Learning a Lexical Simplifier Using Wikipedia

Colby Horn, Cathryn Manduca and **David Kauchak**
Computer Science Department
Middlebury College
{chorn,cmanduca,dkauchak}@middlebury.edu

Abstract

In this paper we introduce a new lexical simplification approach. We extract over 30K candidate lexical simplifications by identifying aligned words in a sentence-aligned corpus of English Wikipedia with Simple English Wikipedia. To apply these rules, we learn a feature-based ranker using SVM^{rank} trained on a set of labeled simplifications collected using Amazon's Mechanical Turk. Using human simplifications for evaluation, we achieve a precision of 76% with changes in 86% of the examples.

1 Introduction

Text simplification is aimed at reducing the reading and grammatical complexity of text while retaining the meaning (Chandrasekar and Srinivas, 1997). Text simplification techniques have a broad range of applications centered around increasing data availability to both targeted audiences, such as children, language learners, and people with cognitive disabilities, as well as to general readers in technical domains such as health and medicine (Feng, 2008).

Simplifying a text can require a wide range of transformation operations including lexical changes, syntactic changes, sentence splitting, deletion and elaboration (Coster and Kauchak, 2011; Zhu et al., 2010). In this paper, we examine a restricted version of the text simplification problem, lexical simplification, where text is simplified by substituting words or phrases with simpler variants. Even with this restriction, lexical simplification techniques have been shown to positively impact the simplicity of text and to improve reader understanding and information retention (Leroy et al., 2013). Additionally, restricting the set of transformation operations allows for more straightforward evaluation than the general simplification problem (Specia et al., 2012).

Most lexical simplification techniques rely on transformation rules that change a word or phrase into a simpler variant with similar meaning (Biran et al., 2011; Specia et al., 2012; Yatskar et al., 2010). Two main challenges exist for this type of approach. First, the lexical focus of the transformation rules makes generalization difficult; a large number of transformation rules is required to achieve reasonable coverage and impact. Second, rules do not apply in all contexts and care must be taken when performing lexical transformations to ensure local cohesion, grammaticality and, most importantly, the preservation of the original meaning.

In this paper, we address both of these issues. We leverage a data set of 137K aligned sentence pairs between English Wikipedia and Simple English Wikipedia to learn simplification rules. Previous approaches have used unaligned versions of Simple English Wikipedia to learn rules (Biran et al., 2011; Yatskar et al., 2010), however, by using the aligned version we are able to learn a much larger rule set.

To apply lexical simplification rules to a new sentence, a decision must be made about which, if any, transformations should be applied. Previous approaches have used similarity measures (Biran et al., 2011) and feature-based approaches (Specia et al., 2012) to make this decision. We take the latter approach and train a supervised model to rank candidate transformations.

2 Problem Setup

We learn lexical simplification rules that consist of a word to be simplified and a list of candidate simplifications:

$$w \rightarrow c_1, c_2, ..., c_m$$

Consider the two aligned sentence pairs in Table

Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Short Papers), pages 458–463,
Baltimore, Maryland, USA, June 23-25 2014.

The first school was **established** in 1857. The first school was **started** in 1857.
The district was **established** in 1993 by merging the former districts of Bernau and Eberswalde. The district was **made** in 1993 by joining the old districts of Bernau and Eberswalde.

Table 1: Two aligned sentence pairs. The bottom sentence is a human simplified version of the top sentence. Bold words are candidate lexical simplifications.

1. The bottom sentence of each pair is a simplified variant of the top sentence. By identifying aligned words within the aligned sentences, candidate lexical simplifications can be learned. The bold words show two such examples, though other candidates exist in the bottom pair. By examining aligned sentence pairs we can learn a simplification rule. For example, we might learn:

$$established \rightarrow began, made, settled, started$$

Given a sentence $s_1, s_2, ..., s_n$, a simplification rule applies if the left hand side of the rule can be found in the sentence ($s_i = w$, for some i). If a rule applies, then a decision must be made about which, if any, of the candidate simplifications should be substituted for the word w to simplify the sentence. For example, if we were attempting to simplify the sentence

```
The ACL was established in 1962.
```

using the simplification rule above, some of the simplification options would not apply because of grammatical constraints, e.g. *began*, while others would not apply for semantic reasons, e.g. *settled*. This does not mean that these are not good simplifications for *established* since in other contexts, they might be appropriate. For example, in the sentence

```
The researcher established a new paper
writing routine.
```

began is a reasonable option.

