A Dataset for the Evaluation of Lexical Simplification

Jan De Belder and Marie-Francine Moens

Katholieke Universiteit Leuven
Department of Computer Science
Celestijnenlaan 200A, B-3001 Heverlee, Belgium
jan.debelder@cs.kuleuven.be, sien.moens@cs.kuleuven.be

Abstract. Lexical Simplification is the task of replacing individual words of a text with words that are easier to understand, so that the text as a whole becomes easier to comprehend, e.g. by people with learning disabilities or by children who learn to read.
Although this seems like a straightforward task, evaluating algorithms for this task is not so. The problem is how to build a dataset that provides an exhaustive list of easier to understand words in different contexts, and to obtain an absolute ordering on this list of synonymous expressions.
In this paper we reuse existing resources for a similar problem, that of Lexical Substitution, and transform this dataset into a dataset for Lexical Simplification. This new dataset contains 430 sentences, with in each sentence one word marked. For that word, a list of words that can replace it, sorted by their difficulty, is provided. The paper reports on how this dataset was created based on the annotations of different persons, and their agreement. In addition we provide several metrics for computing the similarity between ranked lexical substitutions, which are used to assess the value of the different annotations, but which can also be used to compare the lexical simplifications suggested by an algorithm with the ground truth model.

1 Introduction

The Lexical Simplification (LS) problem can be defined as substituting words with easier alternatives, so that the text becomes easier to comprehend. Important is that the meaning of the original text is not altered, and that it remains fluent.

There are several reasons why we would want to make text easier to understand. Different groups of readers are confronted with the difficulty of texts: adults who suffered a brain injury, deaf persons [13], young readers, non-native speakers [12] and readers with low literacy skills [15, 1]. Although these different groups find texts difficult for different reasons, the causes that make texts hard to comprehend overlap to a large degree. What makes a sentence difficult to understand can usually be attributed to one of two factors or both: the lexical difficulty (i.e. difficult words and phrases) and/or the syntactic difficulty (i.e. complex grammatical constructs). After more than a decade since its first appearance in the literature [5], Lexical Simplification is receiving a renewed interest [18, 17, 1]. However, the evaluation still remains problematic.

In this paper we discuss how to create ground truth models used in the evaluation of the Lexical Simplification task. Previous research usually only performed a partial evaluation, e.g. by determining whether a replacement is simpler without taking the context into account, thereby bypassing the difficult Word Sense Disambiguation aspect. Furthermore, it is difficult to compare between methods, since it requires human judgments, which are hard to reproduce. Evaluating a different parameter means running a whole set of evaluations again, which is a tedious and expensive process.

We aim to overcome these problems by developing a corpus on which Lexical Simplification methods can be evaluated. We start from an existing corpus for a related task: that of Lexical Substitution. More specifically, we started from the LexSub dataset from the SemEval 2007 Lexical Substitution task [11]. For a given word, annotators provided alternative words that could replace the original word, without changing the meaning of the sentence (too much). This solves the problem of generating words that fit in the context. We extend this dataset by ordering the alternative words by difficulty. With this dataset, we can evaluate different methods.

This paper reports on how this dataset was created based on the annotations of different persons, and their agreement. We also show how we combine the different annotations to a single list of sorted words. In addition we provide several metrics for computing the similarity between ranked lexical substitutions, which are used to assess the value of the different annotations, but which can also be used to compare the lexical simplifications suggested by the machine with a ground truth model.

In the next section, give an overview of the evaluation methodologies in previous research. Section 3 provides the details of the origin of the dataset and the Lexical Substitution task. In section 4, we provide details on our annotation process. In section 5, we analyze the results of the annotation process, and determine the quality. After assessing how reliable it is, we suggest some metrics of evaluating algorithms with it, presented in section 6. We end with our conclusions in section 7.

2 Previous evaluations

In previous work, the focus was mainly on the methods for Lexical Simplification. In this section, we will not discuss the different methods and their results, but instead concentrate on the evaluation methodologies.

