

Report 1

1. **Data.** Describe the file you selected and what the source was, as well as anything you found interesting about it.

I went to [Project Gutenberg](#) and merged [An Interrupted Night](#) by Pansy and [The Structure of the English Sentence](#) by Lillian Kimball Stewart into one text file for about 118,000 words. The first book was a straightforward storybook, with lots of dialogue and punctuation, such as quotation marks and em dashes (—). It was a very safe text file to start with. I looked over the second book, which was more of an educational book on the English language, which I found interesting because there would probably be more problems I would run into as the writing was more complex. This book was noticeably lengthier, with underscores, asterisks, digits, semicolons, dollar signs (\$), and more.

2. **Methodology.** Describe, at a high level, the approach you took to writing your software and generating your figures. What are the options that the user can select when performing the text normalization steps. What is the additional option that you added beyond the defined requirements? Why do you think this is a useful option?

I read the text file and processed it line by line as it was the only approach I was familiar with. The arguments were passed from function to function as a boolean to retain the options that the user wants. Some options were easier than others, such as lowercase, since all I had to apply a built-in function to the entire string. The rest of the options, such as stemming were more difficult since I would have to split the string and apply functions to each word individually to satisfy the criteria. I had to search on the internet and use generative AI for explanations and definitions since I didn't have a solid understanding of stemming lemmatization, and stopwords. Generative AI was extremely useful during lemmatization since I used it to confirm the predefined rules that I thought of, and at some point, I realized that lemmatization would require hard-coding since there are irregular words that require special handling no matter what. Also, I had to use generative AI to create a list of stopwords, since I didn't want to waste time thinking of 100 stopwords myself.

The additional option I added was to remove punctuation, as it is quite obvious that these punctuations hinder the software in identifying words, which is the main goal of this assignment. For example, we're interested in the word "black", but the textfile could have this word written like "black.", "black,", "black?", "black-eyed", etc. Since my approach was not perfect at first (and never will be), my word count list could have variations of a word, which would hinder the true word count for that particular word. Below is a visualization of the atrocity that could happen:

Word Counts:

black? 2

black-meteor 3

black. 2

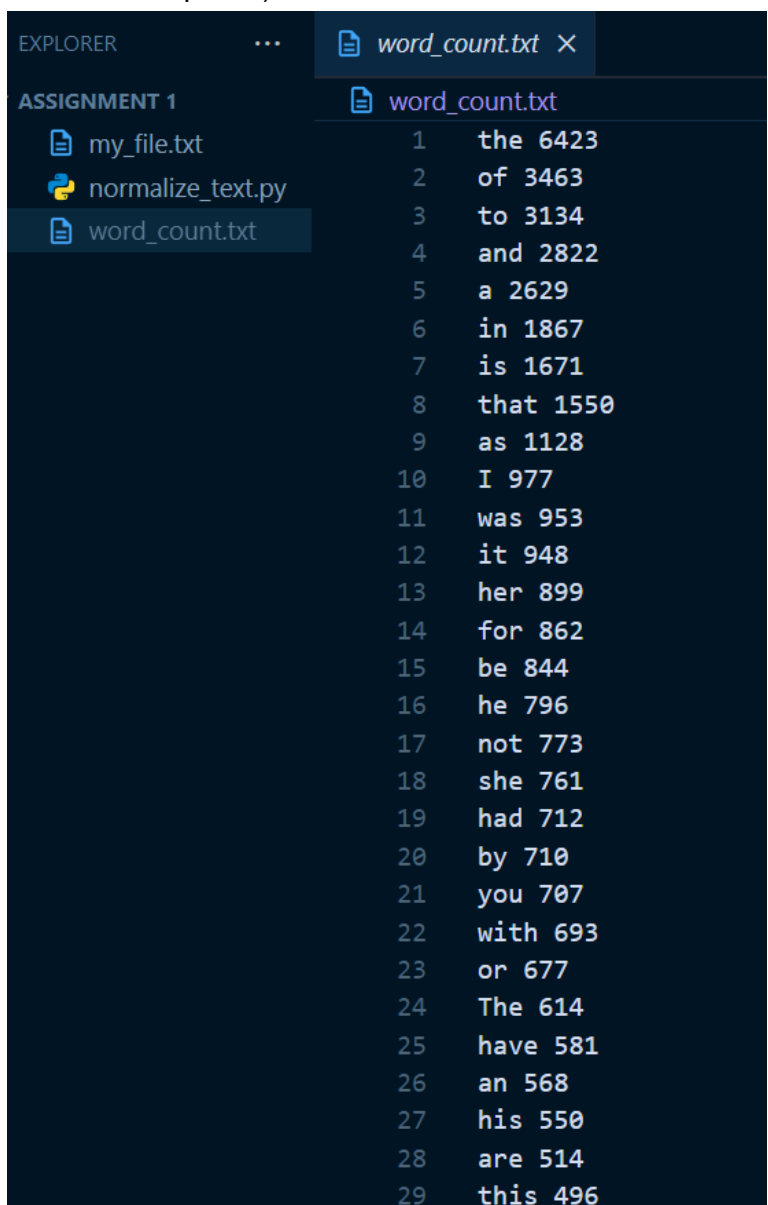
-black 6

3. **Sample Output.** You should show the first and last 25 words and their counts from your program's output, as well as the figures that you generated. The sample output should correspond to an input file that meets the requirements.

After running, the output file is generated successfully and the total tokens are shown.

```
derek@LAPTOP-M8CGD3CK ~/Downloads/4NL3 Natural Language/Assignment 1
● $ python normalize_text.py my_file.txt
-----
word_count.txt file written successfully | Total tokens: 118515
```

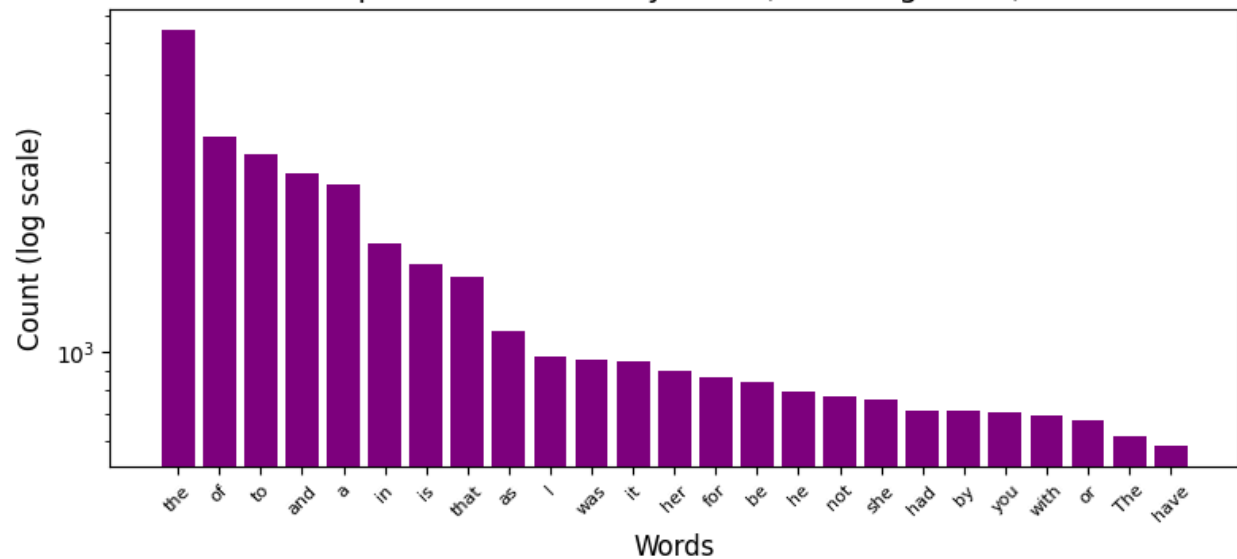
The word_count file that is generated contains all words and their counts (18862 lines total without options)



