Class project: Complex Word Identification

Pierre Finnimore

Abstract

1 Introduction

The task is as follows: given a target word (or set of words) within a sentence, identify if the target is "complex". The data given for this task is a set of labelled sentences with targets. The labels were derived from a survey of both Native and Non-native speakers of two languages: English and Spanish.

We are interested in identifying word complexity for several reasons. Automatic extraction of complex terms could help with automated tutoring systems, Natural Language Generation, writing editing software, studies into second-language acquisition, political speech analysis, Machine Translation, as well as linguistic or psychological studies into the what people find complex.

2 Baseline system description

System descriptions in enough detail for the reader to be able to understand how to reimplement your baseline models and to appreciate why they are suitable for the task at hand.

The baseline system we developed was somewhat more advanced than the initial baseline provided. Both languages used an identical model and set of features. We focused on features that can be extracted using only the target word; no context words were considered. First, we improved the basic architecture of the provided baseline to allow easy pattern-matching of prefixes, infixes and suffixes, and for further features of this type to be easily added.

Our initial hypotheses to try for the baseline were as follows:

1. Letter rarity - this feature was chosen because words with rarer letters might be harder to understand. This feature could be seen as a vague approximation of word rarity, which would otherwise require more data to learn.

2. Max consecutive consonants, max consecutive vowels - the idea behind this was that dense combinations of letters might be difficult to parse. For example "queueing" or "rhythms". These words deviate from the more straightforward consonant-vowel-consonant-vowel pattern.

3. Uniquely English vowel/consonant diagraphs/consonant blends - this was chosen because a non-English speaker might find these tricky. This feature could be expanded (and made more adaptable to different languages) by analysing a corpus for particularly rare combinations.

4. POS tags - because some POS are part of closed sets, and so potentially easier to understand. In addition, certain rarer tags might be harder to comprehend, especially in languages where the overall form of the word is changed, depending on its POS.

5. Number of synonyms - The idea was that ambiguous words might be more confusing. Conversely, perhaps ambiguous words are actually less likely to be regarded as complex, since it is more likely that the person knows at least one of the meanings.

While individually, these features did allow the system to predict with greater-than-random accuracy, most of them did not improve over the initial baseline. Some of the features we tried, such as spanish prefixes, reduced the testing accuracy if they were added. This may just be noise in the

data, or it may be that learning is made more difficult if irrelevant or common features are added. This may be less of a problem with a different learning model.

We quickly identified that POS tags were computationally costly to include, while not providing improvements to performance, and so removed them. Our final Baseline System used the following features:

1. Number of characters in the target
2. Number of words in the target
3. "Rarity" score based on letters of the target
4. Maximum number of consecutive vowels
5. Maximum number of consecutive consonants
6. Number of synonyms
7. Common English and Spanish Suffixes
8. Common English and Spanish Prefixes
9. Common English and Spanish Infixes
10. Common Latin and Greek prefixes

Language	LR	RFC
English	0.73	0.81
Spanish	0.74	0.75

Table 1: Macro-F1 Score for baseline system, with logistic regression (LR) and random forest classifier (RFC)

Language	LR	RFC
English	0.77	0.82
Spanish	0.77	0.74

Table 2: Macro-F1 Score for advanced system, with logistic regression (LR) and random forest classifier (RFC)

It achieved performance of 0.73 on English and 0.75 on Spanish with a logistic regression approach, and

3 Improved system motivation and description

Since basic logistic regression was more effective for Spanish, we separated the model approaches for the two languages.

One of the features added for

We found a list of "1000 simple English words" on Wikipedia, and used these to extract sub-word features of two or three letters long. The idea was that these words may contain certain patterns that people regard as simple, and each sub-word feature was weighted by how often it occurred in the text. Interestingly, introducing these features improved the Spanish F-score, but not the English F-score. This may be because those combinations also happened to correlate with easy or difficult morphological features in Spanish, or it may be that the participants of the original study found that combinations that they recognised from English were easier or more difficult to identify in Spanish.

The single largest improvement to the model was obtained by using a Random Forest Classifier rather than a Logistic Regression classifier. To understand why it is effective, it is easiest to start with thinking about a simple decision tree. A decision tree is a sequence of decisions, based on the features supplied, which partition the result space. If we were making our own decision tree for this task, we might think something like "Is the word longer than 10 letters? If yes, it is complex. If no, does the word end with 'ough'? If yes, it is complex. Otherwise, it isn't.". This set of rules, which partition the classes effectively, can be learned by the computer.

