Lexical Simplification with Pretrained Encoders

Jipeng Qiang,[1] Yun Li,[1] Yi Zhu,[1] Yunhao Yuan,[1] Xindong Wu[2,3]
[1]Department of Computer Science, Yangzhou University, Jiangsu, China
[2]Key Laboratory of Knowledge Engineering with Big Data (Hefei University of Technology), Ministry of Education, Anhui, China
[3] Mininglamp Academy of Sciences, Mininglamp Technology, Beijing, China
{jpqiang, liyun, zhuyi, yhyuan}@yzu.edu.cn, xwu@hfut.edu.cn

Abstract

Lexical simplification (LS) aims to replace complex words in a given sentence with their simpler alternatives of equivalent meaning. Recently unsupervised lexical simplification approaches only rely on the complex word itself regardless of the given sentence to generate candidate substitutions, which will inevitably produce a large number of spurious candidates. We present a simple LS approach that makes use of the Bidirectional Encoder Representations from Transformers (BERT) which can consider both the given sentence and the complex word during generating candidate substitutions for the complex word. Specifically, we mask the complex word of the original sentence for feeding into the BERT to predict the masked token. The predicted results will be used as candidate substitutions. Despite being entirely unsupervised, experimental results show that our approach obtains obvious improvement compared with these baselines leveraging linguistic databases and parallel corpus, outperforming the state-of-the-art by more than 12 Accuracy points on three well-known benchmarks.

1 Introduction

Lexical Simplification (LS) aims at replacing complex words with simpler alternatives, which can help various groups of people, including children (De Belder and Moens 2010), non-native speakers (Paetzold and Specia 2016), people with cognitive disabilities (Feng 2009; Saggion 2017), to understand text better. The popular LS systems still predominantly use a set of rules for substituting complex words with their frequent synonyms from carefully handcrafted databases (e.g., WordNet) or automatically induced from comparable corpora (Devlin and Tait 1998; De Belder and Moens 2010). Linguistic databases like WordNet are used to produce simple synonyms of a complex word. (Lesk 1986; Sinha 2012; Leroy et al. 2013). Parallel corpora like Wikipedia-Simple Wikipedia corpus were also used to extract complex-to-simple word correspondences (Biran, Brody, and Elhadad 2011; Yatskar et al. 2010; Horn, Manduca, and Kauchak 2014). However, linguistic resources are scarce or expensive to produce, such as WordNet and Simple Wikipedia, and it is impossible to give all possible simplification rules from them.

Copyright © 2020, Association for the Advancement of Artificial Intelligence (www.aaai.org). All rights reserved.

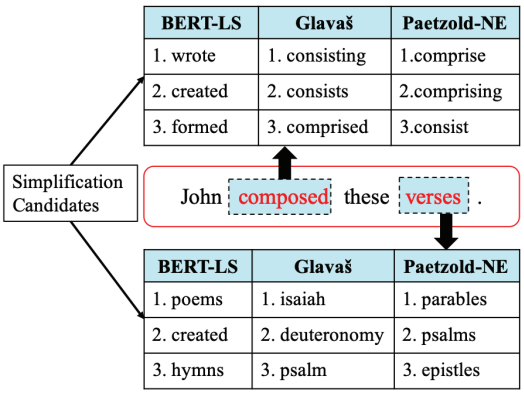

Figure 1: Comparison of simplification candidates of complex words. Given one sentence "John composed these verses." and complex words 'composed' and 'verses', the top three simplification candidates for each complex word are generated by our method BERT-LS and the state-of-the-art two baselines based word embeddings (Glavaš(Glavaš and Štajner 2015) and Paetzold-NE (Paetzold and Specia 2017a)).

For avoiding the need for resources such as databases or parallel corpora, recent work utilizes word embedding models to extract simplification candidates for complex words (Glavaš and Štajner 2015; Paetzold and Specia 2016; 2017a). Given a complex word, they extracted from the word embedding model the simplification candidates whose vectors are closer in terms of cosine similarity with the complex word. This strategy achieves better results compared with rule-based LS systems. However, the above methods generated simplification candidates only considering the complex word regardless of the context of the complex word, which will inevitably produce a large number of spurious candidates that can confuse the systems employed in the subsequent steps.

Therefore, we present an intuitive and innovative idea completely different from existing LS systems in this paper.

We exploit recent advances in the pre-trained transformer language model BERT (Devlin et al. 2018) to find suitable simplifications for complex words. The masked language model (MLM) used in BERT randomly masks some percentage of the input tokens, and predicts the masked word based on its context. If masking the complex word in a sentence, the idea in MLM is in accordance with generating the candidates of the complex word in LS. Therefore, we introduce a novel LS approach BERT-LS that uses MLM of BERT for simplification candidate generation. More specifically, we mask the complex word w of the original sentence S as a new sentence S', and we concatenate the original sequence S and S' for feeding into the BERT to obtain the probability distribution of the vocabulary corresponding to the masked word. The advantage of our method is that it generates simplification candidates by considering the whole sentence, not just the complex word.

