Complex Word Identification: SemEval 2018 Shared Task

170122023

1 Abstract

This report outlines the methodology and final system used in the SemEval 2018 Complex Word Identification task. Two languages, English and Spanish, were evaluated and a binary classification of simple or complex for each word was provided. Semantic features of the words were extracted and used to predict complexity. Three *scikit-learn* classification algorithms were used, a Random Forest Classifier, a Neural Network and Logistic Regression. The best performing system used a Random Forest Classifier and achieved an F1-Score of 0.83.

2 Introduction

The task was to predict which words had been deemed to be complex by a sample of native and non-native English and Spanish speakers. Complex Word Identification (CWI), is a component of lexical simplification that focuses on identifying certain words or short phrases that are challenging to understand. By identifying the complex words, simpler words can be used to replace them, thus making it easier for a non-native, dyslexic or young person to understand. It can also be extended to show what are the common characteristics of words which certain groups finds challenging. The data consisted of news articles, witten by both professionals and amateurs, as well as Wikipedia pages. 10 of both native and non-native speakers were given a sentence and asked to annotate which words they thought would be found complex by other readers. Both individual words and phrases could be tagged as complex.

3 Features

Features from the text that were deemed to be indicative of the complexity of the word were extracted and used to train the model. A number of features were tested but only the following were chosen to be included in the final model.

3.1 Baseline

These were the two initial features provided. Firstly, there was the relative word length compared to the average word length for that language, with the rationale that longer words tended to be more complex. The other was the number of words in the target phrase.

3.2 Frequencies

The frequencies of words, letter unigrams, bigrams and trigrams in the training set were each implemented as a feature. The training data was first read through to build four dictionaries which shored the counts for each of the above. The reasoning for these features was that the less common a certain word or letter sequence was, the more likely it is that it is a complex word. For example the letter trigram *ing* would be very common and therefore is likely not as complex as another letter trigram such as *syn*. The average count for each word or letter in the target sentence was used as the feature for the word.

3.3 Average Senses

(Davoodi and Kosseim, 2017) raised the idea that the number of different senses a word has could be used to determine its complexity. It was suspected that words with many other meanings were likely to be less complex than those with only a few. The NLTK lexical database WordNet was used to identify the number of senses each word in the phrase had. The implementation of this involved taking the average number of senses for the words in the sentence with the expectation what a sentence with a lower number would indicate complexity. The total could not be used as it would weight longer sentences much higher that shorter ones. The minimum value was also tried but the resulting F1 score was slightly lower than when taking the average. One of the limitations of this method was that words such as and, the and to had been tagged as having no other senses, but are clearly not complex words. Also, words that were not present in the corpus were tagged as 0. This was a problem as a sentence with many of these words would have a far lower average than another equally complex sentence. The solution was to not include these words in the averaging, effectively ignoring their presence.

3.4 Average Syllables

The number of syllables in a word was deemed to be an indicator of the words complexity, with more syllables indicating a complex word. The NLTK lexical database cmudict was used as a means of counting the syllables in the target word. Both the average number of syllables in the target words and the number of syllables of the word with the most syllables was tested with the maximum returning the highest f1-score, and so was used instaed af the average.

3.5 Part of Speech Tagging

Another feature that was implemented was extracting the parts of speech tags from the sentence. It was thought that a certain part of speech such as a comparative adverb may be more complex than a conjunction. This could be extended to comparing the complexity of the same word but used in a different sense. For example, the sentence *she banked the plane* is likely to be deemed more complex than the sentence *she went to the bank*. This could be identified by extracting the verb and noun POS tags for the word bank depending on the context. The NLTK part of speech tagger was used and the feature was a one hot encoded vector of the POS tags of the words. POS tagging was also implemented by (Kumar Sanjay and K P, 2016) in a similar way, but extended to look at POS tag sequences.

3.6 Capital Letters

The capital letters feature was included as it was seen from the dataset that there were many instances of words which have a capitalized first letter. It is likely that some link could be drawn between the complexity of the word. Perhaps names and locations would be deemed to be less complex than other words. Both the sum of capital first letters and the average was tested with the sum performing better and subsequently was selected for the final model.

4 Results

The following results were acheived using a training set size of 27299 and a testing set size of 4252. The results for how each algorithm performed when using all of the possible features, for each language are shown in *Table 1* below. A comparison of how each feature performed individually on the English dataset and using a random dforest classifier are presented in *Table 2*.

Model	English	Spanish
Random Forest Classifier	0.829	0.733
Neural Network	0.744	0.719
Logistic Regression	0.759	0.731

Table 1: F1-Scores using all features

Feature	F1-Score
Baseline	0.707
Word Frequency	0.683
Letter Unigrams	0.767
Letter Bigrams	0.759
Letter Trigrams	0.731
Average Syllables	0.621
Average Senses	0.620
POS Tags	0.631
Capital Letters	0.402