3 Learning a Lexical Simplifier

We break the learning problem into two steps: 1) learn a set of simplification rules and 2) learn a ranking function for determining the best simplification candidate when a rule applies. Each of these steps are outlined below.

3.1 Rule Extraction

To extract the set of simplification rules, we use a sentence-aligned data set of English Wikipedia sentences (referred to as *normal*) aligned to Simple English Wikipedia sentences (referred to as *simple*) (Coster and Kauchak, 2011). The data set contains 137K such aligned sentence pairs.

Given a normal sentence and the corresponding aligned simple sentence, candidate simplifications are extracted by identifying a word in the simple sentence that corresponds to a different word in the normal sentence. To identify such pairs, we automatically induce a word alignment between the normal and simple sentence pairs using GIZA++ (Och and Ney, 2000). Words that are aligned are considered as possible candidates for extraction. Due to errors in the sentence and word alignment processes, not all words that are aligned are actually equivalent lexical variants. We apply the following filters to reduce such spurious alignments:

- We remove any pairs where the normal word occurs in a stoplist. Stoplist words tend to be simple already and stoplist words that are being changed are likely either bad alignments or are not simplifications.
- We require that the part of speeches (POS) of the two words be the same. The parts of speech were calculated based on a full parse of the sentences using the Berkeley parser (Petrov and Klein, 2007).
- We remove any candidates where the POS is labeled as a proper noun. In most cases, proper nouns should not be simplified.

All other aligned word pairs are extracted. To generate the simplification rules, we collect all candidate simplifications (simple words) that are aligned to the same normal word.

As mentioned before, one of the biggest challenges for lexical simplification systems is generalizability. To improve the generalizability of the extracted rules, we add morphological variants of the words in the rules. For nouns, we include both singular and plural variants. For verbs, we expand to all inflection variants. The morphological changes are generated using MorphAdorner (Burns, 2013) and are applied symmetrically: any change to the normal word is also applied to the corresponding simplification candidates.

3.2 Lexical Simplification as a Ranking Problem

A lexical simplification example consists of three parts: 1) a sentence, $s_1, s_2, ..., s_n$, 2), a word in that sentence, s_i, and 3) a list of candidate simplifications for s_i, $c_1, c_2, ..., c_m$. A labeled example is an example where the rank of the candidate simplifications has been specified. Given a set of labeled examples, the goal is to learn a ranking function that, given an unlabeled example (example without the candidate simplifications ranked), specifies a ranking of the candidates.

To learn this function, features are extracted from a set of labeled lexical simplification examples. These labeled examples are then used to train a ranking function. We use SVM^{rank} (Joachims, 2006), which uses a linear support vector machine.

Besides deciding which of the candidates is most applicable in the context of the sentence, even if a rule applies, we must also decide if any simplification should occur. For example, there may be an instance where none of the candidate simplifications are appropriate in this context. Rather than viewing this as a separate problem, we incorporate this decision into the ranking problem by adding w as a candidate simplification. For each rule, $w \rightarrow c_1, c_2, ..., c_m$ we add one additional candidate simplification which does not change the sentence, $w \rightarrow c_1, c_2, ..., c_m, w$. If w is ranked as the most likely candidate by the ranking algorithm, then the word is not simplified.

3.2.1 Features

The role of the features is to capture information about the applicability of the word in the context of the sentence as well as the simplicity of the word. Many features have been suggested previously for use in determining the simplicity of a word (Specia et al., 2012) and for determining if a word is contextually relevant (Biran et al., 2011; McCarthy and Navigli, 2007). Our goal for this paper is not feature exploration, but to examine the usefulness of a general framework for feature-based ranking for lexical simplification. The features below represent a first pass at candidate features, but many others could be explored.

