMN, SSMP, or SSMV})

Thus, for sentence rewriting, confusions between tags of the set specified in 2 are irrelevant (e.g. confusion between SSEV and SSCM). By contrast, confusion between tags of *different* sets specified in 1-2 is relevant (e.g. confusion between CEV and SSMA). Table 10 displays the accuracy with which the sign tagger assigns the two sets of class labels relevant to rewriting Type 1 sentences (containing compound clauses). There, row SSX pertains to signs tagged with any of the class labels in the set listed in 2. Considered over the full set of signs, the tagger assigns these class labels with a micro-averaged F_1-score of 0.9318.

Table 10: *Evaluation of the sign tagger over tags exploited in the rewriting of Type 1 sentences*

Tag	P	R	F_1	Support	True-Pos	False-Pos	False-Neg
CEV	0.7991	0.7991	0.7991	876	700	176	176
SSX	0.9794	0.9251	0.9515	6076	5621	118	455
Micro-average:							
All	0.9556	0.9092	0.9318	6952	6321	294	631

The rules comprising the function $rewrite_{SSEV}$ (Section 5.2) depend on accurate detection of four classes of signs:

1. noun phrase coordinators (CMN1),
2. right boundaries of bound relative clauses (ESEV),
3. right boundaries of direct quotes (ESCM), and
4. left boundaries of subordinate constituents ({SSEV, SSCM, SSMA, SSMAdv, SSMI, SSMN, SSMP, or SSMV}).

Thus, for sentence rewriting, confusions between tags in the set specified in 4 are irrelevant (e.g. confusion between SSEV and SSCM). By contrast, confusion between tags in the sets specified in 1-4 are relevant (e.g. confusion between SSMA and ESEV or between CMN1 and SSMP). Table 11 displays the accuracy with which the sign tagger assigns these four sets of class labels. There, row SSX pertains to signs tagged with any of the class labels in the set specified in 4. The sign tagger

Table 11: *Evaluation of the sign tagger over tags exploited in the rewriting of Type 2 sentences*

Tag	P	R	F_1	Support	True-Pos	False-Pos	False-Neg
CMN1	0.7286	0.6628	0.6942	1041	690	257	351
ESEV	0.5261	0.4789	0.5014	1041	272	245	296
ESCM	0.9207	0.9379	0.9292	322	302	26	20
SSX	0.9794	0.9251	0.9515	6076	5621	118	455
Micro-average:							
All	0.9142	0.8598	0.8862	8480	6885	646	1122

assigns class labels belonging to these four sets to signs with a micro-averaged F_1-score of 0.8862.

Over all the tags exploited in the two types of sentence rewriting, the tagger assigns class labels with a micro-averaged F_1-score of 0.9075.

6.2 Evaluation of Sentence Transformation

In this section, we present our evaluation of the sentence transformation process. This includes the results of experiments in which system output is compared with simplifications made by human experts (Section 6.2.1), the readability of system output is estimated using selected readability indices (Section 6.2.2), and humans rate the grammaticality and accessibility of system output and the extent to which this output conveys the same meaning as the original sentences in the texts (Section 6.2.3).

6.2.1 Comparison with Human-Produced Transformations

We evaluated automatically rewritten sentences generated by our system in terms of accuracy, assessed by reference to gold standards produced by linguists. We developed resources to support automatic evaluation of our system. This type of evaluation can be replicated easily to facilitate development of the system. In our experiments, we also compared system performance with that of two baseline methods.

Gold Standards

We developed two datasets constituting gold standards for these tasks against which system output could be compared. They were developed by a linguist who was a native speaker of English and was well-versed in English grammar. She was presented with output generated by the sentence simplification system when it

was used to automatically simplify Type 1 sentences (1009 sentences of the three registers – 325 Health, 419 Literature, and 265 News) and when used to simplify Type 2 sentences (885 sentences of the three registers – 137 Health, 379 literature, and 369 news). The linguist produced the gold standard by manually correcting automatically transformed sentences generated by the system. She was asked to undo transformations involving the arguments of clause complement verbs and transformations triggered by the mis-classification of signs without coordinating or bounding functions. She was also asked to correct grammatical errors in the output sentences. The goal of the task she was undertaking and the way in which the algorithm worked were verbally explained to the linguist and the sentence simplification tool was demonstrated before the post-editing task began.

The sentence simplification method was applied to texts of all three registers. Table 12 contains information about the subset of data used to test the method when reducing the number of compound clauses in sentences. The column *Signs* contains two sub-columns: *All*, which displays the number of signs of syntactic complexity in the data, and *CEV*, which displays the number of signs tagged CEV by human annotators (*Oracle*) and by the automatic tagger described in Section 4 (*OB1*). *Compound Clauses* displays the number of clauses in the data set that contain one or more coordinated clause conjoins. It comprises one column (*Gold*) which displays the number of compound clauses identified by linguists in the data set (the gold standard) and another (*OB1*) which displays the number identified by the sentence transformation method described in Section 5. *Derived Sentences* is the number of sentences generated as a result of simplifying Type 1 sentences. Sub-column *Gold* displays the number of sentences generated by the linguists in the gold standard while sub-column *OB1* displays the number generated by the automatic sentence simplification tool. In our evaluation, we filtered sentences that did not contain signs manually tagged as being of class CEV.

Table 12: *Characteristics of the test data used to evaluate the method to simplify Type 1 sentences*

		Signs			**Compound Clauses**		**Derived Sentences**	
			CEV					
Register	**Tokens**	**All**	**Oracle**	**OB1**	**Gold**	**OB1**	**Gold**	**OB1**
Health	7 198	885	375	265	364	229	698	470
Literature	15 067	2 181	442	511	425	291	1 154	686
News	7 270	898	311	294	293	276	607	564

Table 13 contains information about the subset of data used to test the sentence simplification method when reducing the number of bound relative clauses in sentences. In many cases, the meanings of the column headings are the same as those provided about Table 12. In Table 13, sub-column *SSEV* of *Signs* displays

the number of left boundaries of bound relative clauses in the data set. *Bound Relatives* displays the number of sentences in the data set that contain one or more bound relative clauses. *Derived Sentences* is the number of sentences generated as a result of simplifying Type 2 sentences. We filtered sentences that did not contain signs manually tagged as being of class SSEV.

Table 13: *Characteristics of the test data used to evaluate the method to simplify Type 2 sentences*

		Signs			Bound Relatives		Derived Sentences	
			SSEV					
Register	**Tokens**	**All**	**Oracle**	**OB1**	**Gold**	**OB1**	**Gold**	**OB1**
Health	3 481	501	214	229	176	125	260	129
Literature	13 280	1 967	430	525	404	206	482	260
News	25 850	2 534	531	619	401	372	598	501

Evaluation Using Overlap Metrics

We used an existing implementation of the SARI metric (Xu, Napoles, Pavlick, Chen, and Callison-Burch, 2016)[23] to evaluate the sentence simplification systems described in this article. Xu et al. (2016) note that SARI "principally compares [s]ystem output [a]gainst [r]eferences and against the [i]nput sentence." It is based on a comparison of each simplified sentence generated by a system in response to an input sentence with both the original form of the input sentence and with the set of simplified sentences generated by human simplification of the input sentence. This metric is preferred to BLEU for the evaluation of sentence simplification systems because it is noted to correspond better with human judgements of simplification quality (Xu et al., 2016). We adapted the implementation of SARI, which evaluates sentences in isolation, to compute an average score over all simplified sentences output by the systems.

