BIG DATA ANALYSIS OF TWITTER DATA, NEW YORK TIMES ARTICLES AND COMMON CRAWL DATA FOR CRIME IN UNITED STATES

BY
SARANYA NUKA ILLA
RAVALI PINNAKA

1. Introduction

There has been a lot of crime occurring in the United States, recently. The major happenings are “2019 Morrisville killings”, “2019 Nevada killing spree”, “Beating of Dnigma Howard” etc. This made us to select the topic “crime” for our project.
The 5 sub-topics selected are Murder, Assault, Abuse, Rape and Robbery.

2. Implementation

This project has three main parts called Data Aggregation, Big Data Analysis and Visualization. Let us look at each of them in detail.

2.1 Data Aggregation

Data from three sources namely - Twitter, New York Time and Common Crawl has been collected in the following way.

2.1.1 Twitter Data

Tweets related to crime topics are collected using ***searchTwitter*** API of R. During the collection of tweets, it is made sure that only 2019 tweets from the United States region are collected by giving specific parameters such as “since” and “geocode” while fetching using the API.
The following keywords have been used to collect tweets from the start of 2019 and we were able to collect 25000 Tweets in total.

- Murder
- Assault
- Abuse
- Rape
- Robbery
- woman
- child
- kidnap

For each keyword, 5000 tweets have been collected. The extracted tweets have been converted to a dataframe and the tweet text has been saved to a .txt file.

2.1.2 New York Times

We have used NYT API to fetch the articles from various topics as query words. Though the minimum requirement is only like 100 articles per topic and 500 overall, we have collected **1996** articles in total, only during the period of 2 weeks from March 16 –March 31.

We have used python script that uses requests library to hit the API and fetch the results as JSON. The NYT article API gives only ten results per page, so we have written the code in a way that the pagination is maintained to fetch all the results automatically. The URL of the articles is present at the web_url field of docs object in the response. Beautiful soup library of python is used to fetch the paragraph tags from the web pages of the collected urls and are written to text files which are later processed.

2.1.3 Common Crawl

Owing to the minimum requirements of 500 articles in total, only two archives for the dates 2019-04-15 and 2019 are used in collecting them in order to make sure that only 2019 articles are collected. We were able to collect **993** articles overall.

Using the ***aws command-line interface*** and the command

aws s3 ls --recursive s3://commoncrawl/crawl-data/CC-NEWS/ ,a full-list of zipped warc files is obtained from which we have chosen only 3 files which are unzipped as warc files to the local machine using python's gzip library.

#crawl-data/CC-NEWS/2019/04/CC-NEWS-20190414063114-00197.warc.gz

#crawl-data/CC-NEWS/2019/04/CC-NEWS-20190413184753-00195.warc.gz

#crawl-data/CC-NEWS/2019/04/CC-NEWS-20190415106109-00198.warc.gz

The warc files which consists of thousands of webpages (one of the files of one day's retrieval) are parsed to find the warc-target-url for each record of web page. In order to capture most relevant results to our topics, we chose to pick only those urls that contain the keywords even though it leads to less results. The web urls collected are hit and only the paragraph tags within those web pages are fetched and stored in text files.

Individual text files corresponding to a singles topic are formed for each webpage. All of these are also combined into a single file using python scripts. We preserve both the combined file and all the individual files for future use.

2.2 Big Data Analysis

The data we have collected is in its raw-format. As we have only preserved all of the data collected as text files, we are going to clean this by using libraries of python useful for text-processing.

- First, we wanted to remove all the special characters in the data. Using regex library of python, all the special characters such as [,],(,),..,',",<,>,$,@,&,* and many more are replaced with an empty character.
- Stop words are the natural words which occur more frequently in an article like 'a','and','the'. Nltk package in python is very useful in text processing where we tokenized the text to remove the stopwords in English and perform stemming and lemmatization.
- During the above process, we have added our own set of stopwords as well for the further cleaning of data to get meaningful wordcount. So, these were removed by comparing with a text file which has a list stopwords and ignore that word in the article if it has been found in the text file.
- Also, we used Porter stemmer and WordNet Lemmatizer of nltk.corpus package and tried to preserve the original word as much as we can by taking into account its position and context in the text.

As there are many files with data, we preprocess the text using big data mapper function where the raw data is cleaned by removing all the special characters,stop words and performing stemming and lemmatization after the tokenization for which the count is set to 1 upon each word occurrence in the text files.

