Simplification Using Paraphrases and Context-based Lexical Substitution

Reno Kriz[*], **Eleni Miltsakaki**[†],
Marianna Apidianaki[*△] and **Chris Callison-Burch**[*]
[*] Computer and Information Science Department, University of Pennsylvania
[△] LIMSI, CNRS, Université Paris-Saclay, 91403 Orsay
[†] Choosito, Inc.
{rekriz,marapi,ccb}@seas.upenn.edu, eleni@choosito.com

Abstract

Lexical simplification involves identifying complex words or phrases that need to be simplified, and recommending simpler meaning-preserving substitutes that can be more easily understood. We propose a complex word identification (CWI) model that exploits both lexical and contextual features, and a simplification mechanism which relies on a word-embedding lexical substitution model to replace the detected complex words with simpler paraphrases. We compare our CWI and lexical simplification models to several baselines, and evaluate the performance of our simplification system against human judgments. The results show that our models are able to detect complex words with higher accuracy than other commonly used methods, and propose good simplification substitutes in context. They also highlight the limited contribution of context features for CWI, which nonetheless improve simplification compared to context-unaware models.

1 Introduction

Automated text simplification is the process that involves transforming a complex text into one with the same meaning, but can be more easily read and understood by a broader audience (Saggion, 2017). This process includes several subtasks such as complex word and sentence identification, lexical simplification, syntactic simplification, and sentence splitting. In this paper, we focus on lexical simplification, the task of replacing difficult words in a text with words that are easier to understand.

Lexical simplification involves two main processes: identifying complex words within a text, and suggesting simpler paraphrases for these words that preserve their meaning in this context. To identify complex words, we train a model on data manually annotated for complexity. Unlike previous work, our classifier takes into account both lexical and context features. We extract candidate substitutes for the identified complex words from SimplePPDB (Pavlick and Callison-Burch, 2016), a database of 4.5 million English simplification rules linking English complex words to simpler paraphrases. We select the substitutes that best fit each context using a word embedding-based lexical substitution model (Melamud et al., 2015). An example sentence, along with the complex words identified by our model and the proposed replacements, is shown in Figure 1. We show that our complex word identification classifier and substitution model improve over several baselines which exploit other types of information and do not account for context. Our approach proposes highly accurate substitutes that are simpler than the target words and preserve the meaning of the corresponding sentences.

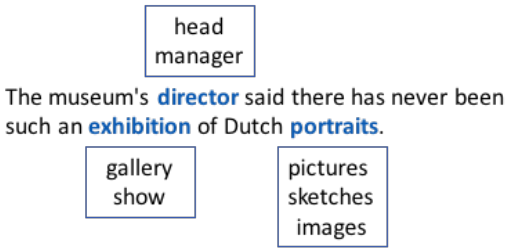

Figure 1: An example sentence with complex words identified by our classifier, and their substitutes proposed by the embedding-based substitution model.

2 Related Work

Prior approaches to text simplification have addressed the task as a monolingual translation problem (Zhu et al., 2010; Coster and Kauchak, 2011; Wubben et al., 2012). The proposed models are trained on aligned sentences extracted from Wikipedia and Simple Wikipedia, a corpus that

contains instances of transformation operations needed for simplification such as rewording, reordering, insertion and deletion. Zhu et al. (2010) propose to use a tree-based translation model which covers splitting, dropping, reordering and substitution. Coster and Kauchak (2011) employ a phrase-based Machine Translation system extended to support phrase deletion, and Wubben et al. (2012) augment a phrase-based system with a re-ranking heuristic.

Woodsend and Lapata (2011) view simplification as a monolingual text generation task. They propose a model based on a quasi-synchronous grammar, a formalism able to capture structural mismatches and complex rewrite operations. The grammar is also induced from a parallel Wikipedia corpus, and an integer linear programming model selects the most appropriate simplification from the space of possible rewrites generated by the grammar. The hybrid model of Angrosh et al. (2014) combines a synchronous grammar extracted from the same parallel corpus with a set of hand crafted syntactic simplification rules. In recent work, Zhang and Lapata (2017) propose a reinforcement learning-based text simplification model which jointly models simplicity, grammaticality, and semantic fidelity to the input. In contrast to these methods, Narayan and Gardent (2016)'s sentence simplification approach does not need a parallel corpus for training, but rather uses a deep semantic representation as input for simplification.

The above-mentioned systems support the full range of transformations involved in text simplification. Other works address specific subtasks, such as syntactic or lexical simplification, which involve identifying grammatical or lexical complexities in a text and rewriting these using simpler words and structures. Syntactic simplification might involve operations such as sentence splitting, rewriting of sentences including passive voice and anaphora resolution (Chandrasekar and Srinivas, 1997; Klerke and Søgaard, 2013).[1] Lexical simplification involves complex word identification, substitute generation, context-based substitute selection and simplicity ranking. To identify the words to be simplified, Shardlow (2013a) proposes to use a Support Vector Machine (SVM) that exploits several lexical features, such as frequency, character and syllable length. Our approach also uses a SVM classifier for identifying complex words, but complements this set of features with context-related features that have not been exploited in previous work.[2]

