Unsupervised Lexical Simplification for Non-Native Speakers

Gustavo H. Paetzold and **Lucia Specia**
University of Sheffield
Western Bank, South Yorkshire S10 2TN
Sheffield, United Kingdom

Abstract

Lexical Simplification is the task of replacing complex words with simpler alternatives. We propose a novel, unsupervised approach for the task. It relies on two resources: a corpus of subtitles and a new type of word embeddings model that accounts for the ambiguity of words. We compare the performance of our approach and many others over a new evaluation dataset, which accounts for the simplification needs of 400 non-native English speakers. The experiments show that our approach outperforms state-of-the-art work in Lexical Simplification.

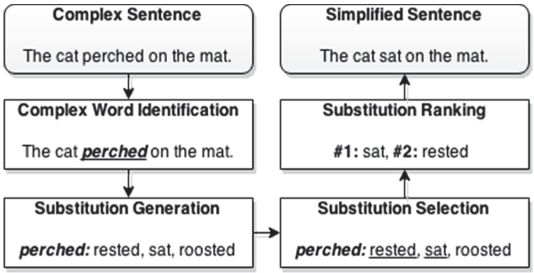
Complex Sentence
The cat perched on the mat.
Complex Word Identification
The cat *perched* on the mat.
Substitution Generation
perched: rested, sat, roosted
Substitution Selection
perched: rested, sat, roosted
Substitution Ranking
#1: sat, #2: rested
Simplified Sentence
The cat sat on the mat.

Figure 1: Lexical Simplification Pipeline

1 Introduction

Vocabulary acquisition is a process inherent to human language learning that determines the rate at which an individual becomes familiarised with the lexicon of a given language. Word recognition, however, is described as a series of linguistic sub-processes that establishes one's capability of identifying and comprehending individual words in a text. It has been shown that individuals with low-literacy levels or who suffer from certain clinical conditions, such as Dyslexia (Ellis 2014), Aphasia (Devlin and Tait 1998) and some forms of Autism (Barbu et al. 2015), can face impairments in either or both processes, often hindering them incapable of recognising and/or understanding the meaning of texts. Impairments that cause the narrowing of one's vocabulary can be severely crippling: the results obtained by (Hirsh, Nation, and others 1992) show that one must be familiar with at least 95% of the vocabulary of a text in order to understand it, and 98% to read it for leisure.

Lexical Simplification (LS) aims to address this problem by replacing words that may challenge a certain target audience with simpler alternatives. It was first introduced in the work of (Devlin and Tait 1998), who inspired further research. The biggest challenge in LS is performing replacements without compromising the grammaticality or changing the meaning of the sentence being simplified.

Most strategies in the literature take LS as the series of cognitive processes illustrated by the pipeline in Figure 1. By following this model, the performance of LS systems has considerably increased in recent years. The approach of (Horn, Manduca, and Kauchak 2014) offers an improvement of 62.9% in accuracy over the earlier work of (Biran, Brody, and Elhadad 2011). However, most recent work is limited to exploiting linguistic resources that are scarce and/or expensive to produce, such as WordNets and Simple Wikipedia. In this paper, we describe an LS approach that focuses on the simplification needs of non-native English speakers. We propose an unsupervised strategy for Substitution Generation, Selection and Ranking. Instead of relying on complex and expensive resources, our approach uses a new context-aware model for word embeddings, which can be easily trained over large corpora, as well as n-gram frequencies extracted from a corpus of movie subtitles. We also introduce a new domain-specific dataset for the task, which accounts for the simplification needs of non-native English speakers. We evaluate the performance of our approaches for each step of the pipeline both individually and jointly, comparing them to several other approaches in the literature.

2 Complex Word Identification

Complex Word Identification (CWI) is the task of determining which words in a text should be simplified, given the needs of a certain target audience. It is commonly performed before any simplification occurs, and aims to prevent an LS system from making unnecessary substitutions. Most existing work, however, do not provide an explicit solution to CWI, and instead model it implicitly (Biran, Brody, and Elhadad 2011; Horn, Manduca, and Kauchak 2014; Glavaš and Štajner 2015). In order to address CWI and still be able to compare our LS approach to others, we have chosen to create a dataset that accounts for the simplification needs of non-native English speakers.

