# Introduction

This is a study on contemporary literature made available in the ConLit database. The study aims to compare three different biographies written about Steve Jobs published in the previous decade to compare what differences in storytelling can be found based on most frequently used words and Supersense annotations. Supersense is a NLP task that consists of 20 specified categories of word classes (Dei Rossi, 2013). Despite managing to carry out most of the data pre-processing, the study failed in the end for problems with downloading the NLTK stopwords list on Jupyter notebooks. Despite this failure, I managed to test visualisations functions with the partially pre-processed data.

# Dataset

ConLit is a database with sampled text data and bookNLP annotated data derived from c. 2700 English language books published over a 20 year period. The database consists of a set of files and a folder database for the different features that have been manually or computationally analysed on each book. In the vast selection of books sampled, I was interested in the three different biographies of Steve Jobs, as they were written by different authors relatively few years apart in the 2010s. On these select books, I was interested in the samples “1,000 words chunks”, unigrams created from the samples of text, and SuperSenses analysed from the text.

# Hypothesis

My hypothesis was that comparing the autobiographies would reveal differences in storytelling choices made by differences in most frequent words and Supersenses.

# Issues with data

As Piper (2022) points out, there are expected to be some errors in the data, as a lot of the preliminary work was done manually and material was abundant. I came across two issues with the data itself. Firstly, not all books mentioned in the metadata list have a “1000 word chunks” folder. The number of those folders is roughly half of the total c. 2700 books estimated by Piper to be in the collection. For this reason, using the metadata file of the collection as a filtering tool for selecting titles of interest – as it lists all categories, authors, genres, and publishing years – may proof frustrating as after selecting suitable titles, not all titles will be available. For instance, only one of the three Jobs biographies had all the samples available. This may not be a real issue however, as a unigram CSV file is available for all of them, which appears to be a processed version of the 1000 word chunks. In fact, using this file removes the step of having to combine the “chunks” files in OpenRefine.

The second, and potentially significant problem is that some documents seem to have more words than others, according to the Supersenses file. From the three Jobs biographies, the Blumenthal book had a significantly lower value in all categories. From the description of the database, I cannot tell why there is so much less data on that one book, as I assumed the same amount of text would have been annotated on each of the books sampled for the database. I suspect either the entire books have been annotated, but just the first thousand words have been made available on select books. Or, some of the books have more repetition of words, leading to more iterations of each of the thousand words. Before making larger scales studies on the data, it should be clarified what is causing the disparity.

Method  
I first pre-processed the data using OpenRefine. In order to parse the .csv files, the option

|  |  |
| --- | --- |
| Parse next | 1 line(s) as column headers |
|  |  |

may need to be selected in order to avoid an error message. To cluster the data, I chose to convert all words to lowercase, at the cost of losing distinction between common and proper nouns.

I used OpenRefine to set all words to lowercase by selecting lemma > edit cell > common transform > set to lowercase. Then I used the following transforms:

to remove the plural s - value.replace(/s$/,"")

to remove number values - replace(value, /\d/, '')

to remove punctuation - value.replace(/\p{Punct}$/, '')

As an afterthought, as removing the suffix -s is different to clearing an entire cell, I should have looked for a command to remove rather than replace the respective regexes.

I then used the *cluster and edit* function in the *key collision* method and keying functions *fingerprint* and *Ngram fingerprint* to merge duplicates and the single letter entries. Then I printed the data as a .csv file, and to remove the fields left empty, I just removed the first 300 or so empty lines. It seems there may have been odd empty lines mixed throughout the document, so this was not the best method. Note: Although prompting the program to print a comma separated file, the files are created as .txt files. I simply renamed them as .csv and the format changed to a table from a literal comma separated .txt file. Perhaps I could have transformed the files to .numbers files first, and removed the empty cells there before printing the files as .csv.

I wanted to use a stopword list, and based on google searches, the best way to achieve this seemed to be by downloading the NLTK stopword list in English on Jupyter Notebook. However, I had multiple issues which I tried to solve with the help of ChatGPT and online articles, and only managed halfway. At first, the **pip install** didn’t work, and after successfully using the magic **%pip**, I still was not able to download the files. I then downloaded the English stopword list manually, uploaded it on the Notebook and set the path, but still was not able to access the wordlist. I finally attempted to download a Scikit stopword list, and it seemed to download but didn’t work, at which point I gave up.

In the end, I managed to create several visualisations of the .csv files of the SuperSenses from each Jobs biography, and a bar chart comparison of the top 10 words from each of the pre-processed unigram files by using pandas, matplotlib and seaborn. However, without removing the stopwords, the visualisations are pointless, as they contain mainly what would be stopwords.

# Conclusion

With access to the NLTK stopword list, it would be possible to finish refining the data. Some of the steps I completed with OpenRefine, e.g. removing punctuation and setting to lowercase could also be done with NLTK. With the properly refined data it would be possible to compare actual top words in the three different biographies, say top 50 or more, and see how much they diverge. My hypothesis was that this could show how differently one can angle a book on the same subject by using different emphases. I relied heavily on Google and ChatGPT to complete this task.

Sources:  
<https://figshare.com/articles/dataset/CONLIT/21166171/1>

Dei Rossi, S., Di Pietro, G., Simi, M. (2013). Description and Results of the SuperSense Tagging Task. In: Magnini, B., Cutugno, F., Falcone, M., Pianta, E. (eds) Evaluation of Natural Language and Speech Tools for Italian. EVALITA 2012. Lecture Notes in Computer Science(), vol 7689. Springer, Berlin, Heidelberg. <https://doi.org/10.1007/978-3-642-35828-9_18>

Piper, A. (2022). The CONLIT Dataset of Contemporary Literature. Journal of Open Humanities Data, 8(0), 24.DOI: <https://doi.org/10.5334/johd.88>