Sprachverarbeitung: Übung

SoSe 24

Janis Pagel
Department for Digital Humanities, University of Cologne

2024-04-23

Please submit your solution via Ilias, either as a Jupyter Notebook (`.ipynb`, you can export your Notebook in Jupyter by going to File > Download) or as a Python script (`.py`) if you are not working in Jupyter.

You can write the written answers to the exercises into the source code as a comment or into a separate document (PDF). In the second case, please submit both your PDF file and your Python source code.

Exercise 6 is a bonus exercise that you can solve if you want to, but don't have to.

Exercise 1.

Download the following corpus files from the *Deutsches Text Archiv* (DTA) into a directory called `dta` and extract them (if you are working on `compute.spinfo.uni-koeln.de`) in Jupyter, it is probably easier to extract the files on your local machine and then upload them to Jupyter via the upload button).

- `https://media.dwds.de/dta/download/dtae/2020-10-23/original/1400-1499.zip`
- `https://media.dwds.de/dta/download/dtae/2020-10-23/original/1500-1599.zip`

This should give you the sub-directories `1400-1499` and `1500-1599`. Write a Python script that reads in all `.txt` files from each directory and stores the content into a dictionary that contains the century ("1400" and "1500") as a key and all tokens from the text files for this century as the value in a list. Hence, your dictionary should look something like the following:

```
{1400: ["token1", "token2", "token3", ...],
 1500: ["token1", "token2", "token3", ...]}
```

You can use "naive" token splitting, i.e. split tokens on whitespaces only, without separating punctuation.

Create a pandas dataframe out of this dictionary. To achieve this, first convert your dictionary into a pandas `Series` object and use the `explode()` and `reset_index()` functions. Your dataframe should contain two columns, one with the century and one with the respective tokens:

century	token
1400	token1
1400	token2
1400	token3
...	
1500	token1
1500	token2
1500	token3

Hint: In order to get a list of all the files in the directories, you can import the `glob` Library and use the following code: `list(glob.iglob("dta/**/*.txt", recursive=True))`. This will give you a list of the file names and paths for each txt-File. You can then iterate over this list to read in all the files' content.

Exercise 2.

On your dataframe from the previous exercise, calculate Type-Token Ratio (TTR) for the 1400 and the 1500 corpus, respectively. To do this, look at the pandas methods `drop_duplicates()` and `size()`. What do you observe? What do the numbers tell you about the vocabulary distribution in the two corpora?

Exercise 3.

You have the hypothesis that the 1500s use much more words related to the topic of "family" than compared to the 1400s. Using the dataframe from Exercise 1, retrieve the absolute counts for German family-related tokens (try at least "kind", "mutter" and "vater") and compare them between the 1400 and 1500 corpora. Do the numbers corroborate or disprove your hypothesis?

Now calulate the relative counts for each century for the same tokens and compare. How have the numbers changed? Do they still corroborate/disprove your hypothesis? Do you think it is generally valid to investigate change in vocabulary usage this way?

Exercise 4.

Retrieve the absolute and relative counts for the token "gott" in both centuries. What does it tell you about the usage of this token across the centuries? Now retrieve absolute and relative counts for the token "Gott" in both centuries. What do these counts tell you, also in regards to the results for the token "gott"?

Exercise 5.

Using the dataframe from Exercise 1, calculate the absolute counts for each token per century and store in a new column. Based on these counts, retrieve the rank of each token per century, such that the token with the highest absolute count has the highest rank (rank 1), decreasing with frequency. To achieve this, look into the usage of the pandas method called `cumcount()`. Use seaborn's `lineplot` to plot the absolute count on the y axis and the rank on the x axis and the century as a group to create two lines, one for 1400 and one for 1500. What do you observe? Can you see a Zipf curve?

Now, only use the 50 highest ranked tokens from each century and plot again. What is the difference? Can you observe a Zipf curve? If not, what is the problem? How could you solve this problem?

Lastly, create a mini corpus with only two texts, `dta/1400-1499/nn_almanach05_1473.txt` and `dta/1500-1599/rechnungsbuch01_1500.txt`. Create the same plots with only these two texts. What do you observe? How do you explain?

Bonus Exercise 6.

Using the dataframe from Exercise 1, calculate Standardized TTR (STTR) for a window of 1000 words per century. Look into the usage of the pandas method "cut" to create the windows. Compare your result to the result of Exercise 2. Are there differences? What do the numbers tell you about the distribution of the vocabulary in the texts of the two centuries?