In addition to the SARI evaluation metric, we calculated the F_1-score of our method as the harmonic mean of precision and recall, given by Algorithm 2. In this algorithm,

$$sim = 1 - (\frac{ld(h,r)}{max(length(h), length(r))})$$

where h and r are sentences occurring in the gold standard and in the system response, respectively; ld is the Levenshtein distance between h and r (Levenshtein, 1966); and $length(x)$ is the length of x in characters. The intuition for use of

[23] Available at `https://github.com/cocoxu/simplification/blob/master/SARI.py` (last accessed on 14th September 2018).

Algorithm 2 is to find the best matches between sentences produced by the system and sentences in the gold standard while still allowing some small differences between them.

Input: H – set of simplified sentences in the gold standard for a given input sentence S
R – set of simplified sentences produced by the system for input sentence S.
$H_0 \leftarrow H$.
$R_0 \leftarrow R$.
Output: *Precision*, *Recall*

$matched\_pairs = 0$
while $|H| \neq 0$ *and* $|R| \neq 0$ **do**
  $h, r \leftarrow \arg\max_{h \in H, r \in R} (sim(h, r))$
  if $sim(h, r) > 0.95$ **then**
    $H = H \backslash \{h\}$
    $R = R \backslash \{r\}$
    $matched\_pairs \mathrel{+}= 1$
  else
    break
  end
end
$Precision = \frac{matched\_pairs}{|H_0|}$
$Recall = \frac{matched\_pairs}{|R_0|}$

Algorithm 2: Evaluation algorithm for sentence simplification

Table 15 displays evaluation statistics for methods to simplify Type 1 sentences. The *Bsln* sub-columns display the performance results of a baseline system exploiting the same rules as *OB1* but with each sign tagged using the majority class label observed for that sign in our annotated data. With the exceptions of those listed in Table 14, all signs were tagged with class label SSEV (left boundaries of subordinate clauses). Comparison of these results with those in the OB1 columns indicates the contribution made by the automatic sign tagger to the simplification task. The *MUSST* column presents evaluation results for a reduced version of the MUSST sentence simplification system (described in Section 2.1.1).[24] MUSST implements several types of syntactic simplification rule. In this table, we focused on performance of the one which splits sentences containing conjoint (compound) clauses, which, like the rules used by OB1, are used to simplify sentences of Type 1. We deactivated the other transformation functions (simplifying relative clauses, appositive phrases and passive sentences). The *Orcl* sub-columns display

[24] Available at `https://github.com/carolscarton/simpatico_ss` (last accessed on 11th September 2018).

the performance of the sentence rewriting method when it exploits error-free sign tagging. The column *OB1* displays the performance of our system when operating in fully-automatic mode, exploiting the sign tagger described in Section 4.

Table 14: *Tags most frequently assigned to the signs in our annotated corpus*

Majority Tag	**Signs**
CEV	[; or], [: but], [: and], [; but], [; and], [, but], [, and]
CLN	[or]
CMN1	[, or]
CMV1	[and]
ESEV	[,]
SPECIAL	[: that]
SSCM	[:]

Table 15: *System performance when simplifying Type 1 sentences*

	F_1**-score**				**SARI**			
Register	**Bsln**	**MUSST**	**OB1**	**Orcl**	**Bsln**	**MUSST**	**OB1**	**Orcl**
Health	0.362	0.281	0.495	0.613	0.201	0.124	0.362	0.514
Literature	0.150	0.101	0.208	0.262	0.203	0.087	0.202	0.229
News	0.233	0.237	0.690	0.706	0.119	0.171	0.596	0.623

When transforming Type 1 sentences in the registers of health and literature, the output of our automatic method is more similar to the gold standard than the output of the baseline system is. In this task, we see that the performance of OB1 also compares favourably with that of the reduced version of MUSST, which exploits a syntactic dependency parser. Calculated by comparing per-sentence Levenshtein similarity between sets of rewritten sentences, two-tailed paired samples t-tests revealed the results of both comparisons to be statistically significant for both F_1 and $SARI$ metrics for texts of all registers ($p \ll 0.01$). The only exception was when comparing the $SARI$ scores obtained by the *bsln* and *OB1* systems ($p = 0.0604$).

By contrast, when simplifying Type 2 sentences in texts of the registers of literature and news (Table 16), the baseline is more accurate than our automatic method. This result was statistically significant for literary texts ($p < 0.0005$). The performance of OB1 was superior to that of the baseline system when processing texts of the health register. This difference in performance was also statistically

Table 16: *System performance when simplifying Type 2 sentences*

	F_1-score				SARI			
Register	**Bsln**	**MUSST**	**OB1**	**Orcl**	**Bsln**	**MUSST**	**Orcl**	**OB1**
Health	0.231	0.063	0.306	0.315	0.207	0.020	0.285	0.296
Literature	0.572	0.000	0.516	0.791	0.168	0.008	0.204	0.289
News	0.583	0.141	0.577	0.629	0.434	0.056	0.451	0.467

significant ($p < 0.0005$). We observe that for the task of simplifying Type 2 sentences, performance of the OB1 system is far superior to that of the reduced version of MUSST.

The SARI evaluation metric indicates few statistically significant differences in the accuracy of the OB1 and *bsln* systems when simplifying Type 2 sentences. A statistically significant difference in performance was only evident for sentences of the health register, where $p = 0.036$. By contrast, differences between the accuracy scores obtained by OB1 and *MUSST* are statistically significant, in favour of OB1, when simplifying Type 2 sentences in texts of all registers ($p \ll 0.01$).

Evaluation of Individual Rules and Error Analysis

We investigated the accuracy of the individual rules exploited by OB1. In this context, accuracy is the ratio of the number of applications of each rule that led to the derivation of correct output sentences to the total number of applications of the rules. Overall, the rules used to transform Type 1 sentences have an accuracy of 0.6990. The rules used to transform Type 2 sentences have an accuracy of 0.5829. Two primary sources of error were found: the specificity of the rules, which limits their coverage; and the inability of the method to discriminate between signs of class CEV linking bound relative clauses and those linking independent clauses.

We categorised and quantified errors made by the *OB1* and *MUSST* systems. For us, each error is a sequence of sentences output by the system in response to a given input sentence that is less than 95% similar to the sequence of sentences produced by linguists when simplifying the same sentence. Across all registers, when transforming Type 1 sentences, information about the five most frequent categories of error made by OB1 is presented in Table 17.

In Table 17, the columns provide error category labels (*error category*), examples of the simplification of a given input sentence by linguists (*human simplified*), examples of the simplification of that sentence by our system (*OB1 simplified*), the similarity of the two simplifications (*Similarity*), and the frequency of errors of this type in our test data (*Freq*). This information is provided for the five most frequent categories of error.

