NLP Assignment 1 - Tokenizer

KARAN MANGLA

201301205

We performed the various experiments and experiments in order to verify the **Zipf's Law** on our corpus of Data . We were provided 3 Datasets one each in English , Telugu and Hindi . So we basically had to prove that the "The frequency of a word is inversely proportional to its statistical rank (in a corpus) "

Basically all the results we got were in accordance with the Zipf's law showing that

$$r \propto i/f$$ (r = statistical rank, f = frequency)

Procedure :

- I read the files line by line , stripping it and doing a basic preprocessing on it so that we can apply the different regex expressions to match the tokens .
- The preprocessing part involves putting spaces around certain punctuations such as ("<" , ">" , ";" , "=" , "|") cause they might have words attached to them and thus might hinder in the further process of tokenization . In this process the extra spaces have also been removed making use of regular expressions . After that we split it by space and pass each string token by token into the tokenizing function .
- Then we pass it to a function to gather out the the tokens by making use of regex expressions . The list of regex that I implemented are :
    - Matching dates
    - Matching numbers [both with decimal(.) and comma(,)]
    - Matching units present at end of numbers like km , ml
    - Matching URLs
    - Matching Email-IDs
    - Matching apostrophe containing words like Karan's
    - Matching fractions
    - Matching abbreviation words like U.S.A
    - Words ending with a [: | , |.] and some more

- We also maintained a list of certain Abbreviations which are used generally such as (‘Dr.' , 'Esq.', 'Hon.' , 'Jr.', 'Mr.' ,'Mrs.','Ms.') , so that we could match them directly and not waste time and computation in further processing .
- Now the matched words are added to a dictionary which maintains a mapping of the string tokens along with their frequency .
- Now coming to the bigrams and trigrams part , we make sure that all the tokens are put in order into the list having the tokens as we have to take two and three adjacent tokens in it .Then the combinations are added to the dictionary along with popping out of the first element in the list .

Difficulties faced :

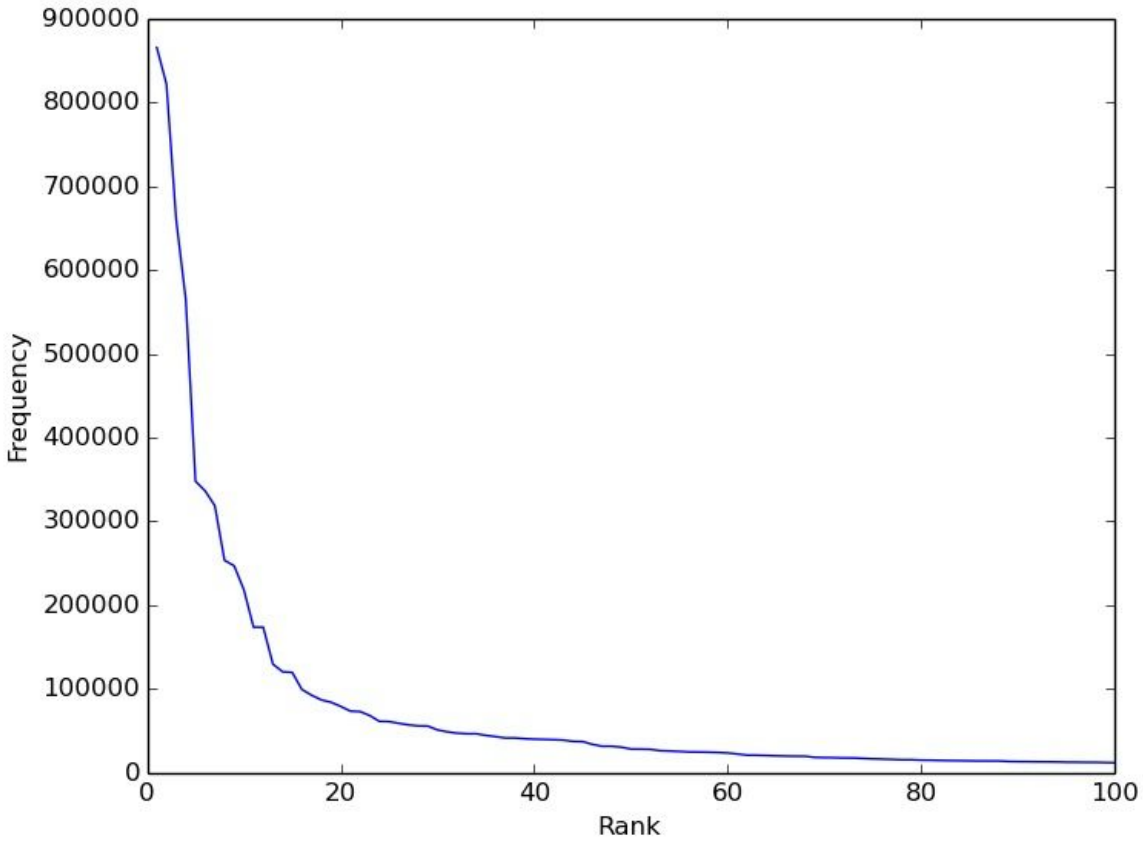
- The size of the data set was the biggest hindrance for me as processing it required a lot of patience . Sitting idle waiting for the graph to be processed .
- The large size of the data set made my laptop hang on many occasions sometimes even requiring to restart the system .
- The large size of the data sets made it difficult to incorporate all the cases and debugging was tough taking into consideration the large size of the data and it felt like I am missing some cases all the time .