Here, we give an example shown in Figure 1 to illustrate the advantage of our method BERT-LS. For complex words 'composed' and 'verses' in the sentence "John composed these verses.", the top three substitution candidates of the two complex words generated by the LS systems based on word embeddings (Glavaš and Štajner 2015; Paetzold and Specia 2017a) are only related with the complex words itself without without paying attention to the original sentence. The top three substitution candidates generated by BERT-LS are not only related with the complex words, but also can fit for the original sentence very well. Then, by considering the frequency or order of each candidate, we can easily choose 'wrote' as the replacement of 'composed and 'poems' as the replacement of 'verses'. In this case, the simplification sentence 'John wrote these poems.' is more easily understood than the original sentence.

The contributions of our paper are as follows:

(1) BERT-LS is a novel BERT-based method for LS, which can take full advantages of BERT to generate and rank substitution candidates. Compared with existing methods, BERT-LS is easier to hold cohesion and coherence of a sentence, since BERT-LS considers the whole sentence not the complex word itself during generating candidates.

(2) BERT-LS is a simple, effective and unsupervised LS method. 1)Simple: many steps used in existing LS systems have been eliminated from our method, e.g., morphological transformation and substitution selection. 2) Effective: it obtains new state-of-the-art results on three benchmarks. 3) Unsupervised: our method cannot rely on any parallel corpus or linguistic databases.

(3) To our best knowledge, this is the first attempt to apply Pre-Trained Transformer Language Models on lexical simplification tasks. The code to reproduce our results is available at https://github.com/anonymous.

2 Related Work

Lexical simplification (LS) contains identifying complex words and finding the best candidate substitution for these complex words (Shardlow 2014; Paetzold and Specia 2017b). The best substitution needs to be more simplistic while preserving the sentence grammatically and keeping its meaning as much as possible, which is a very challenging task. The popular lexical simplification (LS) approaches are rule-based, which each rule contain a complex word and its simple synonyms (Lesk 1986; Pavlick and Callison-Burch 2016; Maddela and Xu 2018). In order to construct rules, rule-based systems usually identified synonyms from WordNet for a predefined set of complex words, and selected the "simplest" from these synonyms based on the frequency of word (Devlin and Tait 1998; De Belder and Moens 2010) or length of word (Bautista et al. 2011). However, rule-based systems need a lot of human involvement to manually define these rules, and it is impossible to give all possible simplification rules.

As complex and simplified parallel corpora are available, especially, the 'ordinary' English Wikipedia (EW) in combination with the 'simple' English Wikipedia (SEW), the paradigm shift of LS systems is from knowledge-based to data-driven simplification (Biran, Brody, and Elhadad 2011; Yatskar et al. 2010; Horn, Manduca, and Kauchak 2014). Yatskar et al. (2010) identified lexical simplifications from the edit history of SEW. They utilized a probabilistic method to recognize simplification edits distinguishing from other types of content changes. Biran et al. (2011) considered every pair of the distinct word in the EW and SEW to be a possible simplification pair, and filtered part of them based on morphological variants and WordNet. Horn et al. (2014) also generated the candidate rules from the EW and SEW, and adopted a context-aware binary classifier to decide whether a candidate rule should be adopted or not in a certain context. The main limitation of the type of methods relies heavily on simplified corpora.

In order to entirely avoid the requirement of lexical resources or parallel corpora, LS systems based on word embeddings were proposed (Glavaš and Štajner 2015). They extracted the top 10 words as candidate substitutions whose vectors are closer in terms of cosine similarity with the complex word. Instead of a traditional word embedding model, Paetzold and Specia (2016) adopted context-aware word embeddings trained on a large dataset where each word is annotated with the POS tag. Afterward, they further extracted candidates for complex words by combining word embeddings with WordNet and parallel corpora (Paetzold and Specia 2017a).

After examining existing LS methods ranging from rules-based to embedding-based, the major challenge is that they generated simplification candidates for the complex word regardless of the context of the complex word, which will inevitably produce a large number of spurious candidates that can confuse the systems employed in the subsequent steps.

In this paper, we will first present a BERT-based LS approach that requires only a sufficiently large corpus of regular text without any manual efforts. Pre-training language models (Devlin et al. 2018; Lee et al. 2019; Lample and Conneau 2019) have attracted wide attention and has shown to be effective for improving many downstream natural language processing tasks. Our method exploits recent advances in BERT to generate suitable simplifications for complex words. Our method generates the candidates of the complex word by considering the whole sentence that

is easier to hold cohesion and coherence of a sentence. In this case, many steps used in existing steps have been eliminated from our method, e.g., morphological transformation and substitution selection. Due to its fundamental nature, our approach can be applied to many languages.

3 Unsupervised Lexical Simplification

In this section, we briefly summarize the BERT model, and then describe how we extend it to do lexical simplification.