Sign tagging errors are those caused when OB1 fails to simplify a sentence

Table 17: *Example errors when rewriting Type 1 sentences (OB1)*

Error category	**Human simplified**	**OB1 simplified**	**Similarity**	**Freq (%)**
Sign tagging	Bloodstained knives were found in the kitchen. An axe was discovered in the bedroom near the bodies of the two men.	Bloodstained knives were found in the kitchen and an axe was discovered in the bedroom near the bodies of the two men.	0.48	405 (53.78)
Incorrect transformation	'How they came to be committed is not clear,' he said. 'That they were committed and committed by you is abundantly clear,' he said.	"How they came to be committed is not clear. They were committed and committed by you is abundantly clear," he said.	0.84	82 (10.89)
Missing pattern	"The organisation was no stranger to the imposition of serious violence against those who might seek to challenge them. Few could afford to trifle with their wishes."	"The organisation was no stranger to the imposition of serious violence against those who might seek to challenge them and few could afford to trifle with their wishes."	0.35	77 (10.23)
Left conjoin too wide	I said 'Maybe' because I wanted to know what he was talking about. I said 'Maybe' because I wanted to know who he was talking with.	I said 'Maybe' because I wanted to know what he was talking about. I said 'who he was talking with.	0.49	66 (8.76)
Left conjoin too narrow	Most are tablets or liquid that you swallow. You may need an injection, a suppository (see page 29) or an inhaler.	Most are tablets or liquid that you swallow. Most are tablets or liquid that you may need an injection, a suppository (see page 29) or an inhaler.	0.84	48 (6.37)

correctly due to a failure to correctly tag the clause coordinator.[25] *Incorrect transformation* errors are those caused when the activated transformation rule fails to generate correct output for some other reason. *Missing pattern* errors are those caused when OB1 makes no transformation of the input sentence despite the fact that the relevant sign of syntactic complexity has been correctly tagged. Overcoming such errors requires the addition of new transformation rules into the set used by OB1. The *left conjoin too wide* and *left conjoin too narrow* errors are those made when the patterns used by the transformation rules incorrectly identify the left boundaries of compound clauses.

In our error analysis, we were able to distinguish *sign tagging* from *missing pattern* errors by examining the tagged versions of input sentences. When the clause coordinator is tagged as being of a different class, the simplification is a sign tagging error. When the clause coordinator is correctly tagged, the simplification is a *missing pattern* error.

Across all registers, when transforming Type 2 sentences, information about the five most frequent categories of error made by OB1 is presented in Table 18.

In Table 18, *sign tagging* errors are those caused when OB1 fails to simplify a sentence correctly due to a failure to correctly classify the left boundary of the relative clause. *Matrix NP too narrow* errors are a subset of those made when the applied transformation rule fails to correctly identify the left boundary of the complex NP that the relative clause modifies. *Relative clause too narrow* errors are a subset of those made when the applied transformation rule fails to correctly identify the right boundary of the complex NP that the relative clause modifies. *Incorrect transformation* errors are those caused when the activated transformation rule fails to generate correct output for some other reason.

We also categorised the errors occurring in 100 sentences of each type processed using *MUSST*. The two main categories of error were caused by inaccurate syntactic parsing. This led to failures in detecting compound clauses and complex NPs in input sentences (91.67% and 97.14% of errors, respectively) and inaccuracies when transformation rules are applied to incorrectly identified syntactic constituents (8.33% and 2.86% of errors, respectively). The first of these categories causes a failure in the system to perform any transformation of a complex input sentence. Examples of erroneous output generated by *MUSST* when transformations are applied to incorrectly parsed sentences (the second category of error) are provided in Table 19. For comparison, human simplifications of these sentences are provided in the *human simplified* column. In this table, column *sim.* displays the similarity of the automatically simplified sentence to the human simplified one, as computed using the *sim* function described in Section 6.2.1.

[25] In this row of Table 17, manual inspection of the automatically tagged sentence revealed that the clause coordinator *and* was misclassified as a coordinator of NPs and as a result, no simplification operation was performed.

Table 18: *Example errors when rewriting Type 2 sentences (OB1)*

Error category	**Human simplified**	**OB1 simplified**	**Similarity**	**Freq (%)**
Missing pattern	"Instead of a bouncy healthy boy his parents had this thrust upon them. This must have been beyond their wildest nightmares."	"Instead of a bouncy healthy boy his parents had this thrust upon them which must have been beyond their wildest nightmares."	0.28	102 (44.15)
Sign tagging	Cotterell was trapped when his fingerprints were found in a room. The women were killed in that room. Cotterell by then was working for Kleeneze.	Cotterell was trapped when his fingerprints were found in the room where the women were killed. Cotterell by then was working for Kleeneze.	0.29	78 (33.77)
Matrix NP too narrow	And the hearing was told that a 14-year-old boy was also flying on the same frequency. That 14-year-old boy cannot be named.	And the hearing was told that a 14-year-old boy was also flying on the same frequency. Boy cannot be named.	0.55	17 (7.36)
Incorrect transformation	Connie rarely left the cottage, but Janice was regularly seen walking her whippets. She bred her whippets as coursing dogs and showed at Crufts.	Connie rarely left the cottage, but Janice was regularly seen walking her whippets. Her she bred as coursing dogs and showed at Crufts.	0.85	16 (6.93)
Relative clause too wide	Staff complained that the subsequent inquiry was serviced by managers and that it failed to call key witnesses. These managers had already defended Francis.	Staff complained that the subsequent inquiry was serviced by managers. These managers had already defended Francis and that it failed to call key witnesses.	0.47	11 (4.76)

Table 19: *Transformations applied to incorrectly parsed sentences (*MUSST*)*

Transformation type	**Human simplified**	***MUSST* simplified**	**Sim.**	**Freq. (%)**
Compound clauses	Elaine Trego never bonded with 16-month-old Jacob, a murder trial was told. He was often seen with bruises, a murder trial was told.	Elaine Trego never bonded with Jacob. And Elaine Trego he was often seen with bruises, a murder trial was told.	0.38	(8.33)
Nominally bound relative clauses	And last night police said fellow officers had reopened their files on three unsolved murders. These police saw Kevin Cotterell caged.	And last night police caged said fellow officers had reopened their files on three unsolved murders. Police saw Kevin Cotterell.	0.73	(2.86)

6.2.2 Automatic Estimation of Readability

We used six readability metrics to estimate the impact of three sentence simplification methods (*MUSST*, *OB1*, and *Orcl*) on propositional density, reading grade level, syntactic complexity, and various aspects of coherence. The selected metrics were propositional idea density (Brown, Snodgrass, Kemper, Herman, and Covington, 2008);[26]; Flesch-Kincaid Grade Level (Kincaid, Fishburne, Rogers, and Chissom, 1975), obtained via the *style* package;[27] and four metrics from the Coh-Metrix package (McNamara, Graesser, McCarthy, and Cai, 2014).[28] These were *syntactic simplicity* and three others providing information about text cohesion:

- *referential cohesion*, which measures the extent to which words and ideas overlap across sentences and across the entire text, forming explicit threads that connect the text for the reader (Lei, 2014),
- *deep cohesion*, which uses frequency counts of causal and intentional connectives and the causal and logical relationships expressed within the text. When the text contains many relationships but few connectives, it is more difficult to process as readers must infer the relationships between ideas in the text, and

[26] Calculated using CPIDR, available for download via a link at `http://ai1.ai.uga.edu/caspr/` (last accessed on 11th September 2018).
[27] *Style* is a Linux command-line utility, part of the GNU software suite.
[28] Calculated using the online demo at `http://tool.cohmetrix.com/` (last accessed on 11th September 2018).

