Pattern-Based Syntactic Simplification of Compound and Complex Sentences

ARCHANA PRAVEEN KUMAR\textsuperscript{1}, ASHALATHA NAYAK\textsuperscript{1}, MANJULA SHENOY K.\textsuperscript{2}, (Member, IEEE), ROSHAN JACOB MANOJ\textsuperscript{1}, AND AKANSHA PRIYADARSHI\textsuperscript{1}
\textsuperscript{1}Department of Computer Science and Engineering, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Karnataka 576104, India
\textsuperscript{2}Department of Information and Communication Technology, Manipal Institute of Technology, Manipal Academy of Higher Education, Manipal, Karnataka 576104, India
Corresponding author: Archana Praveen Kumar (archana.kumar@manipal.edu)

ABSTRACT With the advent of new technologies, simplifying text automatically has been very popular and of high importance among natural language researchers during the last decade. The predominant research done in the area of Automatic Sentence Simplification (ASS) is inclined to either lexical or syntactical simplification of sentences. From the literature survey, it is observed that existing research in lexical simplification makes use of word substitution technique. This causes word sense ambiguity in cases where the word synonyms are not appropriate for a sentence in the given context. In contrast, syntactical simplification though accurate and applicable to Natural Language Processing (NLP) tasks, requires tremendous efforts to construct rules for a given domain. The research proposes a framework called Pattern-based Automatic Syntactic Simplification (PASS) which identifies sentences and applies rules based on grammatical patterns to simplify the sentences thereby making it more generic for NLP tasks. PASS is evaluated by human experts to rate the usefulness of the framework based on fluency, adequacy and simplicity of the sentences. Furthermore, the framework is automatically evaluated with the available online corpus using automatic metrics of SARI, BLEU, and FKGL. The proposed approach generates promising results in the field of ASS and could be used as a preliminary module for NLP tasks as well as other natural language-related applications like summarization, anaphora resolution, question-answering, and many more.

INDEX TERMS Complex sentences, compound sentences, simple sentences, pattern-based syntactic simplification, natural language processing, summarization.

I. INTRODUCTION
Automatic Sentence Simplification (ASS) involves techniques not only to reduce the linguistic complexity of the sentence but also preserves its meaning and original information. Specifically, the simplified form of the sentence is obtained either by substituting the words in the sentence or transforming the sentence structure [1], [2]. ASS is important for both software applications, as well as for certain NLP tasks [2], [3]. Software applications using sentence simplification models assist mainly language learners, people with reading disabilities, or low literacy readers [3]–[5]. Furthermore, the performance of ASS improves the NLP tasks of anaphora resolution [6], machine translation [7], summarization [1], [8] [9] which benefits applications like question answering [10], [11], opinion mining, etc. Due to this reason, the last decade has seen many research efforts targeted in this direction.

Most of the early works done in ASS have extracted sentences that conveyed the most similar meaning to the original sentence called extractive sentence simplification. Later on, with the disposal of knowledge resources like WordNet, research in abstractive sentence simplification increased as compared to extractive sentence simplification [3]. Under the abstractive approach, there are two techniques: Lexical Simplification and Syntactic Simplification [12], [13]. Lexical Simplification is done by identifying and replacing difficult phrases or words using word substitution technique. In contrast, syntactic simplification deals with converting the complex structure of the sentence to simple syntactic
structures [14] retaining the original words. The literature review in the paper discusses research targeted towards ASS in both syntactic as well as lexical simplification.

Though there have been substantial attempts in the literature to simplify the sentences, however most of the models have limitations in generating grammatically correct and simple sentences. Some of the existing approaches, though are able to generate grammatically correct simple sentences, are not able to retain the meaning of the original sentences. Majority of the lexical simplification make use of neural network which would learn based on the rules provided or learn by itself. This approach would be time consuming due to the extensive training time and works only for certain sentences given as input by the application. Furthermore, lexical simplification systems use word substitution technique which cause word sense ambiguity for a given sentence in the given context. This occurs when a word has several meanings and the system is not able to determine which word needs to be substituted for the original word in the given context. In case of syntactic simplification the systems depend on automatically learned rules or hand-crafted rules specific for a given domain. This not only makes the system domain dependent but also requires tremendous efforts to generate or create rules.

To overcome these limitations, the proposed research is based on grammatical patterns which can be further extended to cover complex constructs in a given sentence. The proposed Pattern-based Automatic Syntactic Simplification (PASS) utilizes grammatical patterns existing in the sentences to apply rules thereby simplifying compound and complex sentences. Furthermore the identified grammatical patterns are generic to a given language and applies to any context. The system is able to generate simple sentences which have substantial improvements in terms of grammatical correctness and simplicity. The simplified sentences generated in the approach are also able to preserve and retain the meaning of the original sentences. In this direction, the motivation is to propose a domain-independent system which

- Uses grammatical patterns to identify different kinds of compound and complex sentences
- Apply rules based on grammatical patterns to transform and simplify sentences
- Evaluate PASS concerning its usefulness and compare it with existing models using existing automatic metrics

The rest of the article is subdivided as follows. The basic terminologies is presented in Section II with related work in Section III. The methodology is explained in Section IV. Section V provides the analysis and discussions of the results, section VI gives the evaluation done for PASS and section VII presents the conclusion.

II. BACKGROUND

A sentence is a set or group of words which convey meaning in a given language. English sentences have been classified as simple, compound and complex sentences. The structure of the sentence comprises the information of a topic. Here the topic represents the subject while the details or information of the topic represent the predicate or the verb and object in English grammar. S derives the sentence with either a subject followed by verb as shown in equation 1, while a subject can either comprise of Noun or Pronoun or Restrictor or Determiner as represented in equation 2. The verb can either comprise of verb or complement or object in the sentence as shown in equation 3.

\[
S \rightarrow <\text{subject}> + <\text{verb}>
\]

\[
<\text{subject}> \rightarrow <\text{Noun/Pronoun}> \text{ or }
\]

\[
<\text{verb}> \rightarrow <\text{verb}>	ext{ or }
\]

The sentences can be classified into declarative, imperative, yes-no questions and wh-question type of sentences [15]. Context Free Grammar (CFG), in linguistic terminology, represents the structure of every English sentence into its equivalent mathematical form as stated in [16]. The structure is encoded with a rule or production which expresses the ordering of the words forming a pattern. CFG can represent sentence-level grammatical constructions in English for declarative as shown in equation 4, imperative as shown in equation 5, yes-no question as shown in 6 and wh-question type of sentences as shown in 7. These equations represent sentence S that stands for the nonterminal symbol and its corresponding production rules for the derivation of each of the sentences. Here symbols: - NP refers to Noun Phrase while VP refers to Verb Phrase, Aux refers to the auxiliary verb in the imperative sentence while Wh-NP refers to noun subject in the interrogative sentence containing wh-word e.g. what, who, and so on.

\[
S \rightarrow \text{NP VP}
\]

\[
S \rightarrow \text{VP}
\]

\[
S \rightarrow \text{Aux NP VP}
\]

The grammatical structures for the different sentences have been explored in [17]. This paper gives analysis and structural representations for simple and compound sentences which paved the way to explore grammatical patterns in the sentences used in the research. The criteria for distinguishing simple from the compound and complex sentences have been explored in [18]. This approach provides a way to distinguish the simple from the compound and complex sentences with the number of verbs in the sentence. Sentence classification has paved an important improvement for sentiment classification in [19]. The sentence classification for this kind of task was done by the divide and conquer approach followed by the
sentiment classification using a one-dimensional convolution neural network. Different types of sentences, their structures, and the different conjunctions have been discussed. The conjunctions - coordinating conjunctions, correlative conjunctions, and subordinating conjunctions in the sentences along with their order to illustrate the various types of declarative sentences in English are also discussed in this paper. This paper discussed the different coordinating conjunctions used in compound sentences and the correlative and subordinating conjunctions used in complex sentences. The research is targeted towards simplification of declarative sentences.

