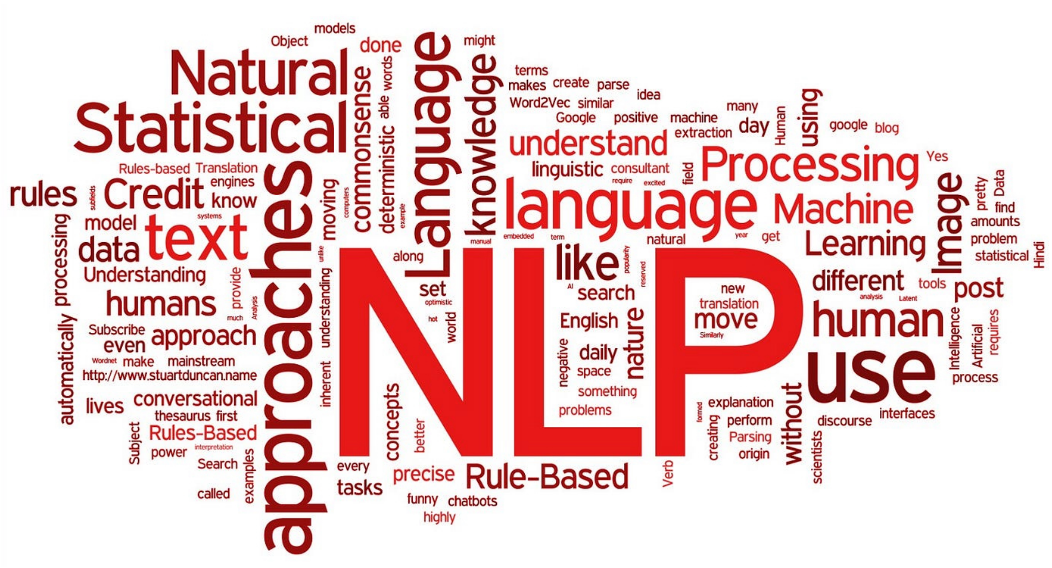
PROJECT ASSIGNMENT 2

Issue Date : 08.11.2024 - Friday
Recitation Date : 08.11.2024 - Friday (14:00) (held on Zoom)
Due Date : 24.11.2024 - Sunday (23:00)
Advisor : R.A. Görkem AKYILDIZ
Programming Language : Python 3.9.18



1 Introduction

Natural language processing (NLP) is a subfield of computer science and especially artificial intelligence. It is primarily concerned with providing computers with the ability to process data encoded in natural language and is thus closely related to information retrieval, knowledge representation and computational linguistics, a subfield of linguistics. Typically data is collected in text corpora, using either rule-based, statistical or neural-based approaches in machine learning and deep learning. Major tasks in natural language processing are speech recognition, text classification, natural-language understanding, and natural-language generation.[1]

Aim of this project is creating a very basic NLP model that makes some basic textual analysis on the given textual data.

2 Requested Statistics

In this project, the aim is generating desired statistics about the given input file (directory of the input file will be given as the first command line argument of the program), statistics to be calculated are as follows:

2.1 Number of Words

Words must be tokenized, tokenizing means parsing the whole input into words part by part. For this purpose, whole text must be purified from the punctuations (except the ones that are in the word itself such as "well-known" "they're" etc.) and then white-spaces can be thought as splitters between the words, then words can be determined by selecting them according to the placement of the white-spaces.

2.2 Number of Sentences

End of the sentence characters can be assumed as ".", "!", "?", or "..."; moreover these characters will not be used unless it is end of the sentence for the sake of easiness, which means there will be no usages such as "Mr.", "Ms.", "1998(?)" etc.

2.3 Average Number of Words per Sentence

The information obtained at previous two parts can be used to obtain this information, the result must be represented with two decimals, it would be beneficial to use `{:.2f}` format for writing the result to the file.

2.4 Number of Characters

All of the characters (including the punctuations and the white-space characters such as newline character) must be counted for this part.