- *temporality*, which uses information on the number of inflected tense morphemes, temporal adverbs, and other explicit cues to estimate the consistency of tense and aspect in the text to estimate the ease with which it can be processed and understood.

Flesch-Kincaid Grade Level was selected because it has been widely used in previous evaluations of syntactic simplification systems (e.g. Woodsend and Lapata (2011); Wubben et al. (2012); Glavas and Stajner (2013); Vu, Tran, and Pham (2014)).

Accessible texts are expected to have small values for propositional density and Flesch-Kincaid Grade Level and large values for the other four metrics. Readability scores were obtained for texts in their original form and the form output by the simplification methods when processing Type 1 sentences (Table 20) and Type 2 sentences (Table 21). In the tables, the *orig* columns present values of each metric obtained for the original versions of the texts.

Table 20: *Estimated readability of text output when transforming Type 1 sentences*

Register	**Orig**	**MUSST**	**OB1**	**Orcl**	**Orig**	**MUSST**	**OB1**	**Orcl**
	Propositional Idea Density				**Flesch-Kincaid Grade Level**			
Health	0.523	0.521	0.510	0.503	8.9	7.4	6.0	5.4
Literature	0.593	0.588	0.588	0.592	10.3	7.1	5.4	6.0
News	0.505	0.502	0.483	0.482	9.6	7.9	5.4	5.3
	Referential Cohesion				**Deep Cohesion**			
Health	41.68	37.45	26.43	23.89	96.16	94.41	92.07	90.66
Literature	90.49	50.00	65.17	72.24	72.91	68.79	64.63	63.68
News	40.90	34.83	35.20	51.99	56.36	54.38	48.4	46.02
	Syntactic Simplicity				**Temporality**			
Health	83.89	91.62	96.78	98.26	52.39	51.2	54.38	53.98
Literature	10.93	58.32	69.50	55.17	63.31	81.86	72.57	76.42
News	46.81	66.64	89.07	85.77	27.76	40.52	30.15	35.94

Inspection of Tables 20 and 21 reveals that all of the automatic systems generate texts that are more readable, in terms of propositional density and Flesch-Kincaid Reading Grade level, than the originals. These metrics also indicate that OB1 compares favourably with the *MUSST* system when simplifying sentences of Type 1 and Type 2. For all registers, the automatic transformation of input sentences led to the generation of texts with reduced *propositional idea density*, making them more readable. When transforming Type 1 sentences in texts of the registers of health and news, of the fully automatic systems, the greatest reduction in propositional

Table 21: *Estimated readability of text output when transforming Type 2 sentences*

Register	**Orig**	**MUSST**	**OB1**	**Orcl**	**Orig**	**MUSST**	**OB1**	**Orcl**
	Propositional Idea Density				**Flesch-Kincaid Grade Level**			
Health	0.500	0.493	0.499	0.500	8.4	8.3	8.3	8.3
Literature	0.597	0.599	0.592	0.594	9.9	9.4	6.8	6.9
News	0.489	0.486	0.478	0.480	10.3	9.6	7.7	7.9
	Referential Cohesion				**Deep Cohesion**			
Health	39.74	45.62	38.97	41.29	87.70	87.90	87.49	87.49
Literature	70.54	55.57	33.72	66.28	81.59	59.48	77.94	77.04
News	24.51	32.64	18.41	44.04	63.31	65.17	61.79	59.87
	Syntactic Simplicity				**Temporality**			
Health	68.44	68.44	69.85	68.79	66.28	62.17	65.17	64.06
Literature	22.36	56.36	58.71	46.41	65.17	28.10	58.71	64.06
News	38.59	41.29	81.86	85.77	28.10	30.50	27.76	30.15

idea density was made by OB1. When transforming Type 2 sentences, the same observation was made for texts of the registers of literature and news.

Inspection of Table 20 reveals that the original versions of the input texts, estimated by the *referential cohesion* and *deep cohesion* metrics, are more readable than those generated by the fully automatic systems that transform Type 1 sentences. The effect on *referential cohesion* may be explained by the fact that the transformation operations increase the numbers of sentences in the texts, reducing the amount of word overlap between adjacent sentences. These findings can be taken as evidence that the transformation operations have a disruptive effect on the coherence of a text. It was noted for texts of all registers. With respect to *referential cohesion*, when transforming Type 1 sentences, only the *orcl* system is able to generate news texts that are more accessible than the originals. For text of the health register, use of *MUSST* harms readability considerably less than OB1, while the reverse is true when transforming literary texts. When transforming Type 2 sentences, *MUSST* was the only automatic system to generate texts that were more referentially cohesive than the originals, in the registers of health and news. From this, we can infer that the conversion of long sentences with many concept mentions into sequences of shorter sentences with fewer concept mentions reduces cohesion by spacing out these mentions over multiple sentences and reducing their repetition in adjacent ones. The data in Tables 20 and 21 shows that, with respect to the *deep cohesion* metric, texts generated by OB1 are not as readable as the originals (sentences of both types) or those generated by *MUSST* (Type 1 sentences). One possible reason for this is that the transformations performed by OB1 generate

texts containing fewer connectives while those performed by *MUSST* do not. When splitting a sentence containing a compound clause into two, *MUSST* preserves the conjunction as the first word of the second output sentence.

The statistics presented in Tables 20 and 21 indicate that for all registers, the sentence simplification systems generate texts with greater *syntactic simplicity* than the originals. Overall, the texts generated by OB1 are indicated to be considerably simpler, syntactically, than those generated by *MUSST*.

When transforming Type 1 sentences, the *temporality* columns in Table 20 indicate that texts generated by the automatic systems are more consistent in terms of tense and aspect than the originals. We observe that the *MUSST* system brings greater improvements in this metric than OB1, except when processing health texts. This implies that the transformation operations applied by OB1 (Section 5.2) introduce more inconsistencies with respect to tense and aspect than those used by *MUSST*. When transforming Type 2 sentences, statistics in Table 21 indicate that, with respect to the *temporality* metric, none of the automatic systems is able to improve the readability of the input texts, save for the *MUSST* system when processing texts of the news register.