The declarative sentences can be categorized as simple, complex, compound and compound-complex sentences. The categories are based on the presence of clauses. A sentence can comprise of one or multiple clauses. If the clause can express a complete thought i.e. it has subject and predicate it is called an independent clause (IC) [19]. In case the clause cannot express a complete thought or is incomplete i.e. it has only the subject and verb it is called a dependent clause (DC). The different categories of sentences are explained in the following subsection.

A. SIMPLE SENTENCES
Sentences which comprise of no DC or in other words contain only one IC with a subject and predicate are termed as simple sentences [20]. An example of the same - ‘Cats are fond of balls’ can be graphically represented with the POS tagging shown in Figure 1 using SPACY [21].

B. COMPOUND SENTENCES
Compound sentences comprise of at least two ICs connected by a coordinating conjunction (CConj) [20]. The different coordinating conjunction are - ‘For’, ‘And’, ‘Nor’, ‘But’, ‘Or’, ‘Yet’, ‘So’ which represent ‘FANBOYS’ stating for the connectors between two or more IC [19]. POS Tagging for a compound sentence ‘He went to the party but she stayed home’ is represented in Figure 2. Compound sentences if not connected by the conjunction can also be separated by a semicolon [22]. On observation, the compound sentences can be categorized into three types:

1) Type I: Each of the IC comprises of one subject (same or different) each performing different actions connected by CConj

- ‘Cats like to drink milk and dogs like to chew bone’
- In the e.g. ‘and’ connects the IC ‘Cats like to drink milk’ with another IC ‘dogs like to chew bone’

2) Type II: The subject of an IC performs two actions connected by a conjunction
- ‘Cats like to drink and play with the ball’
- In the e.g. the subject ‘Cats’ performs two actions of ‘drink’ and ‘play’ connected by ‘and’

3) Type III: When there exists an IC connected using a CConj with another object that further extends the first IC
- ‘Cats like to play with dogs and other cats’
- In the e.g. IC ‘Cats like to play with dogs’ is connected with ‘and’ to object ‘dogs’

C. COMPLEX SENTENCES
Complex sentences are sentences that have one IC and at least one DC [20]. Both the IC and DC are joined in any order with subordinating conjunction(SConj) like ‘even though’, ‘because’, ‘after’, ‘while’, ‘since’ and so on [19]. POS tagging of a complex sentence ‘Whenever it rains I like to wear my blue coat’ is represented in Figure 3.

If the DC comes before the IC they are usually separated by a comma [23]. Based on the position of subordinating conjunction in the sentence and the number of the DC complex sentences can be categorized into the following three types:

1) Type IV: Sentences having have SConj followed by a verb e.g. ‘that’ followed by a verb
- ‘He bought the shoes that were in the shop window’

2) Type V: Sentences having SConj in the beginning or middle of the sentence e.g. ‘Because’ in the beginning or ‘even though’ in the middle of the sentence
- ‘Because my tea was too cold, I heated it in the oven’
- ‘The servants were very obedient to him even though he was very harsh to them’

3) Type VI: Sentences having more than one DCs e.g. DC: ‘drawing on areas of expertise such as applied mathematics’ and ‘thus emerging as the latest technology’
- ‘Artificial intelligence research has been necessarily cross-disciplinary drawing on areas of expertise such as applied mathematics, thus emerging as the latest technology’
The coordinating conjunctions connected to the compound in [27]. Once the parse tree of the sentence is obtained, fying both compound and complex sentences was contributed to simplify the sentences. The approach was targeted towards splitting, reordering, dropping, and substituting operations to handle complex sentences. This model generated the parse tree by basing on statistical machine translation to derive a parse tree for the original sentences till the resultant set comprises of simplified sentences. The model thus applies the rules and transforms the original sentence into small simplified sentences. The approach defines two rules one for the selection of clause which removed the noise before and after the clause. While the second rule selects the entity phrase which simplified the entity without modifying the relations’ truth value. The limitation of this approach was some rules changed the modality of the sentence during simplification.

Another contribution towards syntactic simplification in [31] was done by applying heuristics to the syntactic parse trees. The parse trees thereby created alternatives of the input sentence and its possible several simplified sentences. The limitation being that the parsers output heuristics templates were required which in turn produced simplified sentences having grammatical errors. Syntactic simplification for French texts in [4] was explored using two stages. In the first step all possible simplifications for a given sentence were generated. However, in the second step, the best-simplified sentence was selected which satisfied a certain criteria. The limitation was the pre-requisite of manually hand-crafted rules extracted from a French corpus. An approach by name Simplified Statement Extractor (SSE) in [32] decomposed the original sentence into small simplified sentences. The experiment yielded simple sentences but with lower grammatical accuracy.

Complex sentences from Punjabi texts were identified in [33] using CRF. Here all the accessible features that included interactions between sentences were noted by the framework but the work was limited to Punjabi language. Simplification of compound-complex sentences in [24] was done by using the Stanford dependency parser which generated Modified Stanford Dependency (MSD). This structure is derived from Basic Stanford Dependency (BSD) and Collapsed Stanford Dependency (CSD) structure of the parser. The limitation of the system was it generated incomplete sentences for some test cases of compound-complex sentences. Simplification of complex sentences for building a Spanish corpus has been done in [34]. Depending on number of conjugate verbs, the methodology targets splitting long and complex sentences into simple sentences. The limitation being the splitting of complex sentences is only done for those sentences which were either coordinate or subordinate sentences.
B. LEXICAL SIMPLIFICATION

Lexical simplification of the sentences was tackled in [35] with a dataset comprising of Resource Description Framework (RDF) triples. The RDF using the Sequence-to-Sequence model given the name Split and Rephrase simplified complex sentences. The technique uses probabilistic model to predict the splitting point for a given input sentence. In some cases the wrong splitting point predicted by the model hinders to generate fluent and meaning preserving rephrasings for the given sentence. Another approach in lexical ASS was done in [36] where for a given input sentence a simplified sentence was generated by utilizing a corpus. The technique used here is a variant of phrase-based machine translation system to incorporate phrasal deletion to simplify the text. The approach though has obtained positive results in simplifying the text produced grammatical incorrect sentences in some cases. A simplification model for children was proposed in [37] wherein the difficult words were replaced by easier synonyms. However, the simplification introduced certain errors while simplifying the input sentences. A simplification model using unsupervised machine learning has been implemented in [38]. The limitation of the system was that it could not simplify sentences that had parsing errors. Rule-based sentence simplification model was devised in [39] which incorporated contextual information to replace the original words of the input sentence. However, the simplification of the system effectively worked only for frequently occurring words in the domain.

The sentences were simplified without rules in [38] by splitting them based on their semantic structure using an unsupervised approach. The approach, though was less sensitive to parsing errors, required a large corpus of simplified and standard language with no alignment between them. Lexical simplification of text was also explored in [40] by using sequence to sequence neural network with two Long Short Term Memory (LSTM) networks. Due to the development set, the model would not only predict the perplexity values but also selected the model parameters with the best perplexity. Thus the model could perform both lexical simplification and content reduction.

Lexical simplification was experimented using reinforcement learning technique in [41]. The model called Deep REinforcement for Sentence Simplification (DRESS) explored the possible multiple simplifications of the sentence. The model learned to reward the function which outputs grammatical, fluent, and simple sentences. However, it required optimal method to reward the sentences leading to a significant delay for the training compared to the other neural network models. A basic neural network model in [42] utilized a configuration having both multilayer and multi-head attention architecture for simplification of sentences. The model was trained through paraphrase detection that covered all the simplification rules. However, although the models helped to leverage better rules the system required an additional memory component to maintain the simplification rules. Neural Semantic encoders contributed in [43] utilized greedy and beam search strategy to get the target sentence for simplifying complex sentences. The results of the approach had a positive correlation between adequacy and fluency but a negative correlation between adequacy and simplification. Sentence simplification using an automatic semantic parser has been explored in [44] based on an effective and simple algorithm. Here after the sentence were split the text was fine-tuned to tackle both structural and lexical simplification.