[17] extracts simplification from the edit history in simple Wikipedia. The method is a probabilistic model, so that a distinction can be made between edits that remove spam, edits that correct spelling errors, and actual simplifications. The evaluation was done by selecting 200 edits from each of the models (100 random, and the 100 most probable), e.g. *"annually"* → *"every year"*, and let-

ting 3 native speakers rate these. The possible answers were "simpler", "more complex", "equal","unrelated", and "?" (hard to judge). By collapsing these to "simplification", "not a simplification", and "?", an inter annotator agreement of $kappa = 0.69$ is obtained. However, these simplification are evaluated out of their original context, whereas previous research has shown that this is important [8].

In [18] the authors focused on context-specific lexical paraphrases. The method uses the Web to find and validate alternative wordings. The evaluation is done on a set of 257 news article headlines, taken from Chinese online newspapers. The measures used are precision and recall, although the authors recognize the difficulty of evaluating the recall. The latter is approximated by grouping all the correct answers from all of the methods they evaluated, and assuming this set is an exhaustive set of all the answers. The precision is based on manual judgments, but the authors do not specify by whom this was judged.

The method in [2] makes a distinction between finding pairs of synonyms, and a context aware approach that decides when to substitute, so it can be used in conjunction with e.g. the method from [17]. The dataset consisted of 65 sentences from Wikipedia, for which their method simplified exactly one word, and the baseline (the method from [5]) was also able to simplify that word (but to a different one). The evaluation was done in a thorough way, rating the degree of simplification (simpler or not), meaning preservation (preserves meaning or not) and grammaticality preservation (bad, ok, good) of the substitutions. The annotations were divided among three native English speakers, and a small portion was annotated multiple times to calculate the pairwise inter annotator agreement, which was moderate for all categories (*kappa* between .35 and .53).

[19] describe a more complex method, that also performs syntactic operations, by treating the problem of text simplification as a monolingual machine translation problem, in which sentences from English Wikipedia are translated to a sentences from the Simple Wikipedia. The evaluation was done by simplifying 100 sentences from the English Wikipedia, held out from the training data, and using machine translation measures (BLUE and NIST scores) to compare the generated sentences with the gold standard, i.e. the aligned sentences from Simple Wikipedia.

[16] also perform text simplification, and use the same dataset as [19] for evaluating their method. Next to the machine translation measures, they also engage humans in the evaluation on 64 of the 100 sentences. 45 unpaid volunteers rated the simplifications in three separate experiments: one that decided whether or not the simplified sentence was simpler than the original, a second experiment to rate the grammaticality of the simplified sentences, and a third to indicate how well meaning was preserved. All ratings were on a five point Likert scale.

3 Selecting a dataset

We create ground truth data by building further on another dataset, constructed for a similar task. With this latter we refer to the SemEval 2007 Lexical Substitution task [10]. This task had a similar objective: given a sentence with one marked word, replace this word with another word, so that it still fits the context. The idea behind this task was Word Sense Disambiguation in a practical setting. Lexical Substitution (LEXSUB) is a more general problem than the one we are faced with here. For substitution, any replacement that fits the context is a valid solution, whereas for simplification we want valid replacements that are also easier to understand.

The dataset used for the Lexical Substitution task [10] consists of 201 words, which were chosen at random. For each of the words, 10 sentences were retrieved that contained the word or a conjugated form of the word. The sentences were selected from the English Internet Corpus of English produced by Sharoff [14], obtained by sampling data from the Web[1].

The LEXSUB dataset thus consists of 2010 sentences in total. For each of these sentences, five annotators provided up to three words that could replace the indicated word in each sentence. The annotators also had the possibility of indicating they couldn't think of a better replacement.

To transform this to a Lexical Simplification dataset, we first remove those of the 201 words that are on a list of 'easy words'. We take this list of easy words to be the union of the 'Basic English combined word list' from Simple Wikipedia[2], and the 3000 words from the Dale-Chall readability measure[3]. It is unlikely that those easy words will have to be simplified, or even can be simplified, so we do not include them in the annotation process. After removal there were 43 words, or 430 sentences, remaining. Later we show that these words are almost always ranked highest in the list, and therefore refrain from annotating them.

This dataset offers a great starting point, as it provides an exhaustive list of alternative words that can replace a given word, based on the context (i.e. the sentence).