Candidate Probability
$p(c_i|w)$: in the sentence-aligned Wikipedia data, when w is aligned to some candidate simplification, what proportion of the time is that candidate c_i.

Frequency
The frequency of a word has been shown to correlate with the word's simplicity and with people's knowledge of that word (Leroy and Kauchak, 2013). We measured a candidate simplification's frequency in two corpora: 1) Simple English Wikipedia and 2) the web, as measured by the unigram frequency from the Google n-gram corpus (Brants and Franz, 2006).

Language Models
n-gram language models capture how likely a particular sequence is and can help identify candidate simplifications that are not appropriate in the context of the sentence. We included features from four different language models trained on four different corpora: 1) Simple English Wikipedia, 2) English Wikipedia, 3) Google n-gram corpus and 4) a linearly interpolated model between 1) and 2) with $\lambda = 0.5$, i.e. an even blending. We used the SRI language modeling toolkit (Stolcke, 2002) with Kneser-Kney smoothing. All models were trigram language models except the Google n-gram model, which was a 5-gram model.

Context Frequency
As another measure of the applicability of a candidate in the context of the sentence, we also calculate the frequency in the Google n-grams of the candidate simplification in the context of the sentence with context windows of one and two words. If the word to be substituted is at position i in the sentence ($w = s_i$), then the one word window frequency for simplification c_j is the trigram frequency of $s_{i-1}\ c_j\ s_{i+1}$ and the two word window the 5-gram frequency of $s_{i-2}\ s_{i-1}\ c_j\ s_{i+1}\ s_{i+2}$.

4 Data

For training and evaluation of the models, we collected human labelings of 500 lexical simplification examples using Amazon's Mechanical Turk (MTurk)[1]. MTurk has been used extensively for annotating and evaluating NLP tasks and has been shown to provide data that is as reliable as other forms of human annotation (Callison-Burch and Dredze, 2010; Zaidan and Callison-Burch, 2011).

Figure 1 shows an example of the task we asked annotators to do. Given a sentence and a word to be simplified, the task is to suggest a simpler variant of that word that is appropriate in the context of the sentence. Candidate sentences were se-

[1] https://www.mturk.com/

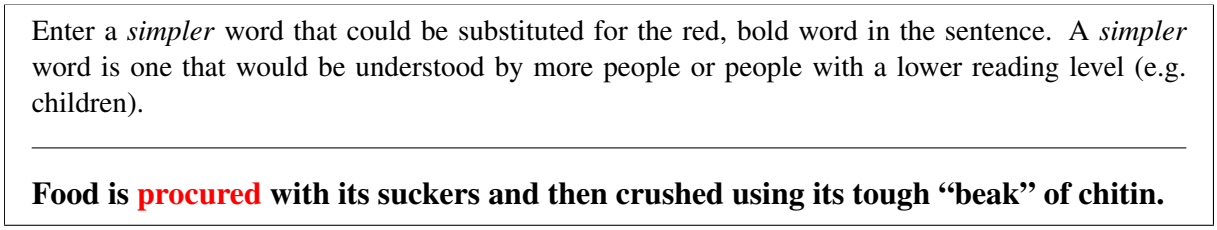

Figure 1: Example task setup on MTurk soliciting lexical simplifications from annotators.

lected from the sentence-aligned Wikipedia corpus where a word in the normal sentence is being simplified to a different word in the simple sentence, as identified by the automatically induced word alignment. The normal sentence and the aligned word were then selected for annotation. These examples represent words that other people (those that wrote/edited the Simple English Wikipedia page) decided were difficult and required simplification.

We randomly selected 500 such sentences and collected candidate simplifications from *50 people per sentence*, for a total of 25,000 annotations. To participate in the annotation process, we required that the MTurk workers live in the U.S. (for English proficiency) and had at least a 95% acceptance rate on previous tasks.