In the lexical simplification subtask, existing methods differ in their decision to include a word sense disambiguation (WSD) step for substitute selection and in the ranking method used. Ranking is often addressed in terms of word frequency in a large corpus since it has been shown that frequent words increase a text's readability (Devlin and Tait, 1998; Kauchak, 2013). Models that include a semantic processing step for substitute selection aim to ensure that the selected substitutes express the correct meaning of words in specific contexts. WSD is often carried out by selecting the correct synset (i.e. set of synonyms describing a sense) for a target word in WordNet (Miller, 1995) and retrieving the synonyms describing that sense. Thomas and Anderson (2012) use WordNet's tree structure (hypernymy relations) to reduce the size of the vocabulary in a document. Biran et al. (2011) perform disambiguation in an unsupervised manner. They learn simplification rules from comparable corpora and apply them to new sentences using vector-based context similarity measures to select words that are the most likely candidates for substitution in a given context. This process does not involve an explicit WSD step, and simplification is addressed as a context-aware lexical substitution task. The SemEval 2012 English Lexical Simplification task (Specia et al., 2012) also addresses simplification as lexical substitution (McCarthy and Navigli, 2007), allowing systems to use external sense inventories or to directly perform in-context substitution.

In our work, we opt for an approach which addresses lexical substitution in a direct way and does not include an explicit disambiguation step. Lexical substitution systems perform substitute ranking in context using vector-space models (Thater et al., 2011; Kremer et al., 2014; Melamud et al., 2015). Recently, Apidianaki (2016) showed that a syntax-based substitution model can successfully filter the paraphrases available in the

[1] For a detailed overview of syntactic simplification works, see (Shardlow, 2014).

[2] Datasets for system training and evaluation have been made available in the SemEval 2016 Complex Word Identification task (Paetzold and Specia, 2016) but present several issues that make system comparison problematic. We explain the drawbacks of the proposed datasets that led to their exclusion from this work in Section 5.

Paraphrase Database (PPDB) (Ganitkevitch et al., 2013) to select the ones that are adequate in specific contexts. In the same line, Cocos et al. (2017) used a word embedding-based substitution model (Melamud et al., 2015) for ranking PPDB paraphrases in context. We extend this work and adapt the Melamud et al. (2015) model to the simplification setting by using candidate paraphrases extracted from the Simple PPDB resource (Pavlick and Callison-Burch, 2016), a subset of the PPDB that contains complex words and phrases, and their simpler counterparts that can be used for in-context simplification.

3 Identifying Complex Words

3.1 Data

The first step for lexical simplification is to identify the complex words that should be simplified. The bulk of prior work on text simplification has addressed the complex word identification problem by training machine learning algorithms on the parallel Wikipedia Simplification (PWKP) corpus (Zhu et al., 2010). The PWKP corpus, however, has several shortcomings, as described in Xu et al. (2015). Namely, it was determined that 50% of the parallel sentences in PWKP were either not aligned correctly, or the simple sentence was not in fact simpler than the complex sentence. Xu et al. (2015) created a more reliably annotated dataset, which uses a corpus consisting of 1,130 articles, manually rewritten by experts at Newsela[3] at four different reading levels. Xu et al. (2015) also aligned sentences from these texts, extracting 141,582 complex/simple sentence pairs.

We use the Newsela corpus to create a gold-standard dataset of complex and simple words for training and testing our models. We do this by hiring crowdsourced annotators through Amazon Mechanical Turk, and asking them to identify complex words in the context of given texts. We randomly select 200 texts from the Newsela corpus, and take the first 200 tokens from each to be labeled by nine annotators. We preprocess the texts using the Stanford CoreNLP suite (Manning et al., 2014) for tokenization, lemmatization, part-of-speech (POS) tagging, and named entity recognition. The annotators are instructed to label at least 10 complex words they deem worth simplifying for young children, people with disabilities, and second language learners. After filtering out stop words (articles, conjunctions, prepositions, pronouns) and named entities, we are left with 17,318 labeled tokens. Tokens identified by at least three annotators are considered as complex, and tokens labeled by less than three or no annotators as simple. This increases the likelihood of complex segments being actually complex; as we can see from Table 1, words identified by only one or two annotators tend to be somewhat noisy.

Annotators	Prevalence	Example Words
0	0.617	heard, sat, feet, shops, town
1	0.118	protests, pump, trial
2	0.062	sentenced, fraction, primary
3	0.047	measures, involved, elite
4	0.035	fore, pact, collapsed
5	0.031	slew, enrolled, widespread
6	0.029	edible, seize, dwindled
7	0.023	perilous, activist, remorse
8	0.023	vintners, adherents, amassed
9	0.015	abdicate, detained, liaison

Table 1: Examples of words identified as difficult to understand within a text by n annotators, where $0 \leq n \leq 9$. Column 2 (Prevalence) shows the proportion of the total number of words identified as complex by n annotators.