In previous work focusing on the evaluation of LS systems, (De Belder and Moens 2012) and (Horn, Manduca, and Kauchak 2014) introduce the LSeval and LexMTurk datasets. The instances in both datasets, 930 total, are composed of a sentence, a target word, and candidate substitutions ranked by simplicity. Using different metrics, one is able to evaluate each step of the LS pipeline over these datasets. There is, however, no way of knowing the profile of the annotators who produced these datasets. In both of them, the candidate substitutions were suggested and ranked by English speakers from the U.S., who are unlikely to be non-native speakers of English in their majority. This limitation renders these datasets unsuitable for the evaluation of our approach because i) the target words used may not be considered complex by non-native speakers ii) the candidate substitutions suggested may be deemed complex by non-native speakers. In order to reuse these resources and create a more reliable dataset, we have conducted a user study to learn more about word complexity for non-native speakers.

2.1 A User Study on Word Complexity

400 non-native speakers participated in the experiment, all university students or staff. They were asked to judge whether or not they could understand the meaning of each content word (nouns, verbs, adjectives and adverbs, as tagged by Freeling (Padr and Stanilovsky 2012)) in a set of sentences, each of which was judged independently. Volunteers were instructed to annotate all words that they could not understand individually, even if they could comprehend the meaning of the sentence as a whole.

All sentences used were taken from Wikipedia, LSeval and LexMTurk. A total of $35,958$ distinct words from $9,200$ sentences were annotated ($232,481$ total), of which $3,854$ distinct words ($6,388$ total) were deemed as complex by at least one annotator.

2.2 A Dataset for Lexical Simplification

Using the data produced in the user study, we first assessed reliability of the LSeval and LexMTurk datasets in evaluating LS systems for non-native speakers. We found that the proportion of target words deemed complex by at least one annotator was only 30.8% for LexMTurk, and 15% for LSeval. As for the candidate substitutions, 21.7% of the ones in LSeval and 13.4% in LexMTurk were deemed complex by at least one annotator.

These results show that, although they may not be used in their entirety, both datasets contain instances that are suitable for our purposes. To create our dataset, we first used the Text Adorning module of LEXenstein (Paetzold and Specia 2015; Burns 2013) to inflect all candidate verbs and nouns in both datasets to the same tense as the target word. We then used the Spelling Correction module of LEXenstein to correct any misspelled words among the candidates of both datasets. Next, we removed all candidate substitutes which were deemed complex by at least one annotator in our user study. Finally, we discarded all instances in which the target word was not deemed complex by any of our annotators. The resulting dataset, which we refer to as NNSeval, contains 239 instances.

3 Substitution Generation

The goal of Substitution Generation (SG) is to generate candidate substitutions for complex words. Most LS approaches in the literature do so by extracting synonyms, hypernyms and paraphrases from thesauri (Devlin and Tait 1998; De Belder and Moens 2010; Bott et al. 2012). The generators described in (Paetzold and Specia 2013; Paetzold 2013) and (Horn, Manduca, and Kauchak 2014) do not use thesauri, but instead extract candidate substitutions from aligned parallel sentences from Wikipedia in Simple Wikipedia. Although parallel corpora can be produced automatically, they still offer limited coverage of complex words, and can thus limit the potential of a simplifier.

The recent work of (Glavaš and Štajner 2015) aims to address these limitations by exploiting word embedding models (Mikolov et al. 2013), which require only large corpora to be produced. Given a target word, they extract the 10 words from the model for which the embedding vectors have the highest cosine similarity to the one of the complex word itself. Traditional embedding models suffer, however, from a very severe limitation: they do not accommodate ambiguous words' meanings. In other words, all possible meanings of a word are represented by a single numerical vector. We propose a new type of embeddings model that addresses this limitation.

3.1 Context-Aware Word Embedding Models

In order to learn context-aware word embeddings, we resort to annotating the training corpus from which the model is learned. If one is able to assign a sense label to all words in the training corpus, then a distinct numerical vector would be assigned to each sense of a word. But as shown by (Navigli 2009), state-of-the-art Word Sense Disambiguation (WSD) systems make mistakes more than 25% of the time, are language dependent and can be quite slow to run. Considering that the training corpora used often contain billions of words, this strategy becomes impractical.

