LAB 2 - Data Aggregation, Big Data Analysis and Visualization

Sahil Pathak
sahilsuh@buffalo.edu
University at Buffalo

Chinmay Swami
chinmayp@buffalo.edu
University at Buffalo

Introduction

The project addresses key challenges in setting up a big data pipelines and involves:

1. Data aggregation from more than one source using the APIs (Twitter API, NYT API, Common Crawl).

2. Applying classical big data analytic method of MapReduce to the unstructured data collected.

3. Store the data collected on WORM infrastructure Hadoop.

4. Building a visualization data product.

1 Directory Structure

- Report.pdf
- Video.mp4
- Part1 - Prototype data collection (folder)
  - Code (folder) - *all scripts and codes for collecting data*
  - Data (folder)
    - Twitter (folder) - *all tweets*
    - NYT (folder) - *all NYT*
    - Common Crawl (folder) - *all CC articles*
- Part2 - Set up big data infrastructure (folder)
  - mapper.py and reducer.py for both Word Count and Word Co-occurrence *(project directory and dependencies required for running part2)*
  - Other files *(screenshots/results/etc)*
- Part3 - Analyze and Visualize (folder)
  - Twitter (folder)
    - Code (folder) - *all scripts and code files required for processing twitter (mapper.py/reducer.py/etc)*
    - Images (folder) - *all visualizations .jpg/.png AND .js /.twbx files to create Twitter images*
  - NYT (folder)
    - Code (folder) - *all scripts and code files required for processing NYT (mapper.py/reducer.py/etc)*
    - Images (folder) - *all visualizations .jpg/.png AND .js /.twbx files to create NYT images*
  - Common Crawl (folder)
  - Code (folder) - *all scripts and code files required for processing CC (mapper.py/reducer.py/etc)*
  - Images (folder) - *all visualizations .jpg/.png AND .js /.twbx files to create CC images*

2 Overview of the Topic

Main topic of interest is Sports and we have decided to consider 5 sub-topics viz. Football (NFL), Hockey (NHL), Soccer (MLS), Baseball (MLB) and Basketball (NBA). These topics are of particular interest when it comes to the United States. We will delve deeper into their data collection, setting up the big data pipeline and finally visualizing the output by using Tableau.

3 Data Aggregation

3.1 Twitter Data

- Each subtopic was targeted using relevant keywords.
- Specific R query was written in order to collect tweets.
- First, we verified our credentials with the Twitter Search API.
- Using the RTweet Twitter Library, we targeted each subtopic using the keywords viz. (“nba”, “mls”, “mlb”, “nhl”, “nfl”, “basketball”, “hockey”, “soccer”, “baseball”).
- For every subtopic, we gathered round 5000 tweets in total.
- The tweets that we received contained irrelevant information in terms of what we required. They had emojis, special character symbols, numbers, etc.
- We decided to pre-process the raw data into meaningful data.
- We removed the numbers, special symbols, brackets and stop-words using Python.
- Each subtopic has tweets into its separate .txt file.
- For the purpose of Word Count and Word Co-occurrence, we combined all the tweets from every subtopic and aggregated it into a single .txt file.

3.2 NYT Data

- Created an account on NYTimes website to fetch the API key.
- The API works by passing queries inside the link.
- Parameters used in the link are:
    - q: Topic Name
    - page: Some value used for pagination
    - begin_date: To restrict results only after certain date
    - API key
- A JSON file was received from the API which consisted of links to the relevant articles.
- These links were extracted and using the url library of python, the articles from those links were downloaded in html format.
- Using BeautifulSoup, the html file was parsed and the article contents were extracted.

3.3 Common Crawl Data

One of the most challenging part was to collect the Common Crawl Data because of its size and advent to which the information was stored into a single file.

Following methodology was followed in order to collect the same:

- The WET .gz file was downloaded onto the local system from the CC website.
- We decided to go ahead with the WET files because the structure of the wet files given on the CC website was well aligned with what we wanted. It was in the format of <record> <payload> where record had some generic information about the article like url, name, length, version, etc. Whereas, the payload had the entire text corresponding to that record.
- Warc and warc3-wet packages were installed using Python.

- The .gz file was extracted to get another file called “wet.paths”.
- This file “wet.paths” contained a list of .gz files. Each line in wet.paths file had a .gz file link.
- Around 50 .gz files were parsed from wet.paths and 50 .gz files were downloaded.
- All those 50 .gz files had .wet files so we extracted them.
- Finally, we had a folder called “WET”. We stored all 50 .wet files into it.
- These .wet files were passed to the python program.
- The python program made us of warc package which made it easy to parse the huge file. (Each .wet file was of ~410 MBs).
- We decided to match two words in the payload section of the data. For eg. If we are parsing one .wet file and trying to collect articles on football (NFL), then we would have two words as (“nfl”, “football”) and if both the words are present into the payload section, then only the whole text will be saved as an article for football.
- Likewise, we did this for every other subtopic.
- We gathered about 500 articles by parsing about 50 .wet files with every subtopic having equal distribution in terms of the number of articles.