4 Annotating the dataset

We started from the same dataset, using the alternative words generated by the annotators as a set of valid alternatives. To convert this dataset from a Lexical Substitution problem to a Lexical Simplification problem, we have to sort the words by their difficulty.

4.1 Methodology

We ask the annotators to rank the different alternatives according to how easy to understand they are in the given sentence. We also include the original word in this list of words to be sorted, so we know which alternatives are easier, equally hard, or harder to understand compared to the original. Furthermore, we allow the annotators to rank different words at the same position, for the cases where they think two words are equally difficult.

[1] http://corpus.leeds.ac.uk/internet.html

[2] http://simple.wikipedia.org/wiki/Wikipedia:Basic_English_combined_wordlist

[3] The percentage of words in a text and not on this list is used as an indicator of difficulty.

4.2 Annotators

We used two different groups for the annotations. The first is Amazon Mechanical Turk[4]. The advantage is that it is cheap, in comparison to hiring professionals, and the results can be obtained very fast since multiple people can work on it. We requested five annotators for each sentence, located in the U.S. and that completed at least 95% of their previous assignments correctly.

However, there are also people on Mechanical Turk who are keen to finish their assignments as quickly as possible, and this might have a negative effect on the quality. Therefore we also had part of the dataset annotated again, by PhD students[5]. Two more annotations for roughly half the sentences were obtained this way. Using these annotations, we can test the quality of the Mechanical Turk annotations.

In total 46 different Turkers participated, each providing on average 29.5 annotations. The other annotations came from 9 different PhD students, with on average 85.9 annotations.

5 Analysis of the dataset and the annotations

5.1 Measuring annotator agreement

To get an idea of the quality of the annotations, we look into methods of calculating the inter annotator agreement. This is not an easy task, since the chance that two rankings are completely identical is very small.

In what follows, let us define ann_i to be the i-th annotator, n_{ann} the number of annotators, w_j the j-th word in the list of alternatives, and $rank_i(w_j)$ the rank given by ann_i to word w_j. All the equations below are based on the replacement of a single word in one sentence.

Fleiss' kappa A typical measure is Cohen's kappa [3], but unfortunately this works only for binary classification problems, with two annotators. To solve the problem of multiple annotators, a solution is to compare each annotator to the majority vote of the other annotators. This is still difficult, since the majority of a ranking is hard to define.

An extension of this measure is Fleiss' kappa [7], that extends to multiple annotators and multiple classes. Although we don't have multiple classes, we can convert our ranking problem into a suitable form. We can do so by taking each two words (w_i, w_j) in the list, and put them in one of three categories: w_i and w_j are ranked equally difficult, w_i is ranked easier than w_j, and w_i is ranked more difficult than w_j. By doing so, we are able to use the Fleiss' kappa measure. Like for Cohen's kappa, a Fleiss' kappa value of 1 means perfect agreement between the annotators.

[4] http://www.mturk.com

[5] Although their native language isn't always English, they have a more than average understanding.

Rank correlation A more appropriate measure would be the Spearman rank correlation coefficient. This takes into account the natural ranking of the words provided by the annotators, rather than having to convert it to a set of pairwise comparisons. The Spearman rank correlation coefficient is defined as

$$\rho = \frac{\sum_j (rank'_i(w_j) - \overline{rank'_i})(rank'_k(w_j) - \overline{rank'_k})}{\sqrt{\sum_j (rank'_i(w_j) - \overline{rank'_i})^2 \sum_j (rank'_k(w_j) - \overline{rank'_k})^2}} \tag{1}$$

where $\overline{rank'_i}$ is the average rank of the words given by annotator i. Often words are ranked at the same position by the annotators, and ties here are solved by assigning them the average of their rank. So a ranking of $((w_1), (w_2,w_3,w_4), (w_5))$ will assign a rank 1 to w_1, and rank 3 $(\frac{2+3+4}{2})$ to w_2,w_3 and w_4. This is indicated by the use of $rank'$ instead of $rank$ in equation 1.

To extend this to a one annotator versus majority case, we define the rank assigned by the second annotator to be the average of the ranks given by the other annotators. The correlation coefficient is a number between -1 and $+1$, with 0 indicating that there is no dependence.