An approach that explored both syntactic and lexical simplification utilized unlabeled text corpora collected from en-Wikipedia dump. The architecture comprised of shared encoder along with a pair of attentional decoders in [45]. The framework simplified complex sentences by training the model with complex sentences along with their corresponding simple sentences. A sequence to sequence model explored in [46] shared all the encoder and decoder parameters for multi-task learning. Here the sentence simplification task, entailment generation, and paraphrase generation task was trained in parallel to get the desired output. An approach for lexical simplification of complex sentences was experimented in [47] that incorporated content word complexities and loss function into their word complexity model during training. Then, their model generated a huge set of candidate diverse simplifications during testing phrases to promote adequacy, simplification, and fluency of the sentences. The model did not perform well for 1. Long and complex sentences having multiple clauses, 2. Sentences that needed anaphora resolution to preserve the meaning, and 3. For sentences where simplification was done for the wrong part of the sentence.

One more approach in [5] adapted a discrete mechanism on sequence to sequence model by providing explicit control on the attributes of the sentence. The attributes such as paraphrasing detail, length of the sentence, were used for simplification. The attributes were chosen carefully by the audience based on the requirement of the simplification. The model was hence termed AudienCe-Centric Sentence Simplification (ACCESS). In the previous models, the operations were not learned by the neural network, in contrast to the approach done in [48]. Wherein the model explicitly was trained to learn the different operations of adding, deleting, and keeping words in the sentence simplification process. By this approach, the intended edit operations on the target sentence were trained thereby resembled the human thinking process for a given sentence while simplifying it. An iterative edit-based unsupervised approach for simplifying the complex sentence was proposed in [49]. Here the approach made use of iterations for editing the word and phrase-level for a given sentence. The method generated multiple sentences iteratively by performing edit operations thereby changing the syntactic and lexical structure of the sentence. However, the limitation being the approach could not paraphrase the sentences properly due to which the meaning of the original sentences was not retained.
IV. METHODOLOGY

PASS comprises of three stages: Segmentation and tagging, Chunk identification and the third stage of analysis and application. Figure 5 shows PASS which simplifies sentences using grammatical patterns. PASS transforms or aligns an input sentence to two or three simplified sentences, so the sentence transformation is categorized as one-to-N as discussed in [3]. In the first stage, each of the input sentences is segmented and for each segmented sentence, POS tagging is done. POS-tagging tags each word of the sentence with its parts-of-speech. Once the tagging is done, chunks of words in the sentence are identified through CFG.

Each identified chunk comprises of VP or NP with the modifiers attached. The NP chunk will also identify the singular/plural number of the phrase, while accordingly, the VP chunk identifies the tense and other aspects of the sentence. After chunk identification, the last stage of analysis and application is done. Here the identified chunks are analyzed and the rules are applied to transform the input sentences into simplified sentences. Analysis and application require grammatical patterns of CFG to be analyzed and then to apply the rules based on the grammatical patterns on the identified chunks.

Once the chunks are identified the rules are chosen depending on the type of the sentence encountered. A rule which can be seen commonly in all the compound sentences is shown in Equation 8 that can be interpreted as follows. If P in the Equation 8 is a complete sentence S that constitutes of NP and VP shown in Equation 4. S is followed by CConj, followed by Z, where Z is an arbitrary sequence of words, the sentence then is simplified into two sentences i.e. X and T. Here X will be the existing sentence formed by separating the NP i.e S_1. While T forms another complete sentence S_2 by identifying and appending the missing subject or predicate or object from X. If Z is a complete sentence by itself after removing the CConj then T = Z.

\[
P \rightarrow S \text{ CConj } Z \quad \text{simplified to}
\]

\[
1. \ X \rightarrow S_1 \\
2. \ T \rightarrow S_2 
\] (8)

The example P → ‘Cats like to play with dogs and other cats’, shown as Type III in will be simplified based on the rule given in equation 8 as follows.

- S → ‘Cats like to play with dogs’
- CConj → ‘and’
- Z → ‘other cats’
- Z has missing subject and predicate which is referred and appended from X i.e. subject → ‘Cats like’
- Missing predicate → ‘to play’
- Simplified sentences are:
  - X → ‘Cats like to play with dogs’
  - T → ‘Cats like to play with other cats’
- Similarly, the other sentence as shown as Type I will be simplified as
  - X → ‘Cats like to drink milk’ and
  - T → ‘Dogs like to chew bone’.
- Sentence shown as Type II will be simplified as
  - X → ‘Cats like to drink’ and
  - T → ‘Cats like to play with the ball’.

Compared to compound sentences, a common rule or grammatical pattern is not observed in complex sentences due to their structure. Rules based on grammatical patterns which can be applied to the sentence of Type IV is shown in Equation 9. Equation 9 is understood as follows. If a complex sentence X begins with an IC followed by that’ followed by the DC, the sentence is divided into two sentences X and T. X is the complete sentence S_1, while T will be the second complete original sentence Z appended with the second NP referred in X. The second NP in X happens to be the object of the sentence. Rule applied to sentence of Type V is given in Equations 10 and 11. In this case, complex sentences have SConj in the middle of the sentence as shown in rule 10. Or at the beginning of the sentence as shown in rule 11, then the original sentence is divided into two simple sentences X having the IC and T having the DC. For complex sentences comprising of more than one DC as shown in Type VI, the split of the original sentence is done as shown in equation 12. The rule states that if X IC (with NP along with the verb VP_1) is followed by the SConj followed by DC. The sentence is divided into two sentences with the initial sentence X having the complete IC. While T will be the second complete sentence with NP of X appended with changed tense of the verb VP_1 represented as VPC and DC of Z. For this sentence, a dictionary is build to generate the past tense of the verb to
be used for the recurring verbs in the complex sentences.

\[ \text{X} \rightarrow \text{IC that DC simplified to} \]
1. \(X \rightarrow S_1\)
2. \(T \rightarrow NP_2 VP_2 DC\) \hspace{1cm} (9)
\[ \text{X} \rightarrow \text{IC} S\text{Conj DC} \text{ simplified to} \]
1. \(X \rightarrow \text{IC}\)
2. \(T \rightarrow \text{DC}\) \hspace{1cm} (10)
\[ \text{X} \rightarrow S\text{Conj IC} \text{ DC} \text{ simplified to} \]
1. \(X \rightarrow \text{IC}\)
2. \(T \rightarrow \text{DC}\) \hspace{1cm} (11)
\[ \text{X} \rightarrow \text{IC} S\text{Conj DC} \text{ simplified to} \]
1. \(X \rightarrow \text{IC}\)
2. \(T \rightarrow \text{NP VP} \text{C}_1 \text{ DC}\) \hspace{1cm} (12)

Using equation 9, the example \(X \rightarrow \) ‘He bought the shoes that were in the shop window.’, shown in Type IV will be simplified as follows.