3.1 The BERT model

BERT (Devlin et al. 2018) is trained on a masked language modeling (MLM) objective, which is combination of a Markov Random Field language model on a set of discrete tokens with pseudo log-likelihood (Wang and Cho 2019). Unlike a traditional language modeling objective of predicting the next word in a sequence given the history, masked language modeling predicts a word given its left and right context. Let $S = w_1, ..., w_l, ..., w_L$ be a set of discrete tokens, $w_l \in V$, where $V = \{v_1, ..., v_M\}$ is the vocabulary. A joint probability distribution of a given sequence X can be calculated by:

$$p(S|\theta) = \frac{1}{Z(\theta)} \prod_{l=1}^{L} \phi_l(w|\theta) \propto exp(\sum_{l=1}^{L} \log \phi_l(w|\theta))$$

where $\phi_l(w)$ is the l'th potential function with parameters θ, and Z is the partition function.

The log potential functions for the l'th token are defined by:

$$\log \phi_l(w|\theta) = w_l^T f_\theta(S_{\setminus l})$$

where w_l is one-hot vector for the l'the token, and $S_{\setminus l} = (w_1, ..., w_{l-1}$,[MASK],$w_{l+1}, ..., w_L)$

The function $f(S_{\setminus l}) \in \mathbb{R}^M$ is a multi-layer bidirectional transformer model (Vaswani et al. 2017). See (Devlin et al. 2018) for details. The model is trained to approximately maximize the pseudo log-likelihood

$$L(\theta) = E_{S \sim D} \sum_{l=1}^{L} \log p(w_l|S_{\setminus l}; \theta)$$

where D is a set of training examples. In practice, we can stochastically optimize the logloss (computed from the softmax predicted by the f function) by sampling tokens as well as training sentences.

Besides, BERT also uses a "next sentence prediction" task that jointly pre-trains text-pair representations. BERT accomplishes this by prepending every sentence with a special classification token, [CLS], and by combining sentences with a special separator token, [SEP]. The final hidden state corresponding to the [CLS] token is used as the total sequence representation from which we predict a label for classification tasks, or which may otherwise be overlooked.

3.2 Simplification Candidate generation

Simplification candidate generation is used to produce candidate substitutions for each complex word. Due to the fundamental nature of masked language modeling (MLM), we mask the complex word w of the sentence S and get the probability distribution of the vocabulary $p(\cdot|S\setminus\{w\})$ corresponding to the masked word w using MLM. Therefore, we can try to use MLM for simplification candidate generation.

For the complex word w in a sentence S, we mask the word w of S as a new sequence S'. If we directly feed S' into MLM, the probability of the vocabulary $p(\cdot|S'\setminus\{t_i\})$ corresponding to the complex word w only considers the context regardless of the influence of the complex word w.

Since BERT is adept at dealing with sentence pairs due to the "next sentence prediction" adopted by BERT. We concatenate the original sequence S and S' as a sentence pair, and feed the sentence pair S, S' into the BERT to obtain the probability distribution of the vocabulary $p(\cdot|S, S'\setminus\{w\})$ corresponding to the mask word. In this way, the higher probability words in $p(\cdot|S, S'\setminus\{w\})$ corresponding to the mask word not only consider the complex word itself, but also can fit the context of the complex word. Finally, we select as simplification candidates the top 10 words from $p(\cdot|S, S'\setminus\{w\})$, excluding the morphological derivations of w. In all experiments, we use BERT-Large, Uncased (Whole Word Masking) pre-trained on BooksCorpus and English Wikipedia [1].

Suppose that there is a sentence "the cat perched on the mat" and the complex word "perched", we can get the top three simplification candidate words "sat, seated, hopped". See Figure 2 for an illustration. We can see the three candidates not only have a strong correlation with the complex word, but also hold the cohesion and coherence properties of the sentence. If we adopt the existing state-of-the-art method based on word embeddings proposed by (Glavaš and Štajner 2015), the top three substitution words are "atop, overlooking, precariously". Very obviously, our method generates better candidates.

3.3 Substitution Ranking

The substitution ranking of the lexical simplification pipeline is to decide which of the candidate substitutions that fit the context of a complex word is the simplest (Paetzold and Specia 2017b). We rank the candidate substitutions based on the following features. Each of the features captures one aspect of the suitability of the candidate word to replace the complex word.

BERT prediction. On this step of simplification candidate generation, we obtain the probability distribution of the vocabulary corresponding to the mask word $p(\cdot|S, S'\setminus\{w\})$. The higher the probability, the more relevant the candidate for the original sentence. The candidate substitutions can be ranked according to their probability.