Results :

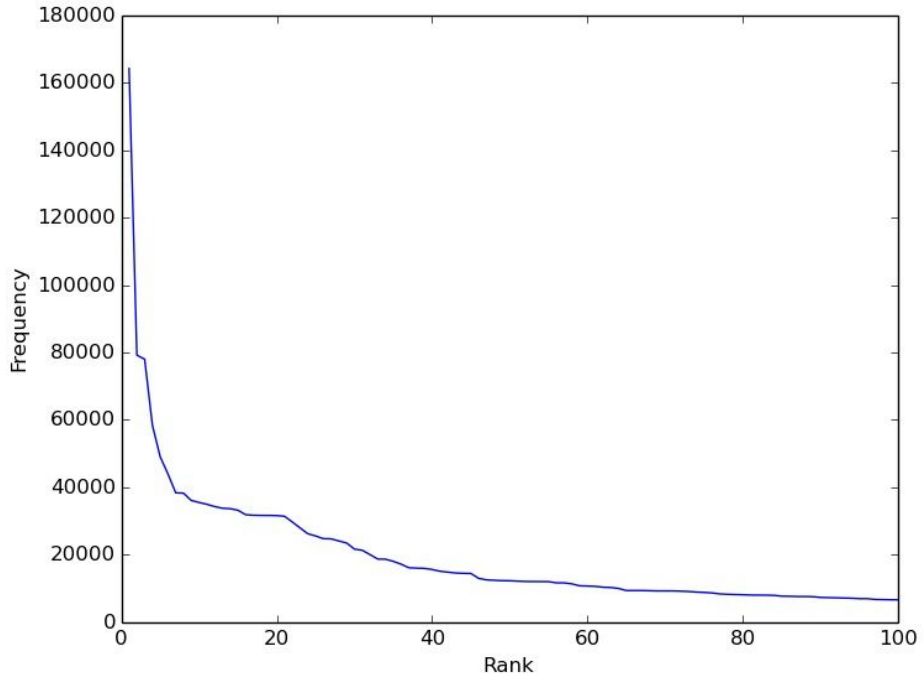
We see clearly that all the cases obey the Zipf's law as the frequency is inversely proportional to the Statistical

