FIT9133 Programming Foundations in Python

Assignment 2 Documentation

- **Subhasish Sarkar (29819253)**

PART 1: Data Cleaning

This program aims to implement a basic language analyser to investigate the linguistic characteristics of children with some form of language disorders. The analyser is able to perform basic statistical analysis on a number of linguistic features and also to present the analysis results using some form of visualisation.

Each of the narrative transcripts is a record of the story-telling task performed by a child for the two groups (SLI and TD), under the supervision of an examiner (investigator). The stories are elicited by presenting pictures with a number of different animal characters to the children participating in the study.

```
@UTF8
@PID:    11312/c-00036109-1
@Begin
@Languages:      eng
@Participants:   CHI Target_Child, EXA Investigator
@ID:     eng|ENNI|CHI|4;11.16|male|SLI||Target_Child|||
@ID:     eng|ENNI|EXA|||||Investigator|||
@Comment:        Birth of CHI is 9-MAY-1995
@Date:   25-APR-2000
@Tape Location:  Disk L10 Track 3
@Bg:     A1
*CHI:    I saw a giraffe and a elephant .
%mor:    pro:sub|I v|see&PAST det:art|a n|giraffe coord|and det:art|
a
         n|elephant .
%gra:    1|2|SUBJ 2|0|ROOT 3|4|DET 4|2|OBJ 5|4|CONJ 6|7|DET 7|5|
COORD 8|2|PUNCT
*CHI:    <that> [/] (.) that (i)s it . [+ bch]
%mor:    pro:dem|that cop|be&3S pro:per|it .
%gra:    1|2|SUBJ 2|0|ROOT 3|2|PRED 4|2|PUNCT
*CHI:    I saw an elephant go swimming .
```

Shown above is an excerpt from the transcript. The main objective of this program is to read these transcripts of the given dataset, for both the SLI and TD groups. We will then conduct a number of pre-processing tasks to extract only the relevant contents or texts needed for analysis in the subsequent tasks.

Upon completing the pre-processing tasks, each of the cleaned transcript will be saved as an individual output file.

The data required for processing and analysis is the narrative produced by the children, which are those statements (or lines) indicated by the label of '*CHI:' in the transcripts. The first step is that, for each original transcript, only those statements which are prefixed or begin with '*CHI:' are extracted.

The next step is to perform a set of pre-processing or filtering tasks. We want to remove certain words (generally referred as tokens) in each statement that consist of some CHAT symbols as either prefixes or suffixes, but retaining certain symbols and words for analysis in Task 2.

The List of Symbols that are filtered out:

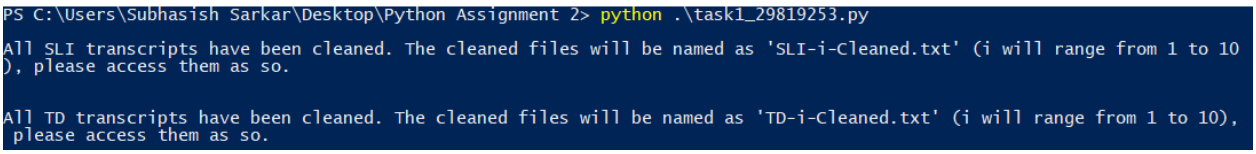
1. Those words that have either '[' as prefix or ']' as suffix are removed but these three symbols: [//], [/], and [*] are retained
2. Those words that have either '<' as prefix or '>' as suffix are retained but these two symbols are removed
3. Those words that have prefixes of '&' and '+' are removed
4. Those words that have either '(' as prefix or ')' as suffix are retained but these two symbols are removed.

It should be noted that there are certain assumptions that have been made while these filtering tasks were carried out:

1. Anything which comes inside '[ ]' have been removed without discrimination.
2. As such elements which have CHAT significance like [* m] and [* m:+ed] were also getting affected. Thus to ensure that the number of grammatical errors did not get affected, both of those elements i.e., both [* m] and [* m:+ed] have been *replaced* with [*], which are then counted.
3. It is assumed that long and medium pauses indicated by (..) and (…) need to be counted as pauses. As such all occurrences of (..) and (…) have been replaced by (.), which are subsequently counted.

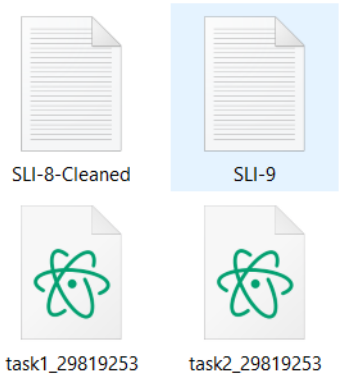
STEPS TO RUN THE PROGRAM:

1. Unzip all the files to a single location. This includes all the SLI, and TD transcripts and the three script files.
2. Using the python keyword, run task1, which then shows the prompt after the script has run.