Table 2: Feature f1-scores using Random Forest

Label	Recall	Precision	F1
0	0.87	0.85	0.86
1	0.79	0.81	0.80

Table 3: Detailed System Performance

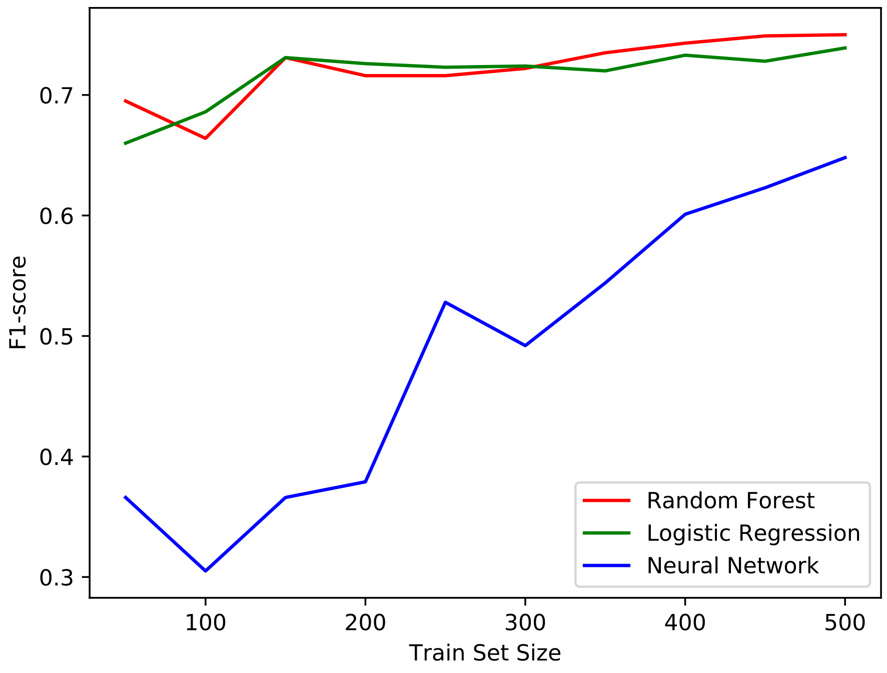

Figure 1: How trainset size affects performance

5 Discussion

The Random Forest Classifier achieved the best results for both languages with English and Spanish yielding F1-scores of 0.829 and 0.733, respectively. This was followed by Logistic Regression, with Neural Networks the least effective of the three.

(Davoodi and Kosseim, 2017) tested a number of algorithms that classified word complexity. They started with a baseline of Naive Bayes and implemented a Neural Network, a Decision Tree Classifier and A Random Forest Classifier. Their findings indicated that the Random Forest was the most effective, with both of their final models being so, with slightly different parameters. These models were placed between 20th and 30th for the 2016 SemEval Task, well above the average.

Each feature is analyzed independently using a Random Forest Classifier. Letter unigrams Achieved the highest F1-score of 0.767. It can be seen that the features which show the frequencies of an aspect of the word compared to the rest of the training data, are more effective than the more complex ones such as POS tags and average senses which require existing semantic information. This is encouraging as it means that a system which only used frequency based features may still be effective on a new language where there is no access to a semantic database of parts of speech, for example.

Figure 1 shows how the f1-scores improve with the size of the training set. The language tested on was English, the whole feature set was used and the testing size remained unchanged. As can be seen, both the Random Forest Classifier perform well on a small data set of only 50 files. Their f1-scores only increases by 0.1 when looking at the full training set. The Neural Network, however, performes significantly worse when only looking at a small dataset. For the system in question the f1-score converges to 0.648 at 500 training instances, with a score of 0.738 achieved with 27299 instances. The conclusion that can be drawn from this is that Neural Networks would not be suitable for learning a new language where there was limited data available, however, it would be interesting to see if its performance surpassed those of Random Forest Classifiers and Logistic Regression if there was a much larger training dataset available than the one provided.

One aspect of the graph that needs highlighting is the variance in the neural network scores. This happens when training on larger trainsets as well as smaller sets, but to a lesser extent. One means of combating this would be to use cross validation to train and test on different samples of the data, and then take the average to return a more representative score.

Table 3 shows a detailed summary of the best performing system. For each tag, 0 for a simple word and 1 for a complex word, it shows the recall, the proportion of the relevant words that were successfully returned, and the precision which indicates the proportion of the results that are returned are relevant. It can be seen that the system is better at classifying simple words as simple than complex words as complex, with better scores for both recall and precision. These results are supported by similar results shown in (Zampieri et al., 2017)

The capital letters feature returned some unexpected results. When tested upon independently, it performed very poorly with an F1 score of 0.402, but when implemented with the rest of the feature set, the system accuracy improved. This suggests that while it is a weak classifier in general, it performed well on classifying words that the rest of the features were not successful with.

6 Future Work

The first extra feature that would be looked to be implemented would be word embeddings. (Yimam et al., 2018) outlined the successful use of these for a number of entries to the 2018 SemEVal task. Word embeddings are features used in natural language processing whereby each word in your vocab set is mapped to a number in a vector. They allow words with similar semantic meaning to be represented numeriacally by computing the cosine similarity between the two vectors.

To further improve on this system, work could be done to investigate the effect of the parameters of the machine learning algorithm on the effectiveness of the system. More algorithms could also be implemented such as a Naive Bayes. Furthermore, an analysis on whether there were certain types of words used in certain contexts that the system is inneffective in classifying would be of merit.

References

Elnaz Davoodi and Leila Kosseim. 2017. Clac at semeval-2016 task 11: Exploring linguistic and psycho-linguistic features for complex word identification. *CoRR*, abs/1709.02843.

Anand sp Kumar Sanjay and Soman K P. 2016. Amritacen at semeval-2016 task 11: Complex word identification using word embedding. In *Proceedings of the 10th International Workshop on Semantic Evaluation (SemEval-2016)*, pages 1022–1027, San Diego, California. Association for Computational Linguistics.

Seid Muhie Yimam, Chris Biemann, Shervin Malmasi, Gustavo H. Paetzold, Lucia Specia, Sanja Stajner, Anaïs Tack, and Marcos Zampieri. 2018. A report on the complex word identification shared task 2018. *CoRR*, abs/1804.09132.

Marcos Zampieri, Shervin Malmasi, Gustavo Paetzold, and Lucia Specia. 2017. Complex word identification: Challenges in data annotation and system performance. *CoRR*, abs/1710.04989.