TALN at SemEval-2016 Task 11: Modelling Complex Words by Contextual, Lexical and Semantic Features

Francesco Ronzano, Ahmed Abura'ed, Luis Espinosa-Anke, Horacio Saggion
Department of Information and Communication Technologies (DTIC)
Universitat Pompeu Fabra
Tangèr 122, Barcelona (08018), Spain
{francesco.ronzano,ahmed.aburaed,luis.espinosa,horacio.saggion}@upf.edu

Abstract

This paper presents the participation of the TALN team in the Complex Word Identification Task of SemEval-2016 (Task 11). The purpose of the task was to determine if a word in a given sentence can be judged as complex or not by a certain target audience. To experiment with word complexity identification approaches, Task organizers provided a training set of 2,237 words judged as complex or not by 20 human evaluators, together with the sentence in which each word occurs. In our contribution we modelled each word to evaluate as a numeric vector populated with a set of lexical, semantic and contextual features that may help assess the complexity of a word. We trained a Random Forest classifier to automatically decide if each word is complex or not. We submitted two runs in which we respectively considered unweighted and weighted instances of complex words to train our classifier, where the weight of each instance is proportional to the number of evaluators that judged the word as complex. Our system scored as the third best performing one.

1 Introduction

Approaches to automatically identify if a target audience will perceive a certain word as complex or not constitute a core component in several language-related areas of research, including *Lexical Simplification* (Bott et al., 2012) and *Readability Assessment* (Collins-Thompson, 2014).

The Complex Word Identification Task of SemEval-2016 proposes a shared framework for evaluating complex word identification systems. Task participants were provided with a set of sentences where, for each sentence, one or more words have been rated as complex or not by 20 human evaluators. An example sentence from this dataset is:

If the growth rate is known , the maximum lichen size will give a minimum age for when this rock was deposited.

In this sentence, the words 'lichen' and 'deposited' were classified as complex by at least one out of the 20 evaluators, unlike e.g. 'growth', which did not received this label by any of them.

In our participation in Task 11, we cast the identification of complex words as a binary classification problem in which each word is evaluated as complex or not, given the sentence in which it occurs. We modelled each word by a set of lexical, semantic and contextual features and evaluated distinct binary classification algorithms. Our approach to Task 11 obtained good performance: our team ranked as the second best performing one and one of the two systems we proposed scored as the third best performing system according to the G-score official evaluation metric (harmonic mean between Accuracy and Recall).

In Section 2 we provide an overview of relevant research related to Complex Word Identification. Section 3 and 4 respectively introduce the Task 11 dataset and present the text analysis tools and resources we exploited to characterize complex words. In Section 5 we describe the word features we used to build our complex word classifier. In Section 6 we present and discuss the performance of our Task 11 system. Finally, in Section 7 we formulate our

Proceedings of SemEval-2016, pages 1011–1016,
San Diego, California, June 16-17, 2016. ©2016 Association for Computational Linguistics

conclusions and outline future venues of research.

2 Related work

The identification of complex words constitutes a key aspect of *Lexical Simplification* (Bott et al., 2012). It can be defined as the problem of replacing difficult words by their simpler synonyms taking into account the specific context in which each word is used. Several techniques have been applied so far to identify complex words. In the context of the PSET Project (Devlin and Tait, 1998), the first lexical simplification system for English was developed, aimed at people with aphasia. It relies on a word difficulty assessment based on psycholinguistic evidence (Quinlan, 1992) in order to decide whether to simplify a word. Recent work exploited the availability of comparable corpora of original documents (e.g. English Wikipedia) and their 'simplified' versions (e.g. Simple English Wikipedia pages) to induce measures which can be used to compare and rank 'quasi-synonymic' word pairs (Yatskar et al., 2010). (Shardlow, 2013) compares three techniques to identify complex words in English: a psycholinguistic approach (Devlin and Tait, 1998), frequency thresholding (i.e. words with low frequency are considered complex), and a machine learning algorithm trained only on word features (frequency, syllable count, ambiguity, etc.). In this work, a corpus of complex words is created based on edit histories from the Simple Wikipedia. The authors conclude that the three tested methods perform similarly in terms of F-measure. (Saggion et al., 2016) use the combined evidence of word frequency and word length to assess the word complexity of a list of synonyms so as to select the simpler one in an Spanish lexical simplification system. (Rello et al., 2013) argue that word frequency and length are two important factors affecting readability and understanding for people with dyslexia.