We instead compromise by using Part-Of-Speech (POS) tags as surrogates for sense labels. Although they do not convey the same degree of information, they account for some of the ambiguity inherent to words which can take multiple grammatical forms. Annotating the corpus with raw POS tags in Treebank format (Marcus, Marcinkiewicz, and Santorini 1993), however, could introduce a lot of sparsity to our model, since it would generate a different tag for each inflection of nouns and verbs, for example. To avoid sparsity, we generalise all tags related to nouns, verbs, adjectives and adverbs to N, V, J and R, respectively.

Once the words in the training corpus are annotated with their generalised POS tags, the model can be trained with any tool available, such as word2vec[1] or GloVe[2].

[1] https://code.google.com/p/word2vec/

[2] http://nlp.stanford.edu/projects/glove/

3.2 Candidate Generation Algorithm

Given a target word in a sentence, its POS tag, and a context-aware embeddings model, our generator extracts as candidates the n words in the model with the shortest cosine distance from the target word that satisfy the following constraints:

1. The word must share the same POS tag as the target word.
2. The word must not be a morphological variant of the target word.

These constraints are designed to filter ungrammatical and spurious candidate substitutions. In order to find whether or not a candidate is a morphological variant of the target word, use extract their stem and lemma using the Text Adorning module of LEXenstein, and verify if any of them are identical.

4 Substitution Selection

The step of Substitution Selection (SS) is responsible for deciding which of the generated substitutions can replace a target word in a given context. While some of existing work employs WSD strategies to address this task explicitly (Nunes et al. 2013; Baeza-Yates, Rello, and Dembowski 2015), others choose to address it implicitly, by joint modeling SS and Substitution Ranking (SR) (Horn, Manduca, and Kauchak 2014; Glavaš and Štajner 2015).

As discussed in the previous Section, WSD systems often suffer from low accuracy, and can hence compromise the performance of a simplifier. Most of them also depend on the synonym relations of a thesauri to work, which can severely compromise the potential of the substitution generator being applicable. If the generator produces a candidate that is not registered in the thesauri, it will not able to decide whether or not it is a synonym of a target complex word, which would force the candidate to be discarded. Although the results reported by (Horn, Manduca, and Kauchak 2014) and (Glavaš and Štajner 2015) show that joint modeling selection and ranking can be a viable solution, we believe that it can limit the performance of an LS approach.

We hypothesize that a dedicated SS approach would more efficiently capture the intricacies of grammaticality and meaning preservation, and consequently allow for the next step to model word simplicity more effectively. We hence take SS as a ranking task itself, and assume that all candidates have a likelihood of fitting the context in which a target word is placed. In order to propose an unsupervised setup for the task, we first introduce the technique of Boundary Ranking.

4.1 Boundary Ranking

The goal of a boundary ranker is to, given a set of example ranking instances and a feature space, learn the direction in which the ranking property grows on the feature space provided. To do so, a decision *boundary* must be learned from a binary classification setup inferred from the ranking examples. Consider Example 1, in which four words are ranked by simplicity.

$$1\!:\!\text{sat},\ 2\!:\!\text{rested},\ 3\!:\!\text{roosted},\ 4\!:\!\text{perched} \qquad (1)$$

A boundary ranker will produce binary classification training instances from the example above based on a parameter p, which determines the maximum positive ranking position. If $p=2$, then *sat* and *rested* will receive label 1, while the remaining words will receive label 0. Once this process is applied to all example rankings available, any linear or non-linear model can be fitted to the data in order to learn a decision boundary between positive and negative examples. Finally, an unseen set of words can be ranked according to their distance from the boundary: the furthest they are from the negative portion of the data, the higher their ranking.

But notice that Boundary Ranking is an inherently supervised approach: it learns a model from ranking examples. In order to train a selector in an unsupervised fashion, we resort to the Robbins-Sturgeon hypothesis.

4.2 The Robbins-Sturgeon Hypothesis

In Jitterbug Perfume (Robbins 2003), author Tom Robbins states that “There are no such things as synonyms! He practically shouted. Deluge is not the same as flood”. A similar statement was made by Theodore Sturgeon, another acclaimed book author, during an interview: “Here’s the point to be made - there are no synonyms. There are no two words that mean exactly the same thing.”. As a summary of these quotes, the Robbins-Sturgeon hypothesis states that a word is irreplaceable.