Penalty based agreement A third measure we can use to evaluate the agreement, is based on penalties. For each word that is ranked at a different position by two annotators, a penalty is given, proportional to the difference in distance. For each word, we can calculate the following score:

$$\text{score}(w_j) = 1 - \frac{|rank_i(w_j) - rank_k(w_j)|}{\max_l rank_k(w_l)} \tag{2}$$

This is similar to the measure used in [4], for comparing rankings. The items ranked there were names, and they were ordered according to the importance of them in a picture.

However, in [4], this was used to compare a generated ranking with an expert ranking. To extend this to our case, where we compare one annotator ann_i against the remainder of the annotators, we give a penalty for each annotator:

$$\text{score}(w_j) = 1 - \frac{1}{n_{ann} - 1} \sum_{k=1,k\neq i}^{n_{ann}} \frac{|rank_i(w_j) - rank_k(w_j)|}{\max_l rank_k(w_l)} \tag{3}$$

5.2 Outlier removal

In table 1 we provide the inter annotator agreement measures, discussed above, for the initial annotations retrieved from Mechanical Turk. Although the Fleiss' kappa measure looks low, agreement seems reasonable.

In order to improve the quality of the dataset, we will filter out some of the less accurate annotators. To illustrate we plotted the annotators on a graph, as can be seen in figure 1. On the y-axis there is the inter annotator agreement, as measured with the rank correlation agreement, that would be obtained by

Measure	Score
Fleiss' Kappa	0.486
Rank Correlation	0.592
Penalty Based	0.716

Table 1: Agreement of the annotators, initial dataset.

Measure	Score
Fleiss' Kappa	0.488
Rank Correlation	0.602
Penalty Based	0.724

Table 2: Agreement of the annotators, after filtering the dataset.

removing a specific annotator. On the x-axis, there is the median submit time for a set of 10 sentences. It is interesting to note that there seems to be little correlation between the average submit time and the quality of the work.

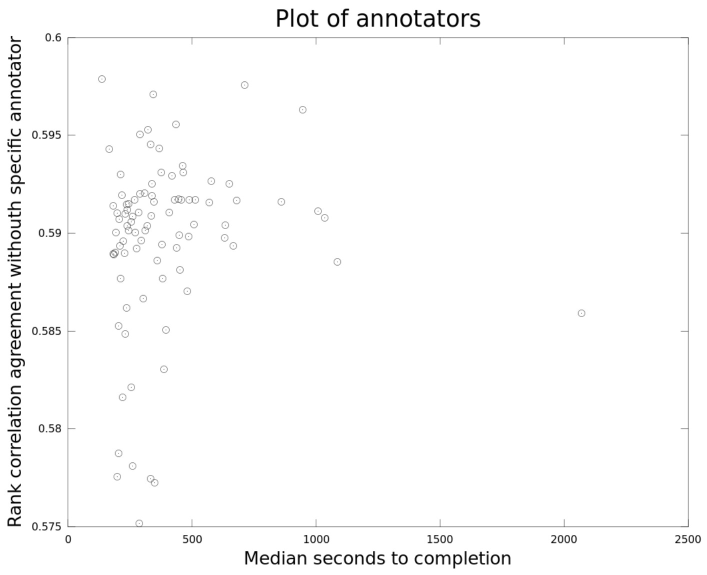

Fig. 1: Graphical representation of the annotators, with on the x-axis the average seconds to completion of 10 sentences, and on the y-axis the change in agreement by removing the annotator.

We removed the four annotators that would result in a maximal increase in agreement between the annotators, and had their annotations redone. This resulted in the agreement scores as can be seen in table 2. Although the agreement scores are higher, the change does not seem to be very remarkable.

5.3 Evaluation of quality

With a Fleiss' kappa score of 0.488, we can assume we have only a moderate agreement [9]. This measure takes into account agreement by chance. However, as noted often in the literature [6], it can be misleading. One factor that reduces the agreement, is that the measure as we use it is very strict: if annotator 1 ranks two words as being equally hard, and annotator 2 ranks them directly below each other, this is a disagreement, although in reality the two answers are closely related.

The Spearman rank correlation ρ of 0.602 indicates there is certainly a correlation between the annotations; if not ρ would be 0.