2.5 Number of Characters (Excluding the Punctuations and White-Spaces)

As opposed to previous part, white-spaces and punctuations (except the ones that are in the word itself such as "well-known" "they're" etc.) must be ignored at this counting, it can be imagined as total number of characters at the all occurrences of the words that was tokenized at Part 2.1.

2.6 The Shortest and Longest Word(s)

Words must be selected according to their length and written to the file with their frequencies, if there is more than one word that is the longest or shortest, they must be sorted according to their frequencies in decreasing order (if there is more than one word that have the same frequency, increasing alphabetical order must be used as the tie-breaker).

2.7 Frequencies of All of the Words

Occurrences of all of the words must be calculated, then they must be divided to the total number of words to get frequency. Later on they must be sorted according to their frequencies in descending order (if there is more than one word that have the same frequency, increasing alphabetical order must be used as the tie-breaker). All of the frequencies must be represented with four decimals, it would be beneficial to use `{:.4f}` at the output step

3 Definition of Output

Output file must follow the exactly same format as the given samples, not obeying the given format may result with getting a grade which is as low as zero, so, please be careful about the formatting. Moreover, path of the output file will be given as second command line argument of the program.

Output file contains the following (All of the following must be separated with a new line character from each other):

- "Statistics about <FILE_NAME>:" where <FILE_NAME> stands for the given input file path as first argument. File name must be formatted according to {:7} format, which puts 7 - (length of the word) times space(s) after the written content for the sake of the alignment. Note that it does not put any space characters if the given string already has at least seven characters.
- "#Words: <NUMBER_OF_WORDS>" where <NUMBER_OF_WORDS> stands for the statistics that are obtained at Part 2.1.
- "#Sentences: <NUMBER_OF_SENTENCES>" where <NUMBER_OF_SENTENCES> stands for the statistics that are obtained at Part 2.2.
- "#Words/#Sentences: <WORDS_PER_SENTENCE>" where <WORDS_PER_SENTENCE> stands for the statistics that are obtained at Part 2.3.
- "#Characters: <NUMBER_OF_CHARACTERS>" where <NUMBER_OF_CHARACTERS> stands for the statistics that are obtained at Part 2.4.
- "#Characters (Just Words): <NUMBER_OF_CHARACTERS_IN_WORDS>" where <NUMBER_OF_CHARACTERS_IN_WORDS> stands for the statistics that are obtained at Part 2.5.
- For this section of output, there are two options, there can either be one shortest word or more than one. If there is just one shortest word, "The Shortest Word: <SHORTEST_WORD> (<FREQ_OF_SHORTEST>)" structure must be used where <SHORTEST_WORD> stands for the shortest word and <FREQ_OF_SHORTEST> stands for the frequency of it, these statistics must be obtained according to Part 2.6. If there is more than one shortest word, then they must be written as in the multiple line format; the first line must be "The Shortest Words:" and the rest must be in "<SHORTEST_WORD> (<FREQ_OF_SHORTEST>)" format, where <SHORTEST_WORD> stands for current shortest word and <FREQ_OF_SHORTEST> stands for frequency of it. The words must be formatted according to {:24} format. The same technique applied for the shortest word(s) must be applied for the longest word(s) too.
- At this section of the output, multiple line format must be followed. The first line must be "Words and Frequencies:" and the rest must be follow the "<WORD> : <FREQ_OF_WORD>" format where <WORD> stands for current word and <FREQ_OF_WORD> stands for frequency of it. {:24} format must be obeyed while printing the words at this part too.

Please be careful about the alignments of the ":" characters (and any other characters), as it is said above, formatting must be obeyed character by character, any mismatch may result with getting a grade which may be as low as zero.

Note that all of the words must be handled as lower-case versions of them while calculating statistics about the frequencies as "Görkem" and "GÖRKEM" are the same but with the different cases. Moreover, the code must `import locale` and the first statement of the code after imports must be `locale.setlocale(locale.LC_ALL, "en_US")`. Otherwise there may be some issues at lowercasing and formatting (such as usage of "," instead of "." or lower-casing "I" as "ı" at Turkish systems) which may cause point deduction.