- \(\text{IC} \rightarrow \) ‘He bought the shoes’
- \(\text{DC} \rightarrow \) ‘were in the shop window’
- \(\text{DC}\) has missing subject (object in \(X\)) which is referred from \(X\) i.e. subject \(\rightarrow \) ‘The shoes’
- Simplified sentences are:
  - \(X \rightarrow \) ‘He bought the shoes’
  - \(T \rightarrow \) ‘The shoes were in the shop window’
- Similarly, the other sentence \(X \rightarrow \) ‘Because my tea was too cold, I heated it in the oven’ as shown in Type V will be simplified as
  - \(X \rightarrow \) ‘My tea was too cold’ and
  - \(T \rightarrow \) ‘I heated it in the oven’.
- Another sentence of Type V ‘The servants were very obedient to him even though he was very harsh to them’ will be simplified as
  - \(X \rightarrow \) ‘The servants were very obedient to him’
  - \(T \rightarrow \) ‘He was very harsh to them’
- Sentence shown in Type VI will be simplified as
  - \(X \rightarrow \) ‘Artificial intelligence research drew on areas of expertise such as applied mathematics’ and
  - \(T \rightarrow \) ‘Artificial intelligence research emerged as the latest technology.

**A. ALGORITHM**

The algorithm PatternAnalysis() describes the steps of analyzing the grammatical patterns in the input sentences and applying the rules to simplify the sentences. Since there are different rules which are applied for different compound and complex sentences based on grammatical patterns, this algorithm has separate functions for each of the sentences. Depending on the requirement of simplification, each function is called separately or sequentially. So consequently the entire dataset is simplified by executing the Compound function or Complex function alone. Or sequentially both based on the order as Compound-Complex or Complex-Compound.

**Algorithm 1 PatternAnalysis()**

**Input:** Declarative Sentences: Text ‘I’ of \(n\) sentences, where \(T = S_1, S_2, \ldots, S_n\)

**Output:** Simple Sentences: Set of Simple sentences \(X\) and \(T\)

for each each sentence \(S_j\) in \(I\) where \(0 < j < n\) do
Call Procedure ComplexS
End for \hspace{1cm} \(\triangleright \) Complex

for each each sentence \(S_j\) in \(I\) where \(0 < j < n\) do
Call Procedure CompoundS\(_j\)
End for \hspace{1cm} \(\triangleright \) Compound

for each each sentence \(S_j\) in \(I\) where \(0 < j < n\) do
\[O = \text{Call Procedure CompoundS} \_j\] \hspace{1cm} \(\triangleright \) Each split sentence from complex algorithm is placed in \(O\)
End for

for each each sentence \(R_i\) in \(O\) where \(0 < i < m\) do
Call Procedure Complex R\(_i\)
End for \hspace{1cm} \(\triangleright \) Compound-Complex

for each each sentence \(S_j\) in \(I\) where \(0 < j < n\) do
\[P = \text{Call Procedure CompoundS} \_j\] \hspace{1cm} \(\triangleright \) Each split sentence from complex algorithm is placed in \(P\)
End for

for each each sentence \(Q_k\) in \(P\) where \(0 < k < m\) do
Call Procedure Compound Q\(_k\)
End for \hspace{1cm} \(\triangleright \) Complex-Compound

**FIGURE 6.** Data flow diagram.

The analysis of each kind of sentence is shown in the Table 1. The table shows the steps applied as per the algorithm with the output for each of the input sentence. The data flow diagram of the entire procedure in Figure 6. Here \(I\) denotes the input sentences given to both Compound and Complex algorithms. While \(O1\) and \(O2\) represents the output of simplified sentences from Compound and Complex algorithm. The individual flowcharts of the Compound and Complex algorithm are shown in Figure 7 and Figure 8.

**B. DATASET DESCRIPTION**

The experimental dataset by name RandomCoCo (Random Compound and Complex) has been designed which comprises of compound and complex sentences taken from publicly available online sites. The dataset is available online on IEEEDataPort [50] consisting of 979 sentences.
A. Praveen Kumar et al.: Pattern-Based Syntactic Simplification of Compound and Complex Sentences

| Procedure | Compound($S_j$) |
|-----------|-----------------|
| if $S_j$ contains CConj then |
| Remove CConj |
| if CConj is ‘nor’ then |
| Change the second sentence structure |
| End If |
| if X and T simple then |
| Exit |
| else if Get missing components of IC then |
| Append missing components to the clause |
| Split to generate X and T |
| End if |
| End Procedure |

| Procedure | Complex($S_j$) |
|-----------|----------------|
| if $S_j$ contains SConj==‘that’ then |
| Split $S_j$ before ‘that’ to form X |
| NP before ‘that’ in $S_j$ is appended to Z as subject of Z |
| Z forms the simple sentence T |
| else if $S_j$ contains SConj in middle then |
| Split $S_j$ before Sconj to form X |
| T is $S_j$ after Sconj |
| else if $S_j$ contains SConj in beginning then |
| Remove SConj |
| Split $S_j$ before the second noun or delimiter whichever comes first to form X |
| T is sentence after the second noun or delimiter whichever comes first |
| else if $S_j$ contains two DC then |
| Split $S_j$ the first complete sentence to form X |
| Append main subject of IC to the first DC with changed tense of verb of first DC to form second sentence |
| Change tense of VP to past tense and append main subject to second DC to form third sentence |
| End if |
| End Procedure |

These 979 sentences consist of 478 compound sentences and 501 complex sentences. The algorithm works with compound-complex sentences but currently the dataset has only compound and complex sentences based on the objective of the research. Among 478 declarative compound sentences which had 435 sentences with conjunctions while the rest 47 were those having a semicolon. Among the 431 sentences 239 sentences comprise of more than two conjunctions. Among the remaining 192 sentences, there are about 90 using ’and’, 20 using ‘for’, 8 using ‘nor’, 50 using ‘but’, 8 using ‘or’, 9 using ‘yet’, and 7 sentences using ‘so’ as conjunctions. Figure 9 shows the statistics of compound sentences in RandomCoCo. Considering Complex sentences, the RandomCoCo comprises 501 declarative sentences. Among this 23 sentences belong to Type IV, 20 belong to Type V and 458 belong to Type VI respectively. Figure 10 shows the analysis of complex sentences in RandomCoco.

The different CFG patterns used for the identified types of sentences is shown in the Table 2. The proposed set of patterns covers the general pattern of the declarative sentences in English language comprising of compound and complex sentences.

V. DISCUSSION OF RESULTS AND ANALYSIS

The approach is implemented using Python libraries of Natural Language Toolkit (NLTK) and SPACY. The compound algorithm was run on the entire RandomCoCo and generated 1058 simple sentences from 478 compound sentences. Table 3 summarizes the sample outputs for different sentences. Some test cases of compound sentences simplified by PASS are shown as follows.

1) As the method deals with the basic form of the sentence i.e. having subject, verb, and object, any other meaning about the sentence is preserved after splitting.
E.g. We have never been to Asia, nor have we visited Africa.

With the use of the algorithm the sentence gets split into X: We have never been to Asia and T: We have never visited Africa.

Hence the meaning is preserved in the sentences after splitting.

2) The steps provided for compound sentences give a robust solution for sentences of the type ‘He likes to run and swim’.