Although modern LS approaches (Horn, Manduca, and Kauchak 2014; Glavaš and Štajner 2015) show that exploring synonymy relations is very useful in making texts easier to read, the Robbins-Sturgeon hypothesis can be used to learn a boundary ranker over unannotated data. If we take this hypothesis to be correct, we can assume that a given target complex word is the only word suitable to replace itself. In the binary classification setup of a boundary ranker, this would mean that the only candidate substitution which would receive label 1 is the target word itself, while any other candidates would receive label 0. With these settings, we would not require annotated data: the candidates to receive label 0 could be automatically produced by a substitution generator, allowing the training of the ranker over unannotated data, and hence unsupervised SS.

5 Substitution Ranking

During Substitution Ranking (SR), the selected candidate substitutions are ranked according to their simplicity. Recently, sophisticated strategies have been introduced for this task, ranging from Support Vector Machines (Jauhar and Specia 2012; Horn, Manduca, and Kauchak 2014), hand-crafted metrics (Biran, Brody, and Elhadad 2011; Bott et al. 2012), and rank averaging (Glavaš and Štajner 2015). Nonetheless, the most popular SR strategy in the literature is frequency ranking: the more frequently a word appears in a corpus, the simpler it is (Devlin and Tait 1998; De Belder and Moens 2010; Leroy et al. 2013). Modern LS approaches also go a step further and account for a word’s context by using not word but n-gram frequencies (Baeza-Yates, Rello, and Dembowski 2015), achieving state-of-the-art simplification results.

Most work assume that the quality of the word and n-gram frequencies produced depend more on the size of the corpus used, rather than on the domain from which it was extracted. Because of this assumption, the most frequently used corpus is the Google 1T, which is not freely available and is composed of over 1 trillion words. (Brysbaert and New 2009), however, has shown that the raw frequencies extracted from movie subtitles capture word familiarity more effectively than the ones extracted from other corpora. Their subtitle corpus was also evaluated on the Lexical Simplification task of SemEval 2012 by (Shardlow 2014). It helped (Shardlow 2014) achieve scores comparable to those obtained by Google 1T, although it is more than four orders of magnitude smaller. These results are very encouraging for our purposes, since the gold-standard used was also produced by non-native English speakers. Inspired by these observations, we have compiled a new corpus for SR.

5.1 Compiling a Corpus of Subtitles

We hypothesize that the type of content from which the subtitles are extracted can also affect the quality of the word frequencies produced. We believe that movies targeting children or young adults, for an example, use a more accessible language than movies targeting older audiences.

In order to test this hypothesis, we exploit the facilities provided by IMDb[3] and OpenSubtitles[4]. While IMDb offers an extensive database of ID-coded and categorised movies and series, OpenSubtitles allows for one to use these IDs to query for subtitles in various languages. IMDb also allows users to create their own lists, and hence help other users with similar taste to find new movies and series to watch.

We compiled our corpus by first parsing 15 lists of movies for children, as well as the pages of all movies and series under the "family" and "comedy" categories. A total of $12,037$ IMDb IDs were gathered. We then queried each ID found in OpenSubtitles, and downloaded one subtitle for each movie. For series, we have downloaded one subtitle for each episode of every season available. All subtitles were parsed, and a corpus of $145,350,077$ words produced. We refer to it as SubIMDB.

6 Experiments

In the following Sections, we describe each of the experiments conducted with our LS approach, which we refer henceforth to as LS-NNS. All other approaches hereon mentioned were replicated to the best of our ability.

6.1 Substitution Generation

We compare ours to five other SG systems:

- **Devlin** (Devlin and Tait 1998): One of the most frequently used SG strategies in the literature, it generates candidates by extracting synonyms from WordNet.
- **Biran** (Biran, Brody, and Elhadad 2011): Extracts candidates from the Cartesian product pairs between the words in Wikipedia and Simple Wikipedia. Any pairs of words that are not registered as synonyms or hypernyms in WordNet are discarded.
- **Yamamoto** (Kajiwara, Matsumoto, and Yamamoto 2013): Queries dictionaries for target words, retrieving definition sentences, and then extracts any words that share the same POS tag as the target word. For this approach, we use the Merriam Dictionary[5].
- **Horn** (Horn, Manduca, and Kauchak 2014): Extracts word correspondences from aligned complex-to-simple parallel corpora. For this system, we use the parallel Wikipedia and Simple Wikipedia corpus provided by (Horn, Manduca, and Kauchak 2014).
- **Glavas** (Glavaš and Štajner 2015): Extracts candidates using a typical word embeddings model. For each target complex word, they retrieve the 10 words for which the embeddings vector has the highest cosine similarity with the target word, except for their morphological variants.