As a check for testing the quality of the annotations, we compare the agreement between the annotations between Turkers and the agreement between the annotations done by the students. This is only for a subset of the data (200 of the 430 sentences), and only two annotations were provided for each sentence. The results are in table 3. It can be seen that the annotators are in larger disagreement than the Turkers, illustrating the difficulty of this task, although the smaller number of annotators has to be taken into consideration.

Measure	**Score**
Fleiss' Kappa	0.393
Rank Correlation	0.451
Penalty Based	0.691

Table 3: Agreement of the student annotators

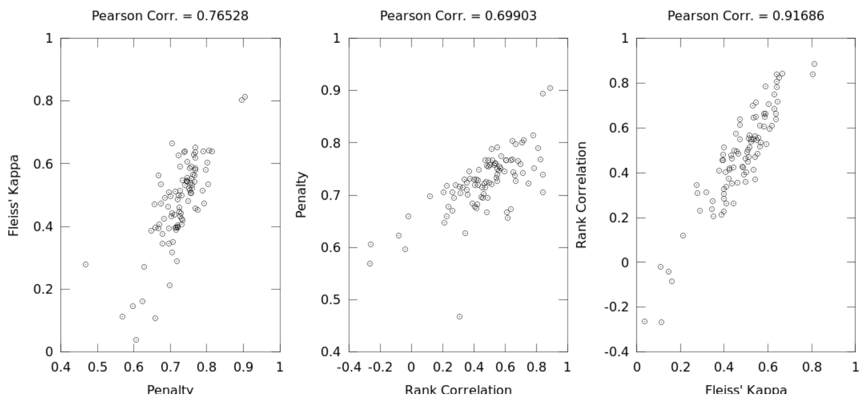

Fig. 2: Correlation between the inter annotator agreement metrics

In figure 2 we created a graphical representation of the correlation between the different measures we used for calculating the inter annotator agreement. Each point on the graphs is an annotator, positioned according to the agreement with the rest of the annotators.

5.4 Merging annotations

In this section we convert the multiple rankings from the annotators to a single gold-standard ranking. One way of doing this, is by taking all the pairwise comparisons from all the annotators, and using the most frequent[6] ordering between each two words. For example, if annotator 1 and 2 rank word w_1 higher than w_2, and annotator 3 ranks them equal, then the most frequent pairwise ranking is $w_1 > w_2$. A problem with this approach is that it can cause inconsistencies: the total ordering is not guaranteed anymore. For example, suppose annotator 1 answered $((w_2), (w_3), (w_1))$, annotator 2 answered $((w_1), (w_2, w_3))$, and annotator 3 $((w_1, w_2, w_3))$. Then the most frequent orderings are $(w_1 = w_2)$, $(w_1 = w_3)$, $(w_2 > w_3)$, which leads to $w_1 = w_2 > w_3 = w_1$, or $w_1 > w_1$.

The previous way of merging the annotations to a single ordering neglects two factors. First, the distance between two words is not taken into account. In the example above, the first annotator ranked w_2 and w_1 further apart then the other annotators, but this is not reflected. A second factor is that the quality of the annotators is not taken into account: the opinion of each annotator weighs equally, but some are more accurate than others.

With these two considerations in mind, we resort to a noisy channel model method of finding the optimal ordering. With this framework, we can define the source model to be the real ordering of the words, that is at this point unknown. The channel through which we observe this real ordering, is in the form of the annotators, that generate their annotations based on the real ordering, but with additional noise (errors).

We can then calculate the optimal real ordering for the alternative words of a sentence as:

$$max_h \prod_{i=1}^{n_{ann}} rel(ann_i) sim(h, annotation_i) \tag{4}$$

in which rel is the reliability of an annotator, and sim is the similarity of the hypothesis real ordering h and the annotation $annotation_i$ provided by the i-th annotator. We can calculate the similarity by simply reusing equation 1 or 3. For the reliability of an annotator, we also use these equations, in the form of the agreement with the combined annotation of all the other annotators, averaged over all the sentences he/she annotated. In the remainder of this section, we report on the orderings obtained by using the penalty based method from section 5.1.

[6] When $w_1 > w_2$, $w_1 = w_2$, and $w_1 < w_2$, we assume $w_1 = w_2$ is the most frequent.