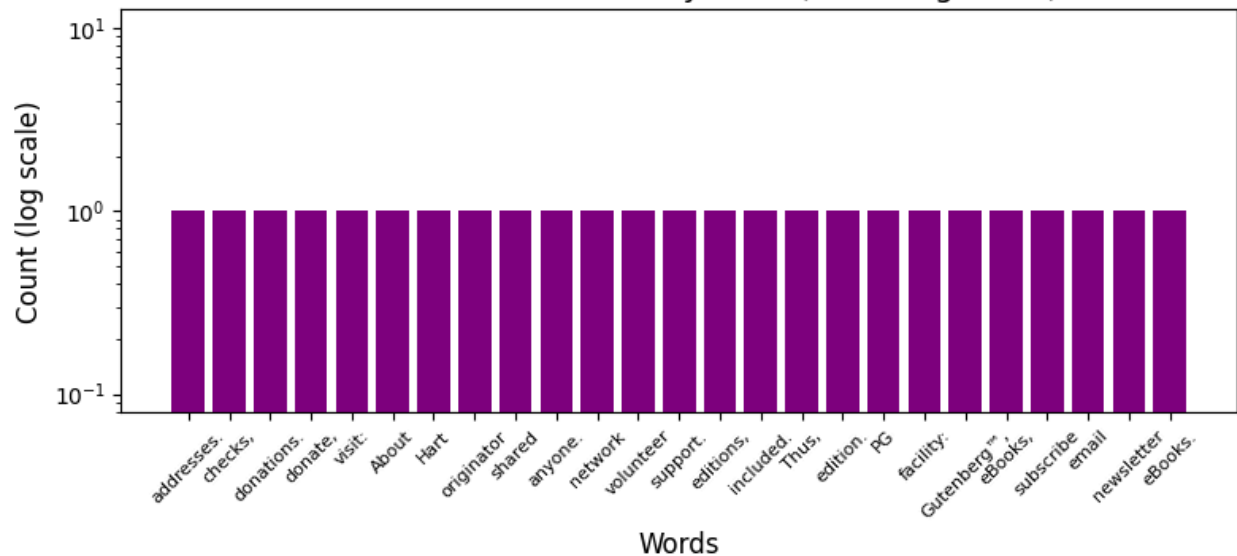
1	the	6423
2	of	3463
3	to	3134
4	and	2822
5	a	2629
6	in	1867
7	is	1671
8	that	1550
9	as	1128
10	I	977
11	was	953
12	it	948
13	her	899
14	for	862
15	be	844
16	he	796
17	not	773
18	she	761
19	had	712
20	by	710
21	you	707
22	with	693
23	or	677
24	The	614
25	have	581
26	an	568
27	his	550
28	are	514
29	this	496

The user will also receive these pop-up plots, which will show the most frequent words and the least frequent words. The code for these bar graphs can easily be adjusted on lines 105 and 106 to show more words, but that would clutter the graph severely at some point (since there are so many distinct words for the graph to display).

Top 25 Word Count by Rank (Semi-Log Scale)



Bottom 25 Word Count by Rank (Semi-Log Scale)



4. **Discussion.** Your discussion should contain two parts. First, include 1-2 paragraphs to address the questions from §2.4. Next, write 1-2 paragraphs about what you learned during the completion of the assignment. You might write about selecting your data, unexpected issues that you ran into and how you resolved them, any new skills or software libraries you learned in order to complete the assignment, or anything else that you learned. Please be specific (i.e., do not write vague statements “The assignment was great, I learned so much about NLP”).

2.4 Findings

Check your results. What do you notice about the words at the top of your list? Other than being “common words”, do you notice any other kinds of properties that they share? Now check the other end of your list. What do you notice about these words? After creating your figure(s), check out this [Wikipedia page](#), especially the introduction and subsection titled “Word frequencies in natural languages”. How do your results compare to the ones you see there? Why do you think that they are similar or different? What does this imply about the impact of removing stopwords like “the” on the total number of tokens in a corpus, compared to removing regular content words (e.g., nouns and verbs)? Answer these questions in your report.