4 Restrictions

- Your code must be able to execute on our department's developer server (dev.cs.hacettepe.edu.tr).
- You must obey given submit hierarchy and get score (1 point) from the submit system.
- **You must benefit from loops and functions.**
- Your code must be clean, do not forget that main function is just a driver function that means it is just for making your code fragments run, not for using them as a main container, create functions in necessary situations but use them as required.
- You must use comments for this project and you must give brief information about the challenging parts of your code. Do not over comment as it is against clean code approach. Design your comments so that they make your code fully understandable and not excessive for others. You can check guides of Python namely PEP-8 and PEP-257 for further information.
- You can benefit from Internet sources for inspiration but do not use any code that does not belong to you.
- You can discuss high-level (design) problems with your friends but do not share any code or implementation with anybody.
- Do not miss the submission deadline.
- Source code readability is a great of importance. Thus, write READABLE SOURCE CODE, comments, and clear MAIN function. This expectation will be graded as “clean code”.
- Use UNDERSTANDABLE names to your variables, classes, and functions regardless of the length. The names of functions, attributes and classes should obey Python naming convention. This expectation will be graded as “coding standards”.
- You can ask your questions through course's Piazza group, and you are supposed to be aware of everything discussed in the Piazza group. General discussion of the problem is allowed, but **DO NOT SHARE** answers, algorithms, source codes and reports.
- All assignments must be original, individual work. Duplicate or very similar assignments are both going to be considered as cheating.
- Submit system for this homework will be opened a few days before deadline, so please be patient.

5 Execution and Test

Your code must be executed under **Python 3.9.18** at **dev.cs.hacettepe.edu.tr**. If your code does not run at department's developer server during the testing stage, then you will be graded as 0 for the code part even if it works on your own machine.

Sample run command is as follows:

- python3 text_analyzer.py input.txt output.txt

6 Grading

Task	Point
Calculating Number of Words (Part 2.1)	10
Calculating Number of Sentences (Part 2.2)	10
Calculating Average Number of Words per Sentence (Part 2.3)	5
Calculating Number of Characters (Part 2.4)	5
Calculating Number of Characters (Excluding the Puncts and W-Ss) (Part 2.5)	10
Finding The Shortest and Longest Word(s) (Part 2.6)	15
Calculating Frequencies of All of the Words (Part 2.7)	25
Clean Code & Comment	20*
Total	100

*** The score of the clean code & comment part will be multiplied by your overall score (excluding clean code & comment part) and divided by the maximum score that can be taken from these parts. Say that you got 60 from all parts excluding clean code & comment part and 10 from clean code & comment part, your score for clean code & comment part is going to be 10*(60/80) which is 7.5 and your overall score will be 60+7.5=67.5.**

Note that you must score one at the submit system, otherwise 20% of your grade will be deducted, moreover, usage of global variables are forbidden and you must implement a main function as advised otherwise 20% of your grade will be deducted! There may also be other point deductions if you do not obey the given rules, such as if you do not use functions and/or loops as necessary.

Also note that you must give the desired outputs to get full credit from the parts, otherwise you may get a grade which is as low as zero for the parts that is not giving the desired output even if your implementation is correct!

7 Submit Format

File hierarchy must be zipped before submitted (Not .rar, only not compressed as .zip files because the system just supports .zip files).

```
- b<StudentID>.zip
    - text_analyzer.py
```

8 Late Policy

You have two days for late submission. You will lose 10 points from maximum evaluation score for each day (your submitted study will be evaluated over 90 and 80 for each late submission day). You must submit your solution in at the most two days later than submission date, otherwise it will not be evaluated. Please do not e-mail to me even if you miss the deadline for a few seconds due to your own fault as it would be unfair for your friends, e-mail submissions will not be considered if you do not have a valid issue.

References

[1] Natural language processing - wikipedia. https://en.wikipedia.org/wiki/Natural_language_processing˙(Last access: 08.11.2024).