3.2 Methods

Following Shardlow (2013a), we use a Support Vector Machine classifier. We also conduct experiments with a Random Forest Classifier. Shardlow (2013a) identified several features that help to determine whether or not a word is complex, including word length, number of syllables, word frequency, number of unique WordNet synsets, and number of WordNet synonyms. Shardlow used word frequencies extracted from SUBTLEX, a corpus of 51 million words extracted from English subtitles.[4] We instead use n-gram frequencies from the Google Web1T corpus Brants and Franz (2006) (henceforth Google n-gram).

Our motivation for using Google n-gram frequencies is based on the hypothesis that word frequency is a strong indicator of word difficulty. More frequent words are more likely to be easy, and less frequent words are more likely to be unknown and therefore hard to understand. The size of the Google n-gram corpus, consisting of a variety of texts across many genres and years, makes

[3]Newsela is a company that provides reading materials for students in elementary through high school. The Newsela corpus can be requested at https://newsela.com/data/

[4]SUBTLEX can be found at: https://www.ugent.be/pp/experimentele-psychologie/en/research/documents/subtlexus

it a good candidate for computing more accurate word frequencies.

In addition to word frequencies and word specific features, we include several context-specific features: average length of words in the sentence, average number of syllables, average word frequency, average number of WordNet synsets, average number of WordNet synonyms, and sentence length. The intuition for including context-specific features is that if a target word is surrounded by simple words, a reader is likely better able to understand the meaning of the target word, which would thus not need it simplified.

4 Lexical Simplification

4.1 Data

For our model and baselines, we consider candidate substitutions from three datasets. The first is WordNet (Miller, 1995), a lexical network encoding manually identified semantic relationships between words, such as synonymy, hypernymy and hyponymy. This resource has been widely used in substitution tasks (McCarthy and Navigli, 2007). We also use paraphrases extracted from the Paraphrase Database (PPDB) and the Simple Paraphrase Database (SimplePPDB). PPDB is a collection of more than 100 million English paraphrase pairs (Ganitkevitch et al., 2013). These pairs were extracted using a bilingual pivoting technique (Bannard and Callison-Burch, 2005), which assumes that two English phrases that translate to the same foreign phrase have the same meaning. PPDB was updated by Pavlick et al. (2015) to assign labels stating the precise entailment relationship between paraphrase pairs (e.g. forward/backward entailment), and new confidence scores (PPDB 2.0 scores) reflecting the strength of paraphrase relations.

SimplePPDB is a subset of PPDB which contains 4.5 million simplification rules, linking a complex word or phrase with a simpler paraphrase with the same meaning. Simplification rules come with both a PPDB 2.0 score and a simplification confidence score (Pavlick and Callison-Burch, 2016), which represent both the strength of the paraphrase relation and how well the replacement word simplifies the target word. These rules were created by sampling 1,000 PPDB phrases, using crowdsourcing to find correct simplifications for each phrase, and building a model to identify rules that simplify the input phrase.

To evaluate the performance of our lexical simplification model, we create a test set from the Newsela corpus. We extract lexical simplification rules from these parallel sentences using two methods. First, we find sentence pairs with only one lexical replacement and use these word pairs as simplification instances. Next, we use a monolingual word alignment software (Sultan et al., 2014) to extract all non-identical aligned word pairs. We only consider word pairs corresponding to different lemmas (i.e. words with different base forms). From this process, we collect a test set of 14,436 word pairs.

4.2 In-context Ranking and Substitution

To accurately replace words in texts with simpler paraphrases and ensure the generated sentences preserve the meaning of the original, we need to take into account the surrounding context. To do this, we adapt the word embedding-based lexical substitution model of Melamud et al. (2015) to the simplification task. Vector-space models have been shown to effectively filter PPDB paraphrases in context while preserving the meaning of the original sentences (Apidianaki, 2016; Cocos et al., 2017).

The Melamud et al. (2015) model (hereafter AddCos) quantifies the fit of substitute word s for target word t in context C by measuring the semantic similarity of the substitute to the target, and the similarity of the substitute to the context:

$$AddCos(s,t,C) = \frac{\cos(s,t)+\sum_{w\in C}\cos(s,w)}{|C|+1} \quad (1)$$

The vectors s and t are word embeddings of the substitute and target generated by the *skip-gram with negative sampling* model (Mikolov et al., 2013b,a). The context C is the set of context embeddings generated by *skip-gram* for words appearing within a fixed-width window of the target t in a sentence. We use a context window of 1; while this seems counter-intuitive, this is the best-performing window found by (Cocos et al., 2017), and we also confirm this result remains true in Section 5.2. We use the AddCos implementation of Cocos et al. (2017)[5], and 300-dimensional word and context embeddings trained over the 4 billion words in the AGiga corpus (Napoles

[5]Available at https://github.com/acocos/lexsub_addcos

et al., 2012) using the gensim word2vec package (Mikolov et al., 2013b,a; Řehůřek and Sojka, 2010). [6]

In our experiments, candidate substitutes for a target word are its paraphrases in the PPDB and SimplePPDB resources. The model needs to select among these candidates the ones that best carry the meaning of target words in specific contexts. We only consider content words (nouns, verbs, adjectives and adverbs) as simplification targets.