Besides lexical simplification, the identification of complex words constitutes a core component of *readability assessment* (Collins-Thompson, 2014), the problem of quantifying the readability of a given text. The presence of complex words usually penalizes readability. Lists of easy words (Dale and Chall, 1948), word characteristics (Kincaid et al., 1975; Gunning, 1952; Mc Laughlin, 1969), or word use in context (e.g. language models) (Si and Callan, 2001) are all techniques or resources which have been used to support the assessment of text readability: these approaches could also be adapted to evaluate word complexity.

3 Dataset

The organizers of SemEval-2016 Task 11 released a *training dataset* composed of 2,236 words together with the sentence in which each word occurs. For each word, the binary complexity judgements of 20 human evaluators were provided (complex word or not complex word). Similarly, Task 11 *testing dataset* consisted of 88,221 words together with the sentence in which each word occurs. In this case, for each word, the binary complexity judgement of only one human annotator was collected.

4 Resources and Tools

In order to identify complex words, we characterize each word by means of a set of lexical, semantic and contextual features. To this purpose, we analyze both the word and the sentence in which it occurs by means of the language resources and text analysis tools described in what follows.

4.1 Language Resources

Information about the frequency of use is important to assess word complexity. Therefore, in our complex word identification approach we exploit the word frequency data of two large corpora: (i) a 2014 English Wikipedia Dump and (ii) the British National Corpus (Leech and Rayson, 2014). We also use WordNet (Miller, 1995) to model semantic word features by relying on word senses and synset relations (e.g. hypernymy). Moreover, we use the Dale & Chall list of 3,000 simple words (Dale and Chall, 1948) in order to incorporate the text readability dimension, as this list contains words which 4th grade students considered understandable.

4.2 Text Analysis Tools

We analyze the sentences in which a word to evaluate occurs by means of the Mate dependency parser (Bohnet, 2010). As a result, we obtain a lemmatized and Part-Of-Speech (POS) tagged version of the sentence, along with its syntactic dependencies. Both POS tags and dependency information are used to compute several features as described in the following Section.

We also processed each sentence by the UKB graph-based Word Sense Disambiguation algorithm (Agirre and Soroa, 2009). Specifically, we benefited from the UKB implementation integrated in the Freeling workbench (Padró and Stanilovsky, 2012). In this way, we may disambiguate single or multiword expressions against WordNet 3.0.

5 Method

In order to evaluate the complexity of a word, we modelled each word as a feature vector. Then, we used such word representation to enable the training and evaluation of distinct binary classification algorithms tailored to determine whether a word is complex or not. To this end, we relied on the Weka machine learning framework (Witten and Frank, 2000). We evaluated the performance of four classification algorithms: Support Vector Machine (with linear kernel), Naïve Bayes, Logistic Regression and Random Forest. For each algorithm, we experimented the effectiveness of the following two training approaches:

- *Simple*: in which complex and non complex word training instances have the same relevance (weight);
- *Weighted*: in which we weighted each non complex word with weight 1 and each complex word with a weight ranging from 1 to 20 with respect to the number of human annotators (over 20) that evaluated the word as complex.

In the remainder of this Section we describe the set of word features we used, and motivate their relevance with respect to the characterization of complex words. When presenting word features, we group subsets of related features in the same subsection (Shallow features, Dependency Tree features, etc.). It is important to note that some of the word features presented are computed by considering, besides the target word, also context words in a $[-3, 3]$ window, where position 0 refers to the target word. If the context word at a specific position cannot be determined, the value of the related feature is set to *undefined*.

5.1 Shallow Features

We exploited the following set of shallow word features:

- **Word length** (CharNumber): the length of the target word (number of characters).
- **Position of the word** (WordPosition): the position of the target word in the sentence. The value of this feature is normalized in the interval $[0, 1]$ by dividing the the position of the target word in the sentence by the length of the same sentence (number of words). The position of the first word of a sentence is 0.
- **Words in sentence** (NumSentenceWords): the number of tokens in the sentence.

