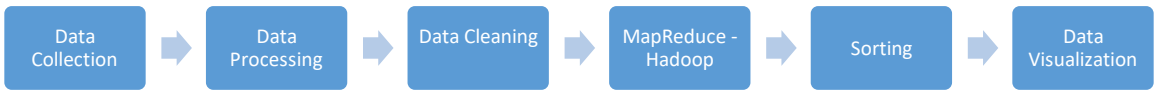
Data Intensive Computing

Lab 2

Ashutosh Ahmad Alexandar Harish Ganesan

This lab is broken down into many phases namely –

- Data Collection – Collect NYT URL and Tweets
- Data Processing – Scrape HTML Body from URL, take only text from tweet information
- Data Cleaning – Take only relevant data from scraped information, split data into several txt files
- Map Reduce – Feed multiple
- Sorting (Post Processing) – Sorting reduced output to get the highest occurances
- Data Visualization – Creation of WordCloud



For this lab, we have considered three topics which are currently trending on the internet –

- **Trump+Russia**
- **NBA**
- **Blockchain**

As we can see from the above block diagram, we start off with the data collection.

We use the TwitterAPI and the NYTimes API to get tweets and article URLs based on keyword search. An example would be “trump+putin”.

Once we have the relevant tweets and URLs, we need to process these URLs using BeautifulSoup in python, to get the HTML body of the URLs we have. Then we must take only the paragraph tags, as these only contain the relevant information that we require. Once we have the paragraph tags, we take all the words in all the paragraphs and store it within a single file. We have one such file for each article.

We also take only the text portion of the tweet data and combine several tweet’s text into a single file, and we have multiple such files.

We can start Hadoop now on the VM, using start-hadoop.sh , which has already been provided.

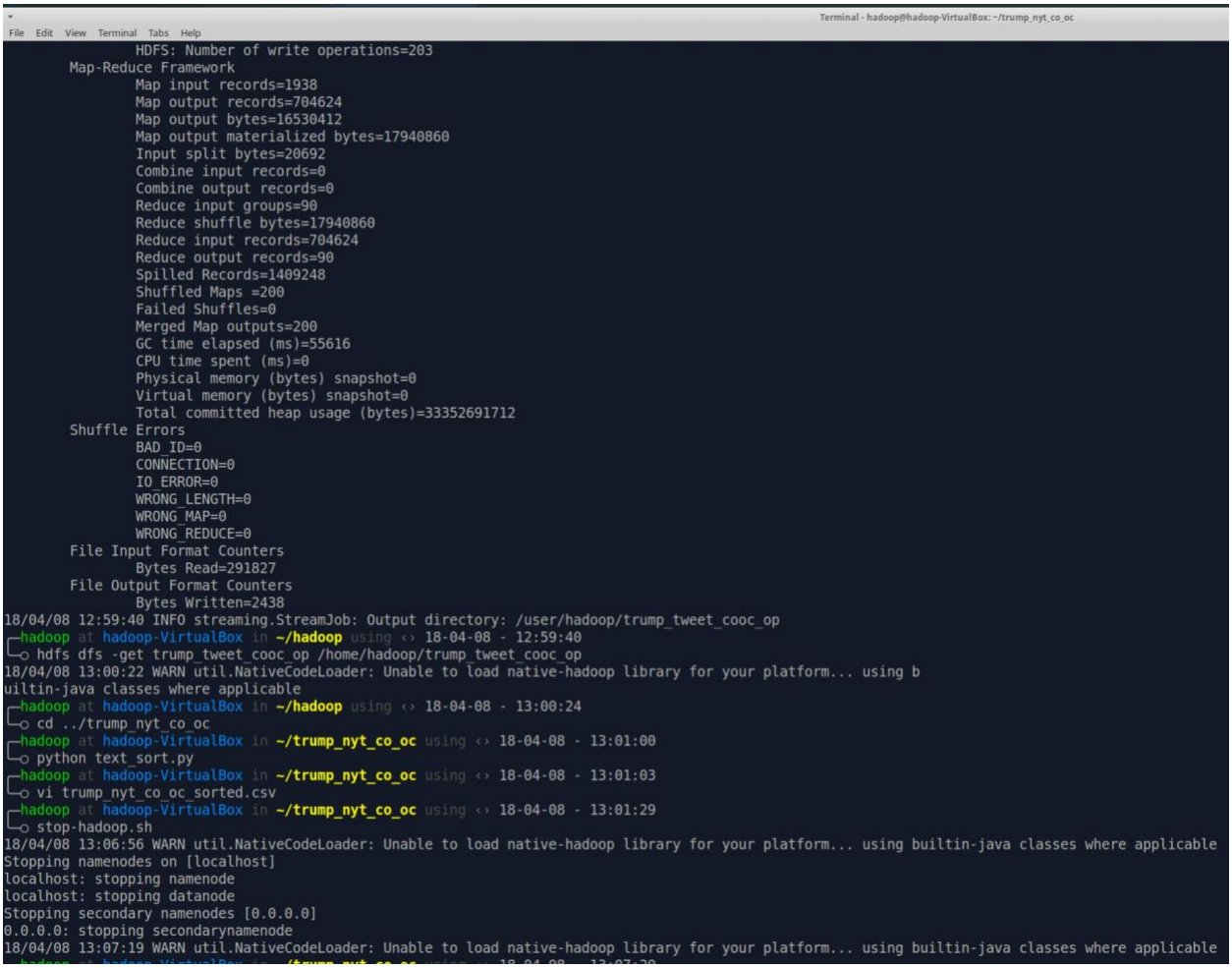
Next, we need to input all these files into Hadoop, using the mapper and reducer code that we have provided. We will Then get an output file in the format ‘part-00000’. This output file contains the reduced output of input we have provided, and this is a set of words and their respective counts.

