Omar Ortiz
Methods in Computational Linguistics 1
FINAL

Introduction:
The goal of my code for this final project is to extract specific tokens, and count how many times those tokens appear in a given corpus. The idea comes from my current MA thesis, where I am looking at THOUGHT and LOT vowels in the speech of native New-Yorkers. A well-known New York city English feature is having a distinction between the vowel sound in THOUGHT (caught, taught, talk, etc.…) and LOT (cot, tot, bot, etc.…). Where the THOUGHT vowels have a higher raised F1 value compared to the LOT vowels. My MA thesis focuses on the potential merger-in-progress that is happening, being led by young people and people of E. Asian and Latinx backgrounds. Figure 1 below shows a person who has the THOUGHT-LOT distinction, where the green color represents THOUGHT tokens and orange bubble represents LOT tokens.

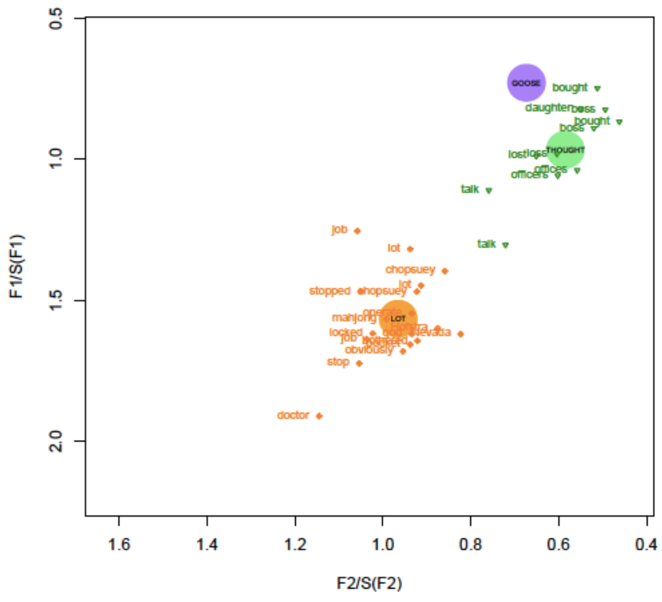

Fig 1 - Wong (2012): The words (tokens) belonging in the THOUGHT category are always distinct (higher) than the tokens in the LOT category for this speaker.

Part of my work has me recording individuals reading a wordlist and sentences, this is done in order to not put focus on the target vowel (distractor words) and to produce natural speech (better for authentication). The code created I imagined would be used if someone sent me a giant corpus (wordlist) and my goal was to 1, see if my target (token) words are in the wordlist, and 2, count how many times each word appears.

The code:
For PART 1, to run my code you'll need to have a document with my wordlist saved on your computer titled "wordlist3" saved as a.txt file There are six THOUGHT vowels I am searching for (these are the six words in my MA thesis I am working with) and I used regular expressions to

find them, referencing MP4. The words “caught, taught, and bought” in the English language never appear with letters before or after them, they are always “free” not “bound” words. So In the code, I was able to just re.search for them. But, the words “talk, dawn, and auto” in the English language can sometimes be “bound” (talkative, dawning, automobile), I included these in my “wordlist3.txt” document to show that these words exist, and that I do not want to extract or count them. That is the reason these words have “^” and/or “$” at the left edge or right edge.
The code prints the THOUGHTVOWELS, and in a separate cell I counted the individual words, with the code “LISTNAME.count(“EXAMPLE”). To confirm this was accurate, I also included some ASSERTS.

After “import re” I did decide to read the entire wordlist and then loop over it, this is because if I were to use the code “with open (……) as source:” I wasn’t able to do the counting. My code has “LISTNAME= []” and at the end “LISTNAME.append” so that I could count how many times a specific word appears. When I tried to use this code using “with open (…..) as source:” it did not work properly. I included this code in part 4 and talk about it below.

Part 2 was done the exact same way as part 1, but they are titled LOT vowels instead of THOUGHT vowels because that is what we are trying to extract. Originally, I tried to have the regular expressions search for clumps like “au” to extract words with these vowels, the problem arose when I realized other words in the English language have these vowels, and they do not belong to the THOUGHT or LOT vowel family. So I had to go more specific with the actual word, adding a “^” or “$” when needed. In particular, all the LOT vowels can be “bound” like *COTton or roBOT* so they all needed a “^” and “$” so that only the LOT vowels printed.

Part 3 was included so that whoever is running my specific code can have the wordlist (to be saved as “wordlist3.txt”. But also to show this is another way to count specific words. This way is more beginner, and requires you to input the wordlist, instead of reading it from a saved file on your computer. The other issue with this style is if you’re given thousands of words, it would not look neat to have to copy and paste a giant word list into one single cell. This code still works, where you can count how many times the word “caught” appears and it gives me the same answer I got in Part 1, it also works to accurately give me a “0” count when looking for a word not in the wordlist, like “potato”.

As mentioned in Part 1, I included Part 4 just to show the possible “simpler” code to do this, that prints out specific words using regular expressions. If there were a way to count the token words like I did with my “append” code – this would possibly be the “better” way to do this, since it is less code. Part 4 just shows how to extract LotVowels, although this could be done the same way to extract the ThoughVowels.

Practical implications:
If given a large wordlist, this can easily extract the words you are looking for, and count how many times they appear. This will help when splicing in Praat to double check the tokens extracted match the count generated in the code.

References

Gorman, K. (2019) ‘Regular Expressions’ [PowerPoint presentation] LING 78100. Available at https://wellformedness.com/courses/LING78100/PDFs/lecture12.pdf

Haddican, Bill, Daniel Ezra, Michael Newman & Faith Kim. The diffusion of the low back merger in New York City. Poster presented at: NWAV 45; 2016; NYC

Wing-mei Wong, Amy. 2012. The Lowering of Raised-THOUGHT and the Low-Back Distinction in New York City: Evidence from Chinese Americans. University of Pennsylvania Working Papers in Linguistics. Volume 18 Issue 2 Selected Papers from NWAV 40.