4 Infrastructure

We have everything working on the Virtual Image using the Oracle VM box. The mapper and the reducer code for WordCount was put in the Hadoop/bin directory. Same method was followed for Word Co-occurrence code. The top 10 words from the Word Count are used to calculate Word Co-occurrence.

Commands used to get things running:

1. rm -r /tmp/*

2. rm -r hdfs/datanode

3. rm -r hadooptmpdata

4. hdfs namenode -format

5. stop-dfs.sh

6. start-all.sh

7. hdfs dfs –mkdir /Tweets

   - hdfs dfs -put /home/cse587/total_tweets.txt /Tweets

   - **Word Count for Twitter Data**:

     hadoop jar /home/cse587/hadoop-3.1.2/share/hadoop/tools/lib/Hadoop-streaming- 3.1.2.jar \

     -file /home/cse587/hadoop-3.1.2/bin/mapper_WordCount.py -mapper 'python3 mapper_WordCount.py' \

     -file /home/cse587/Hadoop-3.1.2/bin/reducer_WordCount.py-reducer 'python3 reducer_WordCount.py' \

     -input /Tweets/total_tweets.txt -output /Tweets/output

   - **Word Co-occurrence for Twitter Data**:

     hadoop jar /home/cse587/hadoop-3.1.2/share/hadoop/tools/lib/Hadoop-streaming-3.1.2.jar \

     -file /home/cse587/hadoop-3.1.2/bin/mapper_WordCo.py -mapper 'python3 mapper_WordCo.py' \

     -file /home/cse587/Hadoop-3.1.2/bin/reducer_WordCo.py -reducer 'python3 reducer_WordCo.py' \

-input /TW_Co/* -output /TW_Co/output

8. hdfs dfs –mkdir /NYT

- hdfs dfs –copyFromLocal /home/cse587/NYT/* /NYT
- **Word Count for NYT Data**:

  hadoop jar /home/cse587/hadoop-3.1.2/share/hadoop/tools/lib/Hadoop-streaming-3.1.2.jar \

  -file /home/cse587/hadoop-3.1.2/bin/mapper_WordCount.py -mapper 'python3 mapper_WordCount.py' \

  -file /home/cse587/Hadoop-3.1.2/bin/reducer_WordCount.py -reducer 'python3 reducer_WordCount.py' \

  -input /NYT/* -output /NYT/output

- **Word Co-occurrence for NYT Data**:

  hadoop jar /home/cse587/hadoop-3.1.2/share/hadoop/tools/lib/Hadoop-streaming-3.1.2.jar \

  -file /home/cse587/hadoop-3.1.2/bin/mapper_WordCo.py -mapper 'python3 mapper_WordCo.py' \

  -file /home/cse587/Hadoop-3.1.2/bin/reducer_WordCo.py -reducer 'python3 reducer_WordCo.py' \

  -input /NYT_Co/* -output /NYT_Co/output

9. hdfs dfs –mkdir /CCDATA

- hdfs dfs –copyFromLocal /home/cse587/CCDATA/* /CCDATA
- **Word Count for Common Crawl**:

  hadoop jar /home/cse587/hadoop-3.1.2/share/hadoop/tools/lib/Hadoop-streaming-3.1.2.jar \

  -file /home/cse587/hadoop-3.1.2/bin/mapper_WordCount.py -mapper 'python3 mapper_WordCount.py' \

  -file /home/cse587/Hadoop-3.1.2/bin/reducer_WordCount.py -reducer 'python3 reducer_WordCount.py' \

  -input /CCDATA/* -output /CCDATA/output

- **Word Co-occurrence for Common Crawl:**

  hadoop jar /home/cse587/hadoop-3.1.2/share/hadoop/tools/lib/Hadoop-streaming-3.1.2.jar \

  -file /home/cse587/hadoop-3.1.2/bin/mapper_WordCo.py -mapper 'python3 mapper_WordCo.py' \

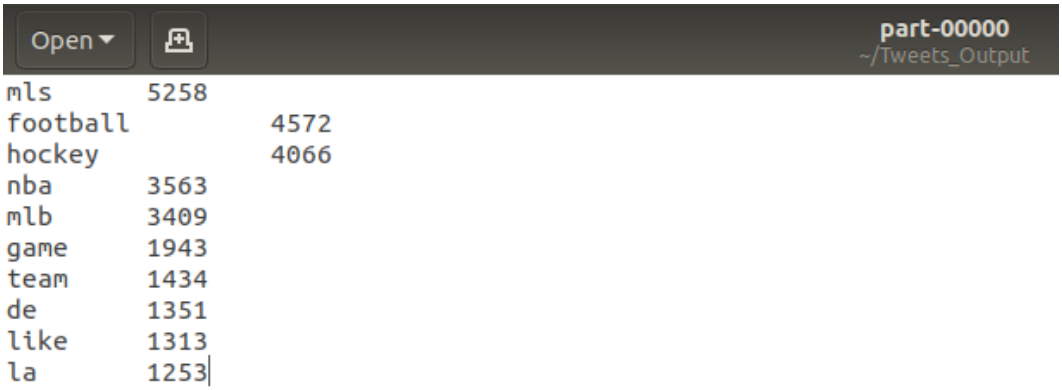
  -file /home/cse587/Hadoop-3.1.2/bin/reducer_WordCo.py -reducer 'python3 reducer_WordCo.py' \