For a "target word-substitute" pair, we include in the model the following features which encode the strength of the semantic relationship between them:

- PPDB 1.0 and 2.0 scores, which represent the overall quality of paraphrases.
- Distributional similarity scores calculated by Ganitkevitch et al. (2013) on the Google n-grams and the AGiga corpus.
- Independence probability, that is the probability that there is no semantic entailment relationship between the paraphrase pair, as calculated by Pavlick et al. (2015).
- SimplePPDB score (Pavlick and Callison-Burch, 2016) – when considering SimplePPDB paraphrases – which reflects the confidence in the simplification rule.

5 Evaluation

5.1 Complex Word Identification

Datasets for training and evaluating Complex Word Identification (CWI) systems were created and released in the SemEval 2016 competition (Paetzold and Specia, 2016) but we decided not to use them for several reasons. Although this was a CWI task, surprisingly only 4.7% of the words in the test data were identified as complex, and all the other words were viewed as simple. As a consequence, none of the systems that participated in the SemEval task managed to beat the accuracy of the "All Simple" baseline which labeled all words in the test set as simple (0.953). As noted by Paetzold and Specia (2016), the inverse problem is present in the corpus developed by Shardlow (2013b), where the "All Complex" baseline achieved higher accuracy, recall and F-scores than all other tested systems, suggesting that marking all words in a sentence as complex is the most effective approach for CWI.

Model	Precision	Recall	F-Score
All-Complex	0.500	**1.000**	0.667
Token Length	0.757	0.900	0.822
n-gram Frequency	0.632	0.862	0.729
SVM-word	**0.880**	0.834	**0.857**
SVM-Context	0.871	0.831	0.850
RF-word	0.805	0.840	0.822
RF-Context	0.824	0.851	0.837

Table 2: Cross-validation performance for four different complex words identification classifiers. Comparison to three baselines. Scores are calculated using **unique** words in our training data.

Another problem in the SemEval-2016 dataset is that although the number of complex words is much higher in the training data (32%), 18% of all words were annotated as complex by only one out of 20 annotators and considered as complex. In addition to the highly different number of complex words in the training and test data, the two datasets are also imbalanced in terms of size, with only 2,237 training instances and 88,211 testing instances. These factors make this dataset a dubious choice for system evaluation. Comparison to the participating systems is also extremely difficult, since the best systems are ones that label most of the data as simple. For these reasons, we decided to create and use our crowdsourced data for training and evaluation.[7]

We compare the performance of an SVM classifier with only word features (SVM-word) to one that exploits both word and context features (SVM-context). We use 5-fold cross validation on unique words from the training data collected through Mechanical Turk (see Section 4.1). We also compare a Random Forest classifier with only word features (RF-word) to one with word and context features (RF-context). We consider three baselines:

- labeling all words as complex (All-Complex).
- thresholding for word length (Token Length), considering longer words as complex; the length threshold with the best performance was 7.

[6] The `word2vec` training parameters we use are a context window of size 3, learning rate *alpha* from 0.025 to 0.0001, minimum word count 100, sampling parameter $1e^{-4}$, 10 negative samples per target word, and 5 training epochs.

[7] We have released the new datasets at https://rekriz11.github.io

		Words with ≥1 paraphrase			All words		
Model	**Coverage**	**Top 1**	**Top 5**	**Oracle**	**Top 1**	**Top 5**	**Oracle**
WordNet frequency	0.911	0.141	0.267	0.291	0.129	0.244	0.265
SimplePPDB Score	0.935	0.180	0.403	0.669	0.168	0.377	0.626
AddCos-PPDB	**0.975**	0.196	0.444	**0.962**	0.191	0.433	**0.938**
AddCos-SimplePPDB	0.819	**0.353**	**0.601**	0.643	**0.289**	**0.492**	0.527

Table 3: Performance of the lexical simplification models on the Newsela aligned test set. Columns 3-5 show the performance of each model on only words with at least one paraphrase in the dataset. Columns 6-8 show the performance of each model on all words; this penalizes for the coverage of the databases.

- thresholding for word frequency using Google n-gram counts (n-gram Frequency), considering more frequent words as simple; the frequency threshold with the best performance was 19,950,000.

The results of this experiment are shown in Table 2. While the Token Length and n-gram Frequency baselines have higher recall, both of our models show substantial improvements in terms of precision and increases overall accuracy and F-score, with SVM outperforming Random Forest. The context-based features seem to have an ambiguous impact, in that they do not improve the performance of the SVM classifier, but they do improve that of the Random Forest classifier. While there are indeed some cases where a relatively simple word is more difficult to understand, due to the size of our corpus, these cases are not found that often in our dataset.

5.2 Lexical Simplification Evaluation

We evaluate the performance of the lexical substitution model using Simple PPDB paraphrases on a test set created from the Newsela corpus, described in Section 4.1. Using the complex word and the corresponding sentence, we find the top suggestions made by our word-embedding based substitution model using SimplePPDB. We compare to three baselines:

- WordNet Frequency: We extract all WordNet synonyms for a complex word, and collect the Google n-gram frequencies for each synonym. We then rank the synonyms in decreasing order of frequency (i.e. the most frequent synonym will be ranked first, and the least frequent one will be ranked last.
- SimplePPDB Score: We extract all SimplePPDB synonyms for a complex word. We then rank the synonyms in decreasing order of their SimplePPDB score.