6.2.3 Reader Opinions

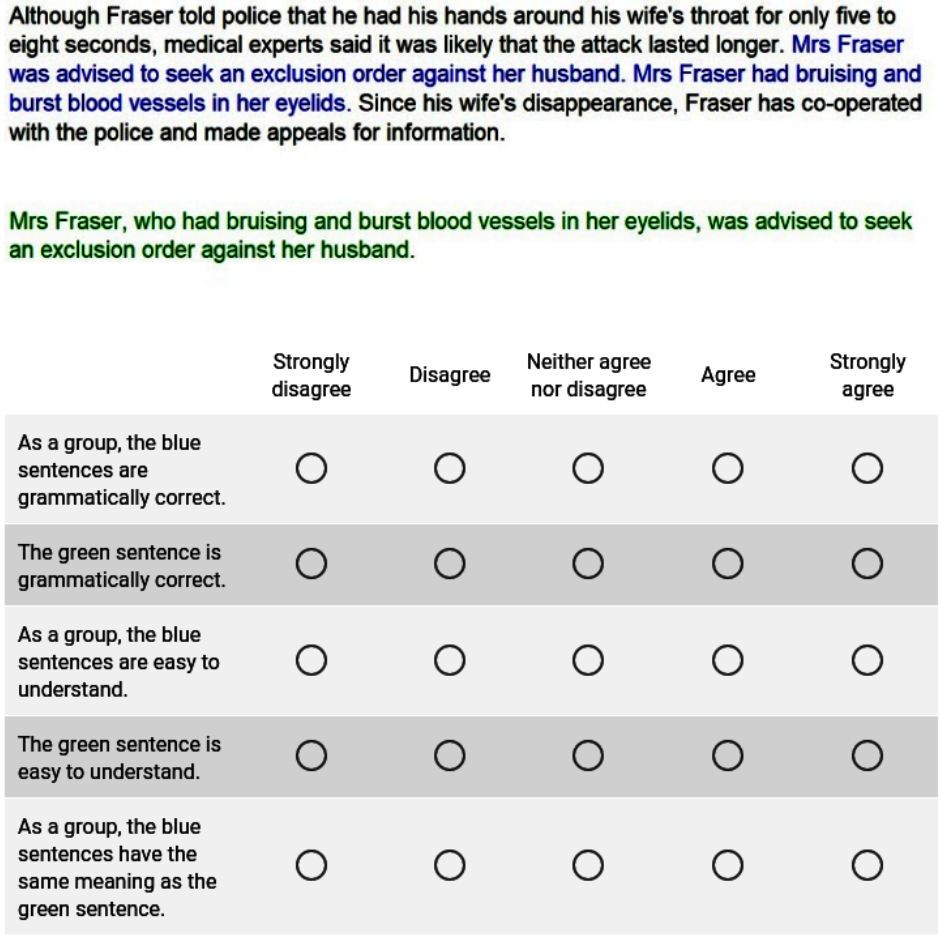
Following human-centred evaluation methods used in previous work (e.g. Angrosh and Siddharthan (2014); Wubben et al. (2012); Feblowitz and Kauchak (2013)), we used the output of OB1 to create items surveying the extent to which readers agreed with five statements about the grammaticality, readability, and meanings of sentences in their original and simplified forms.[29] Figure 1 displays one such survey item. Each participant in this assessment task provided opinions for each of 150 sentences that had been transformed by the sentence simplification method. As a result, this aspect of our evaluation ignores potentially complex sentences that the system failed to transform. These failings are captured by the comparison of system output with human simplifications described in Section 6.2.1.

In our evaluation based on reader opinions, five participants each responded to eight items[30] in nineteen surveys. Four of our participants were fluent non-native speakers of English while one was a native speaker.[31]

We converted participants' extent of agreement with the opinions as integer values ranging from 1 for strong disagreement to 5 for strong agreement. Overall, participants grudgingly agreed that sentences generated by OB1 are easy to understand[32] ([3.789, 4.050]) and collectively have the same meaning as the antecedent sentences ([3.721, 4.017]). Although derived from a smaller number of

[29] These criteria are analogous to *fluency*, *simplicity*, and *meaning preservation*, respectively, used by Angrosh and Siddharthan (2014).
[30] Survey 19 contained six items
[31] The native speaker was Evans, who also performed the error analysis described in Section 6.2.1.
[32] In this section, we provide 95% confidence intervals in square brackets.

Fig. 1: Opinion Survey Item



participants, this compares favorably with agreement scores obtained for various text simplification systems in experiments conducted by Saggion et al. (2015).

Participants rated OB1's output as most readable for Type 1 sentences of the news register ($\mu = 4.053$). They rated it as least readable for Type 2 sentences ($\mu < 3.9$), especially for texts of the literary register ($\mu = 3.683$). Participants perceived that sentence transformations made by OB1 preserved meaning better for Type 1 sentences ($\mu = 3.9853$) than Type 2 sentences ($\mu = 3.7511$). Overall, transformations were felt to best preserve meaning when applied to Type 1 sentences of the news register ($\mu = 4.053$). Our participants were most undecided ($\mu = 3.1111$) about the preservation of meaning in output derived from Type 2 sentences in the health register.

Participants broadly agreed that sentences output by OB1 are grammatical ([4.031, 4.250]) but that the antecedent sentences were already easy to understand

([4.318, 4.443]). They also strongly agreed that the original versions of the sentences were grammatical ([4.686, 4.769]). Opinions expressed in the surveys indicate that participants found the original sentences significantly more readable than those generated using OB1 ($p \ll 0.05$). We noted many cases where participants agreed equally strongly that sentences were easy to understand in both their original and simplified forms, despite the fact that, objectively, the latter contained fewer complex constituents. One possible explanation for this is that survey participants were not first provided with example sentences demonstrating different levels of complexity or readability. Access to such examples may have elicited better informed judgments about the relative complexities of the presented sentences.

We examined correlations (Pearson's correlation coefficient) between different variables in the opinion survey. The three most closely correlated ($0.6840 \leq r \leq 0.8168$) were between:

1. the perceived readability of sentences generated by OB1 and the perceived grammatical correctness of those sentences,
2. the perceived readability of sentences generated by OB1 and the perceived extent to which those sentences preserved the meanings of their antecedents, and
3. the perceived extent to which automatically generated sentences were grammatical and the perceived extent to which they preserved the meanings of their antecedents.

We found no linear relationship between the similarities of system-generated simplifications to gold standard simplifications and the perceived accessibility and grammaticality of those simplifications (Pearson's $r = 0.1716$ and $r = 0.0625$, respectively). There is a small linear relationship between the similarities of system-generated simplifications to gold standard simplifications and the extent to which readers perceived that the system-generated simplifications preserve the meanings of their antecedent sentences ($r = 0.3039$). This correlation was slightly closer for simplifications of Type 2 sentences ($r = 0.4705$).

Our observation from the reader opinion survey is that, overall, participants found the output of our system to be usable. It was generally agreed to be grammatical, to be readable, and to preserve the meanings of the original sentences. The results of the opinion surveys tend to reinforce the findings of a our comparison of system output with human-produced text simplifications (Section 6.2.1).

7 Conclusions and Future Work

In this article, we presented a rule-based approach to automatically simplify English sentences by reducing the numbers of compound clauses and nominally bound relative clauses that they contain. The method exploits components for sentence analysis and iterative sentence transformation.

Our new sentence analysis component, the tool to tag signs of syntactic complexity in accordance with a scheme specifying their syntactic linking and bounding functions, achieved acceptable levels of accuracy. The sign tagger made a

useful contribution to performance of the overall system, outperforming a baseline method in which signs are uniformly tagged with empirically observed majority class labels. Parameters of the sign tagger were set optimally in a performance-driven development process. The sign tagger is able to identify signs of syntactic complexity with a wide range of coordinating and bounding functions, which may be useful for multiple applications. We envisage that it could be used in other NLP tasks such as syntactic parsing, automatic prosody generation, and for additional types of sentence simplification, in a manner similar to that described by Evans (2011).