2.3 Word Count and Co-occurrence using the MapReduce Framework

MapReduce is a framework which allows us to write applications that process huge amounts of data, in parallel, on large clusters of commodity hardware in a reliable manner. As huge amounts of data is already collected, we used MapReduce framework to find the word counts and word co-occurrences in tweets,articles and web pages.
Mapper and reducer are implemented as follows:

- Mapper: In the mapper, python code is written to preprocess the data by removing all the unwanted words. Regex expressions are used to remove special characters, extra spaces etc. Also, The mapper process the data in the above process mentioned in order to tokenize the text and map the word count of individual tokenized word to 1.

- Reducer: The reducer takes in all of these counts to produce final count for each of these tokenized words in the text.

Here we are interested in finding only top 10 word co-occurrences in the data. So,it would be a good start to first find all the word counts and sort them to collect only top 10 of them owing to the fact that it is enough to find co-occurrences of these top 10 in their respective neighborhood.

2.3.1 Word Count using MR Framework

The MR Framework takes the input files ,process them using mapper and reducer to give the output. We specify all the files path to do this. As the MR Framework takes in the output that sits on the hdfs system we first move the data from local machine to hdfs file system using the following command. Before that, we start the hadoop file system.

- To start Hadoop File System

  start-dfs.sh

- To enable run-time permissions for mapper and reducer .py files

  chmod +x mapper.py
  chmod +x reducer.py

2.3.1.1 Commands for generating word count of CommonCrawl data:

- To upload source files to HDFS

  hdfs dfs -put /cc.txt /

- Command to generate word count

  hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming3.1.2.jar -input /cc.txt -output /output/wordcount_cc -file wordcount_mapper.py -mapper wordcount_mapper.py -file reducer.py -reducer reducer.py

The above command is processed on the entire text file of common crawl files collected for these topics which gives the words their count separated by a comma in the sort

- To fetch output from Hadoop File System

  hdfs dfs -get /output/wordcount_cc

- To terminate Hadoop File System

stop-dfs.sh

2.3.1.2 Commands for generating word count of NYTimes data:

- To upload source files to HDFS

  hdfs dfs -put /nyt.txt /

- Command to generate word count

  hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming3.1.2.jar -input /nyt.txt -output /output/wordcount_nyt -file wordcount_mapper.py -mapper wordcount_mapper.py -file reducer.py -reducer reducer.py

The above command is processed on the entire text file of common crawl files collected for these topics which gives the words their count separated by a comma in the sort

- To fetch output from Hadoop File System

  hdfs dfs -get /output/wordcount_nyt

- To terminate Hadoop File System

  stop-dfs.sh

2.3.1.3 Commands for generating word count of Twitter data:

- To upload source files to HDFS

  hdfs dfs -put /twitter.txt /

- Command to generate word count

  hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming3.1.2.jar -input /twitter.txt -output /output/wordcount_twitter -file wordcount_mapper.py -mapper wordcount_mapper.py -file reducer.py -reducer reducer.py

The above command is processed on the entire text file of common crawl files collected for these topics which gives the words their count separated by a comma in the sort

- To fetch output from Hadoop File System

  hdfs dfs -get /output/wordcount_twitter

- To terminate Hadoop File System

stop-dfs.sh

2.3.1.4 Commands for generating word cooccurrence of CommonCrawl data:

- To upload source files to HDFS

  hdfs dfs -put /cc.txt /

- Command to generate word count

  hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming3.1.2.jar -input /cc.txt -output /output/wordcooccur_cc -file wordcooccur_mapper.py -mapper wordcooccur_mapper.py -file reducer.py -reducer reducer.py

The above command is processed on the entire text file of common crawl files collected for these topics which gives the words their count separated by a comma in the sort

- To fetch output from Hadoop File System

  hdfs dfs -get /output/wordcooccur_cc

- To terminate Hadoop File System

  stop-dfs.sh

2.3.1.5 Commands for generating word cooccurrence of NYTimes data:

- To upload source files to HDFS

  hdfs dfs -put /nyt.txt /

- Command to generate word count

  hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming3.1.2.jar -input /nyt.txt -output /output/wordcooccur_nyt -file wordcooccur_mapper.py -mapper wordcooccur_mapper.py -file reducer.py -reducer reducer.py

The above command is processed on the entire text file of common crawl files collected for these topics which gives the words their count separated by a comma in the sort