Context Window	**Top 1**	**Top 5**
0	0.180	0.403
1	0.353	0.601
2	0.352	0.596
3	0.334	0.590
4	0.312	0.585
5	0.291	0.581
6	0.269	0.578
7	0.264	0.577
8	0.252	0.576
9	0.247	0.574
10	0.242	0.572

Table 4: Quality of substitutions proposed by AddCos-SimplePPDB with different context window size as measured by Top 1 and Top 5 accuracy on the Newsela aligned test set.

- AddCos-PPDB: We extract all PPDB synonyms for a complex word and rank them using the AddCos model described above.

The performance of AddCos with SimplePPDB paraphrases (AddCos-SimplePPDB) in the lexical simplification task is compared to performance of the baselines in Table 3. For each model, we calculate Top 1 and Top 5 accuracy scores, which show how often the gold-standard simple word was proposed as the best fitting or among the 5 highest-ranked paraphrases. In addition, we calculate the upper bound performance for each dataset (PPDB, SimplePPDB and WordNet), i.e. how often the gold-standard simple word was found as a paraphrase of the target word in the dataset. This is useful in telling us how well we could potentially do, if we could perfectly rank the paraphrases.

When performing this experiment, we also evaluated the impact of the context window size on the quality of the proposed substitutions. We varied the context window used by the AddCos-SimplePPDB model from 0 to 10. The results of this comparison are found in Table 4. As we can see, the largest effect, as expected, is when the model changes from using no context to choosing a window size of 1 word on either side of the word that is being replaced. As the context window in-

Synonym Rank	Substitution	Simplification	Both
1	**0.396**	**0.280**	**0.227**
2	0.311	0.214	0.153
3	0.278	0.184	0.127
4	0.228	0.142	0.093
5	0.193	0.123	0.075
All	**0.622**	**0.553**	**0.435**

Table 5: Performance of our overall lexical simplification system. We give the proportion of substitutes the system ranked at positions 1 to 5 (i.e. from the top ranked to the fifth-ranked paraphrase in context) which was identified by a majority of workers as (a) a good substitute in context (Substitution); (b) simpler than the target word (Simplification); (c) both a good and simpler substitute (Both). We also show the proportion of complex words where at least one of the top 5 paraphrases satisfies these criteria in the last row.

Baseline	Simple	Complex
n-gram Frequency	dug, sled, chart, lakes, push, tight, harm	estimates, frequent, attributed, isolated, preferred, liability
Token Length	nursing, unknown, squares, feeling, teaching, strength	adorns, asylum, myriad, rigors, nutria, edible
RF-Context	malls, hungry, therefore, hears, heavily, rainy	engaging, secular, gridlock, torrent, sanctions, lobbying
SVM-Context	peacefully, favorite, amazing, websites, harmful, somewhat	swelled, entice, tether, chaotic, vessel, midst

Table 6: Examples of words that were incorrectly classified by the two best performing baselines and the RF-Context model, but were correctly classified by the SVM-Context model. The last row shows examples of words that were incorrectly classified by the SVM-Context model.

creases above 2, however, we see a significant decrease in Top 1 accuracy, and a slower decrease in Top 5 accuracy. Thus, in our model, we chose to use a context window of 1.

We experimented with filtering the substitution candidates using SimplePPDB confidence scores, PPDB paraphrase quality scores, and AddCos context similarity scores, but these all resulted in a slight, non-significant increase in performance, and a significant decrease in coverage. We will also explore other ways for promoting high-quality substitutions without hurting the overall coverage of the system in the future.

One thing to note is that just because a model does not find the gold-standard simple word, does not necessarily mean that it does not find any good substitutes in context. Concrete examples of this are shown in Section 6.

5.3 Overall Simplification System

We integrate the best complex word identification classifier (SVM-context) and the substitution model that provided the best ranking in context (AddCos-SimplePPDB), into a simplification pipeline. The input text is a complex text that needs to be simplified and the output consists of simplification suggestions for experts to choose from in order to create simpler versions of texts. The input text is pre-processed using the Stanford CoreNLP suite (Manning et al., 2014) which performs tokenization, sentence splitting, lemmatization, part-of-speech and named entity tagging. The SVM-Context classifier is used to classify each content word that is not part of a named entity as either simple or complex.

The lexical substitution model then gathers the SimplePPDB substitutes available for the complex target word and ranks them according to how well they fit the corresponding context. We only keep the top five suggestions made by the model as final output.

To evaluate the performance of the overall simplification system, we used the 930 texts from the Newsela corpus that were not used in the training of the CWI classifier. Our model identified over 170,000 complex words that also had paraphrases in SimplePPDB. We again asked crowdsourced annotators to evaluate the suggestions made for a random sample of 2,500 complex words on Amazon Mechanical Turk, in order to determine the number of good substitutions in context, the number of suggested paraphrases that are simpler than the target words, and the suggestions that are both simpler paraphrases and good in-context substitutes.