Language model features. A simplification candidate should fit into the sequence of words preceding and following the original word. Different n-gram language model, we cannot compute the probability of a sentence or sequence of

[1] https://github.com/google-research/bert

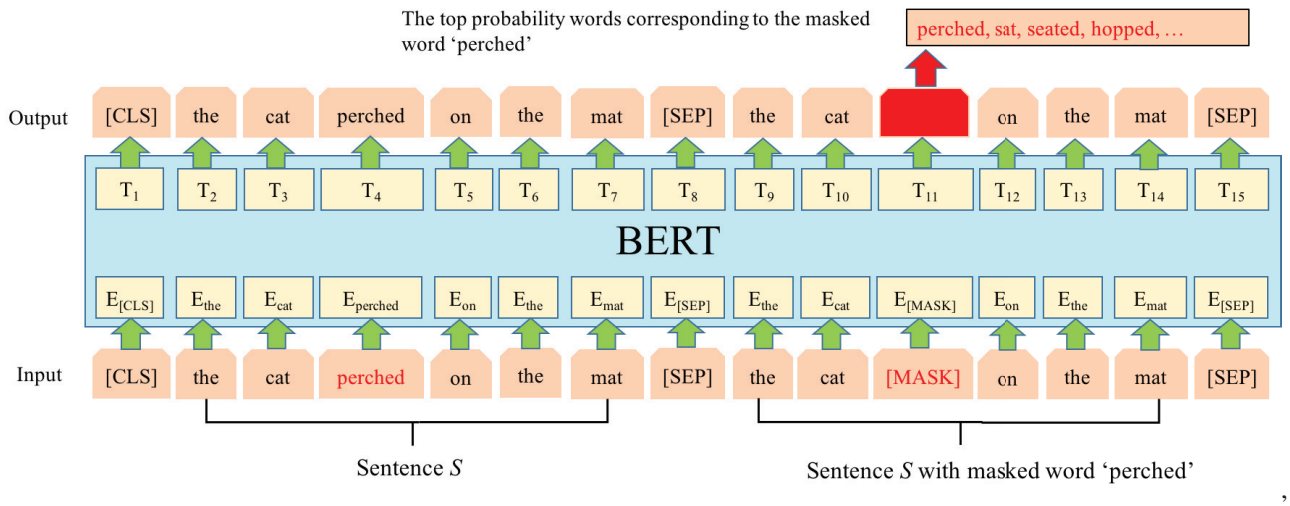

Figure 2: Substitution generation of BERT-LS for the target complex word prediction, or *cloze* task. The input text is "the cat perched on the mat" with complex word "perched". [CLS] and [SEP] are two special symbols in BERT, where [CLS] the token for classification (will not be used in this paper) and [SEP] is a special separator token.

words using MLM. Let $W = w_{-m}, ..., w_{-1}, w, w_1, ..., w_m$ be the context of the original word w. We adopt a new strategy to compute the likelihood of W. We first replace the original word w into the simplification candidate. We then mask one word of W from front to back, and feed into MLM to compute the cross-entropy loss. Finally, we rank all simplification candidates based on the average loss of W. The lower the loss, the simplification candidate is a good substitute for the original word. We use as context a symmetric window of size five around the complex word.

Semantic similarity. The feature is calculated as the cosine between the fastText [2] vector of the original word and the fastText vector of the candidate substitutions. The higher the similarity, the more similar the two words.

Frequency feature. Frequency-based candidate ranking strategies are one of the most popular choices by lexical simplification and can be quite effective. In general, the more frequency a word is used, the most familiar it is to readers. We use word frequency estimates from the Wikipedia [3] and the Children's Book Test (CBT) [4]. As the size of Wikipedia is larger than CBT, we only choose the top 12 million texts of Wikipedia for matching the size.

3.4 Simplification Algorithm

The overall simplification algorithm BERT-LS is shown in Algorithm 1. In this paper, we are not focused on identifying complex words (Paetzold and Specia 2017b), which is a separate task. For each complex word, we first get simplification candidates using BERT after preprocessing the original sequence (lines 1-4). Afterward, BERT-LS computes various rankings for each of the simplification candidates using each of the features, and then scores each candidate by averaging all its rankings (lines 5-13). We choose the top candidate with the highest average rank over all features as the simplification replacement (line 15).

Algorithm 1 Simplify(sentence S, Complex word w)

1: Replace word w of S into [MASK] as S'
2: Concatenate S and S' using [CLS] and [SEP]
3: $p(\cdot|S, S'\backslash\{w\}) \leftarrow BERT(S, S')$
4: $scs \leftarrow top\_probability(p(\cdot|S, S'\backslash\{w\}))$
5: $all\_ranks \leftarrow \varnothing$
6: **for** each feature f **do**
7: $\quad scores \leftarrow \varnothing$
8: $\quad$ **for** each $sc \in scs$ **do**
9: $\quad\quad scores \leftarrow scores \cup f(sc)$
10: $\quad$ **end for**
11: $\quad rank \leftarrow rank\_numbers(scores)$
12: $\quad all\_ranks \leftarrow all\_ranks \cup rank$
13: **end for**
14: $avg\_rank \leftarrow average(all\_ranks)$
15: $best \leftarrow \text{argmax}_{sc}(avg\_rank)$
16: **return** $best$

4 Experiments

We design experiments to answer the following questions:

Q1. The effectiveness of simplification candidates: Does the simplification candidate generation of BERT-LS outperforms the substitution generation of the state-of-the-art competitors?