5.5 Properties of the dataset

After combining the annotations into a single ordering, we can calculate its properties. In 70.5% of the sentences the word can be replaced by one or more simpler words. In 75.6% of the cases, there is also one or more word that is equally hard. Finally, in 71.6% of the cases there are words that are harder.

The average number of alternative words is 5.04. Since we allowed annotators to rank words on the same level of difficulty, there are on average 3.03 levels.

To illustrate what the dataset looks like, in table 4 there are number of example sentences and alternative words, sorted by difficulty. The two last examples are for the same word, severely, showing the dependency on the context.

Finally, to prove our hypothesis that words that are on the list of easy words are already the easiest word, and can't be simplified further, we also had sentences for five of those words annotated, yielding 50 sentences. For those words, there was only a simpler word in 10% of the cases, illustrating that it is probably better to select new words and sentences altogether.

• Rabbits often feed on young, *tender* perennial growth as it emerges in spring, or on young transplants. [[soft], [tender, delicate]]
• Performance test for a system coupled with a locally manufactured station engine model MWM will start *shortly*. [[shortly, soon], [before long], [presently]]
• Perhaps the effect of West Nile Virus is sufficient to extinguish endemic birds already *severely* stressed by habitat losses. [[highly], [seriously, severely, extremely], [gravely], [critically]]
• Mutual Funds are so *severely* conflicted that they will not avail themselves of the alleged benefits of the proposed rule. [[badly], [seriously, severely, heavily], [extremely, gravely]]

Table 4: Examples sentences with alternative words.

6 Metrics

Now that we have merged the different annotations into a single dataset, we can use it for the evaluation of Lexical Simplification methods. In this section, we will define three metrics to do so, but each time with a different goal.

6.1 Binary metric

When practically using the Lexical Simplification algorithms to simplify text, only a single solution can be used (i.e. a word can only be replaced by one other word). Because we included the original word each time in the list of words to be sorted, we can position the other words relative to the original word. We then define a scoring function as follows:

$$score_{bin}(w_j) = \begin{cases} +1, & \text{if } w_j \text{ is easier.} \\ 0, & \text{if } w_j \text{ is equally hard, harder,} \\ & \text{or not in the list of alternatives.} \end{cases} \tag{5}$$

6.2 Rank evaluation

For a more extensive evaluation, multiple words can be generated in a sorted list. This brings us back to calculating the similarity between two rankings, a topic that we investigated in detail in section 5.1. Our penalty based method is based on a method for comparing a generated ranking with an expert ranking, so we can use this in its original form:

$$score_{penalty}(w_j) = 1 - \frac{|rank(w) - rank_{gold}(w_j)|}{\max_l rank_{gold}(w_l)} \tag{6}$$

6.3 Precision and recall

Similar to the evaluation in [18], we can calculate the precision and recall, and use these to compute the F-measure. To determine recall, we define the number of easier alternatives as n_{easier} as $\#\{w_j | rank_{gold}(w_j) < rank_{gold}(w_{orig})\}$

$$P = \frac{\sum_j score_{bin}(w_j)}{\max_l rank(w_l)} \qquad R = \frac{\sum_j score_{bin}(w_j)}{n_{easier}} \tag{7}$$

7 Conclusions

The Lexical Simplification of text entails replacing difficult words with words that are easier to understand. But this is a problem that is hard to evaluate, e.g. because the simplifications are context-dependent, and an exhaustive list of simplifications is hard to generate. In this paper, we have shown how we created a dataset[7] for this problem. By reusing an existing dataset for Lexical Substitution, with an exhaustive enumeration of all possible words that can replace a word, we solve the problem of not being able to measure the recall.

Starting from the Lexical Substitution dataset, we have first filtered out words that were too easy to simplify. Next we had annotators sort the different alternative words according to their simplicity, taking into account the context of the original word in the sentence. For these annotations we calculated several inter annotator agreement measures. The main source of the annotations comes from Mechanical Turkers, and we have shown that their agreement is similar to that of less ‘time biased’ annotators. After removing a number of outliers, we merged the annotations into a single gold standard, by interpreting it as a noisy

[7] Available at `http://people.cs.kuleuven.be/~jan.debelder/lseval.zip`.

channel problem. Finally, we suggested a number of scoring metrics that can be used with this gold standard.