However, there are issues with simple decision trees. Since we do not know ahead of time what the optimal partitions are, we take an approach where we see at each decision point how to partition the features such that it splits the data. But this will only guarantee locally good splits: there may be a different split that leads to globally better partitions. In addition, they are likely to overfit as the partitions get small. This is where Random Forests are useful. They can take a set of simple decision trees, each trained on a different subset of the features, and make a decision based on all of those smaller decisions. In a way, we could think about each simple decision tree extracting some feature or cluster of features, like the "length + ough-suffix", and the Random Forest learns how to make decisions based on these more sophisticated, abstracted features.

This is somewhat analagous to the hidden layers of a neural network, where first we reduce the dimensionality of our data to features, then we reduce the dimensionality of our features to a set

of abstracted feature-combinations, and only *then* make our decision.

There are several reasons that we might think that a Random Forest Classifier, which produces these feature-combinations, would be a promising candidate for a task such as this.

First, (Treeratpituk and Giles, 2009)

Lastly, there is significant precedent for use of this system

When we used the Random Forest Classifier, we took a look at the most important features for each model. As expected, "Number of characters in target" and "Number of tokens in target" were the number 1 and 2 spots for importance for both languages. Interestingly, the "Rarity score", which we developed as a basic measure of the use of rare letters in the word was the 3rd most important feature for English and 4th most important feature for Spanish.

It is important to note that we may be getting the wrong impression from these importance scores, since each feature is measured independently. While there is only one "Number of synonyms" feature, there are 312 "Two-letter infix" features and *1297* "Three-letter infix" features. While any given infix feature may be less significant than these more comprehensive, overall features, the sum total of their effect may be greater.

Rank	Feature	Importance
1.	Number of characters	(0.106)
2.	Synonym count	(0.0397)
3.	Number of tokens	(0.0378)
4.	Rarity score	(0.0358)
5.	Max cons. consonants	(0.0142)
6.	Suffix: ed	(0.00915)
7.	Bi: ed	(0.00894)
8.	Infix: r	(0.00851)
9.	Max cons. vowels	(0.00737)
10.	Bi: ti	(0.00669)

Table 3: Top 10 English Features

4 Experiments on development set

Does your idea work as expected? Evaluate on the test set the baseline and the improved system, is it still the case? Identify examples in development data which help showcase why the improved system works better.

Some of the ideas that we tried worked, but not in the expected way. For example, the case mentioned in the previous section, where adding subword features from 1000 simple English words improved the Spanish score. However, it appears that there is some linguistic research to suggest why this may be the case: Morphology may be particularly important to Spanish learners.

Rank	Feature	Importance
1.	Number of tokens	(0.0894)
2.	Number of characters	(0.076)
3.	Rarity score	(0.0441)
4.	Max cons. consonants	(0.0246)
5.	Bi: de	(0.0115)
6.	Synonym count	(0.0109)
7.	Infix: r	(0.0099)
8.	Max cons. vowels	(0.00914)
9.	Suffix: a	(0.00805)
10.	Suffix: s	(0.00725)

Table 4: Top 10 Spanish Features

Data	System	Language	Score
Dev.	Baseline	English	0.99
Test	Baseline	English	0.99
Dev.	Advanced	English	0.99
Test	Advanced	English	0.99
Dev.	Baseline	Spanish	0.99
Test	Baseline	Spanish	0.99
Dev.	Advanced	Spanish	0.99
Test	Advanced	Spanish	0.99

Table 5: Macro-F1 Score on Development and Test data, for different systems and languages.

5 Learning curves

In general, the systems seen in Figs. 1 and 2 indicate that the Spanish data is more difficult to learn from, with a lower starting point and shallower improvements as more data was added. This may be a peculiarity of the dataset that we had, or representative of the nature of the language itself.

The curves for the logistic regression approach in Fig. 2 are converging to a greater extent than those of the Random Forest Classifier seen in Fig. 1. This suggests that adding more training data would do little for the logistic regression approach, whereas there are still potential gains to be had for the Random Forest Classifier, if more data were available.