Q2. The effectiveness of the LS system: Do the FULL Pipeline of BERT-LS outperforms the full pipeline of the state-of-the-art competitors?

[2] https://dl.fbaipublicfiles.com/fasttext/vectors-english/crawl-300d-2M-subword.zip

[3] https://dumps.wikimedia.org/enwiki/

[4] The dataset can be downloaded from http://fb.ai/babi/

	LexMTurk			BenchLS			NNSeval		
	Precision	Recall	F1	Precision	Recall	F1	Precision	Recall	F1
Yamamoto	0.056	0.079	0.065	0.032	0.087	0.047	0.026	0.061	0.037
Devlin	0.164	0.092	0.118	0.133	0.153	0.143	0.092	0.093	0.092
Biran	0.153	0.098	0.119	0.130	0.144	0.136	0.084	0.079	0.081
Horn	0.153	0.134	0.143	0.235	0.131	0.168	0.134	0.088	0.106
Glavaš	0.151	0.122	0.135	0.142	0.191	0.163	0.105	0.141	0.121
Paetzold-CA	0.177	0.140	0.156	0.180	0.252	0.210	0.118	0.161	0.136
Paetzold-NE	**0.310**	0.142	0.195	**0.270**	0.209	0.236	0.186	0.136	0.157
BERT-LS	0.296	**0.230**	**0.259**	0.236	**0.320**	**0.272**	**0.190**	**0.254**	**0.218**

Table 1: Evaluation results of simplification candidate generation on three datasets.

4.1 Experiment Setup

Dataset. We use three widely used lexical simplification datasets to do experiments.

(1) LexMTurk[5] (Horn, Manduca, and Kauchak 2014). It is composed of 500 instances for English. Each instance contains one sentence from Wikipedia, a target complex word and 50 substitutions annotated by 50 Amazon Mechanical "turkers". Since each complex word of each instance was annotated by 50 turkers, this dataset owns a pretty good coverage of gold simplifications.

(2) BenchLS [6] (Paetzold and Specia 2016). It is composed of 929 instances for English, which is from LexMTurk and LSeval (De Belder and Moens 2010). The LSeval contains 429 instances, in which each complex word was annotated by 46 turkers and 9 Ph.D. students. Because BenchLS contains two datasets, it provides the largest range of distinct complex words amongst the LS dataset of English.

(3) NNSeval [7] (Paetzold and Specia 2017b). It is composed of 239 instances for English, which is a filtered version of BenchLS. These instances of BenchLS are dropped: 1) the target complex word in the instance was not regarded as the complex word by a non-native English speaker, and 2) any candidate in the instance was regarded as a complex word by a non-native speaker.

Notice that, because these datasets already offer target words regarded complex by human annotators, we do not address complex word identification task in our evaluations.

Comparison Systems. We choose the following eight baselines to evaluation: Devlin (Devlin and Tait 1998), Biran (Biran, Brody, and Elhadad 2011), Yamamoto (Kajiwara, Matsumoto, and Yamamoto 2013), Horn (Horn, Manduca, and Kauchak 2014), Glavaš (Glavaš and Štajner 2015), SimplePPDB (Pavlick and Callison-Burch 2016), Paetzold-CA (Paetzold and Specia 2016), and Paetzold-NE (Paetzold and Specia 2017a). The substitution generation strategies of these baselines generate candidates from WordNet, EW and SEW parallel corpus, Merriam dictionary, EW and SEW parallel corpus, word embeddings, context-aware word embeddings, combining the Newsela parallel corpus and context-aware word embeddings.

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

[6]http://ghpaetzold.github.io/data/BenchLS.zip

[7]http://ghpaetzold.github.io/data/NNSeval.zip

4.2 Simplification Candidate Generation

The following three widely used metrics are used for evaluation (Paetzold and Specia 2015; 2016).

Precision: The proportion of generated candidates that are in the gold standard.

Recall: The proportion of gold-standard substitutions that are included in the generated substitutions.

F1: The harmonic mean between Precision and Recall.

The results are shown in Table 1. The results of the baselines on LexMTurk are from (Paetzold and Specia 2017b) and the results on BenchLS and NNSeval are from (Paetzold and Specia 2017a). As can be seen, despite being entirely unsupervised, our model BERT-LS obtains F1 scores on three datasets, largely outperforming the previous best baselines. The baseline Paetzold-NE by combining the Newsela parallel corpus and context-aware word embeddings obtains better results on Precision metric, which demonstrates that substitution generation tends to benefit from the combination of different resources. But, for under-resourced languages, our method is still likely to create competitive candidate generators. The results clearly show that BERT-LS provides a good balance precision and recall using only BERT over raw text.