Evaluation of the iterative rule-based approach to sentence transformation revealed that the rules used to rewrite Type 1 sentences (containing compound clauses) are considerably more accurate than those used to rewrite Type 2 sentences (containing nominally bound relative clauses). This may be due to the fact that in the latter task it is necessary both to identify the spans of nominally bound relative clauses and to discriminate between free and bound relative clauses. There is considerable room for improvement of the sentence transformation process. The number of rules developed and the large number of compound clauses and bound relative clauses that they fail to process indicates to us that a machine learning approach to rule development may have better performance. We envisage a sequence tagging method to identify parts of sentences preceding and following each compound clause. A similar method would be used to identify the parts of sentences following complex noun phrases, the phrases to which relative clauses are bound, and the parts of the sentence preceding those phrases. A substantially smaller set of rules would then be used to combine these elements into a simplified form. These methods would be exploited by the functions $rewrite_{CEV}$ and $rewrite_{SSEV}$, used in the sentence transformation algorithm (Section 5). In pursuit of this task, future work will include developing training data to support such an approach to sentence simplification.

Our evaluation results indicate that the current version of the method is effective at rewriting Type 1 sentences in texts of the registers of *news* and *health*. However, performance is relatively poor when rewriting Type 2 sentences and when performing any kind of sentence simplification in texts of the literary register. This finding is in accord with previous observations about the application of NLP methods to literary texts.

In this work, we have proposed a shallow method for sentence analysis to overcome practical difficulties in the syntactic parsing of long complex sentences. Where feasible, we consider full parsing of input sentences to be preferable to shallow syntactic analysis for the task of sentence simplification. As a result, we are interested in investigating the development of a hybrid approach in which the sign tagger is used as a last resort for the analysis of extremely long sentences and a syntactic parser is used for the analysis of shorter sentences whose parsing is computationally feasible.

Our use of the *deep cohesion* metric to assess the readability of system output indicates that our approach to sentence simplification leads to some loss of rhetorical relations expressed in the original texts. One possible way to address this would be to add new transformation rules to simplify sentences containing conjunctions

such as *but*, which signal contrast between the clause conjoins that they link. In this approach, a sentence-initial "canned" adverb such as *however* could be inserted into the simplified sentence containing the second conjoin, re-establishing the lost rhetorical relation. We plan, in future work, to conduct a more detailed analysis of the impact of sentence simplification operations on the rhetorical structure of output texts.

The human-centred evaluation carried out in this research revealed that participants found the output of the system to be usable. This was determined on the basis of their responses to opinion surveys focused on the grammaticality, accessibility, and meaning of sentences output by the method. Subsequent human-centred evaluations would be improved through the recruitment of larger numbers of participants in the opinion surveys and the use of more objective psycholinguistic evaluation methods such as eye tracking, self-paced reading, rapid serial visual presentation, or reading comprehension testing.

Acknowledgements

This work was supported by the European Commission under the Seventh (FP7-2007-2013) Framework Programme for Research and Technological Development [287607]. We gratefully acknowledge Emma Franklin, Zoë Harrison, and Laura Hasler for their contribution to the development of the data sets used in our research and Iustin Dornescu for his contribution to the development of the sign tagger. For their participation in the user surveys, we thank Martina Cotella, Francesca Della Moretta, Arianna Fabbri, and Victoria Yaneva. We gratefully acknowledge Larissa Sayuri Futino Castro dos Santos for assistance in collating our survey data.

References

Agarwal, R. and Boggess, L. 1992. A simple but useful approach to conjunct identification. In *Proceedings of the 30th annual meeting for Computational Linguistics*, Newark, Delaware, pp. 15–21. Association for Computational Linguistics.

Aluisio, S. M., Specia, L., Pardo, T. A. S., Maziero, E. G., and Fortes, R. P. M. 2008a. Towards Brazilian Portuguese Automatic Text Simplification Systems. In *Proceedings of the eighth ACM symposium on Document engineering, DocEng '08*, Sao Paulo, Brazil, pp. 240–8. ACM.

Aluisio, S. M., Specia, L., Pardo, T. A. S., Maziero, E. G., Caseli, H. M., and Fortes, R. P. M. 2008b. A Corpus Analysis of Simple Account Texts and the Proposal of Simplification Strategies: First Steps Towards Text Simplification Systems. In *Proceedings of the 26th annual ACM international conference on Design of communication, SIGDOC '08*, Lisbon, Portugal, pp. 15–22. ACM.

Angrosh, M. A. and Siddharthan, A. 2014. Text simplification using synchronous dependency grammars: Generalising automatically harvested rules. In *Proceedings of the 8th International Natural Language Generation Conference*, Philadelphia, Pennsylvania, pp. 16–25. Association for Computational Linguistics.

Angrosh, M., Nomoto, T., and Siddharthan, A. 2014. Lexico-syntactic text simplification and compression with typed dependencies. In *Proceedings of COLING 2014, the*

25th International Conference on Computational Linguistics: Technical Papers,, Dublin, Ireland, pp. 1996–2006.

Bennetto, L., Pennington, B. F., and Rogers, S. J. 1996. Intact and impaired memory functions in autism. *Child Development*, *67 (4)*.

Bos, J. 2008. Wide-coverage Semantic Analysis with Boxer. In *Proceedings of the 2008 Conference in Semantics in Text Processing*, Venice, Italy, pp. 277–86.

Bott, S., Saggion, H., and Figueroa, D. 2012. A Hybrid System for Spanish Text Simplification. In *Proceedings of the NAACL-HLT 2012 Workshop on Speech and Language Processing for Assistive Technologies (SLPAT)*, Montréal, Canada, pp. 75–84.

Brill, E. 1994. Some Advances in Transformation-Based Part of Speech Tagging. In *Proceedings of the Twelfth National Conference on Artificial Intelligence*, Seattle, Washington, pp. 722–7.

Brouwers, L., Bernhard, D., Ligozat, A.-L., and Francois, T. 2014. Syntactic Sentence Simplification for French. In *Proceedings of the 3rd Workshop on Predicting and Improving Text Readability for Target Reader Populations (PITR) at EACL 2014*, Gothenburg, Sweden, pp. 47–56. Association for Computational Linguistics.

Brown, C., Snodgrass, T., Kemper, S. J., Herman, R., and Covington, M. A. 2008. Automatic measurement of propositional idea density from part-of-speech tagging. *Behavior Research Methods*, *40 (2)*.

Canning, Y. 2002. *Syntactic Simplification of Text*. Ph.d. thesis, University of Sunderland.

Caplan, D. and Waters, G. S. 1999. Verbal working memory and sentence comprehension. *Behavioural and Brain Sciences*, *22*, pp. 77–126.

Cavender, A., Trewin, S., and Hanson, V. 2015. Accessible writing guide. Accessed from http://www.sigaccess.org/welcome-to-sigaccess/resources/accessible-writing- guide/ on 8th December 2015.

Chandrasekar, R., Doran, C., and Srinivas, B. 1996. Motivations and methods for text simplification. In *Proceedings of the 16th International Conference on Computational Linguistics (COLING '96)*, Copenhagen, Denmark, pp. 1041–4.

Chomsky, N. 1970. Remarks on nominalization. In Jacobs, R. and Rosenbaum, P. (eds.) *Reading in English Transformational Grammar*, Ginn and Company, pp. 184–221.

Cohen, J. 1960. A Coefficient of Agreement for Nominal Scales. *Educational and Psychological Measurement*, *20*(1).