- Instead of splitting the first IC into subject, verb, and object, the first clause is used entirely, then the verb and words after the verb are combined to form the second sentence.
- This preserves the infinitives and the other adverbs and adjectives that might precede the verb.
- So the sentences split into X: He likes to run and T: He likes to swim.
- In this case, the verbs ‘run’ and ‘swim’ with infinitives ‘likes to’ are appended with the subject to form the two simple sentences.
A. Praveen Kumar et al.: Pattern-Based Syntactic Simplification of Compound and Complex Sentences

TABLE 3. Summary of sample outputs of compound sentences in RandomCoCo.

| Sample of Compound Sentence | Type | Pattern Applied |
|-----------------------------|------|-----------------|
| Cathy and John visited us on Thanksgiving | III | The subject of X is appended with the predicate and object of Z |
| Mary never wrote the letter, nor did she call him. | II | Both X and Z are ICs with change done with verb connected to the subject of Z |
| I like tea and he likes coffee | I | Both X and Z are simple after removal of conjunction |

TABLE 4. Summary of sample outputs of complex sentences in RandomCoCo.

| Sample of Complex Sentence | Type | Pattern Applied |
|---------------------------|------|-----------------|
| However, the duties of governor for Kashmir were generally discharged by deputy and after the virtual separation of Kashmir and Jammu, the title alone was retained by the voter | V | Both the subordinating conjunction in middle and beginning of the sentence are removed to simplify the sentence |
| The duties of governor for Kashmir were generally discharged by deputy. The title alone was retained by the voter. | V | For simplifying the sentences, the conjunction is removed |
| The museum was very interesting, as I expected. | V | For simplifying the sentences, the conjunction is removed |
| The museum was very interesting. I expected. | IV | The object missing in Z is appended from X to simplify the sentences |
| India won the tournament defeating Pakistan by 5 wickets in the final at Eden Stadium in Kolkata, thus becoming the first country to win Cricket World Cup final on home soil | VI | The sentence consists of more than one DC, a dictionary is built with the modified tense of the verb in the IC to split the sentences |
| India won the tournament. India defeated Pakistan by 5 wickets in the final at Eden Stadium in Kolkata. India became the first country to win Cricket World Cup final on home soil. | VI | The sentence consists of more than one DC, a dictionary is built with the modified tense of the verb in the IC to split the sentences |

With 501 complex sentences, 1048 simple sentences have been generated from the complex algorithm. Table 4 summarizes the sample outputs for different sentences.

Some test cases of complex sentences used and simplified using the algorithm are shown below.

1) For sentences having ‘that’ the method not only simplifies the sentences but also retains the meaning by appending the object of the first sentence with the required pronoun to the second sentence
   - The complex sentences of the type: ‘My mom asked me to find her sweater that was torn and required stitching.’
   - This kind of sentence is of Type IV for which the steps give the simple sentences as X: My mom asked me to find her sweater and T: Her sweater was torn and required stitching.

2) For the complex sentences having subordinate conjunction in middle the method simplifies the sentences by splitting the first IC and then removing the conjunction to give the second sentence
   - The complex sentence: ‘Let’s go back to Shimla because there we had our first meeting.’
   - The sentence is of Type V wherein the sentence is split into X: Let’s go back to Shimla Nous. and T: There we had our first meeting.

3) For complex sentences where there are more than one DC, the method employs to build a dictionary thereby splitting the sentence by appending the main subject to the next sentence
   - The complex sentence of the type: ‘Teresa got Indian citizenship, spent many months in Pune to receive primary medical training at Holy Hospital, embarking into the slums.’
   - The sentence is of Type VI wherein the sentence is split into X: Teresa got Indian citizenship and T: Teresa spent many months in Pune to receive primary medical training at Holy Hospital and S3: Teresa embarking into the slums.

VI. EVALUATION

Evaluation of PASS has been done by both human experts as well as with the automatic metrics. The manual evaluation has been done for checking the correctness and usefulness of the algorithm. The output of the dataset given to evaluators for evaluation is available online [50]. Automatic evaluation has been done with the automatic metrics along with the publicly available corpus to compare PASS with other existing systems. This section focuses on the basis on which the domain experts have evaluated the system and the reason for choosing the appropriate corpus and automatic metrics.

A. HUMAN EVALUATION

The best reliable method to evaluate a simplification system is to rate the system’s output by human experts as stated in [3]. Therefore PASS has been evaluated by considering five linguistic experts’ judgments to validate the correctness of the system. The research is inclined towards the syntactic simplification of the complex and compound sentences, so the human evaluators have been given the simplified sentences. These simplified sentences are the results after running the Compound and Complex function separately. The evaluators are presented with the input sentences along with simplified...
TABLE 5. Criteria of 5-point Likert scale evaluation.

| Rating | Criteria Given |
|--------|---------------|
| 5      | Exceptional   |
| 4      | Good          |
| 3      | Average       |
| 2      | Below Average |
| 1      | Very Poor     |

sentences and are asked to rate them based on criteria given as follows.

- Fluency – To check whether the decomposed sentence is grammatical and fluent
- Simplicity – To check whether the decomposed sentences simplify the input sentences in terms of structural transformation ignoring lexical simplifications
- Adequacy – To check whether the decomposed sentences preserve the meaning

Evaluator 1 is asked to rate the output for fluency, evaluator 2 rates concerning simplicity. Table 6 tabulates the individual metrics evaluated by each of the evaluators for the algorithm. The evaluation has been done using the 5-point Likert Scale which is the prominent and common technique used for evaluation of simplification systems [3]. The highest score in the Likert scale indicates the correctness of the sentence concerning the metric, whereas the lowest score means vice versa. Table 5 shows the criteria of the 5-point Likert scale given for human evaluation. The sentences having a scale of 3, 4, and 5 have been chosen to be satisfying the given metric.

Once the simplification is done, the system’s output should be useful for other NLP tasks. One such task which the approach looked into as an example is Anaphora Resolution to support the usefulness criteria. Anaphora resolution is a task where the pronouns are resolved to their antecedents. In cases where a sentence after simplification is incorrect, there are high chances of anaphora resolution being erroneous in the simplified sentences [51]. For this the system has been evaluated for two aspects: 1. Usefulness 2. Effect of Anaphora resolution after simplification. The judgments of all evaluators based on fluency and simplicity are considered for evaluating the usefulness of PASS. In terms of evaluating the effect of anaphora resolution, it is very important that the meaning of the original sentence i.e. adequacy along with the correct fluency and simplicity of the output sentences is preserved. This is because in the course of simplification process there are chances of some nuances of meaning that might be lost. Hence to evaluate the algorithm concerning the anaphora resolution the judgments of all evaluators with respect to simplicity, fluency and adequacy are considered. Table 7 shows how the system is evaluated based on the criteria of usefulness and effect on anaphora resolution.

Among 1058 simplified sentences generated from 486 compound sentences in RandomCoCo, evaluator 1 found 944 sentences having the correct fluency. Table 8 gives the statistical details of the outputs for compound sentences for all the evaluators. Considering the fluency and simplicity judgments by the evaluators, the average number of sentences that satisfy fluency is 943 while 967 are in their simplest form. With this, the usefulness of the system gives an accuracy of 90.26% by considering the judgments of four evaluators as reflected in Table 9. Similarly, based on Table 7 the effect of simplified sentences on anaphora resolution is computed thereby giving the accuracy of 90.29%. Table 10 provides the statistical details for compound sentences with effect to anaphora resolution.

Among the 1048 output sentences for 501 complex sentences in RandomCoCo, evaluator 1 found 928 sentences having the correct fluency. Table 8 gives the statistical details of the outputs for complex sentences for all the evaluators. Considering the fluency and simplicity judgments by the evaluators, the average number of sentences that satisfy fluency is 932 while 946 are in their simplest form. With this, the usefulness of the system gives an accuracy of 90.74% by considering the judgments of four evaluators as reflected in Table 9. Similarly, based on Table 7 the effect of simplified sentences on anaphora resolution is computed thereby giving the accuracy of 90.96%. Table 10 provides the statistical details for complex sentences with effect to anaphora resolution.
TABLE 12. Statistical summary for 501 complex sentences in RandomCoCo with respect to usefulness.

| Metric      | Average of Evaluators Judgments | Usefulness of the system |
|-------------|----------------------------------|--------------------------|
| Fluency     | 930                              | 91.65%                   |
| Simplicity  | 991                              |                          |

TABLE 13. Statistical summary for 501 complex sentences in RandomCoCo with respect to Anaphora resolution.

| Metric      | Average of Evaluators Judgments | Usefulness of the system |
|-------------|----------------------------------|--------------------------|
| Fluency     | 943                              | 90.61%                   |
| Simplicity  | 987                              |                          |
| Adequacy    | 939                              |                          |

TABLE 14. Weighted Cohen Kappa coefficients.

| Type of sentences | Fluency | Simplicity | Adequacy |
|-------------------|---------|------------|----------|
| Compound          | 0.94    | 0.98       | 0.99     |
| Complex           | 0.98    | 0.95       | 0.98     |

satisfying fluency, 989 were in their simplest form as judged by evaluator 2. Table 11 gives the detailed analysis for each evaluator. The algorithm gives an accuracy of 91.65% for its usefulness considering the parameters of fluency, simplicity from the four evaluators. Table 12 provides the statistical details for complex sentences concerning the usefulness of the algorithm. To judge the capability of the algorithm and to check the effect of anaphora resolution over simplified sentences, the accuracy of the system considering all evaluator’s judgments comes to 90.61%. Table 13 provides the statistical details for complex sentences to check the effect of anaphora resolution over simplified sentences.

