Contents

1. Introduction

This program is used for investigation, basic data extraction and statistical analysis of the natural-language posts from Q&A (Question and Answering) site. The input is given in the form of a file (here data.xml file). A set of pre-processing tasks are done on the raw data to obtain clean dataset and segregate the posts into two separate files, based on whether a post is a question or answer. Each post or individual lines in the input file has various components like row Id, postTypeId, creation date and body. By passing each input line through a Parser we can extract all these components. For statistical analysis of the whole input file, two graphs are generated, one graph showing the number of posts spread across different vocabulary sizes and another graph showing the number of post(different type: Question/Answer) spread across quarters of subsequent years.

NOTE: While running the three tasks keep data.xml file and the programs of the three tasks in the same folder .

2. Task 1: Handling with File Contents and Preprocessing

Task 1 deals with pre-processing of the file contents to convert it into a readable format. The input file data.xml in raw format consists of special sequences of characters known as character reference. Below figure 2.1 shows the list of character references that has been converted into its original form in task 1. Also special characters like “
”, “
” has been replaced by a single empty space in this task.

Character reference	Original form
&	&
"	"
'	'
>	>
<	<

Figure 2.1 : Character reference transformation

Originally, the individual lines in the input file is of the form as shown in figure 2.2 which after character reference transformation look like the form shown in figure 2.3. After character reference transformation, the data file consists of HTML tags which are of the form **“<*>”** as observed in figure 2.3.

```
<row Id="10695" PostTypeId="2" CreationDate="2019-02-28T14:16:03.543" Body="&lt;p&gt;The &lt;strong&gt;Detroit DIY Electronics Wifi-53
<row Id="10696" PostTypeId="1" CreationDate="2019-03-02T04:17:52.833" Body="&lt;p&gt;I'm looking for the cheapest way to recognise text
<row Id="10697" PostTypeId="1" CreationDate="2019-03-02T08:07:24.133" Body="&lt;p&gt;We have Verizon service with a FiOS Quantum Gatewa
<row Id="10698" PostTypeId="2" CreationDate="2019-03-02T08:13:03.600" Body="&lt;p&gt;A &lt;a href=&quot;https://www.bing.com/search?q=p
<row Id="10700" PostTypeId="1" CreationDate="2019-03-02T18:37:20.453" Body="&lt;p&gt;I am concerned that lg gram is not a matte screen
```

Figure 2.2 : Data.xml file

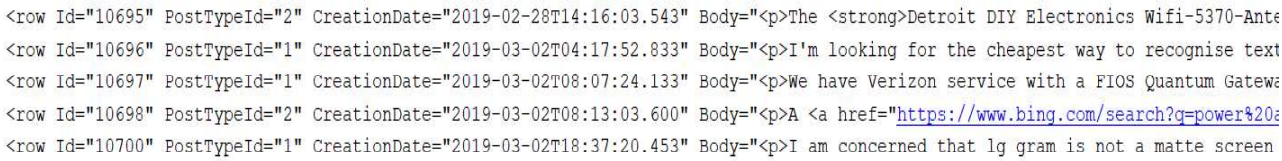

Figure 2.3 : After character reference transformation

The first two lines and the last line are also ignored as a part of data pre-processing. After removal of HTML tags, each line of the input file is checked for its PostTypeId and if the PostTypeId is 1 the cleaned body is saved into a text file Question.txt or if the PostTypeId is 2 then the cleaned body is saved into a text file Answer.txt as shown in figure 2.4 and 2.5. Question.txt and Answer.txt are the two output files obtained after running the program of task 1.

