CONTEXT BASED LEXICAL SIMPLIFICATION

Piyush Pandey[1], Tejas Patil[2], Tanvi Sutar[3], Ankita Ugale[4]

**Dr S.F. Sayyad, Computer Dept. .AISSMS College of Engineering, Pune, Maharashtra. India*
[1]Piyush Pandey. AISSMS College of Engineering, Pune, Maharashtra. India
[2]Tejas Patil., AISSMS College of Engineering, Pune, Maharashtra. India
[3]Tanvi Sutar, AISSMS College of Engineering, Pune, Maharashtra. India
[4]Ankita Ugale, AISSMS College of Engineering, Pune, Maharashtra. India

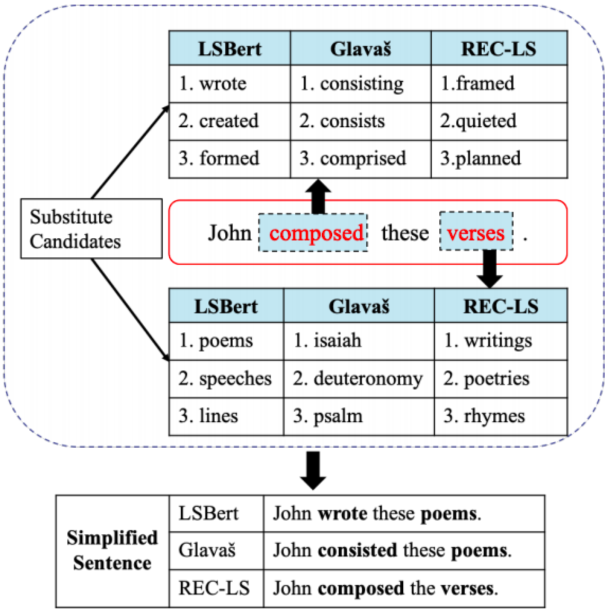
Abstract - *It's tough to decipher text that contains sophisticated and infrequently used vocabulary. Thus, it calls for a mechanism or tool for simplification of the text without changing its meaning. Thus, the motive of this paper is to facilitate non-native (English) language speakers[2], people with dyslexia and children[1] to better understand textual information. For the accomplishment of such a simplification, the approach that is the most appropriate and favored is natural language processing. This task is broadly divided into 3 significant inter-dependent steps which perform identification of complex words[3], generation of appropriate synonyms for the complex words, filtering through all synonyms to find the best substitute and finally the substitution of the generated simpler alternative into the sentence*

***Key Words*: Lexical simplification, BERT, Unsupervised, Pretrained language model.**

1. INTRODUCTION

The main objective of lexical simplification (LS) is to replace complex words with simpler equivalents in order to aid non-native speakers, small kids and specially abled people in comprehending textual data. Lexical simplification can help people in an effective way, because research shows that people can understand the text, in spite of the confusing grammatical context, if they are familiar with the words used. Complex word identification (CWI)[3], substitution generation (SG), and substitute filtering and ranking are all part of the LS framework (SFR).

Existing LS systems use handmade word databases (e.g. Wordnet)[4] or para databases to replace complex words with simpler ones using a set of rules. A technique called word embedding is being heavily used now-a-days to generate substitutes for words. These models produce nearly ten synonyms for the given term. Current state-of-the-art model, REC-LS [5], makes use of both word databases and embedding models. Though the accuracy of generated substitutes is very high, there is a drawback in this model's approach. Due to the little importance given to the context, the model cannot produce substitutes which are meaningful in the given context. This disadvantage of the model, eventually leads to more confusion and has a larger possibility of causing misinterpretation.



LSBert	Glavaš	REC-LS
1. wrote	1. consisting	1.framed
2. created	2. consists	2.quieted
3. formed	3. comprised	3.planned

Substitute Candidates

John composed these verses .

LSBert	Glavaš	REC-LS
1. poems	1. isaiah	1. writings
2. speeches	2. deuteronomy	2. poetries
3. lines	3. psalm	3. rhymes

Simplified Sentence		
	LSBert	John **wrote** these **poems**.
	Glavaš	John **consisted** these **poems**.
	REC-LS	John **composed** the **verses**.