We also present the qualitative evaluation of simplification candidates. We randomly choose 10 short sentences from LexMTurk as examples. Table 2 shows the top six candidates generated by the three approaches. We chooses the state-of-the-art two baselines based word embeddings (Glavaš(Glavaš and Štajner 2015) and Paetzold-NE (Paetzold and Specia 2017a)) as comparison. Candidates that exist in labels (annotated by people) are highlighted in bold.

From Table 2, we observe that BERT-LS achieves the best simplification candidates for complex words compared with the two baselines based word embeddings. The baselines produce a large number of spurious candidates since they only consider the similarity between the complex and the candidate regardless of the context of the complex word. The candidates generated by BERT-LS can maintain the cohesion and coherence of a sentence without the need of morphological transformation.

4.3 FULL LS Pipeline Evaluation

In this section, we evaluate the performance of various complete LS systems. We adopt the following two well-known

Sentence	it originally aired on the Fox network in the United States on October 10, 1991.
Labels	showed, played, shown, appeared, ran, televised, broadcasted, premiered, opened
Glavaš	**broadcasted**, broadcast, re-aired, taped, unbroadcast, al-sumariya
Paetzold-NE	broadcasting, taped, **broadcasted**, re-broadcast, telecast, re-air
BERT-LS	**appeared**, **ran**, **premiered**, **opened**, screened, broadcast
Sentence	a good measure of notability is whether someone has been featured in multiple, independent sources.
Labels	many, several, different, numerous, trotting, some
Glavaš	simultaneous, discrete, simultaneously, predefined, types, overlapping
Paetzold-NE	discrete, simultaneous, overlapping, pre-defined, various, other
BERT-LS	**several**, **numerous**, two, **many**, **different**, single
Sentence	he normally appears in a running gag, where he usually suffers unfortunate, nearly always fatal, events.
Labels	bad, unlucky, sad, terrible, unhappy, hapless, tragic, unlikely, damaging, disasterous
Glavaš	inexplicable, regrettable, distressing, lamentable, galling, obvious
Paetzold-NE	lamentable, galling, obvious, dreadful, inexplicable, **terrible**
BERT-LS	**terrible**, unacceptable, exceptional, unpleasant, **tragic**, unexpected
Sentence	Tolstoy was born in Yasnaya Polyana, the family estate in the Tula region of Russia.
Labels	home, property, house, residence, ands, land, domain, grounds
Glavaš	**property**, realty, freehold, klondikes, real-estate, estate-
Paetzold-NE	condominium, leasehold, realty, xf.com, financier-turned-felon, mirvac
BERT-LS	**property**, **house**, seat, plantation, farm, station
Sentence	Gygax is generally acknowledged as the father of role-playing games.
Labels	usually, normally, mainly, often, mostly, widely, commonly, traditionally, roughly
Glavaš	however, are, therefore, **usually**, furthermore, conversely
Paetzold-NE	therefore, furthermore, uniformly, moreover, as, reasonably
BERT-LS	**widely**, **usually**, **commonly**, **often**, typically, largely
Sentence	he is notable as a performer of Australian traditional folk songs in an authentic style.
Labels	singer, entertainer, musician, artist, player, actor, worker
Glavaš	soloist, pianist, vocalist, flautist, **musician**, songwriter
Paetzold-NE	showman, soloist, pianist, dancer, vocalist, **musician**
BERT-LS	**singer**, practitioner, vocalist, soloist, presenter, **player**
Sentence	Aung San Suu Kyi returned to Burma in 1988 to take care of her ailing mother.
Labels	sick, ill, sickly, dying, suffering, distressed, sickening
Glavaš	troubled, foundering, beleaguered, cash-starved, cash-strapped, debt-ridden
Paetzold-NE	embattled, cash-strapped, beleaguered, foundering, struggling, debt-laden
BERT-LS	**dying**, **sick**, **ill**, elderly, injured, ai
Sentence	the Convent has been the official residence of the Governor of Gibraltar since 1728.
Labels	home, house, dwelling, abode, owner, people
Glavaš	dormitory, domicile, haseldorf, accommodation, non-student, mirrielees
Paetzold-NE	15-bathroom, dormitory, ashenhurst, student-housing, storthes, domicile
BERT-LS	**home**, seat, **house**, mansion, housing, haven
Sentence	Kowal suggested the name and the IAU endorsed it in 1975.
Labels	supported, approved, accepted, backed, okayed, authorized, favored, passed, adopted, ratified
Glavaš	**adopted**, **backed**, rejected, unanimously, drafted, **supported**
Paetzold-NE	drafted, advocated, lobbied, rejected, championed, **supported**
BERT	**approved**, **adopted**, sanctioned, **supported**, **ratified**, **backed**
Sentence	it overwinters in conifer groves .
Labels	tree, pine, evergreen, fir, wood, cedar, grassy, nice
Glavaš	redcedar, groundlayer, multistemmed, coniferous, needle-leaf, deciduous
Paetzold-NE	oak-hickory, castanopsis, pseudotsuga, douglas-fir, menziesii, alnus
BERT-LS	**pine**, cypress, **fir**, redwood, **cedar**, forested

Table 2: Examples of various sentences from LexMTurk dataset. The complex word of each sentence is underlined. We choose the top six substitution candidates for each algorithm. "Labels" is annotated by turkers. Words generated by the algorithms that are exist in the labels are highlighted in bold.

