Zipf's Law verification and Construction of Word Cloud

Stuti Chaturvedi
202161006

Haripriya Goswami
202161003

3rd Yashita Vajpayee
202162012

***Abstract*—In this experiment, we aim to validate Zipf's law and generate word cloud on a text file from Project Gluttenberg.We will also use MapReduce in hadoop to process large amount of data.**

I. Introduction : Word frequency and Zipf's Law

Frequency of a word/term in a corpus plays an important role in determining the relevancy of a document with respect to the given query.
Zipf's law make use of this and ranks each term according to the number of times of its occurrence in the corpus.
Zipf's law states that, if t1 is the most common term in the collection, t2 is the next most common, and so on, then the collection frequency cfr of the rth most common term is proportional to 1/r:

$$cfr \propto 1/r$$

If there is same number of occurrence for two words, dictionary order is preferred. The Zipf's law follows the Power Law as:

$$cfr = C.r^{-\alpha}$$

$$log\ cfr = log\ C - \alpha\ log\ r$$

Where, cfr is the frequency of occurrences, C is a corpus dependent constant, r is the rank of words and α is for the exponential distribution of power law

II. Approach and Implementation

A. Implementation

Python implementation is done to carry out this experiment. Libraries required for this experiment are numpy, scipy,nltk,matplotlib and wordcloud.

B. Importing text and tokenization

We selected Carrol-allice text from Project Guttenberg for this experiment. After importing, tokenization is performed and unique words are listed out.

C. Frequency calculation

After listing of the unique words, calculation of frequency of each word is done and this data is entered into a dictionary.

D. Ranking

Ranks are assigned to each word according the frequency obtained in the previous step. *scipy.stats.rankdata* is used to rank and the list is then sorted in increasing order of rank. A histogram is plotted depicting the *count vs words* distribution of first fifty ranked words.

E. Zipf's law validation

In order to validate the Zipf's law, calculation of exponential constant and Corpus constant is done. This can be achieved by performing least squares method. This is calculated by plotting a graph *log(rank) vs log (frequency)*

F. Word Cloud

Word cloud visually represent the frequency of words. The word with most occurrences has the highest font size while the one with lowest frequency has smallest font size. We implemented the word cloud on the same txt file *"carroll-alice.txt"* using the python module WordCloud.

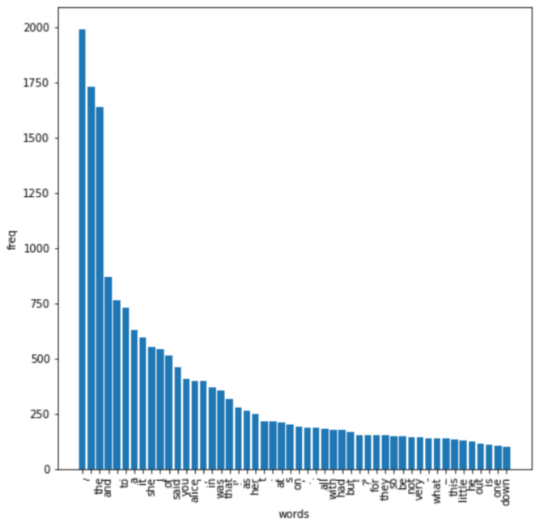
G. MapReduce

MapReduce can basically divide into two parts: Map where the input data is break into smaller pieces and Reduce takes these pieces as inputs to reduce data into smaller framework. In Hadoop, we implemented MapReduce through Python by using mapper.py and reducer.py .

III. Results and Visualization

A. ***Freq vs Word plot***

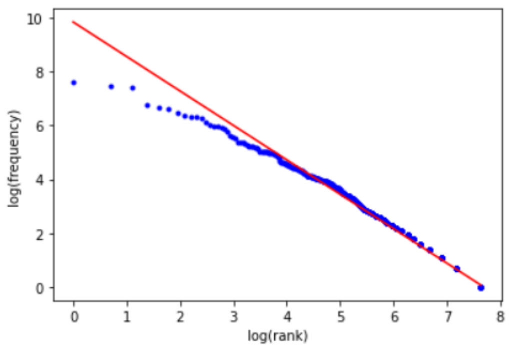
Following is the histogram showing the distribution of first fifty ranked words according to their frequency:



Since the graph has not excluded stopwords, count of less important terms like ’the’, ’and’ etc is higher.

B. *log(frequency) vs log(rank) plot*

Following is the plot:

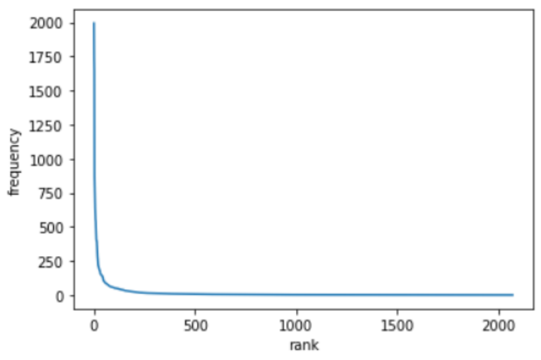


```
Slope: -1.278932182411863
Intercept: 9.834397362103873
Text(0, 0.5, 'frequency')
```

After calculating intercept and slope through code, we see that the slope of graph is around -1.3. The most frequent words are showing highest deviation(error) from the ideal curve(red line)

C. *frequency vs rank plot*

Following show the rank of the words according to their frequency:



The plot shows the inverse relation between the frequency and rank. We can observe that higher the frequency, lower is the rank.

D. *Word Cloud- I*

The following word cloud is including the stopwords.

and
the
to
a
she
of
it
said
you
alice
was
that
in

Stopwords are the words which occur frequently, but does not provide much information to our text.

E. *Word Cloud- II*

The following word cloud is excluding the stopwords. The less meaningful words like ’the’,’and’,’to’ are missing from this cloud.

alice
said
little
one
queen
know
thought
went

F. *Word Cloud- III*

The following word cloud is generated after performing stemming:

said
alic
littl
know
one
went
look
thought
queen

Stemming is reducing a word to its word stem. Here we used PorterStemmer module to execute this.

IV. Conclusion

After performing the experiment, we conclude that Zipf's law is validated as the inverse relationship is witnessed between frequency and rank and the slope of log(freq) vs log(rank) is around -1.

References

[1] Introduction to information retrieval by Christopher D Manning Prabhakar Raghavan Hinrich Schutze

[2] https://nlp.stanford.edu/IR-book/html/htmledition/zipfs-law-modeling-the-distribution-of-terms-1.html