In the future, we will use this dataset to evaluate Lexical Simplification algorithms. A weakness is that the original dataset replaced words mostly by other single words, i.e. multi word expressions are not very common.

Acknowledgments

This research is funded by the EU project *PuppyIR*[8] (EU FP7 231507) and the EU project *TERENCE*[9] (EU FP7 257410).

References

1. Aluísio, S., Gasperin, C.: Fostering digital inclusion and accessibility: the porsimples project for simplification of Portuguese texts. In: Proceedings of the NAACL HLT 2010 Young Investigators Workshop on Computational Approaches to Languages of the Americas. pp. 46–53 (2010)
2. Biran, O., Brody, S., Elhadad, N.: Putting it simply: a context-aware approach to lexical simplification. In: Proc. of the 49th Annual Meeting of the ACL: HLT. pp. 496–501. Association for Computational Linguistics (2011)
3. Cohen, J., et al.: A coefficient of agreement for nominal scales. Educational and psychological measurement 20(1), 37–46 (1960)
4. Deschacht, K., Moens, M., Robeyns, W.: Cross-media entity recognition in nearly parallel visual and textual documents. In: Large Scale Semantic Access to Content (Text, Image, Video, and Sound). pp. 133–144. Le Centre De Hautes Etudes Internationales D'informatique Documentaire (2007)
5. Devlin, S., Tait, J.: The use of a psycholinguistic database in the simplification of text for aphasic readers. Linguistic Databases pp. 161–173 (1998)
6. Eugenio, B., Glass, M.: The kappa statistic: A second look. Computational Linguistics 30(1), 95–101 (2004)
7. Fleiss, J.: Measuring nominal scale agreement among many raters. Psychological Bulletin 76(5), 378 (1971)
8. Lal, P., Ruger, S.: Extract-based summarization with simplification. In: DUC 2002: Workshop on Text Summarization, July 11–12, 2002, Philadelphia, PA, USA (2002)
9. Landis, J., Koch, G.: The measurement of observer agreement for categorical data. Biometrics 33(1), 159 (1977)
10. McCarthy, D., Navigli, R.: Semeval-2007 task 10: English lexical substitution task. In: Proc. of the 4th International Workshop on Semantic Evaluations (SemEval-2007). pp. 48–53 (2007)
11. McCarthy, D., Navigli, R.: The English lexical substitution task. Language Resources and Evaluation 43(2), 139–159 (2009)
12. Petersen, S.: Natural language processing tools for reading level assessment and text simplification for bilingual education. Ph.D. thesis, University of Washington (2007)
13. Quigley, S., Paul, P.: Language and deafness. College Hill Books (1984)

[8] http://www.puppyir.eu
[9] http://www.terenceproject.eu/

14. Sharoff, S.: Open-source corpora: Using the net to fish for linguistic data. International Journal of Corpus Linguistics 11(4), 435–462 (2006)
15. Shewan, C., Canter, G.: Effects of vocabulary, syntax, and sentence length on auditory comprehension in aphasic patients. Cortex: A Journal Devoted to the Study of the Nervous System and Behavior (1971)
16. Woodsend, K., Lapata, M.: Learning to simplify sentences with quasi-synchronous grammar and integer programming. In: Proc. of the 2011 Conference on Empirical Methods in Natural Language Processing. pp. 409–420 (2011)
17. Yatskar, M., Pang, B., Danescu-Niculescu-Mizil, C., Lee, L.: For the sake of simplicity: Unsupervised extraction of lexical simplifications from Wikipedia. In: Human Language Technologies: The 2010 Annual Conference of the NAACL. pp. 365–368 (2010)
18. Zhao, S., Liu, T., Yuan, X., Li, S., Zhang, Y.: Automatic acquisition of context-specific lexical paraphrases. In: Proc. of the IJCAI. pp. 1789–1794 (2007)
19. Zhu, Z., Bernhard, D., Gurevych, I.: A monolingual tree-based translation model for sentence simplification. In: Proc. of the 23rd International Conference on Computational Linguistics (2010)