	LexMTurk		BenchLS		NNSeval	
	Precision	Accuracy	Precision	Accuracy	Precision	Accuracy
Yamamoto	0.066	0.066	0.044	0.041	0.444	0.025
Biran	0.714	0.034	0.124	0.123	0.121	0.121
Devlin	0.368	0.366	0.309	0.307	0.335	0.117
PaetzoldCA	0.578	0.396	0.423	0.423	0.297	0.297
Horn	0.761	0.663	0.546	0.341	0.364	0.172
Glavaš	0.710	0.682	0.480	0.252	0.456	0.197
PaetzoldNE	0.676	0.676	**0.642**	0.434	**0.544**	0.335
BERT-LS	**0.776**	**0.776**	0.607	**0.607**	0.423	**0.423**

Table 3: Full pipeline evaluation results using Precision and Accuracy on three datasets.

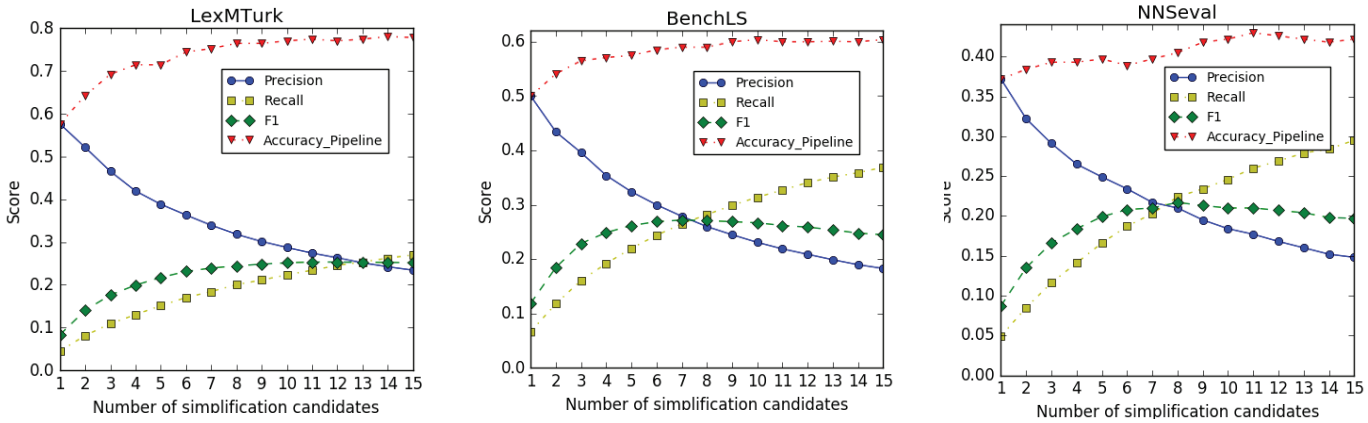

Figure 3: Influence of number of simplification candidates.

metrics used by these work (Horn, Manduca, and Kauchak 2014; Paetzold and Specia 2017b).

Precision: The proportion with which the replacement of the original word is either the original word itself or is in the gold standard

Accuracy: The proportion with which the replacement of the original word is not the original word and is in the gold standard.

The results are shown in Table 3. The results of the baselines on LexMTurk are from (Paetzold and Specia 2017b) and the results on BenchLS and NNSeval are from (Paetzold and Specia 2017a). We can see that our method BERT-LS attains the highest Accuracy on three datasets, which has an average increase of 12% over the former state-of-the-art baseline (Paetzold-NE). It suggests that BERT-LS is the most proficient in promoting simplicity. Paetzold-NE is higher than BERT-LS on Precision on BenchLS and NNSeval, which means that many complex words in two datasets are replaced by the original word itself, due to the shortage of simplification rules in parallel corpora. In conclusion, although BERT-LS only uses raw text for pre-trained BERT without using any resources, BERT-LS remains the best lexical simplification method.

4.4 Influence of the Number of Simplification Candidates

In this part, we try to investigate the influence of the number of simplification candidates to the performance of BERT-LS. The number of candidates ranges from 1 to 15, respectively. Figure 3 shows the performance of simplification candidates (Precision, Recall and F1) and full LS pipeline (Accuracy) with different number of candidates on three benchmarks. When increasing the number of candidates, the score of precision decreases and the score of recall increases. When increasing the number of candidates, the score of F1 first increases, and declines finally. The best performance of simplification candidates through the experiments is achieved by setting the number of candidates equals 7 to 11 for a good trade-off between precision and recall. The score of the accuracy of full LS pipeline first increases and converges finally, which means that the whole LS method is less sensitive to the number of candidates.

