# LAB 2: DATA AGGREGATION, BIG DATA ANALYSIS AND VISUALIZATION

Monisha Balaji, Roopa Chandrashekar
University at Buffalo
mbalaji@buffalo.edu, roopacha@buffalo.edu

**Abstract**

This is the age of Big Data. The amount of data in our world has escalated to such high volumes in the last decade. Hence, critical efforts need to be structured for powerful usage of the available data. Big Data Analysis is thus the study of pervasive data to help recognize behavioral patterns. Data is generally categorized based on multiple parameters such as the high speed they are generated with, the huge volume they represent, the variety of information and the veracity. This project involves every individual to work on their data exploration skills. The computed data is exposed to data analytics and data visualization processing. This report records the working and results of the MapReduce word-count and word co-occurrence on the gathered data.

## 1. INTRODUCTION

Apache Hadoop provides the use of a selection of open-source software facilities that enable using many computers to manipulate the massive amounts of data to help with problem-solving. It provides the usage of one of the most popular software framework MapReduce is a computing paradigm for processing data, which aids distributed storage and processing of data. This project involves using Hadoop MapReduce structure to compute the word-count for the data collected and also execute word co-occurrence and retrieve the top 10 most frequently occurring words.

## 2. DATA COLLECTION

As a requisite for this lab, we are required to gather articles for New York Times, accumulate twitter data and perform web crawling to further gather more documents. It is required to collect data for any particular topic from any domain and also for five sub-topics for the chosen. The topic chosen for this project is “CRIME”. The sub-topics are “MURDER”, “RAPE”, “SMUGGLING”, “THEFT”, “TERRORISM”. The procedure followed to gather data for the respective categories are given below:

## 2.1 NYTIMES ARTICLE COLLECTION

New York Times news articles record the day–to-day happenings and make them available to the world. In order to be able to scrap the original articles, we require an NYTimes Developer account to acquire access to the APIs. This process also requires the installation and importing of packages such as ‘nytimesarticle’. This allows you to use the article API thus enabling you to assemble the required list of articles. It is essential to gather 100 articles for each of the headings chosen; a total summing to a minimum of 600 articles. Hitting the API returns JSON results. The next step involves extracting the article URLs from the ‘response’ field. Once the URL list is acquired, it is necessary to remove duplicate URLs from the list before data extraction. Now that we have the final URL list in our hands, we hit each URL from the list and pluck out the text field from the article.

## 2.2 TWITTER DATA COLLECTION

Twitter Data collection can be expanded from the procedure followed in the previous project. Here, we have used the rtweet package to help with Twitter data collection and processing. It allows the use of functions which after the authentication of your twitter key credentials, helps collect streaming tweets using the Twitter Search API. The procedure is as follows:

1. Used 'search_tweets()' function to collect recent tweets. This is done using multiple keywords related to our topic 'CRIME'. One of the parameters we use while searching is lookup_coords()' which makes use of the Google API key; it basically helps getting tweets from the specified location.

2. Everytime, the tweets are collected and stored in dataframes. Now, all the dataframes are combined into one to hold all the data together. For this, we used the 'bind_rows()' function.

3. From the collected data, we select only the necessary fields using the 'select()' function.

4. Now, we write the collected data into a CSV file for storage.

## 2.3 COMMON CRAWL

Common Crawl refers to the organization that provides massive sets of data collected from the internet to the public for free; which is collected by crawling. It involves crawling through the required domains in this case 'CRIME' from www.crimedaily.com , www.crimereports.com etc.. 'WGET' function helps download the 'war.gz' files from the domain. The next step involves extracting the required URLs from the WARC files. The processing was done in python. This can be done by mentioning the topic and sub-topics as keywords. We collected a total of 599 URLs for 'CRIME'.