English

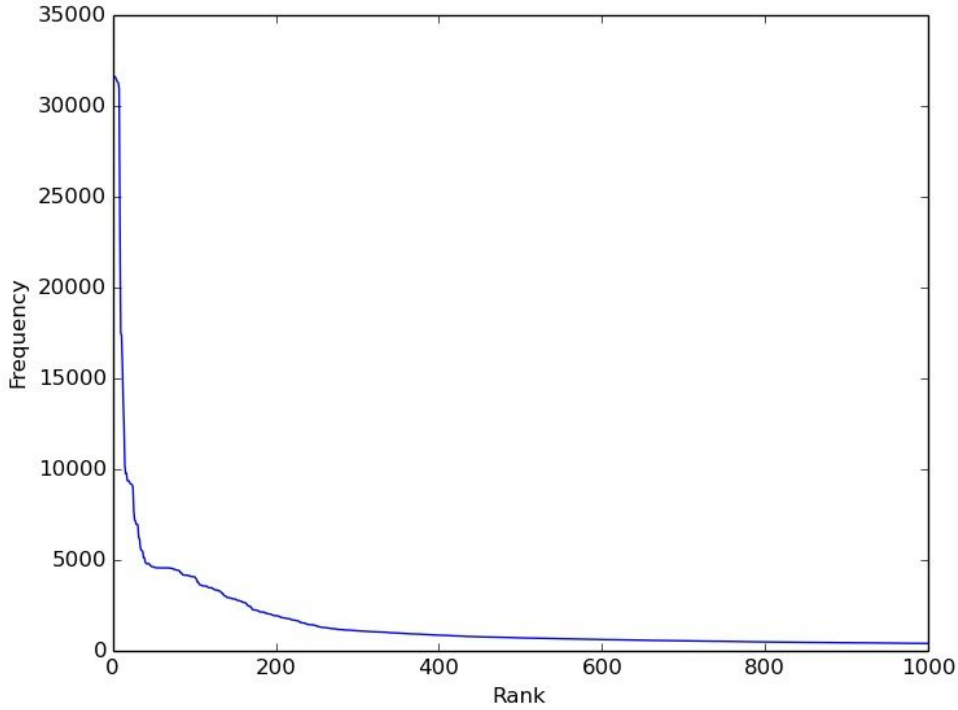
1 . Unigram



2. Bigrams

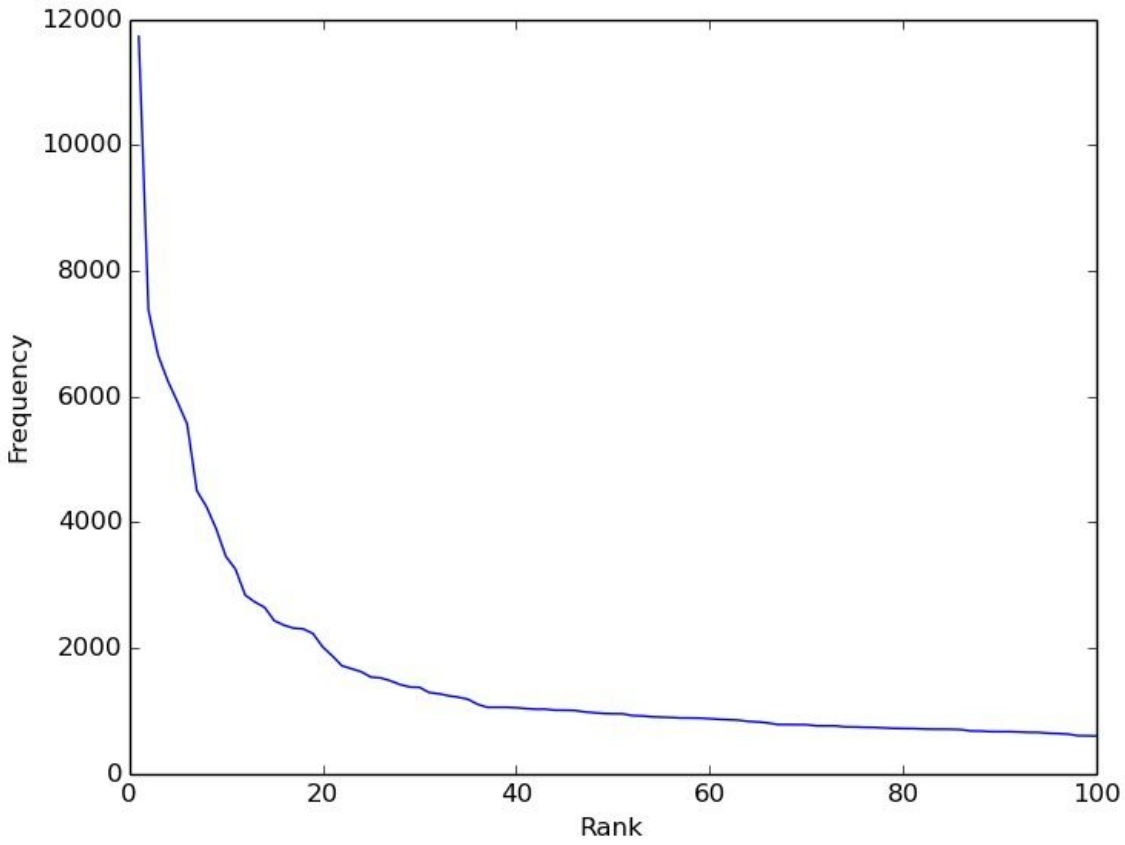


3 . Trigrams

