Introduction to Python : Mini project

Descriptive statistics of the corpus

bingzhi2013@gmail.com

October 22, 2021

This project is to be done in **pairs**. The submission is due **23 : 59 Friday, December 12, 2021** and counts for **80%** of the final grade. No late submission will be accepted after the deadline. You need to send me the code and data in a compressed **Name1_Name2.zip** file. The code can be a *.py file or a notebook file *.ipynb (If you use Colab, you can share your code with me by adding bingzhi2013@gmail as a collaborator.)

1 Objective of the project

The objective of this project is to apply all the notions seen in class. You will have to manipulate a **Conll** format corpus to obtain some descriptive statistics. You need to save the various data obtained in **.txt** or **.csv** files. The clarity and readability of your code as well as good programming practices such as comments will be particularly appreciated.

2 Data

2.1 Obtaining Data

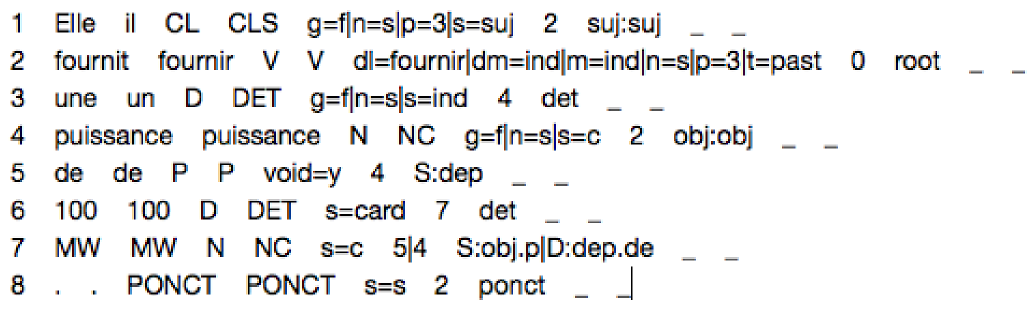
You will find in the folder `mini-project` an archive named `wikip-small.conll.tar.gz`. To extract the corpus(346 MB) from this archive, you can either use a decompression software such as `winrar` or the terminal command `tar -xzvf wikip-small.conll.tar.gz`. Once the archive decompressed, you will get a file named `wikip-small.conll`. This file contains the corpus on which you will work.

Tip : You'll also find a tiny conll file `test_wiki_3_sentences.conll`(4KB) with only 3 sentences. First, use the toy corpus to test and debug your program, if everything goes well, then process the big corpus to retrieve the descriptive statistics.

2.2 Conll Format

The corpus contains a sample of sentences extracted from the **Wikipedia** in `Conll` format. The 'CONLL' file type represents a corpus with one word per line, each line containing the annotation (form, lemma, POS, syntax...) of a word/token in 10 fields, separated by single tab characters. Each word is numbered (ID), and a new sentence starts whenever this word ID is 1 (i.e. Blank lines marking sentence boundaries). An absent entry is represented by the '_' character. For more information about this format, you can go to the official page :

https://universaldependencies.org/format.html.



```
1  Elle  il  CL  CLS  g=f|n=s|p=3|s=suj  2  suj:suj  _  _
2  fournit  fournir  V  V  dl=fournir|dm=ind|m=ind|n=s|p=3|t=past  0  root  _  _
3  une  un  D  DET  g=f|n=s|s=ind  4  det  _  _
4  puissance  puissance  N  NC  g=f|n=s|s=c  2  obj:obj  _  _
5  de  de  P  P  void=y  4  S:dep  _  _
6  100  100  D  DET  s=card  7  det  _  _
7  MW  MW  N  NC  s=c  5|4  S:obj.p|D:dep.de  _  _
8  .  .  PONCT  PONCT  s=s  2  ponct  _  _
```

FIGURE 1 – Example of a sentence in Conll format. Each line corresponds to one word, each column/field, separated by single tabs, contains the morpho-syntactic information.

Column	Information
1	ID: word index
2	Word form
3	Lemma
4	Universal PoS tag
5	Language specific PoS tag
6	Morphological features
7	Head
8	Dependency relation
9	other
10	other

TABLE 1 – Morpho-syntactic information per column

In this project, we only need the information of column 2 (`word form`) and 4 (`UPoS`). Please note that in Python, the index starts from 0, thus in your code, you will need to take the column 1 and 4.

3 Part 1

The first part of the project is to get the basic statistics of the corpus and save them in a `.txt` format file named `corpus_count.txt`. You need to get the following information.

1. The number of the sentences in corpus
2. The number of words in corpus
3. The vocabulary size
4. The number of different PoS (Part-of-Speech) tags
5. The average sentence length

Your file should be in the form :

```
number_of_sentences = XXX
vocabulary_size = XXX
...
```

4 Part 2

In this second part, we will use the `pandas` module to store data in `.csv` format files.

4.1 Frequency of vocabulary

1. Write a function `word_freq()` that takes one argument : the path to a `Conll` file, and returns a dictionary containing the frequency of words in the corpus.
2. Store the frequency of words obtained in a `DataFrame` (see library `pandas`).
3. Store the resulting `DataFrame` in a `.csv` file named `word_freq.csv`.
4. Sort the word frequencies in descending order and save the result in a text file named `sorted_word_freq.txt`.
5. What are the 5 most frequent words in this corpus ?

4.2 Frequency of PoS

1. Write a function `pos_freq()` that takes one argument : the path to a `Conll` file, and returns a dictionary containing the frequency of PoS in the corpus.
2. Store the frequency of PoS obtained in a `DataFrame`.
3. Store the resulting `DataFrame` in a `.csv` file named `pos_freq.csv`.
4. Sort the PoS frequencies in descending order and save the result in a text file named `sorted_pos_freq.txt`.
5. Which PoS tag is the most frequent one in this corpus ?

4.3 Sentences lengths

1. Write a function `length_freq()` that takes one argument : the path to a `Conll` file, and returns a dictionary containing the frequency of sentence lengths in the corpus.
2. Store the frequency of sentence lengths obtained in a `DataFrame`.
3. Store the resulting `DataFrame` in a `.csv` file named `length_freq.csv`.
4. Calculate the mean, the median and standard deviation of the lengths of the sentences and save these results in a text file named `length_info.txt`.

5 Tips

In this mini-project, you are expected to be able to :

— use loops and conditional structures

— create your own functions and re-use them

— manipulate basic data structures such as `list` and `dictionary`

— read and write files

— use external modules, in our case `pandas`

Your code should be as clear as possible and well commented. You should pay attention to the choice of variable and function names. And finally, DO NOT submit code that does not work.