To check the reliability between the evaluators, interrater reliability is tested for the system. Inter-rater agreement scores are the weighted Cohen’s Kappa coefficients [52] done between two evaluators for each metric and type of sentence. This score was done between two evaluators and was above 0.9 indicating perfect reliable agreements between the evaluators. Table 14 provides the weighted Cohen Kappa coefficients for the experiment.

The evaluation of the system’s output for both sentences concerning its usefulness and effect on the anaphora resolution has given promising results which suggest that PASS can be used for ASS. Moreover, the existing neural network models used for lexical simplification predict the probable word for word substitution but do not guarantee to simplify sentences that retain the same meaning compared to PASS. E.g. if the word in the original sentences is ‘Excellent’ which is a superlative word for ‘good’, there are high chances the neural network models might not be able to understand this and can replace the word with ‘good’. This causes the sentences to be simplified but with a meaning which might not correlate with the meaning in the original sentence. However in PASS since the system utilizes the same words as in the original sentences. This assures that the meaning of the original sentences is retained in the simplified sentences generated by the approach.

B. AUTOMATIC EVALUATION

Though human evaluation is the most reliable method to rate the system’s output, it is costly in practice and requires expert annotations as mentioned in [3]. Therefore to obtain quick and standard means to evaluate the system, automatic evaluation metrics are used along with the corpora for simplification. Among the most popular and publicly available corpora such as PWKP [26], TurkCorpus [53], and Newsela [54], the proposed approach has been evaluated with TurkCorpus. The motivation to use this dataset is as follows: The TurkCorpus dataset calculates the simplicity metric by extracting sentences from the PWKP dataset having one to one alignments and eight multiple simplified sentences for each original sentence collected from Amazon Mechanical Turk. Since this corpus provides one to one alignments, chances of misalignments are low and hence there is no creation of noisy data. So it is useful for evaluation of PASS which also aligns one input sentence to one or N sentences. Moreover, since the corpus has eight simplified sentences for each input sentence, PASS can be easily tested with this corpus. Furthermore, this dataset comprises 2359 sentences split into 2000 training and 359 sentences for testing giving a huge and diverse dataset to evaluate the system better. One another major reason for using TurkCorpus is that it allows PASS to be compared with the other existing lexical models.

The existing approaches have been evaluated to different automatic metrics such as BLEU [55], SARI [53], and FKGL [56]. Higher values of BLEU and SARI better the systems, whereas for FKGL lower the value, better the systems. BLEU computes the modified precision for n-gram by computing: (a) The maximum occurrences of n-gram word in the transformed sentence, (b) The clipped absolute count of every n-gram word by its greatest reference count and (c) Total of the clipped counts distributed by the entire unclipped counts of candidate words. With this computation, the BLEU metric is a machine translation-inspired metric that depends on and matches each n-gram in the transformed sentence independent of the position. BLEU metric is calculated with Brevity Penalty (BP) as shown in equation 13, wherein p denotes the candidate translation length and q denotes the length of reference corpus.

\[
BP = \begin{cases} 
1, & \text{if } p > q, \\
\left(\frac{p}{q}\right)^{1-p/q}, & \text{otherwise.} 
\end{cases} 
\]

\[
BLEU = BP \cdot \exp\left(\sum_{n=1}^{T} x_n \log u_n\right) 
\]

BLEU metric as shown in equation 14 computes the geometric average of the modified n-gram precisions \(u_n\) up to length \(T\) with positive weights \(x_n\). As per the equation, the BLEU metric would increase the precision for shorter
simplification algorithm as a metric for the overall comparison of ASS systems. Flesch-Kincaid Grade Level (FKGL). FKGL was used in ASS systems. It is calculated as shown in [3]. One another popular metric evaluated for ASS systems is BLEU. BLEU has a prominent association with human judgments of fluency and adequacy along with readability, PASS is also evaluated based on three metrics: SARI (High value preferred), BLEU (High value preferred), and FKGL (Lower value preferred). This package provides a single access point to evaluate the system outputs. Using the EASSE package with TurkCorpus, the automatic metrics of SARI, BLEU and FKGL are evaluated for PASS and compared against the existing approaches. The evaluation of PASS has been done for both the training and test set of TurkCorpus. Moreover, the automatic evaluation of the simplification results done based on the sentences obtained after running all the four functions - Compound, Complex, Compound-Complex and Complex-Compound.

In the training set it is observed that there are more compound-complex and compound sentences as compared to only complex sentences. Based on this analysis, we expect that SARI values should be lower for the former set of sentences compared to complex sentences. This is because SARI provides low scores for sentences involving structural changes specifically sentence splitting which are more predominant in the former set of sentences. Moreover, PASS transforms compound sentences using more rules compared to complex sentences. Hence it is expected to have lower BLEU scores for the former set of sentences in comparison to complex sentences. The reason for this is more the transformation, the lower is the BLEU score for the sentences. PASS splits the sentences for both compound sentences and complex sentences by removing words thereby reducing the length of the sentence. Therefore it is predicted to have lower FKGL scores in complex-compound as well as compound-complex algorithm compared to complex and compound algorithm alone. The system is evaluated for each of compound, complex, compound-complex, and complex-compound algorithm against each of the metrics as shown in Table 15 concerning the 2000 training dataset of TurkCorpus. As expected, the prediction of having lower SARI and BLEU scores for compound, compound-complex and complex-compound algorithm compared to complex sentences has been justified in Table 15. The prediction of obtaining lower FKGL scores is also proved in Table 15. The system’s prediction of lower BLEU scores for compound sentences is also justified by segregating only compound sentences and evaluating each rule which is shown in Table 16.

In addition to this, evaluation of PASS concerning the TurkCorpus test dataset has also been done. On observing the test dataset, it is seen, there are more compound-complex sentences compared to compound and complex sentences alone. So it is expected to get high SARI values for the former set of sentences. BLEU has a prominent association with human judgments of fluency and adequacy but with simplicity the association is less prominent when sentence is split [57]. Hence using this metric alone is not sufficient to evaluate ASS systems. Another widely used metric used in ASS systems is System output Against References and Input sentence - SARI. This metric assesses how suitable the simplification system is in adding, deleting or keeping the words during transformation compared to the reference sentences. SARI thereby correlates with human judgments of simplicity as mentioned in [53]. Therefore SARI is the most popular metric used for ASS systems. Considering the systems output S, input sentence SI, Reference sentence RS and #f(.) as a binary value of n-grams occurrence f in a given set. The n-gram precision p(n) and recall r(n) for the add, keep and delete operations is given in 15, 16, 17, 18, and 19.