Table 5 shows the quality of the paraphrases ranked by our system in positions from one to five. We can see that the paraphrases our system selects as the best have a higher likelihood of being both good substitutes in context and simpler than the target word. We also show the proportion of target words that had at least one good substitute in context, one simple substitute, and one good and simple substitute.

6 Error Analysis

In this section, we give examples of words for which our models give the correct output and the

Sentence	Gold-Standard	WordNet Frequency	SimplePPDB Score	AddCos-PPDB	AddCos-SimplePPDB
(7.1) Advocates **argue** that including women will help end harassment of female troops.	**say**	reason, fence, debate, contend, indicate	**say**, think, tell, talk, mean	contend, assert, acknowledge, insist, complain	**say**, claim, believe, suggest, debate
(7.2) But in April , detainees covered cameras used to **monitor** them.	**watch**	supervise, proctor, admonisher	find, meet, give, try, allow	track, manipulate, control, analyze, supervise	track, control, check, **watch**, follow
(7.3) Similarly , police can investigate cases and have the **authority** to seize animals.	**power**	agency, potency, bureau, assurance	force, control, permission, office, limit	jurisdiction, discretion, right, prerogative, ability	**power**, responsibility, body, agency

Table 7: Examples of the top-5 substitutes for our three baselines and our best model (AddCos-SimplePPDB). We also provide the gold-standard simplification (Gold-Standard).

Sentence	Bad Substitutions
(8.1)b	basic
(8.2) Russian poultry is more expensive, and U.S. producers enjoy numerous cost **advantages**.	prospect, benefits, revenue, merit, feature
(8.3) Although the calculus may be different with Syrian **refugees**, the parallel for me is politics.	life, right, return, shelter, million
(8.4) He saw them bring in animals to a university, where they'll be cared for and put up for **adoption**.	acceptance, passage, approval, endorsement

Table 8: Examples of words and their context where our model fails to provide any good replacements.

baselines fail to do so. In addition, we give examples of words on which our models perform poorly.

First, we consider examples of words that were incorrectly classified by each of the four best performing CWI models: the RF-Context and SVM-Context models, and the n-gram Frequency and Token Length baselines. (Table 6). In the first three rows, we give words that were correctly identified by SVM-Context, but incorrectly categorized by the two baselines and RF-Context; in the last row, we give examples of words incorrectly classified by SVM-Context. We observe that the n-gram Frequency model tends to incorrectly classify relatively short words that are rare in the Google n-gram corpus as complex. On the other end, the Token Length model shows that using this feature alone leads to incorrectly identifying shorter words such as "adorn" and "myriad" as simple, when these words are relatively complex.

Table 7 presents examples of substitution where the baseline systems did not find the correct paraphrase, but AddCos-SimplePPDB did. As we have mentioned, even when a model did not find the gold-standard paraphrase, they sometimes did find a different paraphrase that works well in the context. In Example 7.2, the top paraphrase identified by both AddCos-PPDB and AddCos-Simple PPDB for the word "monitor" is "track", which is a reasonable substitute. On the other hand, in Example 7.3, AddCos-Simple PPDB model was able to identify a good simple substitute, when none of the other models were able to identify a suitable word with comparable complexity.

Finally, Table 8 shows examples of output of the overall simplification system. Here, the **blue** word is a word that our CWI classifier identified as complex (for simplicity, we only look at one complex word per sentence). From there, we consider the five top-ranked substitutes proposed by AddCos-Simple PPDB, and show which were identified by the majority of annotators as good substitutes for the target word, simpler than the target, good simpler substitutes, and bad substitutes. In row 5 of Table 8, we can see that for the word "adoption", all five words identified by our model are considered to be bad substitutes, since they are all synonyms describing a different sense of adoption. Even though SimplePPDB is quite large, it does not cover all senses of the words represented. Another issue is that SimplePPDB contains some noisy paraphrases, as is the case with all automatically collected synonym banks. We see this with "recognize" being a synonym of "recognition", even though we specified that "recognition" is a noun. Our model does filter out the worst paraphrases (with PPDB2.0 score < 2), but there are still some words that are simply poor substitutes.

We reviewed the examples where our system failed to generate acceptable substitutions for the identified complex words. Below we present the major categories of errors.

- The identified complex term is part of a phrase and no substitution is acceptable. For example, in Example 8.1, Elementary, Mid-

dle or High School is a description of the type of school. *Elementary School* has an alternative name in some cases but *High School* should never become *Tall School*.

- The complex word has no simpler synonym that would be a good substitute. The difficulty of the word might reside in its meaning which can be unknown to the reader. In Example 8.3, it would be more useful to point to the definition of *refugees*.

- The complex word is part of a predicate with arguments that are not accessible to our model. In Example 8.4, the intended meaning of *adoption*, human adoption, is hard to capture in the vicinity of the complex word.

- Finally, in some cases, our annotators were quite strict in admitting a substitute. In Example 8.2, for example, *cost merit* would not be syntactically correct but *cost merits* would be acceptable.