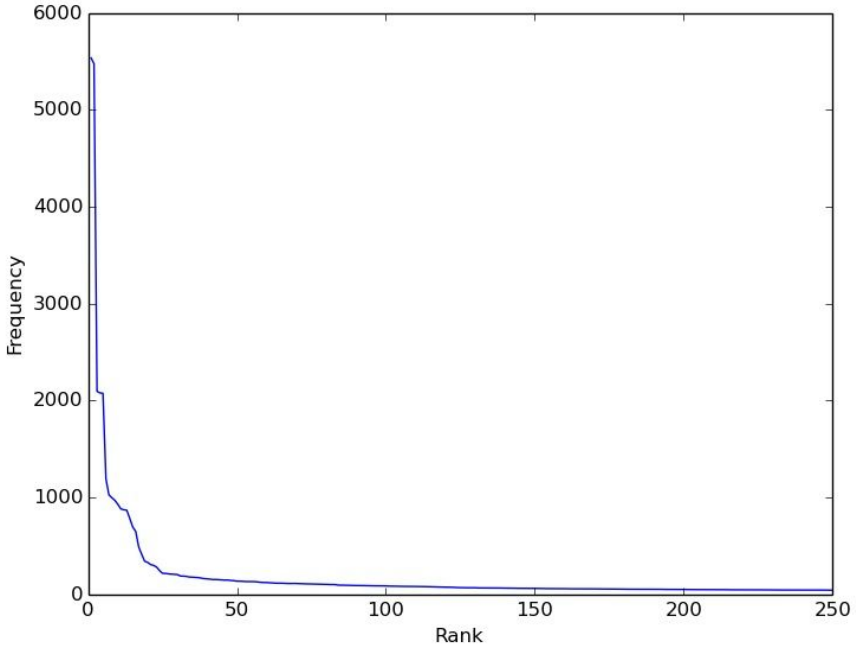


Telugu

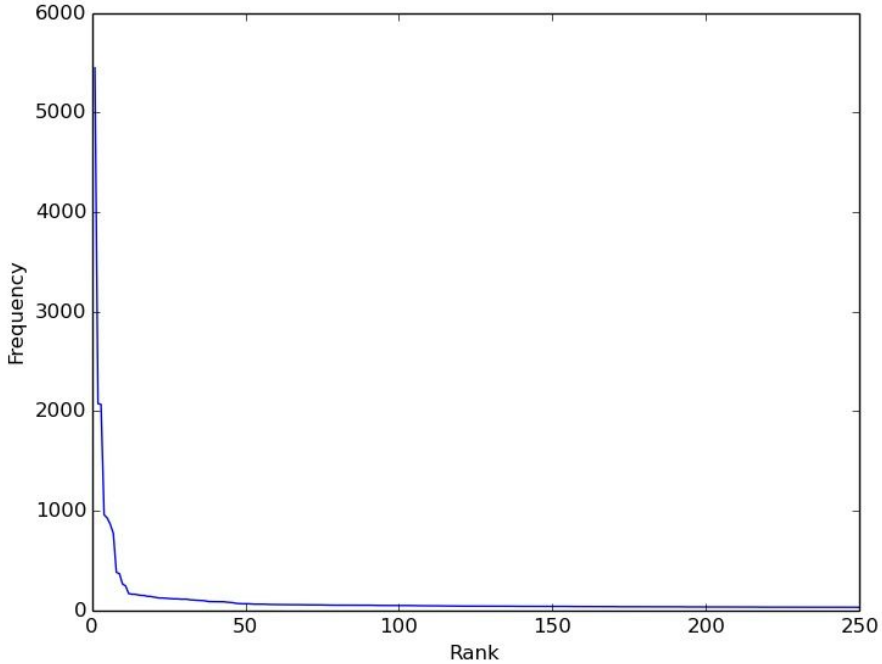
1 . Unigram



2. Bigrams

