Assignment Report

Assignment 3 & 4

Roll No: 2018201035

1 Tokenisation

1.1 Tokenisation of corpus 3 & corpus 4

The output of the tokenisation of corpus 3 and corpus 4 are *.txt* files stored in the folders:

- **Google Drive:**
  https://drive.google.com/drive/folders/1WbiO90_OdBblPEmrnLT2DQHtUnwaydX-?usp=sharing
- **One Drive:**
  https://iiitaphyd-my.sharepoint.com/:f:/g/personal/ankit_pant_students_iiit_ac_in/Eo4Serua-IFPqPaTfLoOOMwBN5ltyhlxxp8CbDTrW1L7nw?e=AyIZkt

The tokeniser separates the various types of tokens using spaces. However the white-spaces in the original text (spaces, tabs, etc.) are ignored. The following types of tokens were identified:

- Words
- Punctuations
- Email addresses
- URLs
- Number / Currency
- Name
- Hashtag
- Mention

1.2 Zipf Graph for corpus 1

To obtain the Zipf graph, the frequency of all words were calculated. Words were identified as sequence of characters delimited by space. The dictionary was then sorted according to frequency (in descending order) and graph was plotted for words that were ranked from 10001 to 11000. The graph can be seen in the following figure 1:

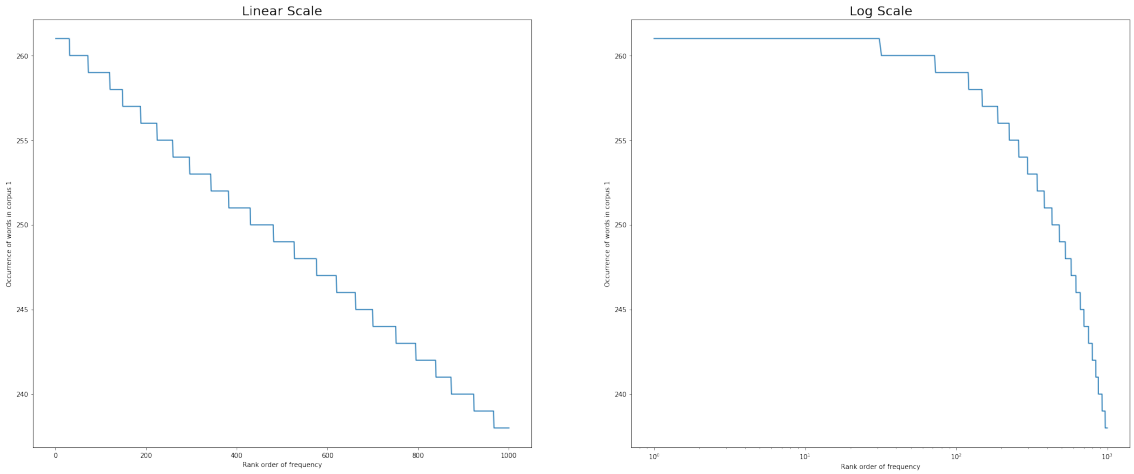

Figure 1: Zipf graph for corpus 1

It can be concluded that the corpus follows the Zipf's Law to some extent. Since the plot is done for 10001 to 11000 ranked words, the graph isn't representative of the ideal Zipf graph. However the notion that the count of words drop sharply can be seen in the graphs above. The sub-graph containing the x-axis on a logarithmic scale represents this phenomenon better.

1.3 Zipf Graph for corpus 2

To obtain the Zipf graph, the frequency of all words were calculated. Words were identified as sequence of characters delimited by space. The dictionary was then sorted according to frequency (in descending order) and graph was plotted for words that were ranked from 10001 to 11000. The graph can be seen in the following figure 2:

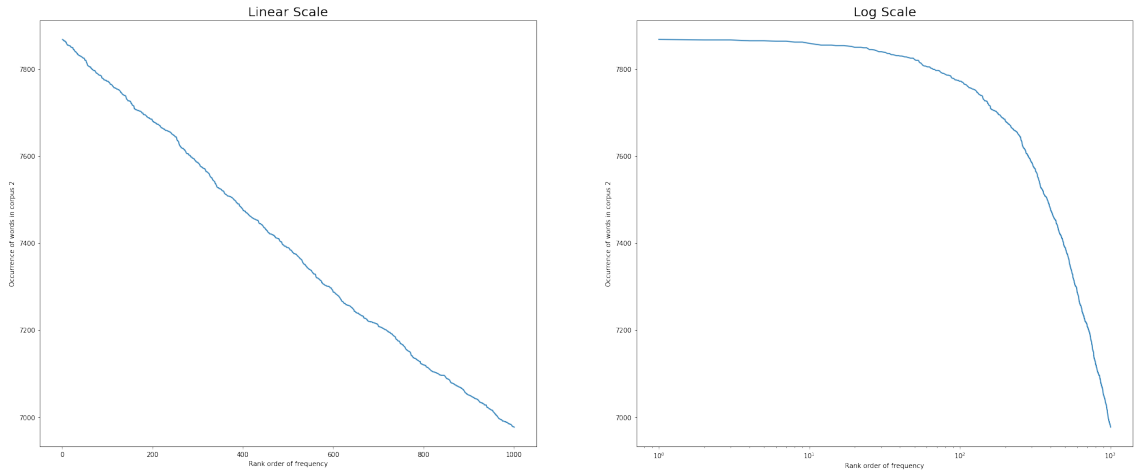

Figure 2: Zipf graph for corpus 2

It can be concluded that the corpus follows the Zipf's Law to some extent. Since the plot is done for 10001 to 11000 ranked words, the graph isn't representative of the ideal Zipf graph. However the notion that the count of words drop sharply can be seen in the graphs above. The sub-graph containing the x-axis on a linear scale represents this phenomenon better.

2 Creating Language Models

2.1 Creating 4-Gram model for corpus1.txt

The model was created and saved in https://drive.google.com/drive/folders/1WbiO90_OdBblPEmrnLT2DQHtUnwaydX-?usp=sharing. Since Smoothing and Interpolation also needed to be done along with 4Gram, 3Gram, 2Gram and 1Gram models were also saved (found in the aforementioned URL).
The Entropy and Perplexity of the model was also calculated as had the following values:

- **Entropy:** 5.634462686484124e-05
- **Perplexity:** 1.000039055881912

2.2 Creating 6-Gram model for corpus1.txt

The model was created and saved in https://drive.google.com/drive/folders/1WbiO90_OdBblPEmrnLT2DQHtUnwaydX-?usp=sharing. Since Smoothing and Interpolation also needed to be done along with 6Gram, 5Gram, 4Gram, 3Gram,

2Gram and 1Gram models were also saved (found in the aforementioned URL). The Entropy and Perplexity of the model was also calculated as had the following values:

- **Entropy:** 0.00011187311204822931
- **Perplexity:** 1.0000775475388517

2.3 Creating 4-Gram model for corpus2.txt

Since the size of the data-set was huge ($\approx 6\,G.B.$), the file was split into 16 files (each $\approx 400\,M.B.$). But creating all 4Gram through 1Gram failed due to memory being exceeded despite individually processing the files. Hence the perplexity of the model could not be calculated.

2.4 Creating 6-Gram model for corpus2.txt

Since the size of the data-set was huge ($\approx 6\,G.B.$), the file was split into 16 files (each $\approx 400\,M.B.$). But creating all 4Gram through 1Gram failed due to memory being exceeded despite individually processing the files. Hence the perplexity of the model could not be calculated.