The simplifications suggested by the annotators were then tallied and the resulting list of simplifications with frequencies provides a ranking for training the candidate ranker. Table 2 shows the ranked list of annotations collected for the example in Figure 1. This data set is available online.[2]

Since these examples were selected from English Wikipedia they, and the corresponding aligned Simple English Wikipedia sentences, were removed from *all* resources used during both the rule extraction and the training of the ranker.

5 Experiments

5.1 Other Approaches

We compared our lexical simplification approach (**rank-simplify**) to two other approaches. To understand the benefit of the feature-based ranking algorithm, we compared against a simplifier that uses the same rule set, but ranks the candidates only based on their frequency in Simple English Wikipedia (**frequency**). This is similar to baselines used in previous work (Biran et al., 2011).

To understand how our extracted rules compared to the rules extracted by Biran et al., we used their rules with *our* ranking approach (**rank-Biran**). Their approach also extracts rules from a corpus of English Wikipedia and Simple English Wikipedia, however, they do not utilize a sentence-aligned version and instead rely on context similarity measures to extract their rules.

5.2 Evaluation

We used the 500 ranked simplification examples to train and evaluate our approach. We employed 10-fold cross validation for all experiments, training on 450 examples and testing on 50.

We evaluated the models with four different metrics:

precision: Of the words that the system changed, what percentage were found in *any* of the human annotations.

precision@k: Of the words that the system changed, what percentage were found in the top k human annotations, where the annotations were ranked by response frequency. For example, if we were calculating the precision@1 for the example in Table 2, only "obtained" would be considered correct.

accuracy: The percentage of the test examples where the system made a change to one of the annotations suggested by the human annotators. Note that unlike precision, if the system does not suggest a change to a word that was simplified it still gets penalized.

changed: The percentage of the test examples where the system suggested some change (even if it wasn't a "correct" change).

5.3 Results

Table 3 shows the precision, accuracy and percent changed for the three systems. Based on all three metrics, our system achieves the best results. Although the rules generated by Biran et al. have reasonable precision, they suffer from a lack of coverage, only making changes on about 5% of the

[2] http://www.cs.middlebury.edu/~dkauchak/simplification/

word	frequency	word	frequency	word	frequency
obtained	17	made	2	secured	1
gathered	9	created	1	found	1
gotten	8	processed	1	attained	1
grabbed	4	received	1	procured	1
acquired	2	collected	1	aquired	1

Table 2: Candidate simplifications generated using MTurk for the examples in Figure 1. The frequency is the number of annotators that suggested that simplification.

	precision	accuracy	changed
frequency	53.9%	46.1%	84.9%
rank-Biran	71.4%	3.4%	5.2%
rank-simplify	76.1%	66.3%	86.3%

Table 3: Precision, accuracy and percent changed for the three systems, averaged over the 10 folds.

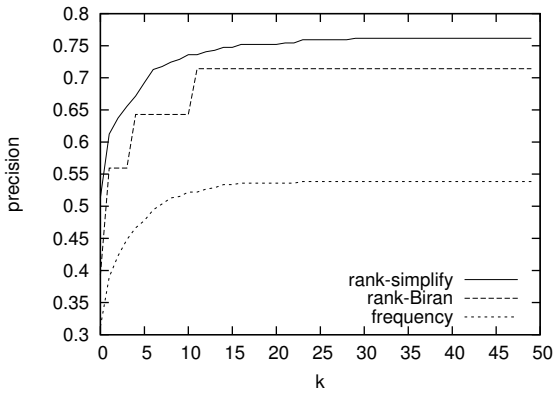

Figure 2: Precision@k for varying k for the three different approaches averaged over the 10 folds.

examples. For our approach, the extracted rules had very good coverage, applying in over 85% of the examples.

This difference in coverage can be partially attributed to the number of rules learned. We learned simplifications for 14,478 words with an average of 2.25 candidate simplifications per word. In contrast, the rules from Biran et al. only had simplifications for 3,598 words with an average of 1.18 simplifications per word.