Cohn, T. and Lapata, M. 2009. Sentence Compression as Tree Transduction. *Journal of Artificial Intelligence Research*, *20*(34).

Coster, W. and Kauchak, D. 2011. Simple English Wikipedia: A new text simplification task. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics (ACL-2011)*, Portland, Oregon, pp. 665–9. Association of Computational Linguistics.

Daelemans, W., Höthker, A., and Tjong Kim Sang, E. 2004. Automatic Sentence Simplification for Subtitling in Dutch and English. In *Proceedings of the 4th International Conference on Language Resources and Evaluation (LREC-2004)*, Lisbon, Portugal, pp. 1045–8.

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

DeFrancesco, C. and Perkins, K. 2012. An analysis of the proposition density, sentence and clause types, and nonfinite verbal usage in two college textbooks. In M. S. Plakhotnik, S. M. Nielsen, and D. M. Pane (Eds.), *Proceedings of the 11th Annual College of Education & GSN Research Conference*, Miami, Florida, pp. 20–5.

Dornescu, I., Evans, R., and Orasan, C. 2013. A tagging approach to identify complex constituents for text simplification. In *Proceedings of the 9th International Conference on Recent Advances in Natural Language Processing (RANLP-2013)*, Hissar, Bulgaria, pp. 221–9.

Evans, R. 2011. Comparing methods for the syntactic simplification of sentences in information extraction. *Literary and Linguistic Computing*, *26 (4)*, 371–88.w

Evans, R. and Orasan, C. 2013. Annotating signs of syntactic complexity to support sentence simplification. In I. Habernal and V. Matousek (Eds.), *Text, Speech and Dialogue. Proceedings of the 16th International Conference TSD 2013*, pp. 92–104. Plzen, Czech Republic: Springer.

Feblowitz, D. and Kauchak, D. 2013. Sentence Simplification as Tree Transduction. In *Proceedings of the 2nd Workshop on Predicting and Improving Text Readability for Target Reader Populations*, Sofia, Bulgaria, pp. 1–10. Association for Computational Linguistics.

Ferrés, D., Marimon, M., and Saggion, H. 2015. A Web-Based Text Simplification System for English. *Procesamiento del Lenguaje Natural*, *55*, 191–4.

Gaizauskas, R., Foster, J., Wilks, Y. Arundel, J., Clough, P., and Piao, S. 2001. The Meter corpus: A corpus for analysing journalistic text reuse. In *Proceedings of Corpus Linguistics 2001 Conference*, Lancaster, UK, pp. 214–23. Lancaster University Centre for Computer Corpus Research on Language.

Glavas, G. and Stajner, S. 2013. Event-centered simplification of news stories. In *Proceedings of the Student Workshop held in conjunction with RANLP 2013*, Hissar, Bulgaria, pp. 71–8.

Gonzalez-Dios, I., Aranzabe, M. J., and Díaz de Ilarraza, A. 2018. The corpus of Basque simplified texts (CBST). *Language Resources and Evaluation*, *52*.

Grover, C., C. Matheson, A. Mikheev, and M. Moens 2000. LT TTT - a flexible tokenisation tool. In *Proceedings of Second International Conference on Language Resources and Evaluation*, Athens, Greece, pp. 1147–54.

Hepple, M. 2000. Independence and commitment: Assumptions for rapid training and execution of rule-based POS taggers. In *Proceedings of the 38th Annual Meeting of the Association for Computational Linguistics*, Hong Kong, pp. 278–85. Association for Computational Linguistics.

Jay, T. B. 2003. *The psychology of language*. Upper Saddle Rive, NJ: Pearson.

Jelínek, T. 2014. Improvements to Dependency Parsing Using Automatic Simplification of Data. In *Proceedings of LREC-2014*, Reykjavik, Iceland, pp. 73–7. European Language Resources Association.

Jonnalagadda, S., Tari, L., Hakenberg, J., Baral, C., and Gonzalez, G. 2009. Towards Effective Sentence Simplification for Automatic Processing of Biomedical Text. In *Proceedings of NAACL HLT 2009: Short Papers*, Boulder, Colorado, pp. 177–80. Association for Computational Linguistics.

Kincaid, J. P., Fishburne, R. P., Rogers, R. L., and Chissom, B. S. 1975. Derivation of new readability formulas (Automatic Readability Index, Fog Count and Flesch Reading Ease Formula) for Navy enlisted personnel. CNTECHTRA Research Branch Report 8-75, CNTECHTRA.

Kintsch, W. and Welsch, D. M. 1991. The construction–integration model: A framework for studying memory for text. In W. E. Hockley and S. Lewandowsky (Eds.), *Relating theory and data: Essays on human memory*, pp. 367–85. NJ: Erlbaum: Hillsdale.

Klerke, S., Goldberg, Y, and Søgaard, A. 2016. Improving sentence compression by learning to predict gaze. In *Proceedings of NAACL-HLT 2016*, San Diego, California, pp. 1528–33. Association for Computational Linguistics.

Kudo, T. 2005. Crf++: Yet another crf toolkit. *Software available at http://crfpp. sourceforge. net*.

Lafferty, J., McCallum, A., and Pereira, F. C. 2001. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *Proceedings of the 18th International Conference on Machine Learning*, Rümlang, Switzerland, pp. 282–9. Morgan Kaufmann.

Lei, C.-U., Man, K. L., and Ting, T. O. 2014. Using Coh-Metrix to Analyse Writing Skills of Students: A Case Study in a Technological Common Core Curriculum Course. In *Proceedings of the International MultiConference of Engineers and Computer Scientists 2014 Vol II, IMECS 2014*, Hong Kong, pp. 3–6. IMECS.

Levenshtein, V. I. 1966. Binary Codes Capable of Correcting Deletions and Insertions and Reversals. *Soviet Physics Doklady*, *10 (8)*.

Madnani, N. 2011. iBLEU: Interactively debugging and scoring statistical machine translation systems. In *Proceedings of the Fifth IEEE Conference on Semantic Computing (ICSC)*, Palo Alto, California, pp. 213–4. IEEE.

Maier, W., Kübler, S., Hinrichs, E., and Kriwanek, J. 2012. Annotating coordination in the penn treebank. In *Proceedings of the Sixth Linguistic Annotation Workshop*, Jeju, Republic of Korea, pp. 166–74. Association for Computational Linguistics.

Marcus, M. P., Santorini, B., and Marcinkiewicz, M. A. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics*, *19*(2).

de Marneffe, M.-C., MacCartney, W, and Manning, C. D. 2006. Generating typed dependency parses from phrase structure parses. In *Proceedings of the International Conference on Language Resources and Evaluation (LREC)*, Genoa, Italy, pp. 449–54. ELDA.

Martos, J., Freire, S., González, A., Gil, D., Evans, R., Jordanova, V., Cerga, A., Shishkova, A., and Orasan, C. 2013. User preferences: Updated. Technical Report D2.2, Deletrea, Madrid, Spain.

Max, A. 2000. *Syntactic simplification - an application to text for aphasic readers*. Mphil in computer speech and language processing, University of Cambridge, Wolfson College.

McDonald, R. T. and Nivre, J. 2011. Analyzing and integrating dependency parsers. *Computational Linguistics*, *37*(1).