5.2 Dependency Tree Features

The following set of features is derived by processing the dependency tree of the sentences that include the word to evaluate:

- **Word depth in the dependency tree** (DepthInTree_*position* - 7 features): we considered the depth in the dependency tree of the target word (*position* equal to 0), the three previous words and the three following words.
- **Parent word length** (ParentCharNumber): the length (number of characters) of the parent of the current (target) word in the dependency tree.

5.3 Corpus-based Features

Word frequency data derived from the British National Corpus and the 2014 English Wikipedia was used to compute the following set of features:

- **British National Corpus frequency** (BNCFrequency_*position* - 7 features): we considered the BNC frequency[1] of the target word lemma (*position* equal to 0), the three previous word lemmas and the three following word lemmas.

[1] http://ucrel.lancs.ac.uk/bncfreq/lists/1_1_all_fullalpha.txt.Z

- **English Wikipedia frequency** (ENwikiFrequency_*position* - 7 features): we considered the 2014 English Wikipedia frequency of the target word (*position* equal to 0), the three previous words and the three following words. Word frequencies were computed over a tokenized and lower-cased version of the English Wikipedia.

- **Simple word list** (Dale_Chall): a binary feature to point out the presence of the target word in the Dale & Chall list.

5.4 WordNet features

We used WordNet 3.0 to compute the following features. Given a target word, we refer as *target-word-synsets* the set of synsets that have the same POS of the target word and include the target word among their lexicalizations (all the senses of the target word). Note that this set of features is computed without relying on Word Sense Disambiguation.

- **Number of Synsets** (WNSynsetN): the number of synsets in *target-word-synsets* (i.e. number of senses of the target word).

- **Number of Senses** (WNSenseN): the sum of the number of word senses (i.e. the number of lexicalizations) of each *target-word-synset*.

- **Depth in the hypernym tree** (WNDepth): the average depth in the WordNet hypernym hierarchy among all the *target-word-synsets*.

- **Number of Lemmas** (WNLemma): the average number of synset lexicalizations among all the *target-word-synsets*.

- **Gloss length** (WNGloss): the average length of synset Glosses among all the *target-word-synsets*, in terms of number of tokens.

- **Number of relations** (WNRelation): the average number of semantic relations among all the *target-word-synsets*.

- **Number of Distinct POSs** (WNDistinctPOS): the number of distinct POS represented by at least one *target-word-synset*.

- **Part of Speech** (WN_*POS* - 4 features): for each WordNet POS (*POS* equal to Noun, Verb, Adjective and Adverb) we counted the number of synsets with that POS among the *target-word-synsets*, thus generating four features.

5.5 WordNet and corpus frequency features

The following set of features was computed by combining WordNet data, the word frequencies of the British National Corpus (BNC) and the results of the UKB WordNet-based Word Sense Disambiguation algorithm applied to the sentences where complex words appear. Thanks to the UKB algorithm, we identify the WordNet 3.0 synset that characterizes the sense of each target word (*WSD-synset*). Besides the target word, each *WSD-synset* usually has other lexicalizations, i.e. other synonyms. We retrieve the BNC frequency of all the lexicalizations of the *target-word-WSD-synset* and compute the following features:

- **Percentage of lexicalizations with higher / lower frequency than target word** (Lexic*Higher/Lower*FreqWSD - 2 features): the percentage of the lexicalizations of the *WSD-synset* with a BNC frequency higher / lower than the target word BNC frequency.

- **Ratio of total lexicalizations' frequencies related to lexicalizations with higher / lower frequency than target word** (Lexic*Higher/Lower*SumFreqWSD - 2 features): the ratio between the sum of BNC frequencies of the lexicalizations of the *WSD-synset* with a frequency higher / lower than the target word frequency and the sum of BNC frequencies of all the lexicalizations of the *WSD-synset*.

We also computed the previous set of 4 features without relying on the results of the UKB Word Sense Disambiguation algorithm: we considered for each target word all the lexicalizations of all the synsets that represent possible senses and have the same POS of the same target word. Similarly to the UKB based features, these features are referred to as: Lexic*Higher/Lower*FreqALL and Lexic*Higher/Lower*SumFreqALL.