\[
\begin{align*}
p_{add}(n) &= \frac{\sum_{g \in S} \min(#f(S \cap SI), #f(RS))}{\sum_{g \in S} #f(S \cap SI)}, \\
#f(S \cap SI) &= \max(#f(S) - #f(SI), 0) \quad (15) \\
r_{add}(n) &= \frac{\sum_{g \in SI} \min(#f(S \cap SI), #f(RS))}{\sum_{g \in SI} #f(RS \cap SI)}, \\
#f(RS \cap SI) &= \max(#f(R) - #f(SI), 0) \quad (16) \\
p_{keep}(n) &= \frac{\sum_{g \in SI} \min(#f(SI \cap S), #f(SI \cap RS'))}{\sum_{g \in SI} #f(SI \cap SI)}, \\
#f(SI \cap S) &= \min(#f(SI), #f(S)) \quad (17) \\
r_{keep}(n) &= \frac{\sum_{g \in SI} \min(#f(SI \cap S), #f(SI \cap RS'))}{\sum_{g \in SI} #f(SI \cap SI)}, \\
#f(SI \cap RS') &= \min(#f(SI), #f(RS)/r), 0 \quad (18) \\
p_{del}(n) &= \frac{\sum_{g \in SI} \min(#f(SI \cap S), #f(SI \cap R'))}{\sum_{g \in SI} #f(SI \cap S)}, \\
#f(SI \cap S) &= \max(#f(SI) - #f(S)), 0 \quad (19) \\
#f(SI \cap R') &= \max(#f(SI) - #f(RS)/r), 0 \quad (19)
\end{align*}
\]

A limitation of SARI is the controlled count of transformations taking to account thereby evaluating only one to one paraphrased sentences [3]. Furthermore, system is highly penalized by SARI in cases, where only one reference similar to the initial sentence is available. Hence the model’s result does not modify the initial sentence thus giving a less score as stated in [3]. One another popular metric evaluated for ASS system corresponds to reading grade coefficients is Flesch-Kincaid Grade Level (FKGL). FKGL was used in reading tests of Naval personnel as shown in Equation 20. FKGL gives good scores for shorter sentences even if they are grammatically wrong and does not preserve meaning as stated in [3]. Moreover, as mentioned in [57], this score can be used to evaluate superficial simplicity but cannot be used as a metric for the overall comparison of ASS systems.

\[
FKGL = 0.39\left(\frac{\text{number of words}}{\text{number of sentences}}\right) - 15.59 \quad (20)
\]

To evaluate the system as mentioned in [3], there is a need to use the individual metric in conjunction with other metrics. Moreover, to correlate the human judgments with simplicity, fluency, adequacy along with readability, PASS is also evaluated based on all three metrics: SARI (High value preferred), BLEU (High value preferred), and FKGL (Lower value preferred). For this, the Easier Automatic Sentence Simplification Evaluation (EASSE) a Python package is used which aims at facilitating, standardizing automatic comparison and evaluation of Sentence Simplification systems [58].
Table 15. Automatic evaluation of TurkCorpus training dataset for Compound, Complex-Compound, Complex and Compound-Complex algorithm.

| Evaluation     | SARI (High) | BLEU (High) | FKGL (Low) |
|----------------|-------------|-------------|------------|
| Compound       | 22.350      | 8.982       | 7.956      |
| Complex-Compound| 22.085      | 9.283       | 6.516      |
| Complex        | 31.228      | 93.124      | 7.030      |
| Compound-Complex| 22.140      | 8.457       | 6.588      |

Table 16. Compound sentences evaluation concerning rules with TurkCorpus training dataset.

| Evaluation Metric | First Rule | All Rules |
|-------------------|------------|-----------|
| SARI (High)       | 30.3881    | 22.3509   |
| BLEU (High)       | 87.5671    | 8.98225   |
| FKGL (Low)        | 3.7072     | 7.9567    |

Table 17. Automatic evaluation of TurkCorpus test dataset for Compound, Complex-Compound, Complex and Compound-Complex algorithm.

| Evaluation     | SARI (High) | BLEU (High) | FKGL (Low) |
|----------------|-------------|-------------|------------|
| Compound       | 35.245      | 59.529      | 6.440      |
| Complex-Compound| 35.413      | 55.971      | 5.430      |
| Complex        | 30.305      | 92.890      | 7.174      |
| Compound-Complex| 35.412      | 55.962      | 5.566      |

Table 18. Compound sentences evaluation concerning rules with TurkCorpus test dataset.

| Evaluation Metric | First Rule | All Rules |
|-------------------|------------|-----------|
| SARI              | 30.7271    | 35.2454   |
| BLEU              | 87.5296    | 39.3297   |
| FKGL              | 3.9483     | 6.5400    |

Table 19. Performance comparison with existing models.

| Model             | SARI (High) | BLEU (High) | FKGL (Low) |
|-------------------|-------------|-------------|------------|
| SBSMT (PPDB+ SARI)| 39.96       | 73.08       | 7.89 (HSplit) |
| NTS+ SARI         | 37.30       | 79.82       | 7.89 (HSplit) |
| DRESS-LS          | 37.27       | 80.12       | 7.58 (HSplit) |
| DMASST-DCSS       | 40.42       | 73.29       | 7.66 (HSplit) |
| PBSMT-R           | 38.56       | 81.11       | 8.78 (HSplit) |
| Hybrid            | 31.40       | 48.97       | 5.12 (HSplit) |

PASS has been human evaluated by RandomCoCo (private dataset) while the automatic evaluation has been done by a publicly available dataset TurkCorpus. The reason for this is RandomCoCo is essentially good for human evaluation as it has less number of sentences. Therefore it is possible for the human experts to go through each of the sentence correctly based on their expertise and evaluate them. But since the dataset is small hence it cannot be taken forward for automatic evaluation as it would not give accurate results. However, on the contrary using a dataset which is large like Turkcorpus provides good and accurate results through automatic evaluation. Due to its size, this corpus cannot practically be used for human evaluation. As it is not possible for the human expert to manually go through each sentence and evaluate to check whether it has been correctly simplified. Therefore RandomCoCo has been used for human evaluation while, Turkcorpus has been used for automatic evaluation.

C. APPLICABILITY OF PASS

The advantage of splitting the long sentence into the shorter sentence by PASS might have the possibility of aiding the
online anaphora resolution tool. Using these split sentences the publicly available online anaphora resolution tool can possibly resolve the longer sentences with better context. Thus make the sentences grammatically correct which is shown in the following example. The improvement is seen in sentences where there exists an IC connected with conjunction with another object that further extends the first IC as given in – ‘Manuscripts are placed together by rich people, monarchs, hermitages, and temples. They are placed in collections and stores.’ Using an automatic anaphora resolution tool the resolved output was: ‘Manuscripts are placed together by rich people, monarchs, hermitages, and temples. Rich people, monarchs, hermitages, and temples are placed in collections and stores.’ After using PASS method the original sentence gets split as ‘Manuscripts were placed together by rich people, monarchs, hermitages. Manuscripts were placed in collections. Manuscripts were placed in stores.’. With this example, the observation is that anaphora resolution of the simplified sentences generated from PASS might get resolved better thereby improving the efficiency of anaphora resolution.

The possible improvement in the efficiency of an online anaphora resolution tool after simplification using PASS can be a potential future work. Moreover, the second advantage of using PASS is that as the sentences are split into small sentences followed by their anaphora getting resolved. So they can be easily used to create multiple choice questions for assessing factual cognitive levels.

VII. CONCLUSION
For extracting information in the NLP applications simple sentences are pre-requisite. However, the given text document could include various types of compound and complex sentences requiring its simplification. Hence transformation of compound and complex sentences to simple sentences is essential for natural language tasks. Furthermore, such kind of system can be useful for people having disability in reading and language learning. In this direction PASS - a syntactic sentence simplification system has been proposed in this paper. PASS identifies the different types of the compound and complex sentences through grammatical patterns. The system applies rules based on grammatical patterns to simplify the sentences. The generated simple sentences can be utilized to for other systems or automatic tools or modules for assisting other NLP tasks.