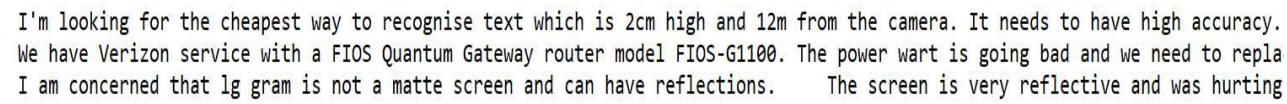

Figure 2.4 : Question.txt

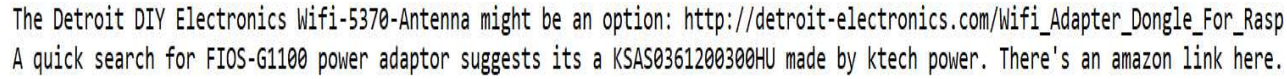

Figure 2.5 : Answer.txt

3. Task 2: Building a Class for Data Analysis

In task 2 we are collecting and returning the required data for further analysis in task 3. A class called Parser is built which extracts and encapsulates the components of an input string (individual post) for example its ID, post type, creation date, cleaned body and vocabulary size.

Cleaned body is obtained by using preprocessLine function from task 1 which removes all the character references and HTML tags to return the body of a post in a readable format.

Vocabulary size is obtained by taking count of unique words in the cleaned body for a particular post.

The __str__ () method is called when print() function is invoked and the output can be observed as shown in figure 3.1, 3.2 and 3.3 for Question, Answer and Others respectively. Figure 3.3 shows a case where Vocabulary size is zero.

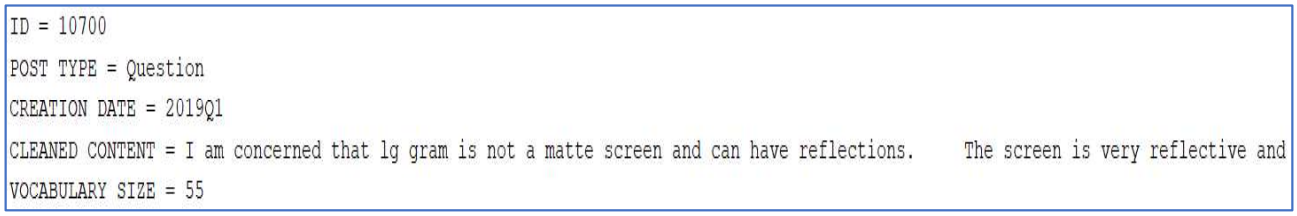

Figure 3.1 : Parser class output for row Id 10700

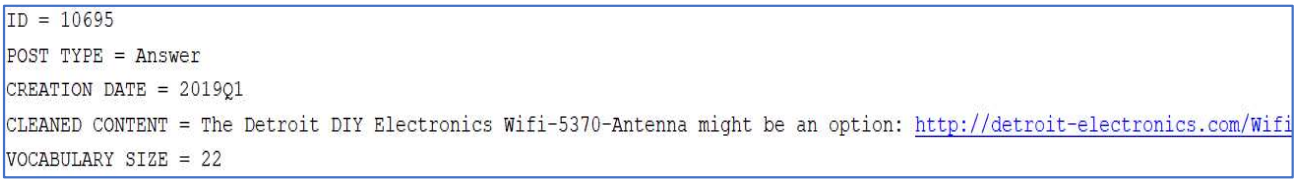

Figure 3.2 : Parser class output for row Id 10695

Figure 3.3 : Parser class output for row Id 373

4. Task 3: Analyzing the File for Data Visualization

In task 3 we are visualizing the input data for further statistical analysis. This is done by plotting two graphs one bar graph showing the number of posts spread across different vocabulary sizes calculated using task 2 and another line graph showing the number of posts (different type: Question/Answer) spread across quarters of subsequent years. The ranges in figure 4.1 are all left inclusive and the range for “others” is greater than equal to 100.
The input file to generate the two plots is data.xml and the output files are the plots as .png files (wordNumberDistribution.png and postNumberTrend.png) as shown in figure 4.1 and 4.2.

For completion of task 3 we are required to import python modules numpy, pandas and pyplot function from matplotlib module.