Fig -1: Comparison of substitutes generated by Our model, Glavas and REC-LS model

Context plays a vital role in substitute generation. Fig - 1 depicts a scenario and helps to back the claim. In the figure, Fig -1, substitutes generated by three different models are considered. Models under consideration are LS-Bert, Glavas[6] and RES-LS. Top three substitutes generated by the models for the given complex word can be seen in the figure. Though substitutes generated by Glavas[6] and REC-LS are quite accurate synonyms for the given complex word, the generated synonyms do not fit well with the sentence. They fail to convey the sentence's correct meaning. On the contrary, the substitutes generated by LS-Bert are not only accurate alternatives for the given complex word considered independently but also perfectly fits into the sentence and does not create any ambiguity and lowers the chances of misinterpretation.

Because word complexity is context-dependent, LSBert uses Bidirectional long-short term memory a.k.a Bi-LSTM to recognise complicated words. The Bert Model masks the difficult words that have been identified. Then, both the sentences, with masked and unmasked complex words, are concatenated and fed to the Bert model. The Bert model then produces multiple substitutes for the masked words. To assure grammaticality and meaning equivalence to the original sentence in the output, five high-quality features are employed to rate the substitutions: the frequency of words and their similarity, the ordering of prediction, language models which are based on Bert model and the paraphrase database PPDB. LS-Bert uses a recursive approach to simplify the sentence. It considers one complex word at a time and iterates until all complex words in the sentence are simplified.

2. LITERATURE REVIEW

Previously, lexical simplification tasks were performed by analyzers. [8]Practical Simplification of English Text(PSET) was one of the models studied, and the system was broadly divided into two components: An analyzer component that performs the syntactic analysis and disambiguation of text and a simplify component that eventually takes the output of the analyzer and performs the simplification. The analyzer has 3 main components: lexical tagger, morphological analyzer, parser. The simplifier consists of two components: lexical simplifier and syntactic simplifier. This approach to text simplification was quite primitive and had several flaws. Syntactic simplifiers replaced a big sentence with multiple smaller sentences for better understandability. But it had an adverse effect on the length of the sentences and number of words being used. When used alone, several of the words become confusing. Thus such a simple approach to simplification was not sufficient to counter ambiguity issues. Due to the simplicity of the system, it fails to incorporate the context of the sentence in the lexical simplification process.

Supervised text-data simplification systems are utilized in recently developed text simplification models. The availability of parallel sentences dataset is critical to the performance of such systems. Resources for such parallel sentences are available but they contain a lot of inaccurate data which eventually makes this approach inefficient.

Rule-based[9][10][11], in which each rule contains a difficult word and its related synonyms, is another prominent lexical simplification strategy. In a rule-based approach, synonyms are identified from WordNet or a linguistic dataset that consists of predefined complex words and their simpler counterparts. However, rule-based systems have a key drawback in that it is hard to provide all possible simplification rules for each word.

To avoid such a dependence on parallel corpora or word to synonym mapping datasets, [5] Recursive Context-Aware Lexical Simplification uses this approach where LS systems started using word embeddings. In this approach, the top 10 words were extracted whose vectors were closer to that of the complex word's vector in terms of cosine similarity.

Following a review of existing simplification models ranging from rule-based to embedding-based, the most significant flaw discovered was that the system generated a large number of substitute candidates, making further processing, i.e. determining the best fit substitute, extremely difficult and labor-intensive. This is because these systems either completely ignored the context or treated it with less significance.

Thus, to eliminate these drawbacks, the context-based lexical simplification model is proposed. This model leverages the advanced capacity of BERT to generate substitutes which are accurate and in accordance with the context of the sentence. An LS framework incorporating complicated word identification, substitution generations, and substitute ranking is proposed in this research, which is based on the LSBert-simple framework of lexical simplification. The framework can simplify one sentence recursively.

3. LEXICAL SIMPLIFICATION PIPELINE

We detail each phase of our lexical simplification architecture in this section, which comprises the three steps of complex word identification (CWI), substitution generation (SG), filtering, and substitute ranking (see Figure 2). (SR). Our model iteratively simplifies the text by simplifying one complicated word at a time. Each stage will be described in full below.

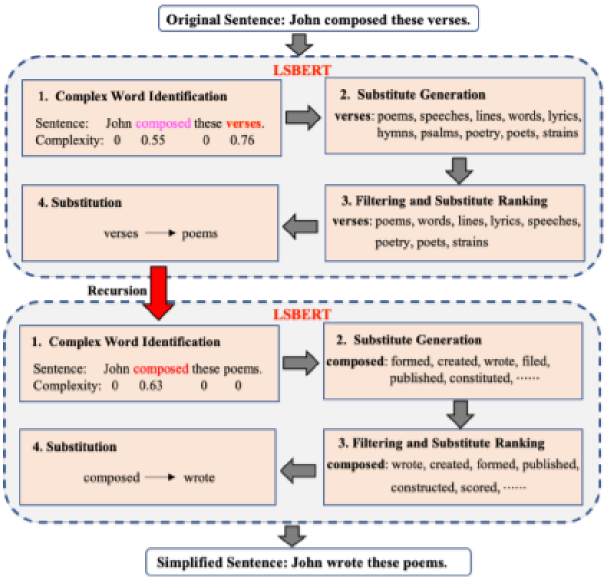

Original Sentence: John composed these verses.
LSBERT
1. Complex Word Identification
Sentence: John composed these verses.
Complexity: 0 0.55 0 0.76
2. Substitute Generation
verses: poems, speeches, lines, words, lyrics, hymns, psalms, poetry, poets, strains
3. Filtering and Substitute Ranking
verses: poems, words, lines, lyrics, speeches, poetry, poets, strains
4. Substitution
verses → poems
Recursion
LSBERT
1. Complex Word Identification
Sentence: John composed these poems.
Complexity: 0 0.63 0 0
2. Substitute Generation
composed: formed, created, wrote, filed, published, constituted, ……
3. Filtering and Substitute Ranking
composed: wrote, created, formed, published, constructed, scored, ……
4. Substitution
composed → wrote
Simplified Sentence: John wrote these poems.

Fig -2: Overview of the lexical simplification framework.

3.1. Complex Word Identification

The goal of identifying complex words from a single sentence has been explored for years, with the purpose of determining which words in a sentence should be simplified.

The CWI was framed as a sequence labeling task, and a SEQ based on bi-directional long short-term memory units (BiLSTM) was trained to predict the binary complexity of words as marked in the dataset. The SEQ model, in contrast to the other CWI models, has two advantages: it takes word context into consideration and it helps eliminate the need for substantial feature engineering because it just uses word embeddings as input information.

The SEQ method assigns a lexical complexity score (p) to each word, indicating the chance that it belongs to the complicated class. If the lexical complexity of a word exceeds a set threshold, it is classified as a complicated word. In Figure 2, for example, the example "John produced these verses" is shown. The two words "assembled" (with p = 0.55) and "verses" (with p = 0.76) will be the challenging words to be simplified if the complexity threshold is set to 0.5. To simplify, our model begins with the word "verses" and the highest p value above the predefined threshold. We'll recalculate the complexity of each word in the sentence when we've completed the simplification process, eliminating terms that have been simplified. Furthermore, by conducting named entity identification on the phrase, we eliminate the simplification of entity words.

3.2. Substitute Generation

The goal of substitue generation (SG) is to generate substitute candidates for the difficult word w given a sentence S and a complex word w. Using the pre-trained language model Bert, our model prefers to produce substitution candidates for the complicated term. We give a quick overview of the Bert model before describing how we use it to perform lexical simplification.

Bert [7] is a self-supervised technique to deep transformer encoder training involving two training shots on goal: masked language modeling (MLM) and the next sentence prediction (NSP). Unlike classical language modeling, which predicts the next word in a series based on its history, MLM predicts missing tokens in a sequence based on its left and right context. Bert completes the NSP task by appending a special categorization token, [CLS], to each sentence and combining sentences with a special separator token, [SEP]. The last hidden state relating to the [CLS] token is used as the total sequence representation from which we estimate a label for classification tasks or may be ignored otherwise.