## 3. DATA PREPROCESSING/ DATA CLEANING

Data pre-processing is used since real-world data is often incomplete, inconsistent, and/or lacking in certain behaviors or trends, and is likely to contain many errors. The data collected from NYTimes, Twitter and Common Crawl need to be stripped clean before running the HDFS structure. The steps involved for data cleaning are:

1. The text extracted from URLs are stored in text files. In case of Twitter, the text field is extracted from the CSV file and stored separately.
2. This text file is subjected to a series of processing. Initially, we tokenize the text i.e. the text is broken down into single words.
3. Now, each word is tested for the following:
    - Check if it alphabetic using isalpha() function. We strip out all numeric terms.
    - Remove punctuations.
    - Replace multiple white-spaces by a single white-space.
    - Remove the word if it belongs to the list of stopwords.
    - Remove the words that have a high word count because of good usage in conversations. Eg: "said", "could", "would" etc.
4. In case of twitter data, it requires some additional data processing like:
    - Remove the links that are a part of the tweet text field starting with http or https.
    - Remove emoticons.
    - Remove screen names if present i.e. the names starting with '@'.
5. In case of common crawl data, we use Beautiful Soup package to help with html parsing.

6. Finally, once the clean text is secured, stemming/lemmatization is applied. This concluding data is stored and used as an input to the MapReduce structure.

## 4. WORD COUNT

Word count is computed using a MapReduce Structure. This is done with the help of the mapper.py and reducer.py scripts given. The final cleaned text is given as input to the mapper. The mapper produces (key, value) pairs as output i.e. it emits <word, 1> list. The reducer gets input from the shuffling phase. It aggregates the list of values for each key and finally produces the list of co-occurring words with sums. From this, we retrieve the top 10 most frequently used words in the document. This is done for the texts from NYTimes, Twitter and Common Crawl.