- To fetch output from Hadoop File System

  hdfs dfs -get /output/wordcooccur_nyt

- To terminate Hadoop File System

  stop-dfs.sh

2.3.1.6 Commands for generating word cooccurrence of Twitter data:

- To upload source files to HDFS

  hdfs dfs -put /twitter.txt /

- Command to generate word count

  hadoop jar $HADOOP_HOME/share/hadoop/tools/lib/hadoop-streaming3.1.2.jar -input /twitter.txt -output /output/wordcooccur_twitter -file wordcooccur_mapper.py -mapper wordcooccur_mapper.py -file reducer.py -reducer reducer.py

The above command is processed on the entire text file of common crawl files collected for these topics which gives the words their count separated by a comma in the sort

- To fetch output from Hadoop File System

  hdfs dfs -get /output/wordcooccur_twitter

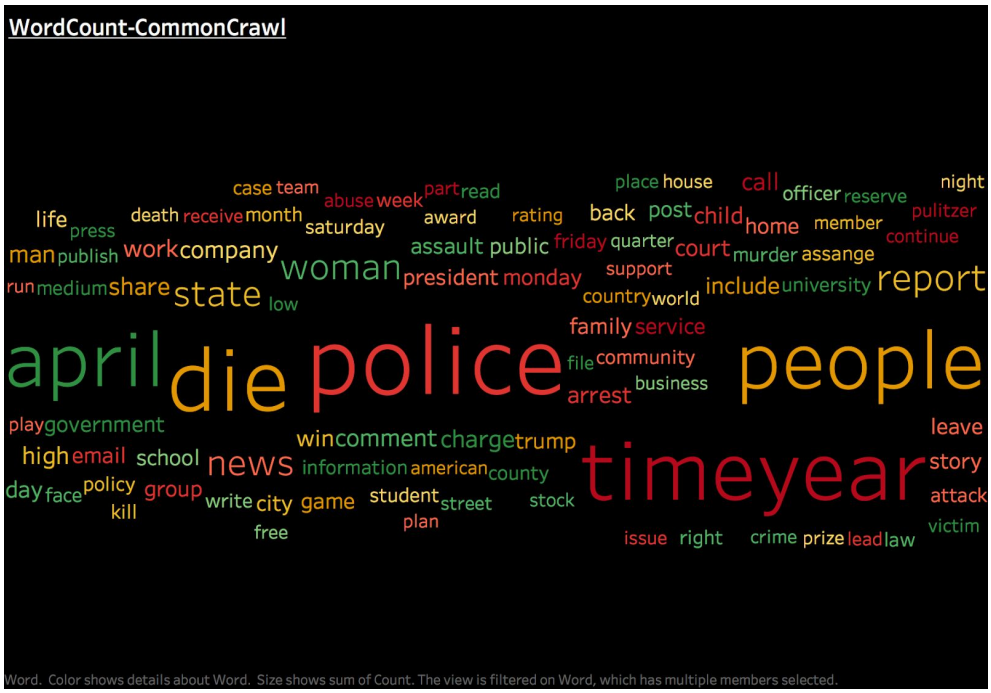
- To terminate Hadoop File System

  stop-dfs.sh

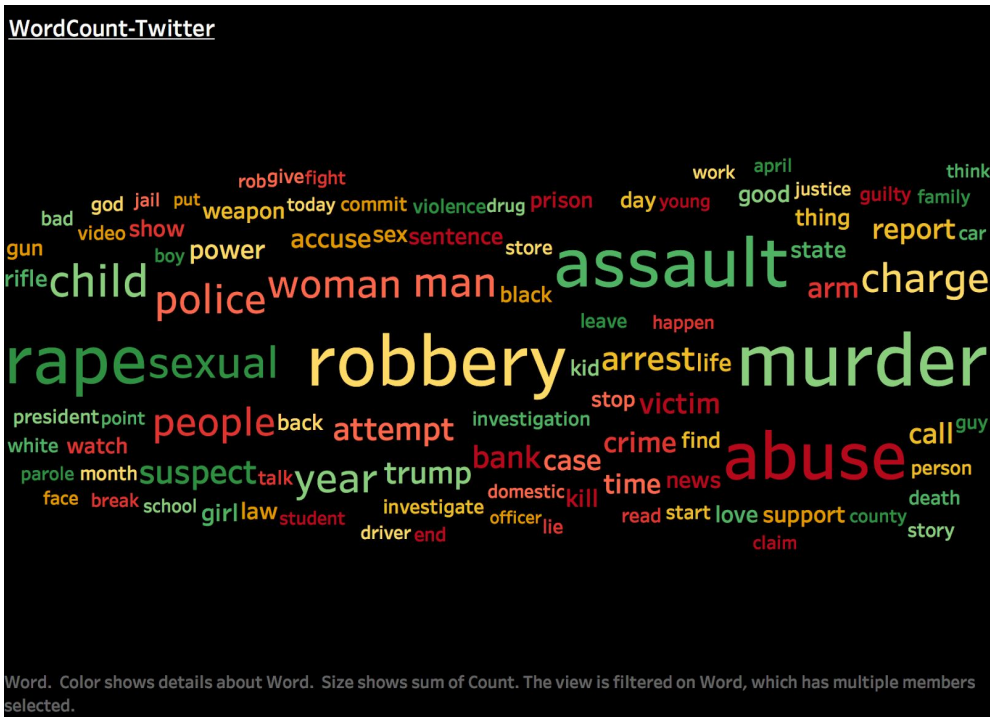
2.4 Visualizations using Tableau

To display the word cloud associated with each topics, we have used Tableau server. The output files obtained from the HDFS have been given as the data source for the Tableau Workbook. There are 2 columns in the file, Word and Count. Word column is considered for text and colors of the word cloud. Count (frequency) is considered for the size of each word in the word cloud. The word cloud for the top 100 words have been visualized for the 3 sets of data - namely, Twitter, NYTimes and CommonCrawl.

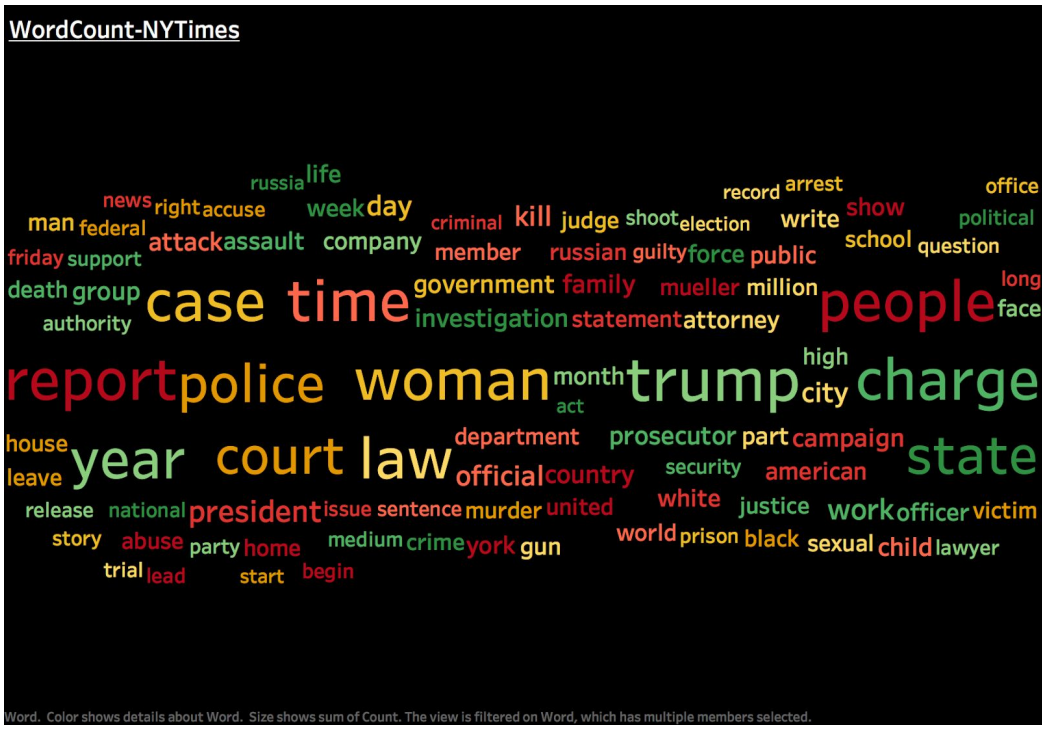
The word cloud for word count of CommonCrawl data:



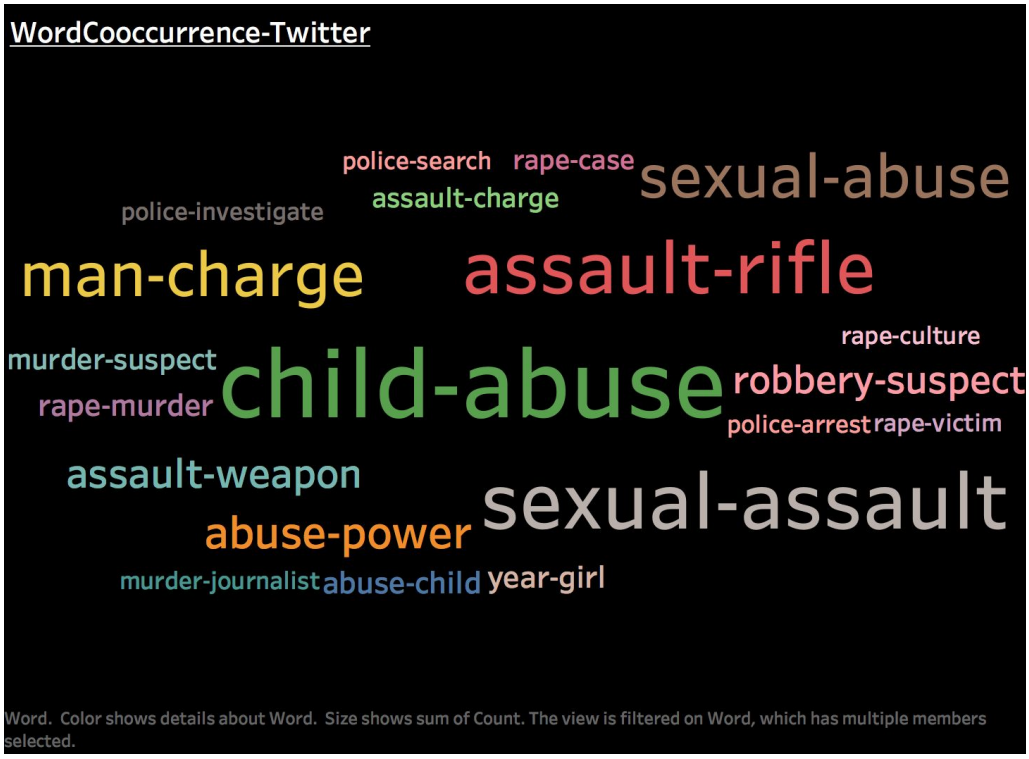
The word cloud for word count of Twitter data:



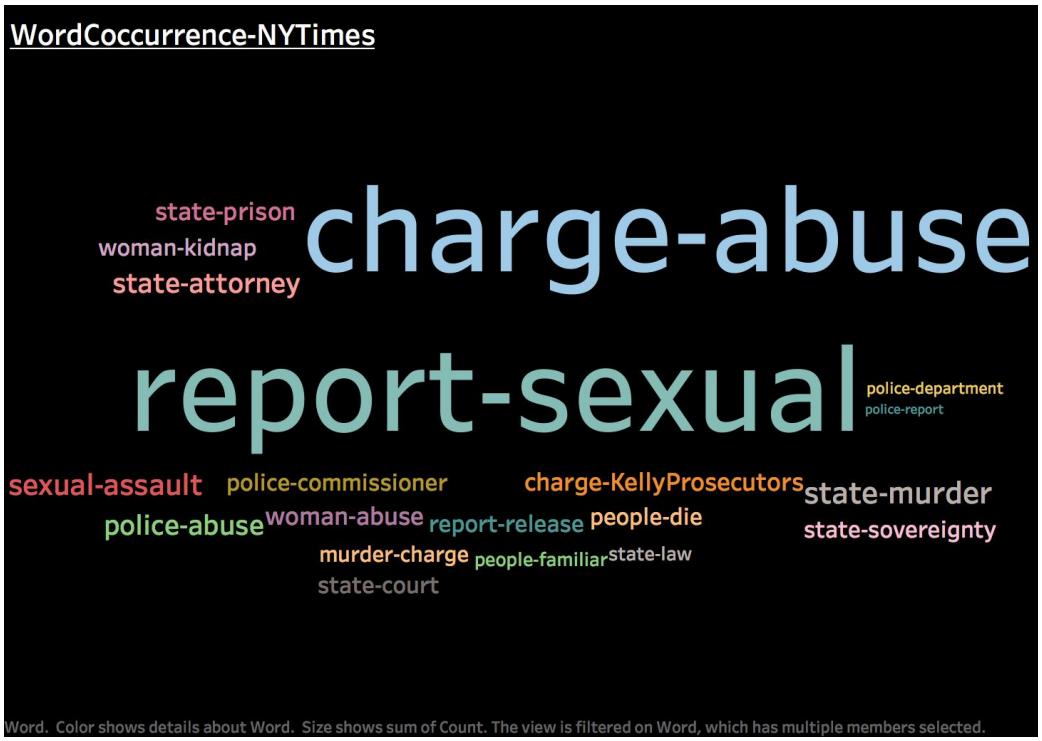
The word cloud for word count of NYTimes data:



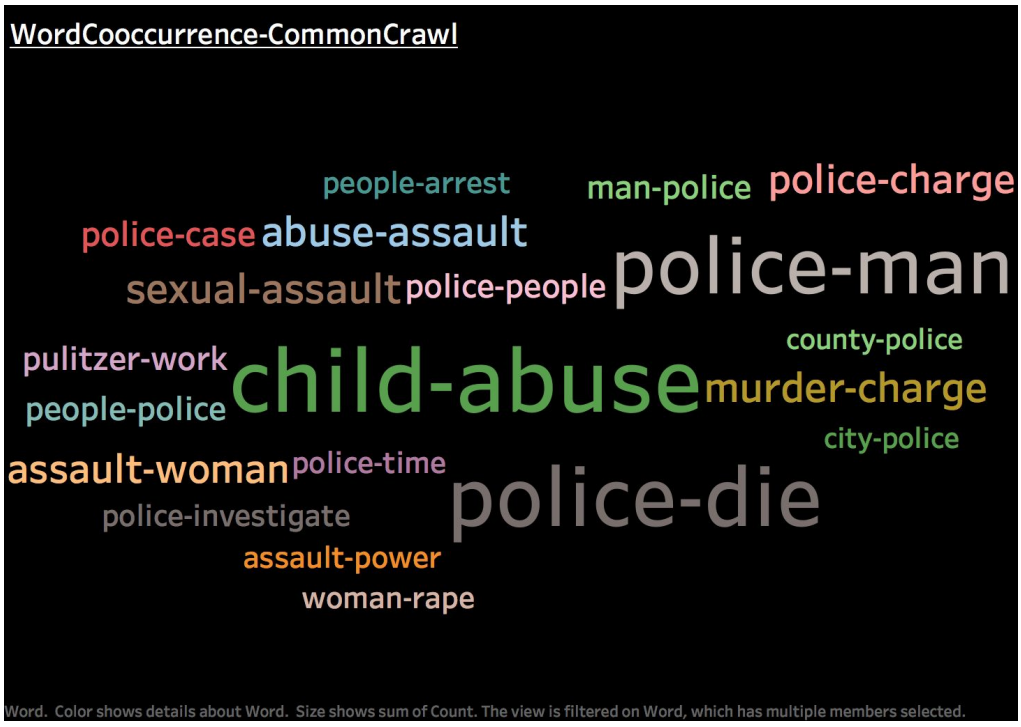
The word cloud for word co-occurrence of Twitter data:



The word cloud for word co-occurrence of NYTimes data:



The word cloud for word co-occurrence of CommonCrawl data



4.1 Analysis and Results

4.1.1 Tabulation for Word Count

The top 10 frequently occurring words from the Twitter data is:

	Word	Count (Frequency)
1	Murder	4009
2	Robbery	3724
3	Assault	3594
4	Rape	3481
5	Abuse	3265
6	Child	1451
7	Sexual	1428
8	Man	1277
9	Police	1233
10	Year	1166

The top 10 frequently occurring words from the NYTimes data is:

	Word	Count (Frequency)
1	Year	5386
2	People	4081
3	Trump	3825
4	State	3640
5	Time	3325
6	Woman	3288
7	Police	3000
8	Report	2670
9	Charge	2554

10	Case	2522

The top 10 frequently occurring words from the CommonCrawl data is:

	Word	Count (Frequency)
1	Year	1924
2	Police	1441
3	Die	1221
4	Time	1156
5	April	1018
6	People	1015
7	Report	930
8	News	879
9	State	789
10	Woman	767

4.1.2 Key words for Word Cooccurrence

For each of the word cooccurrence, the top 10 words which have the highest count are considered.

- The top 10 frequently occurring words in Twitter data are:
  [Murder, Robbery, Assault, Rape, Abuse, Child, Sexual, Man, Police, Year]
- The top 10 frequently occurring words in NYTimes data are:
  [Year, People, Trump, State, Time, Woman, Police, Report, Charge, Case]
- The top 10 frequently occurring words in Common Crawl data are:
  [Year, Police, Die, Time, April, People, Report, News, State, Woman]

5. References

- https://www.tableau.com/support/help
- https://developer.nytimes.com/get-started
- http://commoncrawl.org/the-data/get-started/
- https://developer.twitter.com