Ho Vinh Thinh (2562630)
Prabal Agarwal (2561419)
Nurzat Rakhmanberdieva (2561148)

1.1

Common text processing techniques are:

1. **Tokenization**
This tool is used to reduce the text corpus to word level data. In English texts there are frequent occurrences of hyphenated words, emoticons, html tags, etc which needs to be converted into separate tokens. This tools also removes special characters from the text. In other languages such as Chinese where a number of words are combined together in a phrase or a sentence, designing a tokenizer is even more challenging and important.

2. **Stop word removal**
Each language uses some words belonging to the category of prepositions in order to grammatically represent the relation between entities. Such words have very high frequencies. When designing a information retrieval system, such words don't contribute much to the metric used for evaluation of similarity between a query and a documents or between two documents. Hence, such words must be removed during the preprocessing phase. The set of stop words can be identified by using some basic language statistics.

3. **Stemming**
Stemming is used to reduce all the morphological forms of a single word to a unique string.
Eg. words such as call, called, calling can be reduced to a single unique string such as "call".
Stemming is useful in information retrieval as for a query term, the system can search for all the morphological forms and retrieve potential relevant documents.

4. **Dependency parsing**
Parser is an NLP tool that is used to find the dependency relations among the words in a sentence. Implementing a parser is useful in semantic searching. Apart from dependency relation, a full parser also tags each word with the person, tense, number(singular or plural) and this data can be leveraged to extract documents that are more relevant to the query being searched for.

1.2

Documents from three languages English, German and French. Following are the links to the documents:

`http://www.gutenberg.org/cache/epub/611/pg611.txt`
`http://www.gutenberg.org/cache/epub/5072/pg5072.txt`
`https://www.gutenberg.org/files/47743/47743-0.txt`

Instructions to run the tokenizer:
For Linux:

```
tar -xzvf code.tar.gz
cd code
bash build
bash run [inputTextFile] [outputTokenizedTextFile] [freqDataFile]
```

bash run takes 3 parameter:
inputTextFile: input file contains the input document.
outputTokenizedTextFile: output file contains preprocessed document.
freqDataFile: output file contains frequency of words in preprocessed document.

The tool will generate [outputTokenizedTextFile] and [freqDataFile], then draw Zipf's Law chart.
In addition, a file called "StemmerTermMap.txt" is generated which contains all tokens with the same meaning that are explored by stemmer.

Ho Vinh Thinh (2562630)
Prabal Agarwal (2561419)
Nurzat Rakhmanberdieva (2561148)

2.1

The first rank tokens are the following:

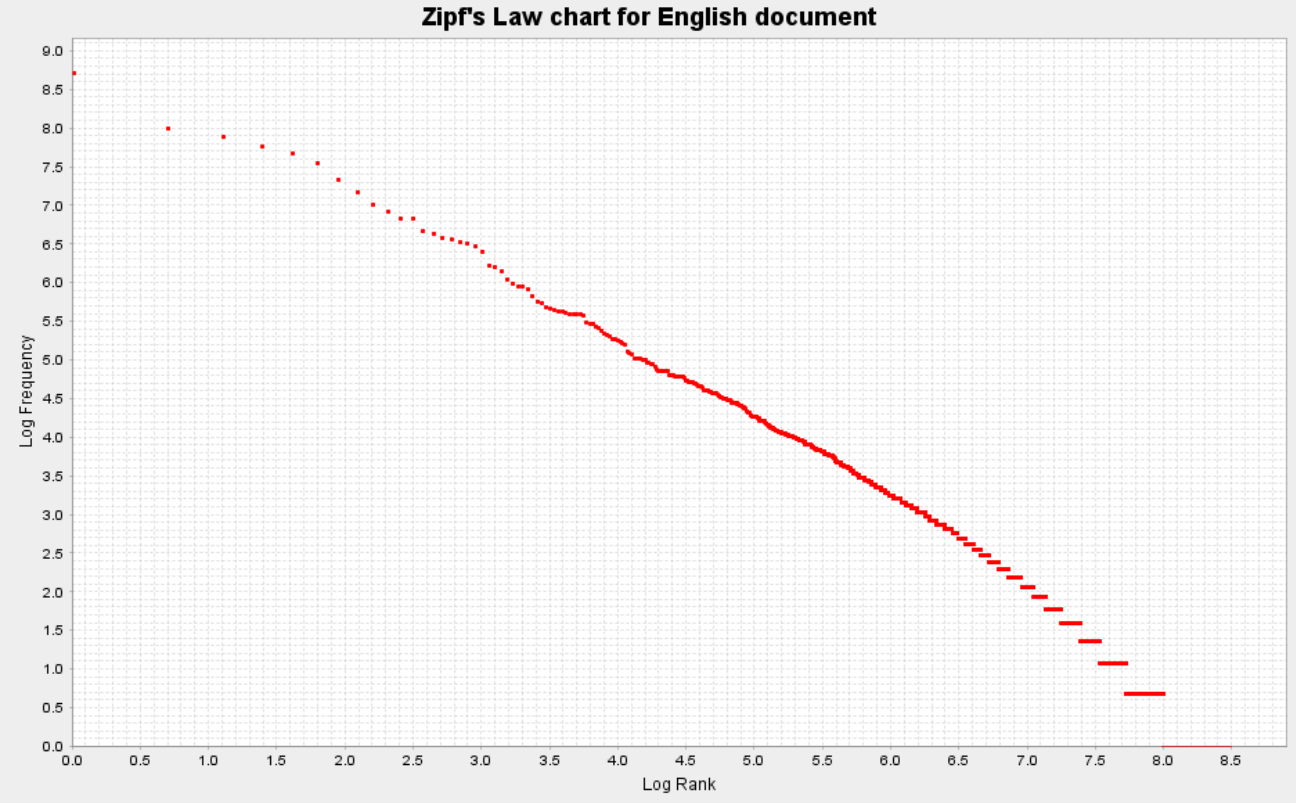
English: the
German: die
French: de

2.2

Log transformation is done on data because values can range over several orders of magnitude. In the present case, the there are around 1000 different words in the text corpus and some of the words have frequencies in the order of 10000. Hence, taking a log, reduces the values to comparable magnitude and good statistical models can be built on such a transformed data due to low variance.

Frequency vs rank plot for the documents are the following:

Figure 1: Plot for English language



Ho Vinh Thinh (2562630)
Prabal Agarwal (2561419)
Nurzat Rakhmanberdieva (2561148)

Figure 2: Plot for German language

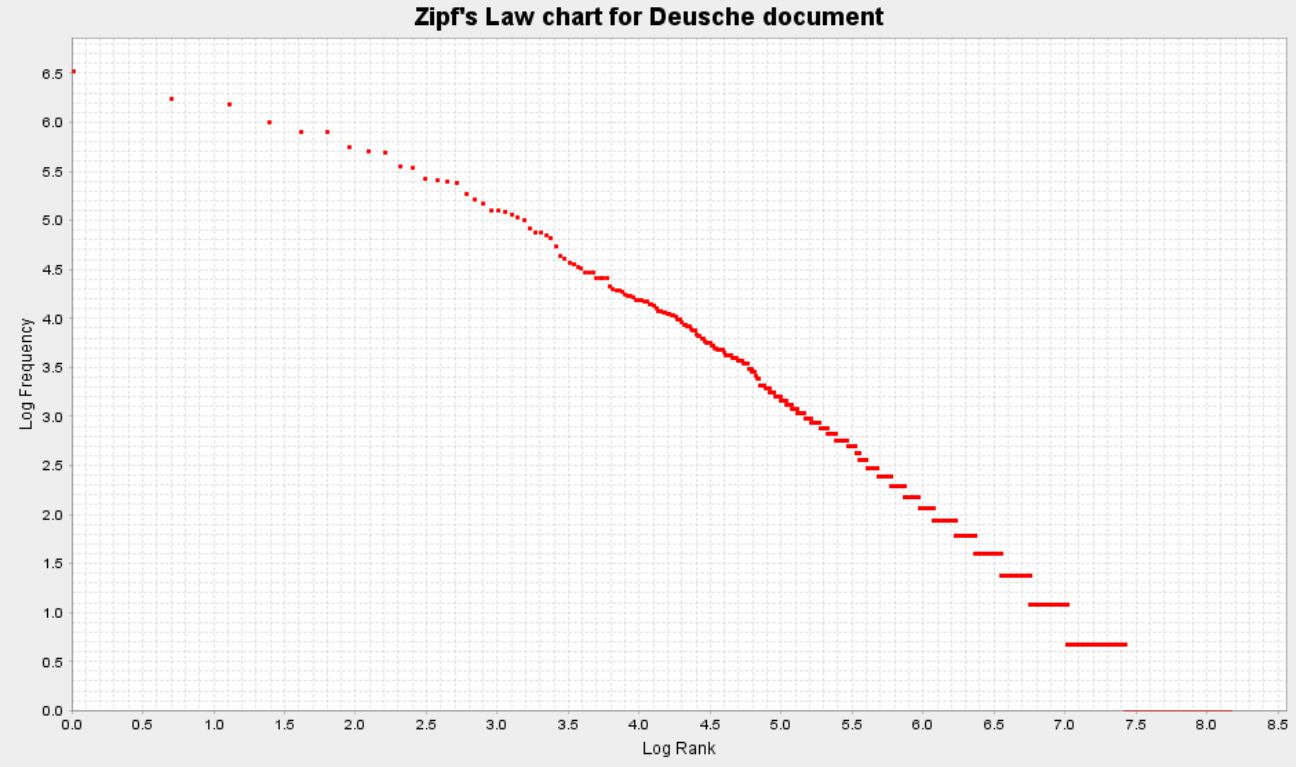


Figure 3: Plot for French language