The limitation of PASS is it cannot split compound sentences having semicolons along with one conjunction. In some cases, the meaning of the split sentences is not preserved due to the original sentence structure needing human intervention. Furthermore, in the case of complex sentences PASS cannot split the sentences if there is more than one noun in the sentence. Currently, such sentences are transformed into simple sentences by compound-complex algorithm as the research provides a generalized approach to rule-based simplification. Moreover, since cases of repeated nouns are rare in complex sentences this could be a potential extension of the system in future.

One major advantage of PASS over the increasingly popular neural network algorithms is that it allows for granular tuning and modifications as per requirement of the user. Neural network algorithms tend to be a black-box model that do not allow the user to direct the simplification of the sentences. However syntax-based simplification ensures simplification of only those sentences which conform to the rules set, allowing for addition of further rules and modification of existing rules.

This paper makes an unique attempt to check the effect of PASS on anaphora resolution after simplification. In this direction PASS has been evaluated by linguistic experts for its usefulness and influence on anaphora resolution after simplification. An automatic evaluation of PASS along with the automatic metrics has shown that the system gives promising results. Furthermore, the comparison of PASS with existing approaches has given good improvement in terms of the metrics. An extension is to make the system handle compound sentences having multiple conjunctions and complex sentences with more than one noun. This can be achieved by exploring the grammatical patterns of such sentences in CFG. Another long term objective is to check the performance of PASS as it depends on SPACY and NLTK libraries of Python. The observation of the results of simplification show potential benefit in anaphora resolution but needs further investigation not only in anaphora resolution but also in other NLP tasks in the future. Furthermore, future plan includes to check on the system’s efficiency in aiding the anaphora resolution tool and generate multiple-choice questions from the simple sentences generated from PASS.

REFERENCES
[1] R. Chandrasekar, C. Doran, and S. Bangalore, “Motivations and methods for text simplification,” in Proc. 16th Int. Conf. Comput. Linguistics (COLING), vol. 2, 1996, pp. 1–4. [Online]. Available: https://aclanthology.org/C96-2
[2] A. Siddhartham, “Syntactic simplification and text cohesion,” Res. Lang. Comput., vol. 4, no. 1, pp. 77–109, 2006.
[3] F. Alva-Manchego, C. Scarton, and L. Specia, “Data-driven sentence simplification: Survey and benchmark,” Comput. Linguistics, vol. 46, no. 1, pp. 135–187, Mar. 2020.
[4] L. Brouwers, D. Bernhard, A.-L. Ligozat, and T. Francois, “Syntactic sentence simplification for French,” in Proc. 3rd Workshop Predicting Improving Text Readability Target Reader Populations (PITR), 2014, pp. 47–56.
[5] L. Martin, E. de la Clergerie, B. Sagot, and A. Bordes, “Controllable sentence simplification,” in Proc. 12th Lang. Resour. Eval. Conf., Marseille, France, May 2020, pp. 4689–4698. [Online]. Available: https://aclanthology.org/2020.lrec-1.577
[6] V. Ng, “Machine learning for coreference resolution: From local classification to global ranking,” in Proc. 43rd Annu. Meeting Assoc. Comput. Linguistics (ACL), 2005, pp. 157–164. [Online]. Available: https://aclanthology.org/P05-1
[7] C. Poornima, V. Dhanalakshmi, and K. M. Anand, “Rule based sentence simplification for English to Tamil machine translation system,” Int. J. Comput. Appl., vol. 25, no. 8, pp. 38–42, Jul. 2011.
[36] A. Bawakid and M. Oussalah, “Sentences simplification for automatic summarization,” in Proc. IEEE 10th Int. Conf. Cybernetic Intel. Syst. (CIS), Sep. 2011, pp. 59–64.

[39] Y. Dong, Z. Li, M. Rezagholizadeh, and J. C. K. Cheung, “EditSIM: A lightweight syntactic text simplification engine to improve recall in automatic biomedical information extraction,” in Proc. AMIA Annu. Symp., 2010, pp. 351.

[41] J. De Belder and M.-F. Moens, “Text simplification for children,” in Proc. SIGIR Workshop Accessible Search Syst., New York, NY, USA, 2010, pp. 19–26.

[44] S. Yang and M. Lapata, “Sentence simplification with deep reinforcement learning,” in Proc. Conf. Empirical Methods Natural Lang. Process., Copenhagen, Denmark, 2017, pp. 584–594. [Online]. Available: https://aclanthology.org/D17-1062

[47] D. Mantas and F. Soricut, “Static and structure-based sentence simplification for question generation,” in Proc. Int. Conf. Empirical Methods Natural Lang. Process., Brussels, Belgium, 2018, pp. 3164–3173. [Online]. Available: https://aclanthology.org/D18-1355

[50] A. Kumar. (2022). [Online]. Available: https://www.yourdictionary.com/simplified-sentence

[53] J. De Belder and M.-F. Moens, “Text simplification for children,” in Proc. SIGIR Workshop Accessible Search Syst., New York, NY, USA, 2010, pp. 19–26.

[56] A. Kumar, “Sentence simplification for automatic summarization,” in Proc. IEEE 10th Int. Conf. Cybernetic Intel. Syst. (CIS), Sep. 2011, pp. 59–64.

[59] Y. Dong, Z. Li, M. Rezagholizadeh, and J. C. K. Cheung, “EditSIM: A lightweight syntactic text simplification engine to improve recall in automatic biomedical information extraction,” in Proc. AMIA Annu. Symp., 2010, pp. 351.

[62] J. De Belder and M.-F. Moens, “Text simplification for children,” in Proc. SIGIR Workshop Accessible Search Syst., New York, NY, USA, 2010, pp. 19–26.

[65] A. Kumar, “Sentence simplification for automatic summarization,” in Proc. IEEE 10th Int. Conf. Cybernetic Intel. Syst. (CIS), Sep. 2011, pp. 59–64.

[68] Y. Dong, Z. Li, M. Rezagholizadeh, and J. C. K. Cheung, “EditSIM: A lightweight syntactic text simplification engine to improve recall in automatic biomedical information extraction,” in Proc. AMIA Annu. Symp., 2010, pp. 351.
ARCHANA PRAVEEN KUMAR received the B.E. degree in computer science and engineering from Mumbai University, India, in 2002, and the M.Tech. degree in computer science from the Manipal Institute of Technology, MAHE, Manipal, India, in 2011, where she is currently pursuing the Ph.D. degree in computer engineering. She is also working as an Assistant Professor with the Computer Science and Engineering Department, Manipal Institute of Technology. Her research interests include machine learning, semantic web, ontologies, and natural language processing.

ASHALATHA NAYAK received the B.Tech. degree in computer science from the NMAM Institute of Technology, the M.Tech. degree in computer science from the Manipal Institute of Technology, Manipal, and the Ph.D. degree in computer science from the IIT Kharagpur. She is currently a Professor with the CSE Department, Manipal Institute of Technology. She has authored/coauthored more than 35 technical papers in refereed journals and conferences. Her research interests include UML models and object oriented system development, model-based analysis and model-based testing, information security, and enterprise architecture.

MANJULA SHENOY K. (Member, IEEE) received the B.Tech. degree in computer science and engineering from Mangalore University, in 1994, and the M.Tech. and Ph.D. degrees in computer science and engineering from MAHE University, in 2002 and 2014, respectively. She has about 20 indexed journals and about 25 conference papers to her credit. Her research interests include semantic web, cloud computing, data mining, and big data analytics.

MANJULA SHENOY K. (Member, IEEE) received the B.Tech. degree in computer science from the Manipal Institute of Technology, Manipal, India. He has previously interned with Defence Research and Development Organization (GOI) and a start-up that focused on data science. His research interests include machine learning and natural language processing.

AKANSHA PRIYADARSHI received the B.Tech. degree in computer science engineering from the Manipal Institute of Technology, Manipal, Karnataka. She has interned at Goldman Sachs Services Pvt. Ltd. Her research interests include machine learning and natural language processing.

* * *