Once we have the output file, we run the text_sort.py script that we have provided, and this sorts the ‘part-00000’ file into a CSV file which is in descending order with respect to the word count.

As we can see from the screenshot below, an output folder is created in the HDFS. Then we need to copy that to our local file system using “hdfs dfs –get /output /home/Hadoop/op”.

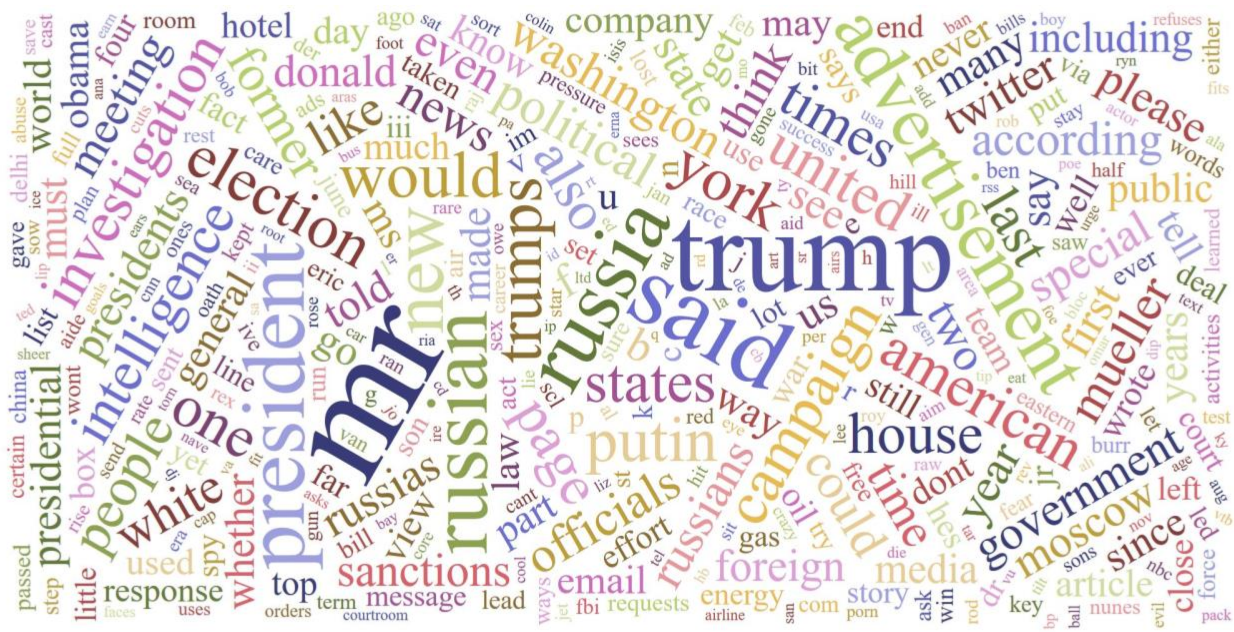
Then we run the text_sort.py script to sort the file and obtain a new CSV.