McNamara, D. S., Graesser, A. C., McCarthy, P. M., and Cai, Z. 2014. *Automated Evaluation of Text and Discourse with Coh-Metrix*. Cambridge University Press.

Mishra, K., Soni, A., Sharma, R., and Sharma, D. 2014. Exploring the effects of Sentence Simplification on Hindi to English Machine Translation System In *Proceedings of the Workshop on Automatic Text Simplification: Methods and Applications in the Multilingual Society*, Dublin, Ireland, pp. 21–9. Association for Computational Linguistics.

Miwa, M., Sætre, R., Miyao, Y., and Tsujii, J. 2010. Entity-Focused Sentence Simplification for Relation Extraction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, Beijing, China, pp. 788–96. Association for Computational Linguistics.

Narayan, S. and Gardent, C. 2014. Hybrid Simplification using Deep Semantics and Machine Translation. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics*, Baltimore, Maryland, pp. 435–45. Association for Computational Linguistics.

Ogden, C. K. 1932. *Basic English: a general introduction with rules and grammar*. London: K. Paul, Trench, Trubner & Co., Ltd.

Paetzold, G. H. and Specia, L. 2013. Text Simplification as Tree Transduction. In *Proceedings of the 9th Brazilian Symposium in Information and Human Language Technology*, Fortaleza, CE, Brazil, pp. 116–25. Sociedade Brasileira de Computação.

Papineni, K., Roukos, S., Ward, T., and Zhu, W. J. 2002. BLEU: a method for automatic evaluation of machine translation. In *Proceedings of the 40th annual meeting for Computational Linguistics*, Philadelphia, Pennsylvania, pp. 311–8. Association for Computational Linguistics.

Quirk, R., Greenbaum, S., Leech, G., and Svartvik, J. 1985. *A comprehensive grammar of the English language*. Longman.

Rennes, E. and Jönsson, A. 2015. A Tool for Automatic Simplification of Swedish Texts. In *Proceedings of the 20th Nordic Conference of Computational Linguistics (NODALIDA 2015)*, Vilnius, Lithuania, pp. 317–20. LiU Electronic Press.

Rindflesch, T. C., Rajan, J. V., and Hunter, L. 2000. Extracting molecular binding relationships from biomedical text. In *Proceedings of the sixth conference on Applied natural language processing*, Seattle, Washington, pp. 188–95. Association of Computational Linguistics.

Saggion, H., Štajner, S., Bott, S., Mille, S., Rello, L., and Drndarevic, B. 2015. Making It Simplext: Implementation and Evaluation of a Text Simplification System for Spanish. *ACM Transactions on Accessible Computing (TACCESS) - Special Issue on Speech and Language Processing for AT (Part 2)* , *6*(4).

Scarton, C., Palmero Aprosio, A., Tonelli, S., Martin-Wanton, T., and Specia, L. 2017. MUSST: A Multilingual Syntactic Simplification Tool. In *The Companion Volume of the IJCNLP 2017 Proceedings: System Demonstrations*, Taipei, Taiwan, pp. 25–8. AFNLP.

Seretan, V. 2012. Acquisition of Syntactic Simplification Rules for French. In *Proceedings of the Eighth International Conference on Language Resources and Evaluation (LREC'12)*, pp. 4019–26, Istanbul, Turkey, May 2012. European Language Resources Association (ELRA).

Sheremetyeva, S. 2014. Automatic Text Simplification For Handling Intellectual Property (The Case of Multiple Patent Claims). In *Proceedings of the Workshop on Automatic Text Simplification: Methods and Applications in the Multilingual Society*, Dublin, Ireland, August 24th 2014, pp. 41–52, Association for Computational Linguistics.

Siddharthan, A. 2004. *Syntactic Simplification and Text Cohesion*. Ph.d. thesis, University of Cambridge.

Siddharthan, A. 2006. Syntactic simplification and text cohesion. *Research on Language and Computation*, *4 (1)*.

Siddharthan, A. 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 (ENLG '11)*, Nancy, France, pp. 2–11. Association for Computational Linguistics.

Siddharthan, A. and Angrosh, M. A. 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*, Gothenburg, Sweden, pp. 722–31. Association for Computational Linguistics.

Štajner, S., Calixto, I., and Saggion, H. 2015. Automatic Text Simplification for Spanish: Comparative Evaluation of Various Simplification Strategies. In *Proceedings of Recent Advances in Natural Language Processing (RANLP-2015)*, Hissar, Bulgaria, pp. 618–26.

Suter, J., Ebling, S., and Volk, M. 2016. Rule-based Automatic Text Simplification for German. In *Proceedings of the 13th Conference on Natural Language Processing (KONVENS 2016)*, Bochum, Germany, pp. 279–87. Bochumer Linguistische Arbeitsberichte (BLA).

Sutton, C. and McCallum, A. 2011. An introduction to conditional random fields. *Foundations and Trends in Machine Learning*, *4 (4)*.

Tomita, M. 1985. *Efficient Parsing for Natural Language: A Fast Algorithm for Practical Systems*. Norwell, MA, USA: Kluwer Academic Publishers.

Van Delden, S. and Gomez, F. 2002. Combining finite state automata and a greedy learning algorithm to determine the syntactic roles of commas. In *Proceedings of the 14th IEEE International Conference on Tools with Artificial Intelligence*, ICTAI '02, Washington, DC, USA, pp. 293–301. IEEE Computer Society.

Vickrey, D. and Koller, D. 2008. Sentence Simplification for Semantic Role Labeling In *Proceedings of the ACL-08: HLT*, ACL '08:HLT, Columbus, Ohio, USA, pp. 344–52. Association for Computational Linguistics.

Vu, T. T., Tran, G. B., and Pham, S. B. 2014. Learning to Simplify Children Stories with Limited Data. In Nguyen N. T., Attachoo B., Trawiński B., and Somboonviwat K. (eds), *Intelligent Information and Database Systems. ACIIDS 2014*, pp. 31–41. Bangkok, Thailand: Springer.

Woodsend, K. and Lapata, M. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the Conference on Empirical Methods in Natural Language Processing*, Edinburgh, Scotland, pp. 409–20. Association for Computational Linguistics.

Wubben, S., van den Bosch, A., and Krahmer, E. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics*, Jeju Island, Korea, pp. 1015–24. Association for Computational Linguistics.

Xu, W., Callison-Burch, C., and Napoles, C. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics*, *3*, pp. 283–97.

Xu, W., Napoles, C., Pavlick, E., Chen, Q., and Callison-Burch, C. 2016. Optimizing Statistical Machine Translation for Text Simplification. *Transactions of the Association for Computational Linguistics*, *4*, pp. 401–15.

Yatskar, M., Pang, B. Danescu-Niculescu-Mizil, C., and Lee, L. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from wikipedia. In *Proceedings of Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the ACL*, Los Angeles, California, pp. 365–8. Association of Computational Linguistics.

Zhang, X. and Lapata, M. 2017. Sentence Simplification with Deep Reinforcement Learning. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Copenhagen, Denmark, pp. 584–94.

Zhu, Z., Bernhard, D., and Gurevych, I. 2010. A Monolingual Tree-based Translation Model for Sentence Simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, Beijing, China, pp. 1353–61.