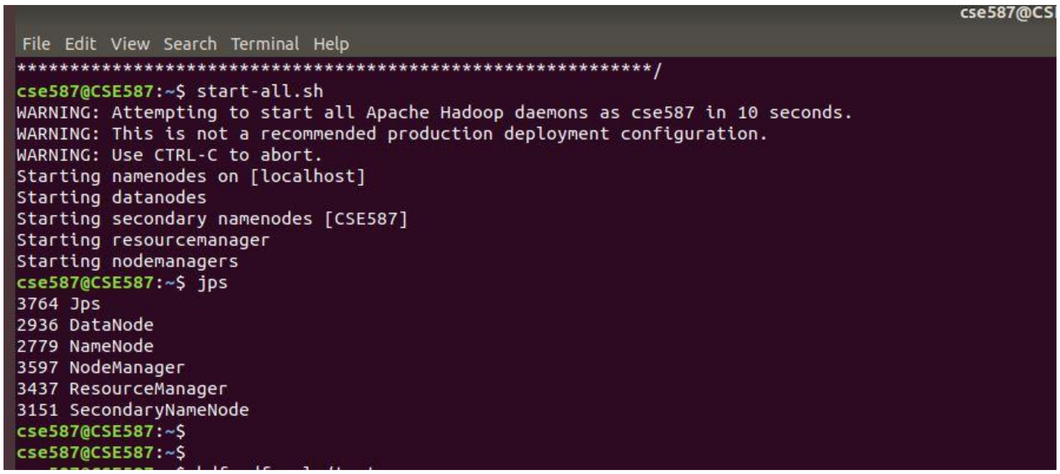


**Fig 1.** Starting namenode and datanode on hdfs

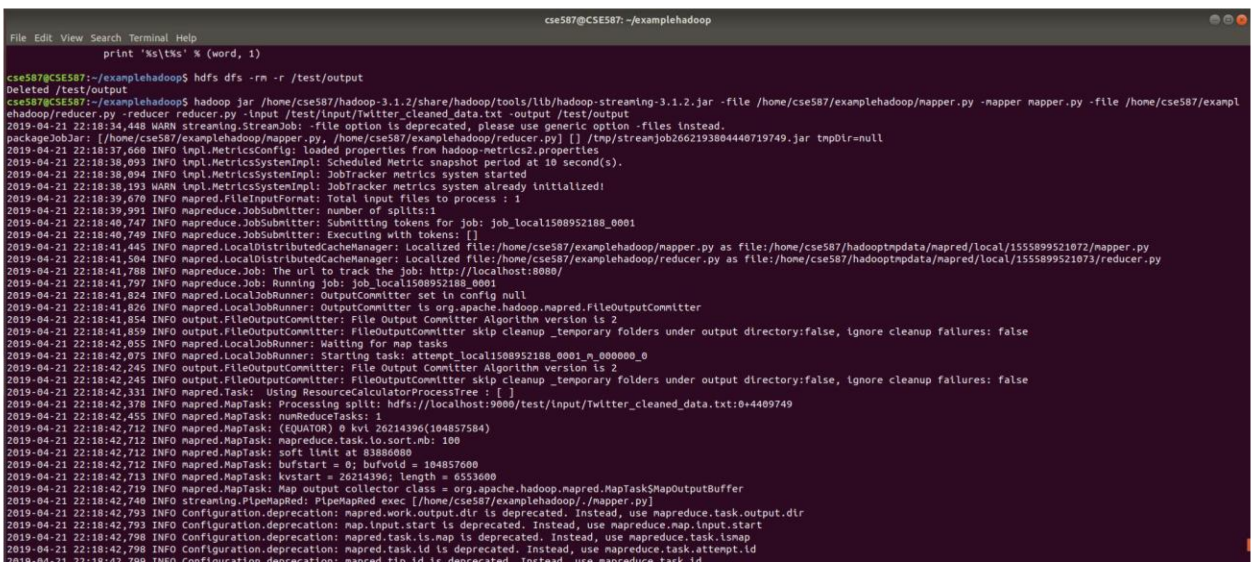


**Fig 2.** Executing word count for Twitter data

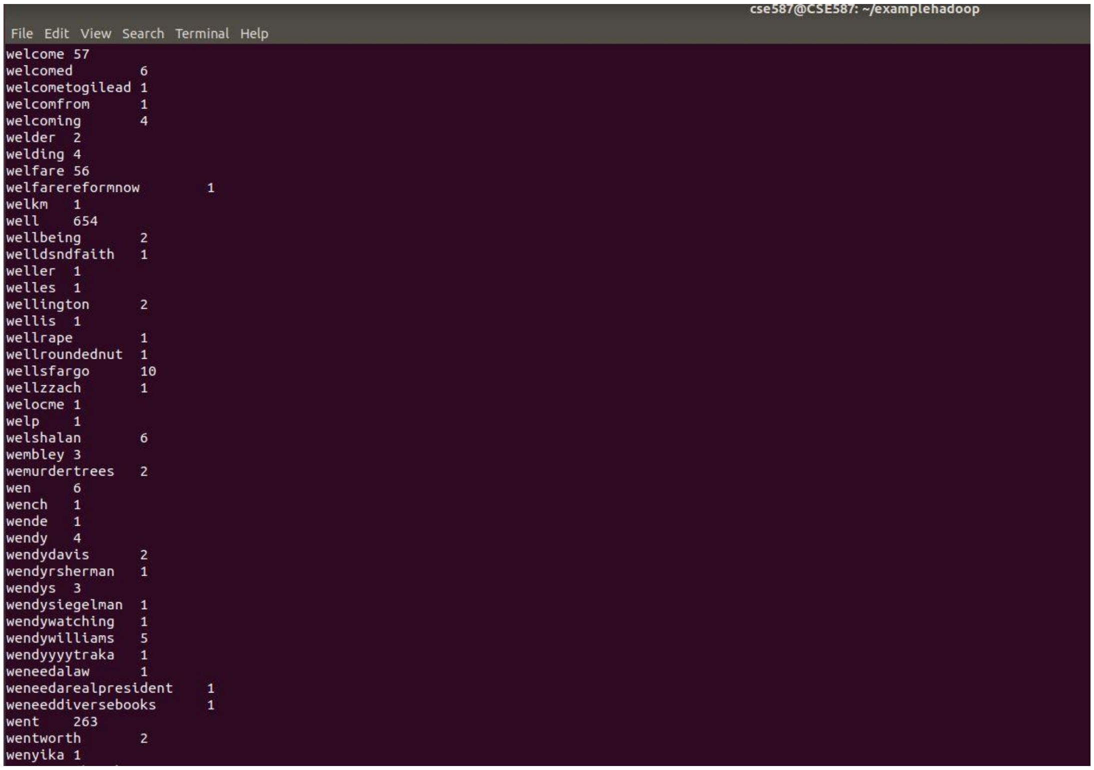


**Fig 3.** Word Count on Twitter Data

## 5. WORD CO-OCCURRENCE

Word co-occurrence involves the phenomenon of computing the frequency of highly co-occurring words in the given document. This is also done using the mapper and reducer scripts which are written in python. The reducer gets input from the shuffling phase. It aggregates the list of values for each key and finally produces the list of co-occurring words with sums. From this, we retrieve the top 10 most frequently appearing co-occurring words. This is done for the text from NYTimes, Twitter and Common Crawl.