Like the Glavas generator, ours selects 10 candidates for each target word. We use the word2vec toolkit to train our word embeddings model. The corpus used contains 7 billion words, and includes the SubIMDB corpus, UMBC webbase[6], News Crawl[7], SUBTLEX (Brysbaert and New 2009), Wikipedia and Simple Wikipedia (Kauchak 2013). To tag our corpus, we use the Stanford Parser (Klein and Manning 2003), which offers over 97% accuracy in consolidated datasets. For training, we use the bag-of-words model (CBOW), and 500 dimensions for the embedding vectors. We use the same resources and parameters to train the model for the Glavas generator.

For evaluation, we use the the following four metrics over the NNSeval dataset:

- **Potential:** The proportion of instances for which at least one of the substitutions generated is present in the gold-standard.
- **Precision:** The proportion of generated substitutions that are present 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 obtained, which are illustrated in Table 1, reveal that our generator is more effective than all other approaches evaluated. They also highlight the potential of context-aware word embedding models: they offer a 2.4% F1 improvement over the traditional embeddings model used by the Glavas generator.

6.2 Substitution Selection

A Substitution Selection system requires a set of candidate substitutions to select from. For that purpose, we use the

[3] http://www.imdb.com/

[4] http://www.opensubtitles.org/

[5] http://www.merriam-webster.com/

[6] http://ebiquity.umbc.edu/resource/html/id/351

[7] http://www.statmt.org/wmt11/translation-task.html

	Potential	Precision	Recall	F1
Yamamoto	0.314	0.026	0.061	0.037
Biran	0.414	0.084	0.079	0.081
Devlin	0.485	0.092	0.093	0.092
Horn	0.464	0.134	0.088	0.106
Glavas	0.661	0.105	0.141	0.121
LS-NNS	**0.699**	**0.118**	**0.161**	**0.136**

Table 1: Substitution Generation evaluation results

candidate substitutions produced by the highest performing generator from the previous experiment, which is LS-NNS. We compare ours to five other SS approaches:

- **No Selection**: As a baseline, we consider the approach of not performing selection at all.
- **Lesk** (Lesk 1986): Uses the Lesk algorithm to select the word sense in WordNet that best describes a given target word with respect to its context.
- **Leacock** (Leacock and Chodorow 1998): Uses a more sophisticated interpretation of the Lesk algorithm. It takes into account not only the overlap between a target words' context and sense examples in WordNet, but also their semantic distance.
- **Belder** (De Belder and Moens 2010): Candidate substitutions are filtered with respect to the classes learned by a latent-variable language model. For this approach, we use the algorithm proposed by (Brown et al. 1992), which learns word clusters from large corpora. We learn a total of $2,000$ word clusters.
- **Biran** (Biran, Brody, and Elhadad 2011): Filters candidate substitutions with respect to the cosine distance between the word co-occurrence vectors of a target word and a candidate substitution. We use a lower-bound of 0.1 and an upper-bound of 0.8. The corpus used to obtain the co-occurrence model is the same used in the training of the word embedding models used by our SG approach.