After this step, we can stop Hadoop, using stop-hadoop.sh that was already provided.



We then consider only the top 30 rows to create the WordCloud.

We use d3.js to create the word-clouds, by inputting these sorted CSV files into the HTML page, we then just take our data and draw a word-cloud with it. An example is shown below.



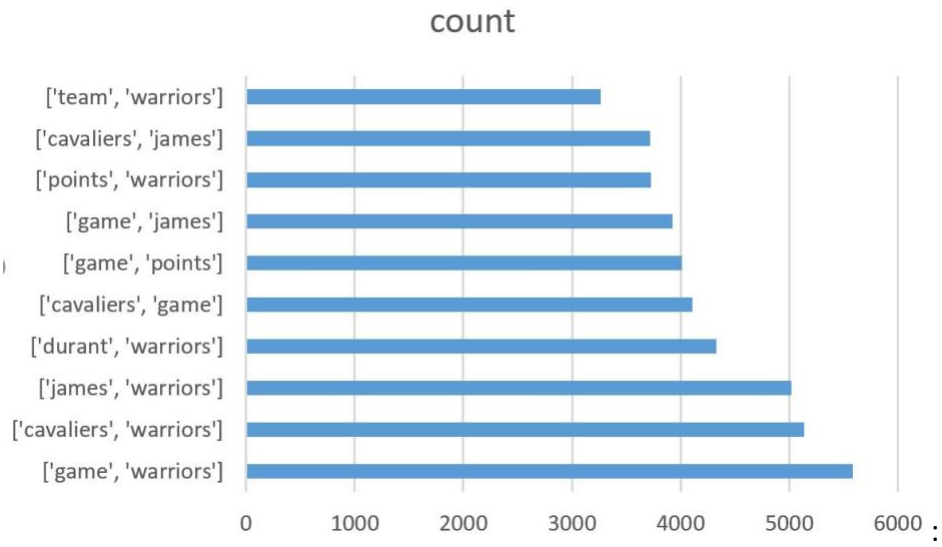
Shown above is one of the word-clouds for the keyword “Trump+Russia” from NY Times.

Now, we must find the co-occurance of the words using the top 10 words we have determined from the last step. For this we need to change the mapper code such that the key is a list with two elements. The two elements will be (word1, word2). Each Pair will have a count associated with itself after reducing.

At the end of the MapReduce process, we are returned with an output file again, but this time, the output will be in the format -> (word1, word2) , count.

We now need to remove duplicates in this CSV file, because (word1, word2) and (word2, word1) are equivalent, but our Map Reduce doesn’t take care of this case.

Upon removing these duplicates, we create a simple bar chart to visualize these co-occurances. An example of these co-occurances are is seen below:



This is the co-occurance count for the pairs, which came from the NBA NY Times data.

Shown below is the final webpage, and depending on which radio button is selected, two wordclouds show up, one for NY Times and one for twitter. This might be a bit slow as the csv file is pretty large, so please be patient while the word-clouds load.

The link to the video created is here on box:

https://buffalo.app.box.com/s/54kva33ijplicafjv23orccifsqot385