6 Experiment and results

In order to identify the best approach to classify words as complex or not, we compared four classifiers by training on both the *Simple* and *Weighted* datasets. We evaluated the classification performance by means of a 10-fold cross-validation. Results are summarized in Table 1.

Classifier	Dataset	Precision	Recall	G-Score	F-Score
Random Forest	Simple	0.746	0.756	0.582	**0.735 (run 1)**
	Weighted	0.836	0.823	0.780	**0.824 (run 2)**
Support Vector Machine	Simple	0.685	0.707	0.707	0.650
	Weighted	0.728	0.718	0.718	0.719
Logistic Regression	Simple	0.667	0.697	0.476	0.659
	Weighted	0.733	0.734	0.745	0.733
Naïve Bayes	Simple	0.654	0.613	0.594	0.626
	Weighted	0.706	0.708	0.750	0.705

Table 1: Comparison of the performance of four complex word binary classifiers by a 10-fold cross-validation over the Task 11 training dataset.

Table 1 shows that the best performance in terms of F-Score was achieved by the Random Forest classifier for both approaches (*Simple* and *Weighted*). As a consequence, the two systems we submitted to Task 11 relied on a Random Forest model respectively trained on *Simple* (unweighted, run 1) and *Weighted* (run 2) instances. Our run based on *Weighted* instances performed quite well, ranking as the third best system in Task 11 with a G-Score of 0.772, where the G-Score of the best performing system is 0.774. With respect to F-Score our best performing run was the one based on *Simple* instances that ranked as sixth.

In Table 2 we show the top 10 features in our feature set in terms of information gain.

Info-gain	Feature name
0.37865	ENwikiFrequency_position_0
0.33303	BNCFrequency_position_0
0.18752	WNSynsetN
0.18439	WNGloss
0.14452	Dale_Chall
0.13596	LemmaLowerFreqALL
0.10567	WNdepth
0.10558	LemmaLowerSumFreqALL
0.08037	WNDistinctPOS
0.06244	WNSenseN

Table 2: Top 10 features with respect to information gain.

We can see that the frequencies of the word to evaluate in the two corpora we considered (English Wikipedia and British National Corpus) constitute the two most informative features. Five of the top 10 features are computed by relying on WordNet, without performing Word Sense Disambiguation: among them we can find the number of synsets (senses) of the word to evaluate (WNSynsetN), the average length of the glosses of these synsets (WNGloss) and the average depth of these synsets in the hypernym tree (WNDepth). Other useful indicators of word complexity are the presence of the word in the simple words list of Dale & Chall and the set of lexicalizations (of the synsets associated to the word) characterized by a frequency in the British National Corpus lower than the frequency of the word to evaluate (LexicLowerFreqALL and LexicLowerSumFreqALL).

In Table 3 we show the performance of the four classification algorithms we considered by training them on the whole *training dataset* (with *Simple* or *Weighted* instances) and testing them on the *testing dataset*. The best performance in terms of both F-Score and G-Score are achieved by the two Random Forest classifiers that were trained respectively on *Simple* (unweighted) and *Weighted* instances. In general, when we train the classifiers on *Weighted* instances in place of *Simple* ones, on the one hand both recall and accuracy improve, thus resulting in a higher G-Score, on the other hand the precision decreases, thus resulting in a lower F-Score.

Classifier	Dataset	Precision	Recall	G-Score	F-Score
Random Forest	Simple	0.186	0.673	0.750	0.292 +
	Weighted	0.164	0.736	0.772 *	0.268
Support Vector Machine	Simple	0.132	0.406	0.549	0.199
	Weighted	0.103	0.720	0.706	0.180
Logistic Regression	Simple	0.131	0.454	0.588	0.203
	Weighted	0.086	0.804	0.682	0.156
Naïve Bayes	Simple	0.083	0.769	0.670	0.151
	Weighted	0.079	0.787	0.656	0.144

Table 3: Comparison of the performance of four complex word binary classifiers. Each classifier is trained on the whole *training dataset* and tested on the annotated *testing dataset*. The asterisk symbol (*) points out the best performing classifier by G-Score while the plus symbol (+) the best performing classifier by F-Score.