As discussed in a previous Section, the boundary ranker used by our approach requires training instances with binary labels. Using the Robbins-Sturgeon hypothesis, we create training instances by assuming a maximum positive ranking position $p = 1$, i.e., we assign label 1 to all target words in the NNSeval dataset, and 0 to all the generated candidates. We use seven features to train the model:

- Language model log-probabilities of the following five n-grams: $s_{i-1}c$, cs_{i+1}, $s_{i-1}cs_{i+1}$, $s_{i-2}s_{i-1}c$ and $cs_{i+1}s_{i+2}$, where c is a candidate substitution, and i the position of the target word in sentence s. We use a 5-gram language model trained over SubIMDB with SRILM (Stolcke 2002).
- The word embeddings cosine similarity between the target complex word and a candidate. For this feature, we employ the same context-aware embeddings model used in the SG experiment.
- The conditional probability of a candidate given the POS tag of the target word. To calculate this feature, we learn the probability distribution $P(c|p_t)$, described in Equation 2, of all words in the corpus used to train our context-aware word embeddings model.

$$P(c|p_t) = \frac{C(c, p_t)}{\sum_{p \in P} C(c, p)}, \tag{2}$$

where c is a candidate, p_t is the POS tag of the target word, $C(c, p)$ the number of times c received tag p in the training corpus, and P the set of all POS tags.

For each instance of the evaluation dataset, we discard the 50% lowest ranked candidate substitutions. We learn the decision boundary through Stochastic Gradient Descent with 10-fold cross validation. For evaluation, we use the same dataset and metrics described in the previous experiment.

The results (Table 2) show that our approach was the only one to obtain higher F1 scores than those achieved by not performing selection at all. This reveals that SS is a very challenging task, and that using an unreliable approach can considerably decrease the effectiveness of the SG used.

	Potential	Precision	Recall	F1
No Selection	0.699	0.118	0.161	0.136
Lesk	0.176	0.060	0.026	0.037
Leacock	0.013	0.011	0.002	0.003
Belder	0.247	**0.201**	0.034	0.058
Biran	0.322	0.122	0.068	0.087
LS-NNS	**0.644**	0.192	**0.131**	**0.156**

Table 2: Substitution Selection evaluation results

6.3 Substitution Ranking

In this experiment, we evaluate the potential of our corpus (SubIMDB) in SR alone. To do so, we first train 5-gram language models over SubIMDB and four other corpora:

- **Wikipedia** (Kauchak 2013): Composed of $97,912,818$ words taken from Wikipedia.
- **Simple Wiki** (Kauchak 2013): Composed of $9,175,446$ words taken from Simple Wikipedia.
- **Brown** (Francis and Kucera 1979): Composed of $1,182,211$ words of edited English prose produced in 1961.
- **SUBTLEX** (Brysbaert and New 2009): Composed of $62,504,269$ words taken from assorted subtitles.

We then rank candidates by their unigram probabilities in each language model. The evaluation dataset used is the one provided for the English Lexical Simplification task of SemEval 2012 (Specia, Jauhar, and Mihalcea 2012), composed of 300 training and $1,710$ test instances. We choose this dataset as opposed to NNSeval because it has also been annotated by non-native speakers and it allows a more meaningful comparison, given that 12 systems have already been tested on this dataset as part of SemEval 2012. Each instance is composed of a sentence, a target word, and candidate substitutions ranked in order of simplicity by non-native speakers. The evaluation metric used is TRank, which measures the ratio with which a given system has correctly ranked at least one of the highest ranked substitutions on the gold-standard. As discussed in (Paetzold 2015), this metric is the one which best represents the performance of a ranker in practice.

The results obtained are illustrated in Table 3. For completeness, we also include in our comparison the best performing approach of SemEval 2012, as well as the baseline, in which candidates are ranked according to their raw frequencies in Google 1T.

	TRank
Wikipedia	0.519
Simple Wiki	0.570
Brown	0.596
SUBTLEX	0.618
Google 1T	0.585
Best SemEval	0.602
SubIMDB	**0.627**

Table 3: Substitution Ranking evaluation results

Our findings show that frequencies from subtitles have a higher correlation with simplicity than the ones extracted from other sources. Our corpus outperformed all others, including the best approach in SemEval 2012, by a considerable margin. We have also found evidence that domain is more important than size: the Brown corpus, composed of 1 million words, outperformed the Google 1T corpus, composed of 1 trillion words.