5 Conclusion

We proposed a simple BERT-based approach for lexical simplification by leveraging the idea of masking language model of BERT. Our method considers both the complex word and the context of the complex word when generating candidate substitutions without relying on the parallel corpus or linguistic databases. Experiment results have

shown that our approach BERT-LS achieves the best performance on three well-known benchmarks. Since BERT can be trained on raw text, our method can be applied to many languages for lexical simplification.

One limitation of our method is that it can only generate a single-word replacement for complex words, but we plan to extend it to support multi-word expressions. In the future, the pre-trained BERT model can be fine-tuned with just simple English corpus, and then we will use fine-tuned BERT for lexical simplification.

6 Acknowledgment

This research is partially supported by the National Key Research and Development Program of China under grant 2016YFB1000900; the National Natural Science Foundation of China under grants 61703362 and 91746209; the Program for Changjiang Scholars and Innovative Research Team in University (PCSIRT) of the Ministry of Education, China, under grant IRT17R32; and the Natural Science Foundation of Jiangsu Province of China under grant BK20170513.

References

Bautista, S.; León, C.; Hervás, R.; and Gervás, P. 2011. Empirical identification of text simplification strategies for reading-impaired people. In *European Conference for the Advancement of Assistive Technology*, 567–574.

Biran, O.; Brody, S.; and Elhadad, N. 2011. Putting it simply: a context-aware approach to lexical simplification. In *ACL*, 496–501.

De Belder, J., and Moens, M.-F. 2010. Text simplification for children. In *SIGIR Workshop*, 19–26.

Devlin, S., and Tait, J. 1998. The use of a psycholinguistic database in the simpli cation of text for aphasic readers. *Linguistic Databases* 1:161–173.

Devlin, J.; Chang, M.-W.; Lee, K.; and Toutanova, K. 2018. Bert: Pre-training of deep bidirectional transformers for language understanding. In *NAACL*.

Feng, L. 2009. Automatic readability assessment for people with intellectual disabilities. *ACM SIGACCESS accessibility and computing* (93):84–91.

Glavaš, G., and Štajner, S. 2015. Simplifying lexical simplification: do we need simplified corpora? In *ACL*, 63–68.

Horn, C.; Manduca, C.; and Kauchak, D. 2014. Learning a lexical simplifier using wikipedia. In *ACL (Short Papers)*, 458–463.

Kajiwara, T.; Matsumoto, H.; and Yamamoto, K. 2013. Selecting proper lexical paraphrase for children. In *ROCLING*, 59–73.

Lample, G., and Conneau, A. 2019. Cross-lingual language model pretraining. *arXiv preprint arXiv:1901.07291*.

Lee, J.; Yoon, W.; Kim, S.; Kim, D.; Kim, S.; So, C. H.; and Kang, J. 2019. Biobert: pre-trained biomedical language representation model for biomedical text mining. *arXiv preprint arXiv:1901.08746*.

Leroy, G.; Endicott, J. E.; Kauchak, D.; Mouradi, O.; and Just, M. 2013. User evaluation of the effects of a text simplification algorithm using term familiarity on perception, understanding, learning, and information retention. *Journal of medical Internet research* 15(7):e144.

Lesk, M. 1986. Automatic sense disambiguation using machine readable dictionaries: How to tell a pine cone from an ice cream cone. In *Proceedings of the 5th Annual International Conference on Systems Documentation*, SIGDOC '86, 24–26. New York, NY, USA: ACM.

Maddela, M., and Xu, W. 2018. A word-complexity lexicon and a neural readability ranking model for lexical simplification. In *EMNLP*, 3749–3760.

Paetzold, G., and Specia, L. 2015. Lexenstein: A framework for lexical simplification. In *Proceedings of ACL-IJCNLP 2015 System Demonstrations*, 85–90.

Paetzold, G. H., and Specia, L. 2016. Unsupervised lexical simplification for non-native speakers. In *AAAI*, 3761–3767.

Paetzold, G., and Specia, L. 2017a. Lexical simplification with neural ranking. In *Proceedings of the 15th Conference of the European Chapter of the Association for Computational Linguistics: Volume 2, Short Papers*, 34–40.

Paetzold, G. H., and Specia, L. 2017b. A survey on lexical simplification. In *Journal of Artificial Intelligence Research*, volume 60, 549–593.

Pavlick, E., and Callison-Burch, C. 2016. Simple ppdb: A paraphrase database for simplification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (Volume 2: Short Papers)*, 143–148.

Saggion, H. 2017. Automatic text simplification. *Synthesis Lectures on Human Language Technologies* 10(1):1–137.

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

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

Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A. N.; Kaiser, Ł.; and Polosukhin, I. 2017. Attention is all you need. In *Advances in neural information processing systems*, 5998–6008.

Wang, A., and Cho, K. 2019. Bert has a mouth, and it must speak: Bert as a markov random field language model. *arXiv preprint arXiv:1902.04094*.

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 *NAACL*, 365–368.