To conceal the difficult word w in a phrase S, we use the special symbol "[MASK]" as a new sequence S'. If we feed S' directly into MLM, the probability of the vocabulary p(·|S'\{ti}) corresponding to the complex word w solely takes the context into account, regardless of the influence of the complex word w. Bert is skilled at dealing with sentence pairs as a result of the NSP task he has chosen. We concatenate the original sequences S and S' to form a sentence pair, and then input the sentence pair (S, S') into the Bert to extract the probability distribution of the vocabulary p (·|S, S'\{w}) corresponding to the mask word. As a result, the higher probability words in p(·|S, S'\{w}) corresponding to the mask word takes into consideration not just the complex word itself, as well as the context of the sophisticated word. Ultimately, after excluding semantic derivations, we select the top ten words of p(·|S, S'\{w}) as substitution alternatives. Furthermore, because the difficult word's contextual information is applied twice, we randomly mask a fixed number of words in S omitting w to suitably reduce the influence of contextual information.

3.3. Filtering and Substitute Ranking

The replacement ranking of the lexical simplification framework identifies which substitutions are the simplest in the context of a complex word. C = c1, c2, ... cn, where n is the number of replacement candidates [14].

To eliminate some complex options, our approach employs threshold-based filtering. We utilise the Zipf frequency [15], which is the base-10 logarithm of the number of times it appears per billion words, to rank the replacement word possibilities. The higher the value of a term for a person, the more common or familiar it is. Our model then computes various ranks depending on their scores for each of the criteria. After obtaining all of the rankings for each feature, our model scores each candidate by average all of its rankings. Finally, as the best substitution, we select the candidate with the greatest ranking.

4. CONCLUSIONS

In this paper, we provide a text simplification system that simplifies text based on its context. In the next three steps, we summarize the paper's key contribution:

1. The SEQ model, which is more optimal than the classic identification model, is used to identify the complicated word.

2. It creates alternatives for the current complicated word, for which it employs Bert's pre-trained model. Bert is enhanced by two training objectives: MLM (masked language modeling) and NSP (next sentence prediction) are two training objectives that improve Bert's contextual accuracy.

3. It chooses and ranks substitution choices based on the output's grammatically and contextual fit.

In short, our system makes use of the context and consider it as one of the most important parameters for simplification which in turn, makes our proposed system more accurate in all three important steps i.e., Complex Word Identification, Substitute Generation, filtering and substitute ranking in lexical simplification.

5. REFERENCES

[1] J. De Belder, M.-F. Moens, Text simplification for children, In Proceedings of the 2010 SIGIR Workshop on Accessible Search Systems (2010)

[2] G. H. Paetzold, L. Specia, Unsupervised lexical simplification for non- native speakers., in: AAAI, 2016

[3] G. Paetzold, L. Specia, Lexical simplification with neural ranking, in: ACL: Volume 2, Short Papers, 2017.

[4] S. Devlin, J. Tait, The use of a psycholinguistic database in the simplification of text for aphasic readers, Linguistic Databases 1 (1998) 161173.

[5] S. Nisioi, S. Stajner, S. P. Ponzetto, L. P. Dinu, Exploring neural text simplification models, in: ACL, Vol. 2, 2017.

[6] G. Glavas, S. Stajner, Simplifying lexical simplification: do we need simplified corpora?, in: ACL, 2015.

[7] J. Devlin, M.-W. Chang, K. Lee, K. Toutanova, Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805.

[8] John Caroll, Guido Minnen, Yvonne Canning , Siobhan Devlin, and John Tait- Practical Simplification of English newspaper Text to assist Aphasic Readers.

[9] E. Pavlick, C. Callison-Burch, Simple ppdb: A paraphrase database for simplification, in: ACL: Volume 2, Short Papers, 2016.

[10] M. Lesk, Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone, in: Proceedings of the 5th Annual International Conference on Systems Documentation, 1986.

[11] M. Maddela, W. Xu, A word-complexity lexicon and a neural readability ranking model for lexical simplification, in: EMNLP, 2018.

[12] Advaith Siddharthan. 2006. Syntactic Simplification and Text Cohesion. Research on Language and Computation.

[13] Xingxing Zhang and Mirella Lapata. 2017. Sentence Simplification with Deep Reinforcement Learning. In Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing.

[14] G. H. Paetzold, L. Specia, A survey on lexical simplification, in: Journal of Artificial Intelligence Research.

[15] M. Brysbaert, B. New, Moving beyond kucera and francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for american english, Behavior Research Methods.