7 Conclusions

In this paper, we described our participation to SemEval-2016 Task 11 concerning Complex Word Identification. We presented and evaluated our system based on both the characterization of words by means of contextual, lexical and semantic features and the exploitation of a Random Forest classifier to decide if a word is complex or not.

As future work we are planning to expand the feature set that we consider to characterize words by relying on new corpora and lexical resources. Moreover, we would like to explore complementary approaches to take advantage of distributional representations of words (i.e. word embeddings) or other language models to determine words complexity.

Acknowledgments

This work is partly supported by the Spanish Ministry of Economy and Competitiveness under the Maria de Maeztu Units of Excellence Programme (MDM-2015-0502) and the ABLE-TO-INCLUDE Project (Competitivity and Innovation Programme of the European Commission, CIP-ICT-PSP-2013-7/621055).

References

Eneko Agirre and Aitor Soroa. 2009. Personalizing Pagerank for Word Sense Disambiguation. In *Proceedings of the 12th Conference of the European Chapter of the Association for Computational Linguistics*, pages 33–41. Association for Computational Linguistics.

Bernd Bohnet. 2010. Very High Accuracy and Fast Dependency Parsing is not a Contradiction. In *Proceedings of the 23rd International Conference on Computational Linguistics (Coling 2010)*, pages 89–97, Beijing, China, August.

Stefan Bott, Luz Rello, Biljana Drndarevic, and Horacio Saggion. 2012. Can Spanish Be Simpler? LexSiS: Lexical Simplification for Spanish. In *Proceedings of the 24th International Conference on Computational Linguistics (CoLing 2012)*, December.

Kevyn Collins-Thompson. 2014. Computational Assessment of Text Readability. A Survey of Current and Future Research. *ITL - International Journal of Applied Linguistics 165:2*, 165(2):97–135.

Edgar Dale and Jeanne S. Chall. 1948. The Concept of Readability. *Elementary English*, 23(24).

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.

Robert Gunning. 1952. *The Technique of Clear Writing*. McGraw-Hill.

J. Peter Kincaid, Robert P. Fishburne Jr., Richard L. Rogers, and Brad S. Chissom. 1975. Derivation of New Readability Formulas for Navy Enlisted Personnel. Technical report, Naval Technical Training Command.

Geoffrey Leech and Paul Rayson. 2014. *Word Frequencies in Written and Spoken English: Based on the British National Corpus*. Routledge.

G. Harry Mc Laughlin. 1969. SMOG Grading - a new Readability Formula. *Journal of Reading*, pages 639–646, May.

George A. Miller. 1995. WordNet: a Lexical Database for English. *Communications of the ACM*, 38(11):39–41.

Lluis Padró and Evgeny Stanilovsky. 2012. FreeLing 3.0: Towards Wider Multilinguality. In *Proceedings of the Language Resources and Evaluation Conference (LREC 2012)*, Istanbul, Turkey, May. ELRA.

Philip T. Quinlan. 1992. *The Oxford Psycholinguistic Database*. Oxford University Press.

Luz Rello, Ricardo Baeza-Yates, Laura Dempere-Marco, and Horacio Saggion. 2013. Frequent Words Improve Readability and Short Words Improve Understandability for People with Dyslexia. In *Human-Computer Interaction - INTERACT 2013*, pages 203–219.

Horacio Saggion, Stefan Bott, and Luz Rello. 2016. Simplifying Words in Context. Experiments with Two Lexical Resources in Spanish. *Computer Speech & Language*, 35:200–218.

Matthew Shardlow. 2013. A Comparison of Techniques to Automatically Identify Complex Words. In *ACL (Student Research Workshop)*, pages 103–109. The Association for Computer Linguistics.

Luo Si and Jamie Callan. 2001. A Statistical Model for Scientific Readability. In *Proceedings of the Tenth International Conference on Information and Knowledge Management*, CIKM '01, pages 574–576, New York, NY, USA. ACM.

Ian H. Witten and Eibe Frank. 2000. *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA.

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 *Proceedings of the Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics*, pages 365–368.