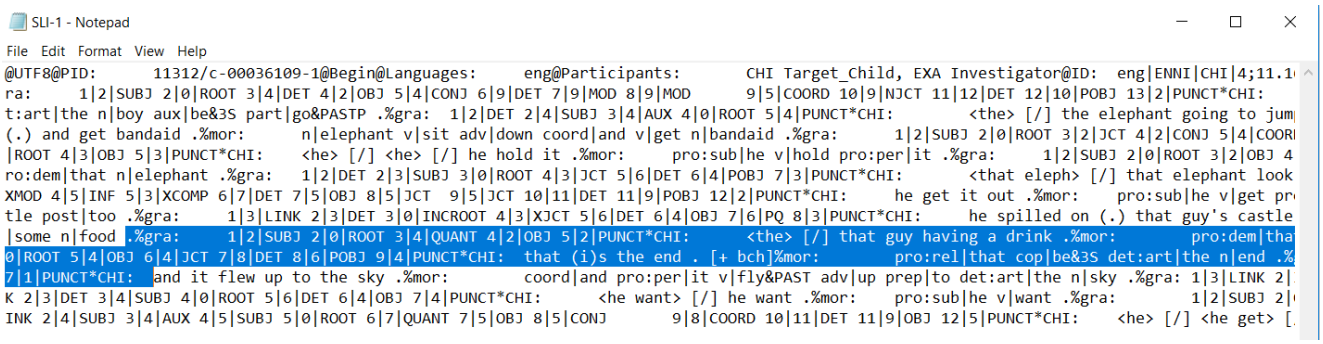
Once they are run on the CLI, all the files are automatically cleaned and stored at the location where the SLI and TD transcripts and scripts are stored.

Like so:

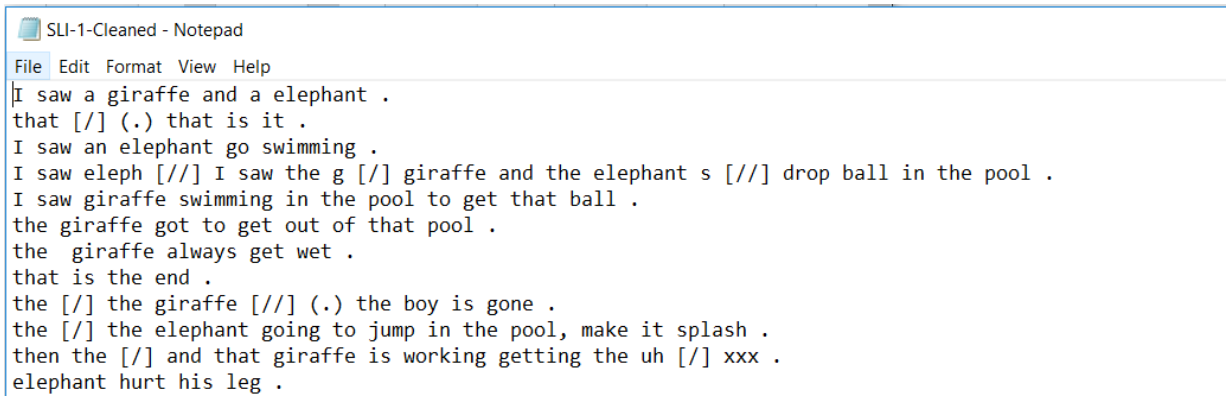


The script merely needs to be run for the cleaned files to be made.

As a result of all the cleaning, the transcript which looked like this:



Now looks like this:



PART 2: Data Analysis

The second task is about collating the required data for analysis. The main task is to produce a number of statistics for the two groups of children transcripts. These statistics are those that might serve as good indicators for distinguishing between the children with SLI and the typically developed (TD) children.

The statistics for each individual file can then be accessed and seen using the menu-based user interaction system built into the script. The user needs to first select a script, and analyse it using the *analyse_script* method defined in the class, by accessing it using an object called from the same class. The user can then show the statistics which then displays all the statistics that were

calculated in the *analyse_script* method.

The statistics for each of child transcript that we are interested in are:

- Length of the transcript — indicated by the number of statements
- Size of the vocabulary — indicated by the number of unique words
- Number of repetition for certain words or phrases — indicated by the CHAT symbol [/]
- Number of retracing for certain words or phrases — indicated by the CHAT symbol [//]
- Number of grammatical errors detected — indicated by the CHAT symbol [*]
- Number of pauses made — indicated by the CHAT symbol (.)