The precision of both of the approaches that utilized the SVM candidate ranking were significantly better than the frequency-based approach. To better understand the types of suggestions made by the systems, Figure 2 shows the precision@k for increasing k. On average, over the 500 examples we collected, people suggested 12 different simplifications, though this varied depending on the word in question and the sentence. As such, at around k=12, the precision@k of most of the systems has almost reached the final precision. However, even at $k = 5$, which only counts correct an answer in the top 5 human suggested results, our system still achieved a precision of around 67%.

6 Future Work

In this paper we have introduced a new rule extraction algorithm and a new feature-based ranking approach for applying these rules in the context of different sentences. The number of rules learned is an order of magnitude larger than any previous lexical simplification approach and the quality of the resulting simplifications after applying these rules is better than previous approaches.

Many avenues exist for improvement and for better understanding how well the current approach works. First, we have only explored a small set of possible features in the ranking algorithm. Additional improvements could be seen by incorporating a broader feature set. Second, more analysis needs to be done to understand the quality of the produced simplifications and their impact on the simplicity of the resulting sentences. Third, the experiments above assume that the word to be simplified has already been identified in the sentence. This identification step also needs to be explored to implement a sentence-level simplifier using our approach. Fourth, the ranking algorithm can be applied to most simplification rules (e.g. we applied the ranking approach to the rules obtained by Biran et al. (2011)). We hope to explore other approaches for increasing the rule set by incorporating other rule sources and other rule extraction techniques.

References

Or Biran, Samuel Brody, and Noemie Elhadad. 2011. Putting it simply: A context-aware approach to lexical simplification. In *Proceedings of ACL*.

Thorsten Brants and Alex Franz. 2006. Web 1T 5-gram version 1. Linguistic Data Consortium, Philadelphia.

Philip R. Burns. 2013. Morphadorner v2: A Java library for the morphological adornment of english language texts.

Chris Callison-Burch and Mark Dredze. 2010. Creating speech and language data with Amazon's Mechanical Turk. In *Proceedings of NAACL-HLT 2010 Workshop on Creating Speech and Language Data with Amazon's Mechanical Turk*.

Raman Chandrasekar and Bangalore Srinivas. 1997. Automatic induction of rules for text simplification. In *Knowledge Based Systems*.

William Coster and David Kauchak. 2011. Simple English Wikipedia: A new text simplification task. In *Proceedings of ACL*.

Lijun Feng. 2008. Text simplification: A survey. CUNY Technical Report.

Thorsten Joachims. 2006. Training linear svms in linear time. In *Proceedings of KDD*.

Gondy Leroy and David Kauchak. 2013. The effect of word familiarity on actual and perceived text difficulty. *Journal of American Medical Informatics Association*.

Gondy Leroy, James E. Endicott, David Kauchak, Obay Mouradi, and Melissa Just. 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 (JMIR)*.

Diana McCarthy and Roberto Navigli. 2007. Semeval-2007 task 10: English lexical substitution task. In *Proceedings of SEMEVAL*.

F. J. Och and H. Ney. 2000. Improved statistical alignment models. In *ACL*.

Slav Petrov and Dan Klein. 2007. Improved inference for unlexicalized parsing. In *Proceedings of HTL-NAACL*.

Lucia Specia, Sujay Kumar Jauhar, and Rada Mihalcea. 2012. Semeval-2012 task 1: English lexical simplification. In *Joint Conference on Lexical and Computerational Semantics (*SEM)*.

Andreas Stolcke. 2002. SRILM - an extensible language modeling toolkit. In *Proceedings of the International Conference on Statistical Language Processing*.

Mark Yatskar, Bo Pang, Cristian Danescu-Niculescu-Mizil, and Lillian Lee. 2010. For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In *NAACL/HLT*.

Omar F. Zaidan and Chris Callison-Burch. 2011. Crowdsourcing translation: Professional quality from non-professionals. In *Proceedings of ACL*.

Zhemin Zhu, Delphine Bernhard, and Iryna Gurevych. 2010. A monolingual tree-based translation model for sentence simplification. In *Proceedings of ICCL*.