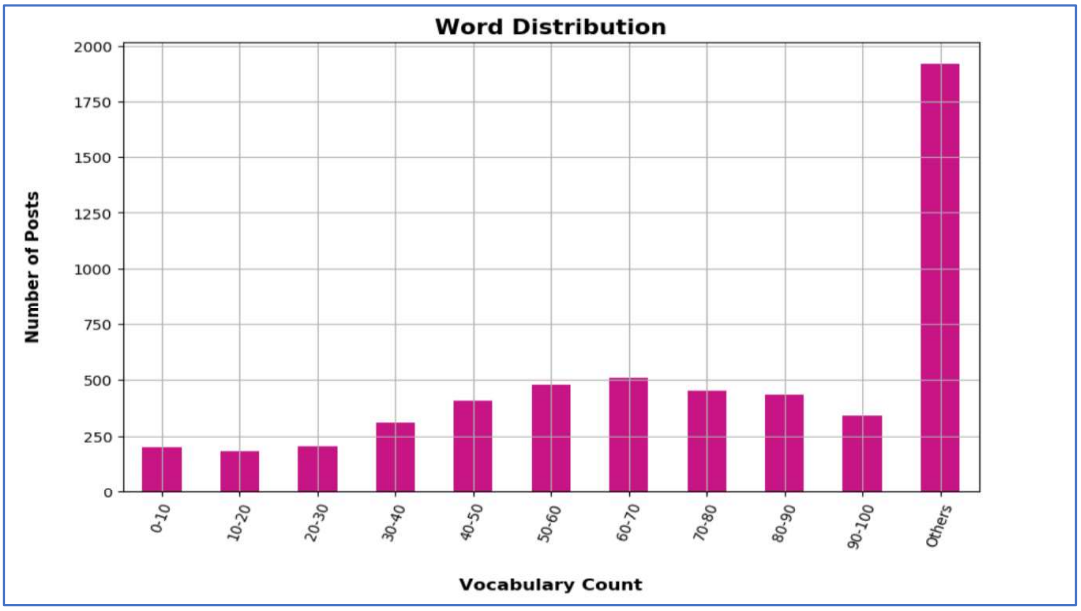

Figure 4.1 : wordNumberDistribution.png

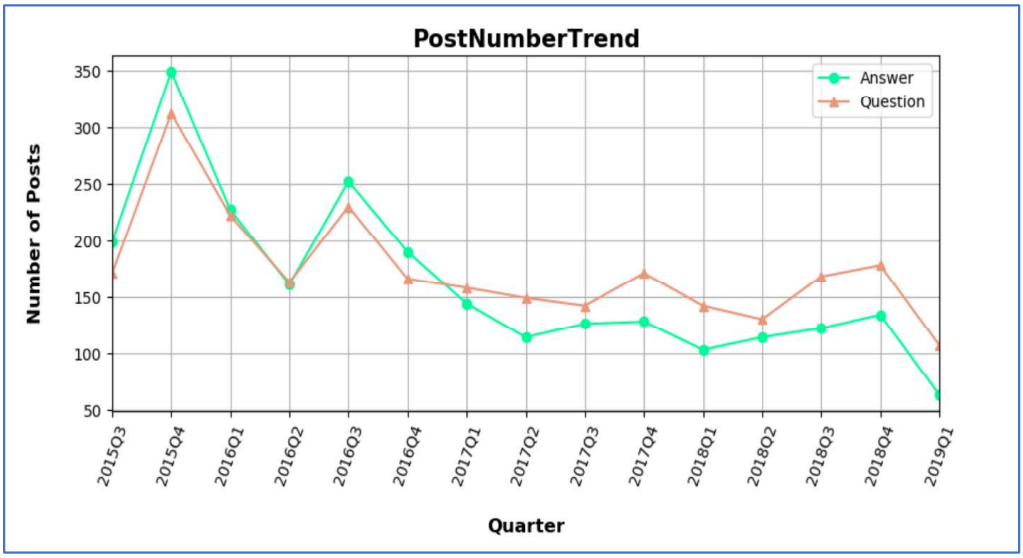

Figure 4.2 : postNumberTrend.png

5. References

- For Regular Expressions module:

  https://docs.python.org/3/library/re.html

  https://www.w3schools.com/python/python_regex.asp

- For matplotlib module:

  https://matplotlib.org/3.1.0/tutorials/introductory/pyplot.html

  https://matplotlib.org/api/pyplot_api.html