The words at the top of my list are all stopwords, which we know are meaningless and carry little information because they are very commonly used. They share very common properties. These words are very short, all top 25 words are 4 or fewer characters long. Most of them are also all words that do not carry actual meaning, other than connecting sentences structurally. For example, the word “the” is something you put before a noun. The word “and” is just to connect two separate nouns or ideas. They do not define anything concrete or have meaning, like the word “cake” which is a baked dessert we eat. To conclude, because of these properties, all of these words are used regardless of context, because they do not carry context. The words at the end of the list are the opposite, they have meanings, and some are more complex than others. Some of them are spelled a bit longer, and some vocabularies are rarely used. For example, Gutenberg is the name of a website, which carries free eBooks. Those words represent very specific things, locations, and ideas. If the input data isn’t cleaned well, there can be symbols that make words extremely unique, such as the GutenBerg™ symbol.

My graph shows a sharp decline between the most frequent word and the second most frequent word, but only gradually decreases after that, which does not align with Zipf’s Law. It makes sense to me that my graph should follow Zipf’s Law in theory, as “the” should be twice as common as “of” since “the” usually appears before most nouns. The sharp decline makes it very clear that it is following Zipf’s Law, as well as the fact that the top 3 most common words are “the”, “of”, and “and”, which are mentioned in the introduction of the Wikipedia page. However, my curve does not perfectly mimic the prediction, as factors play a role. My text file could be too small, and a larger text file may bring my curve closer to Zipf’s Law. The corpus might also be biased, such as a story about a young girl might use the words “she” and “her” more often than other stories. Removing all the stopwords might alter the curve as well since we know that they are heavily common and the Wikipedia introduction paragraph only mentions the stopwords.

This assignment helped me gain a better understanding of NLP concepts. I learned about stemming and lemmatization, which I had never encountered or heard of before. Usually, I've only ever filtered out punctuation, digits, or letters, but I've never had to follow a more complex rule set, such as removing suffixes from words or checking for duplicate characters before suffixes, such as "running" to "run" instead of "runn". I took more time to code lemmatization since I was having trouble differentiating it from the concept of stemming. I had to research online to understand how the rules work, such as having to hard-code irregular words like "children" to "child".

I was more critical when selecting the data, trying to look for a corpus that would give my program a bigger challenge, such as books on English literature. I quickly realized that this assignment is more difficult than it seems, as my program could struggle with more challenging pieces of text out there. I enjoy coding in Python, and this assignment has helped me freshen up my coding ability since it covered a wide range of aspects of a complete project. For example, taking in user arguments is not something I do very often, so I had to look into using Python's sys module. I also do not use regex often, so it was fun to play around with its syntax and admire its flexibility. I thoroughly enjoyed this assignment and look forward to learning more about the next ones.

I used GitHub Copilot to help me generate code snippets, such as the template to read and write files, as well as using arguments and plots. I used ChatGPT to generate a hard-coded list of stopwords and a lemmatization dictionary for irregular words. I also used ChatGPT throughout the assignment to explain foreign concepts to me, such as lemmatization and stemming. I also needed some help to understand regex expressions.

- (1) ChatGPT (GPT 3.5), GitHub Copilot (GPT 3 variant)
- (2) NVIDIA GPUs
- (3) 7 Hours
- (4) OpenAI, GitHub (Microsoft)
- (5) Canada Central (Toronto)
- (6) I did some research online, such as looking at Wikipedia articles and asking ChatGPT to affirm what I've found. I know that OpenAI is heavily supported by Nvidia because stocks.

CO2 Emission Related to Experiments

Experiments were conducted using a private infrastructure, which has a carbon efficiency of 0.432 kgCO₂eq/kWh. A cumulative of 7 hours of computation was performed on hardware of type GTX 1080 (TDP of 180W).

Total emissions are estimated to be 0.54 kgCO₂eq of which 0 percents were directly offset.

Estimations were conducted using the [MachineLearning Impact calculator](#) presented in [?].

@article{lacoste2019quantifying, title=Quantifying the Carbon Emissions of Machine Learning, author=Lacoste, Alexandre and Luccioni, Alexandra and Schmidt, Victor and Dandres, Thomas, journal=arXiv preprint arXiv:1910.09700, year=2019