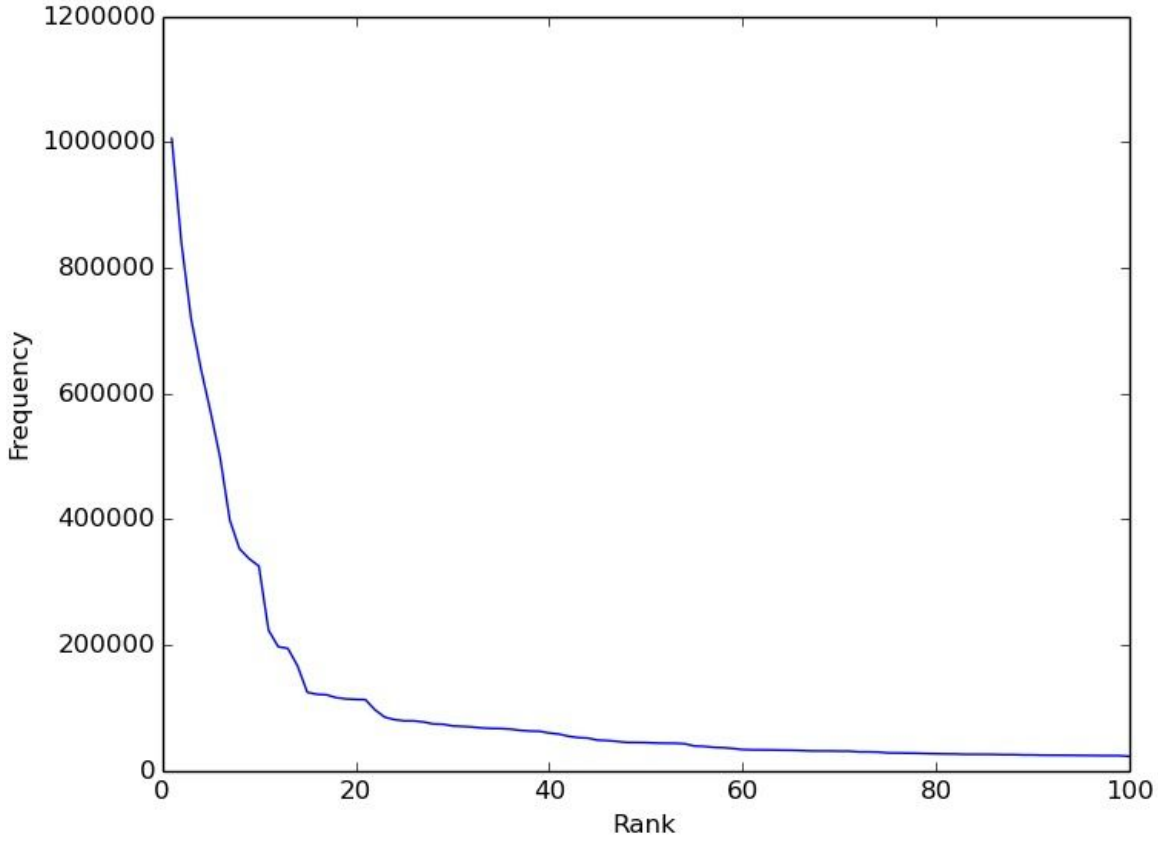


3. Trigrams

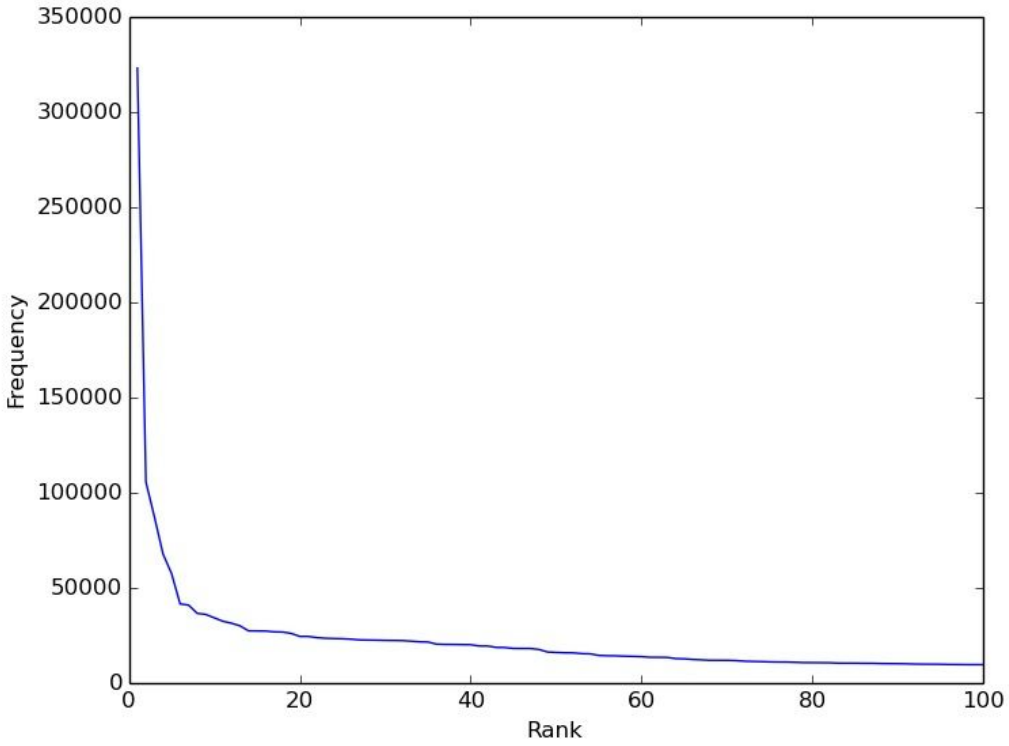


Hindi

1. Unigram



2. Bigrams



3. Trigrams