Since the length of each child transcript is measured by the number of statements, the end of each statement can be determined based on the following punctuation marks: either a full stop '.', a question mark '?', or an exclamation mark '!'

The re module is imported again, as there are certain special symbols like apostrophe (') and colon (:) exist within words and need to be removed from the script to ensure that there are no discrepancies in the statistics obtained.

STEPS TO RUN THE PROGRAM:

1. Run the script using the python keyword, and the following prompt is shown:

```
PS C:\Users\Subhasish Sarkar\Desktop\Python Assignment 2> python .\task2_29819253.py

                1. Select a transcript to analyse
                2. Show Statistics
                3. Exit
```

2. Selecting option 1 will ask the user for the which script they want to analyse, in which case, the user should input the name of a cleaned file.

```
PS C:\Users\Subhasish Sarkar\Desktop\Python Assignment 2> python .\task2_29819253.py

                1. Select a transcript to analyse
                2. Show Statistics
                3. Exit

                Enter your choice here:
                1
Enter the name of the file: SLI-1-Cleaned.txt
```

Note that the file name should include the extension ".txt"
After this the menu is displayed again, and the user can now choose either option 2 or 3.

Option 3 is to exit from the menu system.

Choosing option 2 will give out the calculated statistics of the selected file:

```
Enter the name of the file: SLI-1-Cleaned.txt

                1. Select a transcript to analyse
                2. Show Statistics
                3. Exit

                Enter your choice here:
                2
The length of the transcript is: 67
The size of vocabulary used: 117
The number of words or phrases repeated done: 47
The number of words or phrases retraced: 10
The number of grammatical errors made: 1
The number of pauses made: 12
```

A validation check was put in place to ensure that any other input other than 1,2 or 3 would give an error message.

```
                1. Select a transcript to analyse
                2. Show Statistics
                3. Exit

                Enter your choice here:
                5
Please enter a valid number
```

PART 3: Data Visualisation

In the last task, a class to visualise the statistics collected in Task 2 is implemented, as a form of bar graph. The implementation of this visualiser class makes use of the external Python packages, such as NumPy, SciPy, Pandas, and/or Matplotlib in order to create the suitable graphs for comparing the statistics collected for the two groups of children transcripts.

Since the data stored is in the form of a dataframe, an additional method called as *returnDataFrame* was made inside the visualiser class. This method simply returns the dataframe which allows the representation of the statistics in a tabular format, where the columns denote the statistic types and the rows denote the statistic counts of each child transcript.

In the *visualise_statistics* method, two arguments are being taken (self and other). This is done so because the invocation of this method by an object, should trigger the visualisation of both the bar graphs, which will not be possible unless the other object is also taken as an argument.

This decision assumes that the invocation of the method should immediately trigger the display of the graph rather than plot the graph separately outside of the class definition.

STEPS TO RUN THE PROGRAM:

1. Use the python keyword to run the script on the CLI.
2. Once it runs, It displays the following prompt which contains the total average statistics of each group.
   This is being returned by the *compute_averages* method.

```
TD Statistics:

Average length of Transcript: 86.5
Average Grammatical Errors: 0.1
Average Number of Pauses: 38.4
Average Number of Repetitions of Words/Phrases: 17.9
Average Number of Words/Phrases Retraced: 16.3
Average Size of Vocabulary: 173.9

SLI Statistics:

Average length of Transcript: 71.6
Average Grammatical Errors: 0.5
Average Number of Pauses: 27.5
Average Number of Repetitions of Words/Phrases: 23.0
Average Number of Words/Phrases Retraced: 14.6
Average Size of Vocabulary: 130.5
```

3. The TD and SLI Dataframes are being returned by the *returnDataFrame* method as such:

TD Statistics Data Frame

	gerr	len	pau	rep	ret	voc
Script 1	0	95	39	14	11	115
Script 2	0	90	55	8	11	178
Script 3	1	81	39	21	22	196
Script 4	0	87	20	49	7	173
Script 5	0	90	38	9	21	159
Script 6	0	84	42	18	23	180
Script 7	0	76	52	21	22	175
Script 8	0	90	27	10	11	166
Script 9	0	81	42	23	15	193
Script 10	0	91	30	6	20	204

SLI Statistics Data Frame

	gerr	len	pau	rep	ret	voc
Script 1	1	67	12	47	10	117
Script 2	2	70	41	5	11	111
Script 3	0	106	18	39	5	147
Script 4	1	68	53	21	44	130
Script 5	0	77	40	9	18	155
Script 6	0	61	14	14	10	102
Script 7	0	68	8	28	12	142
Script 8	0	72	43	8	13	145
Script 9	0	70	22	45	10	134
Script 10	1	57	24	14	13	122

4. The graph of the average values of all statistics for the two groups occur as such: