A Customizable Thesaurus

McGill University

LING 550: Computational Linguistics

Rosie Barnes (260622130) and Alice Tilles (260579661)

December 12, 2016

Introduction

For the final project, we decided to make a customizable thesaurus. Like a regular thesaurus, a user can look up words and it will return a list of synonyms. However, this program outputs synonyms based on the user's preferences. Specifically, the program allows the user to input a word, and choose to either British-ify or American-ize it. The results given to the user are: (1) a list of the most common synonyms that a person speaking the goal dialect would use, (2) a list of which of the synonyms are disproportionately used by those speakers, and (3) a similarity analysis to ensure the synonyms provided are accurate.

To run the program, no arguments are needed. After downloading the prerequisite databases and tools, enter on the command line:

```
>> python[3] wordConversion.py
```

The prompts are all user-friendly. Some words with a lot of synonyms to try: *quit, big*

Motivation

There are many applications for our program, ranging from trivial to meaningful. We see it being used in any situation when someone needs to take on someone else's voice. For instance,

an author might use it when writing a character whose accent and word choice they are not perfectly in tune with. Or, a politician (or their speech writer/publicist) may alter a speech with words that will resonate more with the audience they are speaking to. Alternatively, it can be used 'reversely' -- if you come across an unfamiliar word, this program might be able to 'translate' it into a word you know.

Related Research and Tools Used

We could not find any similar synonym-generating programs. The most pertinent research we found was "*Semantic Variation in Idiolect and Sociolect: Corpus Linguistic Evidence from Literary Texts*" which looked at texts written by different demographics, such as age and gender, and the time period it was written in. They tried to find patterns common within each demographic. But, they did not find good results, attributing it to low homogeneity within a text. We added the "disproportionate probability" to overcome this possible set-back.

The databases for word frequencies we used are from Subtlex-UK and Subtlex-US. The British word frequencies are based on a corpus of 201.3 million words from 45,099 BBC broadcasts. The American has database includes 51 million words from 8,388 American films and television show episodes. The base thesaurus we chose was WordNet because of its simple NLTK interface, as well as its extensive network of related words. It contains 117,659 synsets (sets of words considered semantically equivalent) and 155,287 unique lemmas ('words').

Description of the Implementation

To calculate the most common words of the goal dialect, we divided the frequency count in the desired Subtlex database of each synonym by the sum of the all synonym's frequencies in the database. This resulted in a dictionary of each synonym's share percentage in the goal dialect. This was the first set of words outputted to the user, ranked by highest to lowest share.

For the disproportionate words, we first calculated the shares of each synonym in both the original and the goal dialect using the method above. With those, a Python dictionary was compiled, where each key was a synonym, and each value was the disproportionate probability, calculated by dividing the pre-translation share by the post-translation share.

$$Disproportionate\ Probability\ of\ Word\ A\ = \frac{Pre{-}Translation\ Share\ of\ Word\ A}{Post{-}Translation\ Share\ of\ Word\ A}$$

This resulted in a range from just over 0 to over 3000. The higher this number, the less like the goal dialect it is. The lower the number, the more like the goal dialect it is -- these are the words presented to the user. We scaled these numbers into readable analysis for the user: *"[Word] is [description] for an [American/British] person to say."* Descriptions include *much, much more, more*, and *a little bit more*.

We include similarity measures using data outside of WordNet to be extra safe in recommending synonyms generated by WordNet. To calculate similarity, we used the UMBC (University of Maryland, Baltimore County) Semantic Similarity Service to compare the inputted word to each synonym. This algorithm returns a number between 0.0 and 1.0; 0.0 represents completely dissimilar words, and 1.0 represents identical meanings. Then, we made a gradient of similarities based on these numbers. If the number was less than .5, we recommended they use it with discretion. If the number was less than .1, we advised them not to use it all, and recommended them a better word with a higher similarity measure, if available.

Shortcoming and Future Development

One of the main problems with our software application is that words in the frequency databases we used (SUBTLEX-UK and SUBTLEX US) are not marked for word sense. This is a problem as it may be that the output does not have the same sense as the desired translation of the inputted word. Furthermore, the synonyms generated by WordNet are not perfect, hence the translations will not always be accurate. Finally, with regards to thoughts on future development, additional parameters should be implemented. Some additional parameters we had devised include age, gender, and time period.

References

Brysbaert, M. & New, B. (2009) Moving beyond Kucera 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 (4), 977-990.

Fellbaum, Christiane (2005). WordNet and wordnets. In: Brown, Keith et al. (eds.), Encyclopedia of Language and Linguistics, Second Edition, Oxford: Elsevier, 665-670

Van Heuven, W.J.B., Mandera, P., Keuleers, E., & Brysbaert, M. (2014). Subtlex-UK: A new and improved word frequency database for British English. Quarterly Journal of Experimental Psychology, 67, 1176-1190.

"UMBC Semantic Similarity Service." *UMBC Semantic Similarity Service*. Web. 2 Dec. 2016. <http://swoogle.umbc.edu/SimService/>.