6.4 Round-Trip Evaluation

Finally, we assess the performance of our LS approach in its entirety. We compare it to three modern LS systems:

- **Biran** (Biran, Brody, and Elhadad 2011): Combines the SG and SS approaches described in Sections 6.1 and 6.2 with a metric-based ranker. Their metric is illustrated in Equation 3, where $F(c, C)$ is the frequency of candidate c in corpus C, and $||c||$ the length of candidate c.

$$M(c) = \frac{F(c, \text{Wikipedia})}{F(c, \text{Simple Wikipedia})} \times ||c|| \qquad (3)$$

- **Kauchak** (Horn, Manduca, and Kauchak 2014): Combines the generator of Section 6.1 with an SVM ranker (Joachims 2002) that joint models SS and SR. They train their approach on the LexMTurk dataset, and use as features the translation probability between a candidate and the target word, as determined by an alignment model learned over a simple-complex parallel corpus, as well as n-gram frequencies from various corpora.
- **Glavas** (Glavaš and Štajner 2015): Combines the generator of Section 6.1 with a ranker that also joint models SS and SR. It ranks candidates by averaging the rankings obtained with several features. As features they use n-gram frequencies, the cosine similarity between the target word and a candidate, as determined by a typical word embeddings model, as well as the average cosine similarity between the candidate and all content words in the target word's sentence.

For evaluation, we use the the following metrics over the NNSeval dataset:

- **Precision**: The proportion of instances in which the target word was replaced with any of the candidates in the dataset, including the target word itself.
- **Accuracy**: The proportion of instances in which the target word was replaced with any of the candidates in the dataset, except for the target word itself.
- **Changed Proportion**: The proportion of times in which the target word was replaced with a different word.

We train our generation and selection approaches with the same settings used in the previous experiments. For Substitution Ranking we use 5-gram probabilities with two tokens to the left and right of the candidate. This way, we account for context during Substitution Ranking. The results obtained are illustrated in Table 4. They reveal that our approach is the most effective Lexical Simplification solution for non-native English speakers.

	Precision	Accuracy	Changed
Biran	0.121	0.121	**1.000**
Kauchak	0.364	0.172	0.808
Glavas	0.456	0.197	0.741
LS-NNS	**0.464**	**0.226**	0.762

Table 4: Round-trip evaluation results

7 Conclusions

We have proposed a new, unsupervised Lexical Simplification approach. It relies in two resources: a context-aware word embeddings model and a corpus of subtitles, both of which can be easily obtained for multiple languages. We have also introduced NNSeval, a new dataset for the evaluation of LS systems which targets the simplification needs of non-native English speakers. In our experiments, we compare our strategies to several others, and show that ours are the most effective solutions available for Substitution Generation, Selection and Ranking.

In the future, we intend to create lexicon retrofitted context-aware embedding models, explore more sophisticated unsupervised SR solutions, conduct new user studies with non-native English speakers, and investigate whether or not the word frequencies from SubIMDB are capable of capturing elaborate psycholinguistic properties, such as Age of Acquisition and Familiarity.

All methods and resources used in this paper are available in the LEXenstein framework[8].

References

Baeza-Yates, R.; Rello, L.; and Dembowski, J. 2015. Cassa: A context-aware synonym simplification algorithm. In *Proceedings of the 14th NAACL*, 1380–1385.

[8] http://ghpaetzold.github.io/LEXenstein/

Barbu, E.; Martín-Valdivia, M. T.; Martínez-Cámara, E.; and Ureña-López, L. A. 2015. Language technologies applied to document simplification for helping autistic people. *Expert Systems with Applications* 42:5076–5086.

Biran, O.; Brody, S.; and Elhadad, N. 2011. Putting it simply: A context-aware approach to lexical simplification. In *Proceedings of the 49th ACL*, 496–501.

Bott, S.; Rello, L.; Drndarevic, B.; and Saggion, H. 2012. Can spanish be simpler? lexsis: Lexical simplification for spanish. In *Proceedings of the 2012 COLING*, 357–374.

Brown, P. F.; deSouza, P. V.; Mercer, R. L.; Pietra, V. J. D.; and Lai, J. C. 1992. Class-based n-gram models of natural language. *Computational Linguistics* 18:467–479.

Brysbaert, M., and New, B. 2009. Moving beyond kučera and francis: A critical evaluation of current word frequency norms and the introduction of a new and improved word frequency measure for american english. *Behavior Research Methods* 41(7):977–990.

Burns, P. R. 2013. Morphadorner v2: A java library for the morphological adornment of english language texts. *Northwestern University, Evanston, IL*.