7 Conclusions and Future Work

We present a novel model for simplification that first identifies complex words in text, and then ranks lexical simplification candidates according to their adequacy in these specific contexts. We perform experiments showing that our model makes correct simplification suggestions 35% of the time as measured by top-1 accuracy (versus 20% of the time for the best baseline), and produces a good substitution in its top-5 predictions 60% of the time (versus 44% for the best baseline). We perform a detailed error analysis that suggests future improvements, e.g. not replacing words within collocations like *elementary school*, and extending the context model to include the arguments of words that are going to be simplified.

Achieving high performance on single words is crucial for any system that hopes to adequately holistically simplify a text. Our methods can also be extended to the phrase level. SimplePPDB contains phrasal simplification rules, as well as lexical simplification rules. We can assign a vector representation to phrases to be used by the AddCos model, by applying a vector composition method to the vectors of individual words in the phrase. We plan to extend our method in this direction in future work.

Although our system outperforms simpler baselines on both tasks, the performance of the overall system is relatively low. The filtering mechanisms we have experimented with up to now in order to make high-confidence predictions, increased the quality of the proposed substitutions but significantly decreased the coverage. We will explore other ways for promoting high-quality substitutions without hurting the overall coverage of the system in the future.

The AddCos implementation we used in this work does not rely on syntactic annotations and can be easily applied to new languages. In future work, we plan to experiment with syntactic substitution models and with syntax-based word embeddings like the ones used in the initial AddCos implementation (Melamud et al., 2015). We expect syntactic information to further enhance the quality of the proposed substitutions, ensuring the functional similarity of the lexical substitutions to the target word. Furthermore, we intend to integrate lexical and syntactic simplification, both crucial steps towards text simplification.

8 Data and Software

We release the data that we collected, which is of higher quality than the data used in previous shared tasks on Complex Word Identification. We also release our software for performing context-aware paraphrase substitutions. The dataset and the code can be found at https://rekriz11.github.io

9 Acknowledgements

We would like to thank the anonymous reviewers for their helpful comments and feedback on this work, and Anne Cocos for sharing with us her implementation of the AddCos model with PPDB paraphrase substitutes.

This material is based in part on research sponsored by DARPA under grant number FA8750-13-2-0017 (the DEFT program) and HR0011-15-C-0115 (the LORELEI program). The U.S. Government is authorized to reproduce and distribute reprints for Governmental purposes. The views and conclusions contained in this publication are those of the authors and should not be interpreted as representing official policies or endorsements of DARPA and the U.S. Government. The work has also been supported by the French National Research Agency under project ANR-16-CE33-0013. Finally, we gratefully acknowledge the support of NSF-SBIR grant 1456186.

References

Mandya Angrosh, Tadashi Nomoto, and Advaith Siddharthan. 2014. Lexico-syntactic text simplification and compression with typed dependencies. In *Proceedings of COLING 2014, the 25th International Conference on Computational Linguistics: Technical Papers*. Dublin, Ireland, pages 1996–2006.

Marianna Apidianaki. 2016. Vector-space models for PPDB paraphrase ranking in context. In *Proceedings of the 2016 Conference on Empirical Methods in Natural Language Processing*. Austin, Texas, pages 2028–2034.

Colin Bannard and Chris Callison-Burch. 2005. Paraphrasing with Bilingual Parallel Corpora. In *Proceedings of the 43rd Annual Meeting of the Association for Computational Linguistics (ACL'05)*. Ann Arbor, Michigan, pages 597–604.

Or Biran, Samuel Brody, and Noemie Elhadad. 2011. Putting it simply: a context-aware approach to lexical simplification. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*. Portland, Oregon, USA, pages 496–501.

Thorsten Brants and Alex Franz. 2006. Web 1t 5-gram version 1 ldc2006t13 .

R Chandrasekar and B Srinivas. 1997. Automatic induction of rules for text simplification 10:183–190.

Anne Cocos, Marianna Apidianaki, and Chris Callison-Burch. 2017. Word Sense Filtering Improves Embedding-Based Lexical Substitution. In *Proceedings of the 1st Workshop on Sense, Concept and Entity Representations and their Applications*. Valencia, Spain, pages 110–119.

Will Coster and David Kauchak. 2011. Learning to simplify sentences using wikipedia. In *Proceedings of the Workshop on Monolingual Text-To-Text Generation*. Portland, Oregon, pages 1–9.

Siobhan Devlin and John Tait. 1998. The use of a psycholinguistic database in the simplification of text for aphasic readers. *Linguistic Databases* pages 161–173.

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

David Kauchak. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics*. Association for Computational Linguistics, Sofia, Bulgaria, pages 1537–1546.

Sigrid Klerke and Anders Søgaard. 2013. Simple, readable sub-sentences. In *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop*. Sofia, Bulgaria, pages 142–149.

Gerhard Kremer, Katrin Erk, Sebastian Padó, and Stefan Thater. 2014. What substitutes tell us - analysis of an "all-words" lexical substitution corpus. In *Proceedings of the 14th Conference of the European Chapter of the Association for Computational Linguistics*. Gothenburg, Sweden, pages 540–549.

Christopher Manning, Mihai Surdeanu, John Bauer, Jenny Finkel, Steven Bethard, and David McClosky. 2014. The Stanford CoreNLP Natural Language Processing Toolkit. In *Proceedings of 52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. Baltimore, Maryland, pages 55–60.

Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of the Fourth International Workshop on Semantic Evaluations (SemEval-2007)*. Prague, Czech Republic, pages 48–53.

Oren Melamud, Omer Levy, and Ido Dagan. 2015. A Simple Word Embedding Model for Lexical Substitution. In *Proceedings of the 1st Workshop on Vector Space Modeling for Natural Language Processing*. Denver, Colorado, pages 1–7.

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

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

George A. Miller. 1995. WordNet: A lexical database for English. *Communications of the ACM* 38(11):39–41.

Courtney Napoles, Matthew Gormley, and Benjamin Van Durme. 2012. Annotated Gigaword. In *Proceedings of the Joint Workshop on Automatic Knowledge Base Construction and Web-scale Knowledge Extraction (AKBC-WEKEX)*. Montreal, Canada, pages 95–100.

Shashi Narayan and Claire Gardent. 2016. Unsupervised sentence simplification using deep semantics. In *Proceedings of the 2016 International Conference (INLG)*. Edinburgh, Scotland, pages 111—120.

Gustavo Paetzold and Lucia Specia. 2016. SemEval 2016 Task 11: Complex Word Identification. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*. Association for Computational Linguistics, San Diego, California, pages 560–569.

Ellie Pavlick and Chris Callison-Burch. 2016. Simple PPDB: A Paraphrase Database for Simplification. In *Proceedings of the 54th Annual Meeting of the Association for Computational Linguistics (ACL 2016)*. Berlin, Germany.

Ellie Pavlick, Pushpendre Rastogi, Juri Ganitkevitch, Benjamin Van Durme, and Chris Callison-Burch. 2015. PPDB 2.0: Better paraphrase ranking, fine-grained entailment relations, word embeddings, and style classification. In *Proceedings of the 53rd Annual Meeting of the Association for Computational Linguistics and the 7th International Joint Conference on Natural Language Processing (Volume 2: Short Papers)*. Beijing, China, pages 425–430.

Radim Řehůřek and Petr Sojka. 2010. Software Framework for Topic Modelling with Large Corpora. In *Proceedings of the LREC 2010 Workshop on New Challenges for NLP Frameworks*. Valletta, Malta, pages 45–50.

Horacio Saggion. 2017. *Automatic Text Simplification*. Synthesis Lectures on Human Language Technologies. Morgan & Claypool Publishers.

Matthew Shardlow. 2013a. A Comparison of Techniques to Automatically Identify Complex Words. In *51st Annual Meeting of the Association for Computational Linguistics Proceedings of the Student Research Workshop*. Association for Computational Linguistics, Sofia, Bulgaria, pages 103–109.

Matthew Shardlow. 2013b. The cw corpus: A new resource for evaluating the identification of complex words. In *Proceedings of the Second Workshop on Predicting and Improving Text Readability for Target Reader Populations*. Sofia, Bulgaria, pages 69–77.

Matthew Shardlow. 2014. A Survey of Automated Text Simplification. *International Journal of Advanced Computer Science and Applications(IJACSA), Special Issue on Natural Language Processing* 2:58–70.

Lucia Specia, Sujay Kumar Jauhar, and Rada Mihalcea. 2012. Semeval-2012 task 1: English lexical simplification. In **SEM 2012: The First Joint Conference on Lexical and Computational Semantics – Volume 1: Proceedings of the main conference and the shared task, and Volume 2: Proceedings of the Sixth International Workshop on Semantic Evaluation (SemEval 2012)*. Montréal, Canada, pages 347–355.

Md Arafat Sultan, Steven Bethard, and Tamara Sumner. 2014. Back to Basics for Monolingual Alignment: Exploiting Word Similarity and Contextual Evidence. *Transactions of the Association for Computational Linguistics* 2:219–230.

Stefan Thater, Hagen Fürstenau, and Manfred Pinkal. 2011. Word meaning in context: A simple and effective vector model. In *Proceedings of 5th International Joint Conference on Natural Language Processing*. Asian Federation of Natural Language Processing, Chiang Mai, Thailand, pages 1134–1143.

S. Rebecca Thomas and Sven Anderson. 2012. Wordnet-based lexical simplification of a document. In Jeremy Jancsary, editor, *Proceedings of KONVENS 2012*. ÖGAI, pages 80–88.

Kristian Woodsend and Mirella Lapata. 2011. Learning to simplify sentences with quasi-synchronous grammar and integer programming. In *Proceedings of the 2011 Conference on Empirical Methods in Natural Language Processing*. Edinburgh, Scotland, UK., pages 409–420.

Sander Wubben, Antal van den Bosch, and Emiel Krahmer. 2012. Sentence simplification by monolingual machine translation. In *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Jeju Island, Korea, pages 1015–1024.

Wei Xu, Chris Callison-Burch, and Courtney Napoles. 2015. Problems in current text simplification research: New data can help. *Transactions of the Association for Computational Linguistics* 3:283–297.

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*. Copenhagen, Denmark, pages 584–594.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A Monolingual Tree-based Translation Model for Sentence Simplification. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*. Coling 2010 Organizing Committee, Beijing, China, pages 1353–1361.