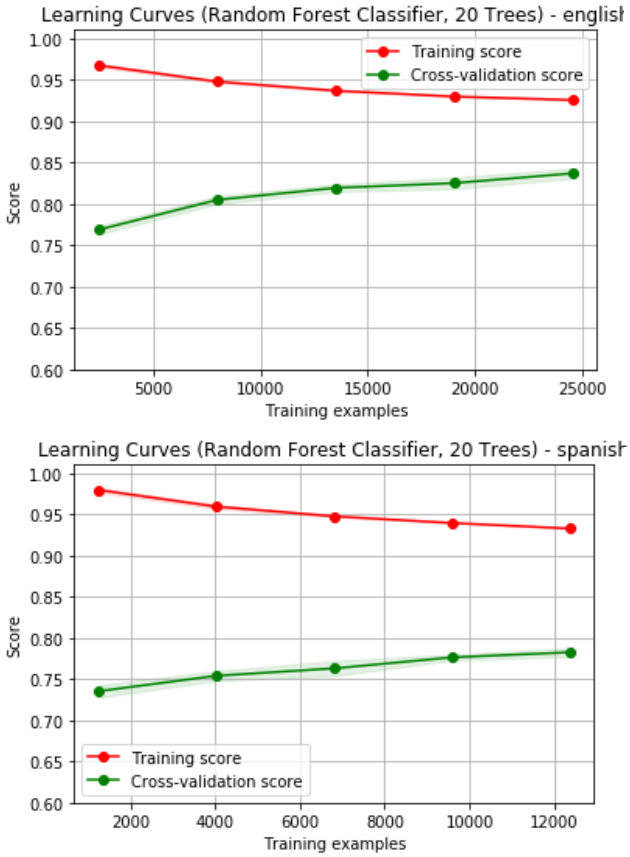

Figure 1: 5-fold cross validation performance on English and Spanish training data for a Random Forest Classifier with 20 Trees

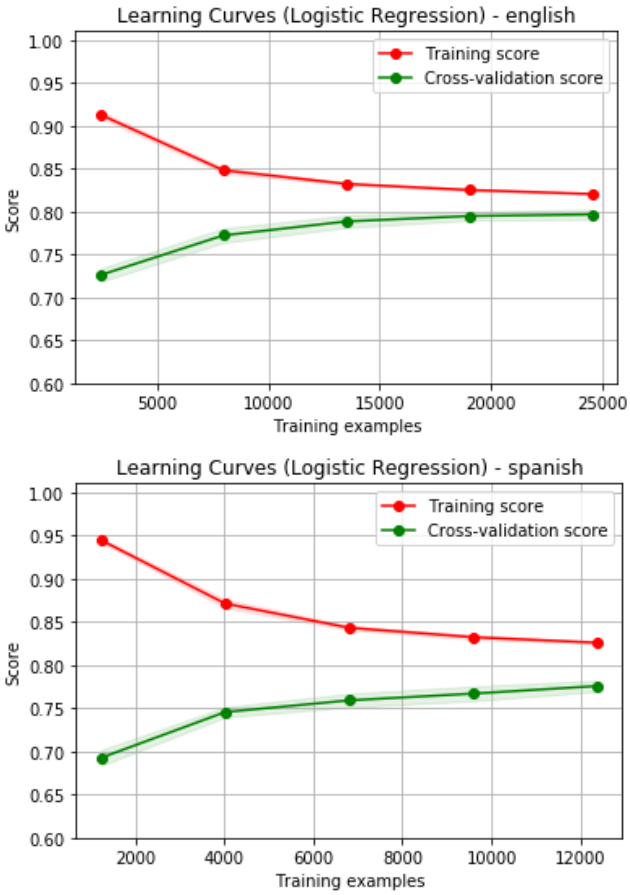

Figure 2: 5-fold cross validation performance on English and Spanish training data for a Logistic Regression

With little data available, the Random Forest Classifier still outperforms the Logistic Regression approach. In general, the approaches that we would expect to take the most data would be those based on neural networks, since back-propagation requires very small steps along the gradient. However, since we did not implement these techniques, we did not encounter this difficulty.

6 Examples of failed predictions

Identify examples where your improved system fails to predict correctly and propose ideas for future work to address them.

7 Conclusions

what have we learnt from your experiments that could inform future work

Our main approach to this task was to find some data, then try to extract as much as we could from that data. We primarily focussed on subword features, as these effectively reduce sparsity. These subword features proved to be effective at this task, suggesting that at least one component of perceived complexity is simply the structure of the word's constituent letters.

References

Pucktada Treeratpituk and C Lee Giles. 2009. Disambiguating authors in academic publications using random forests. In *Proceedings of the 9th ACM/IEEE-CS joint conference on Digital libraries*, pages 39–48. ACM.