De Belder, J., and Moens, M.-F. 2010. Text simplification for children. In *Proceedings of the SIGIR Workshop on Accessible Search Systems*, 19–26.

De Belder, J., and Moens, M.-F. 2012. A dataset for the evaluation of lexical simplification. In *Computational Linguistics and Intelligent Text Processing*. Springer. 426–437.

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

Ellis, A. W. 2014. *Reading, Writing and Dyslexia: A Cognitive Analysis*. Psychology Press.

Francis, W. N., and Kucera, H. 1979. Brown corpus manual. *Brown University*.

Glavaš, G., and Štajner, S. 2015. Simplifying lexical simplification: Do we need simplified corpora? In *Proceedings of the 53rd ACL*, 63–68.

Hirsh, D.; Nation, P.; et al. 1992. What vocabulary size is needed to read unsimplified texts for pleasure? *Reading in a Foreign Language* 8:689–689.

Horn, C.; Manduca, C.; and Kauchak, D. 2014. Learning a Lexical Simplifier Using Wikipedia. In *Proceedings of the 52nd ACL*, 458–463.

Jauhar, S. K., and Specia, L. 2012. Uow-shef: Simplex–lexical simplicity ranking based on contextual and psycholinguistic features. In *Proceedings of the 6th SemEval*, 477–481.

Joachims, T. 2002. Optimizing search engines using clickthrough data. In *Proceedings of the 8th ACM*, 133–142.

Kajiwara, T.; Matsumoto, H.; and Yamamoto, K. 2013. Selecting Proper Lexical Paraphrase for Children. In *Proceedings of the 25th ROCLING*, 59–73.

Kauchak, D. 2013. Improving text simplification language modeling using unsimplified text data. In *Proceedings of the 51st ACL*, 1537–1546.

Klein, D., and Manning, C. D. 2003. Accurate unlexicalized parsing. In *Proceedings of the 41st ACL*, 423–430.

Leacock, C., and Chodorow, M. 1998. Combining local context and wordnet similarity for word sense identification. *WordNet: An Electronic Lexical Database* 49(2):265–283.

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.

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 Conference on Systems Documentation*, 24–26.

Marcus, M. P.; Marcinkiewicz, M. A.; and Santorini, B. 1993. Building a large annotated corpus of english: The penn treebank. *Computational Linguistics* 19:313–330.

Mikolov, T.; Sutskever, I.; Chen, K.; Corrado, G. S.; and Dean, J. 2013. Distributed representations of words and phrases and their compositionality. In *Advances in Neural Information Processing Systems 26*. Curran Associates, Inc. 3111–3119.

Navigli, R. 2009. Word sense disambiguation: A survey. *ACM Computing Surveys* 41(2):10.

Nunes, B. P.; Kawase, R.; Siehndel, P.; Casanova, M. A.; and Dietze, S. 2013. As simple as it gets-a sentence simplifier for different learning levels and contexts. In *Proceedings of the 13th ICALT*, 128–132.

Padr, L., and Stanilovsky, E. 2012. Freeling 3.0: Towards wider multilinguality. In *Proceedings of the 8th LREC*.

Paetzold, G. H., and Specia, L. 2013. Text simplification as tree transduction. In *Proceedings of the 9th STIL*, 116–125.

Paetzold, G. H., and Specia, L. 2015. Lexenstein: A framework for lexical simplification. In *Proceedings of the 53rd ACL*, 85–90.

Paetzold, G. H. 2013. *Um Sistema de Simplificacao Automatica de Textos escritos em Ingles por meio de Transducao de Arvores*. Western Parana State University.

Paetzold, G. H. 2015. Reliable lexical simplification for non-native speakers. In *Proceedings of the 14th NAACL Student Research Workshop*, 9–16.

Robbins, T. 2003. *Jitterbug Perfume*. Random House Publishing Group.

Shardlow, M. 2014. A Survey of Automated Text Simplification. *International Journal of Advanced Computer Science and Applications*.

Specia, L.; Jauhar, S. K.; and Mihalcea, R. 2012. Semeval-2012 task 1: English lexical simplification. In *Proceedings of the 1st SemEval*, 347–355.

Stolcke, A. 2002. Srilm-an extensible language modeling toolkit. In *Proceedings of the International Conference on Spoken Language Processing*, 257–286.