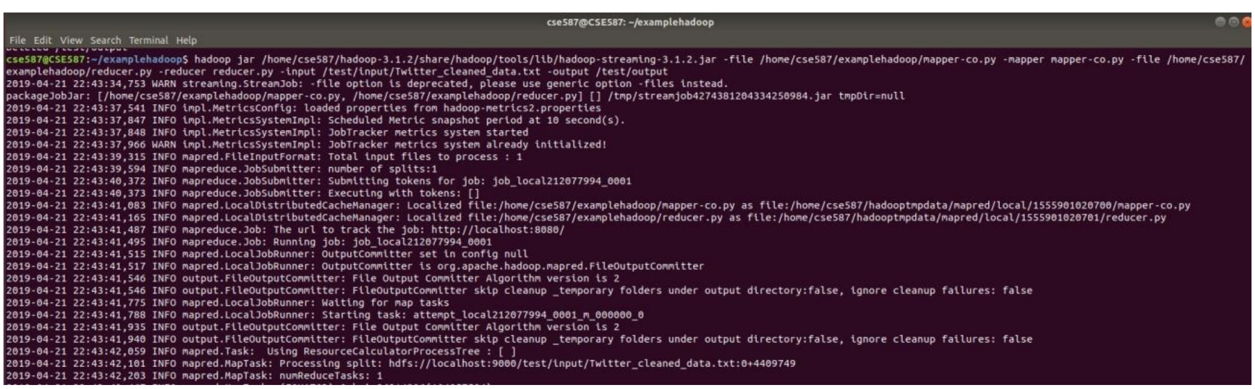


**Fig 4.** Executing Word Co-occurrence on twitter data

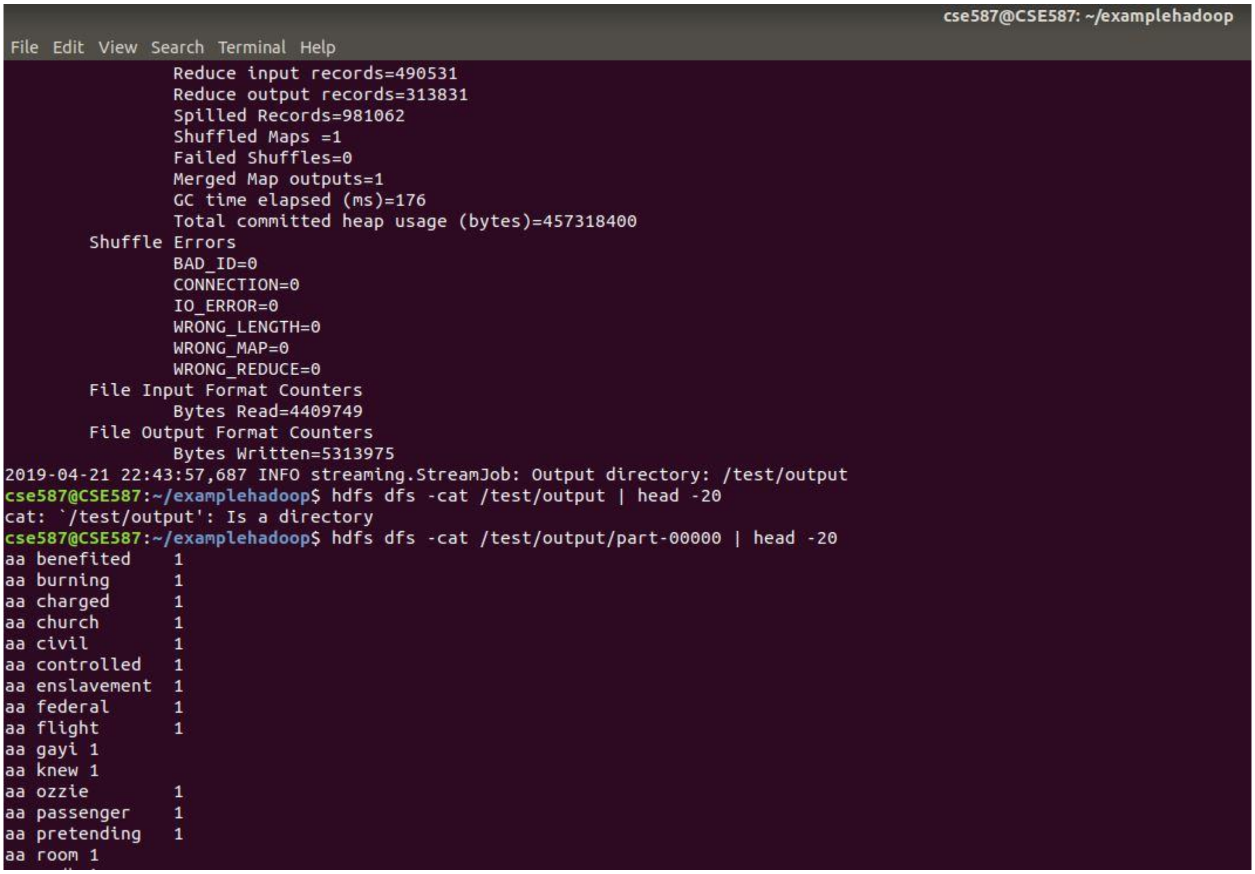


**Fig 5.** Example Word Co-occurrence count

## 7. VISUALIZATION

Visualization is an essential part of this project as we are required to develop word clouds. Word Clouds are visual representation of text data used to portray the highlighted words i.e. the words with the highest frequencies. We use Tableau to perform this. We know that the output from the reducer emits the words list with their frequencies. This is imported into a CSV file and sorted in a non-increasing order based on the sum value. From this, it is easy to extract the top 10 words appearing in Twitter data, NYtimes articles and Common Crawl.

An excel sheet with the top 10 list is imported into Tableau and is used to create the word clouds respectively. Word Clouds are also created for the top 10 list of co-occurring words from the respective sources.

## 8. CONCLUSION

Thus Word-Count and Word Co-occurrence problems have been implemented using MapReduce. This project has helped expand knowledge on data aggregation and visualization. It enabled working with different search APIs for Twitter and NYTimes and also working on the returned results. Thus this lab has aided with working on Big Data and analytical tools.

**Website link:** https://public.tableau.com/profile/monisha5735#!/vizhome/Monisha/Sheet5

## REFERENCES


1. https://pypi.org/project/wget/
2. https://marcobonzanini.com/2015/03/23/mining-twitter-data-with-python-part-4-rugby-and-term-co-occurrences/
3. https://developer.rhino3d.com/guides/rhinopython/python-csv-file/
4. https://www.machinelearningplus.com/nlp/lemmatization-examples-python/
5. https://pythonspot.com/tokenizing-words-and-sentences-with-nltk/