  -input /CC_Co/* -output /CC_Co/output

5 Output of MR Job

5.1 Twitter

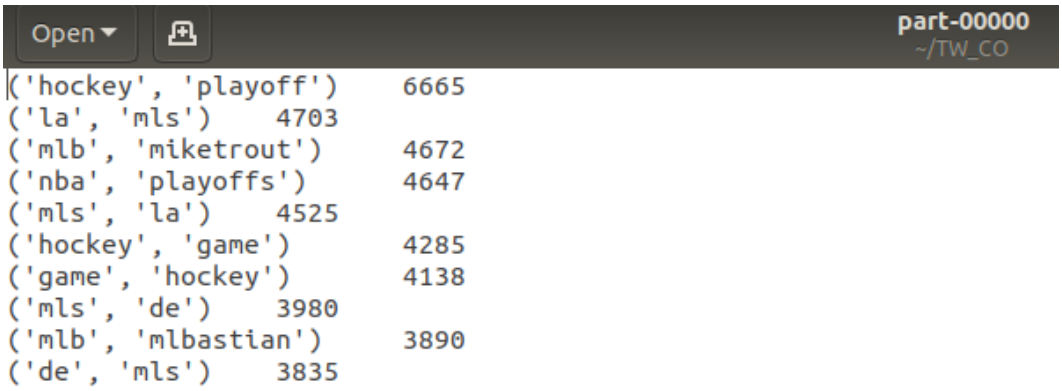
Word Count:



Twitter Data – Top 10 words

From the above figure, we can see that our word count output is well aligned with the topic (Sports). Also, our 5 subtopics are top 5 most occurring words in the whole tweet collection. By looking at the output, we can say that our data pre-processing for tweets was efficiently done.

Word Co-occurrence:

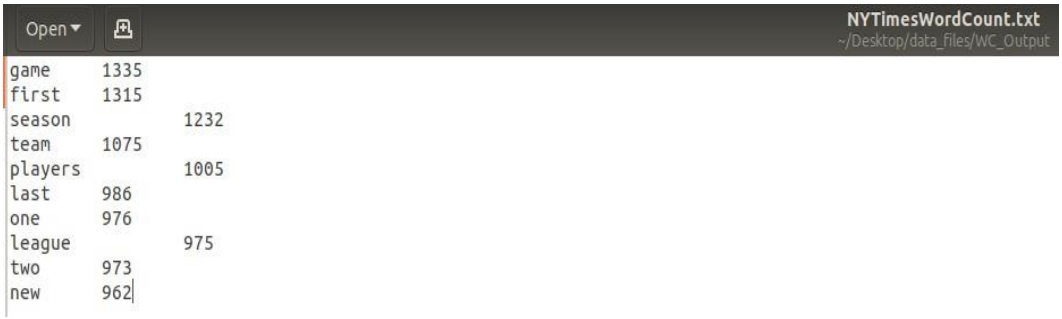


Twitter – Word Co-occurrence

From the above figure, we can see that certain words such as “hockey” are related to the word “playoff” as well as the word “game”. Overall, the output is satisfactory considering the keywords which we used for gathering tweets.

5.2 NYT

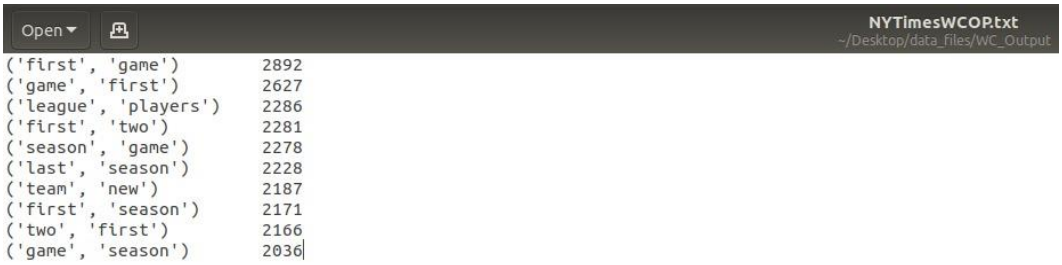
Word Count:



NYT – Top 10 Words

From the above figure, we can see that the top 10 words are related to our main topic "Sports" but they are not very well aligned to our subtopics. One possible reason can be because of the variety of the data which is present in a single article. Though the articles are extracted based on the keywords, we cannot certainly guarantee that the text present in the article is completely matching the subtopic which we have chosen.

Word Co-occurrence:

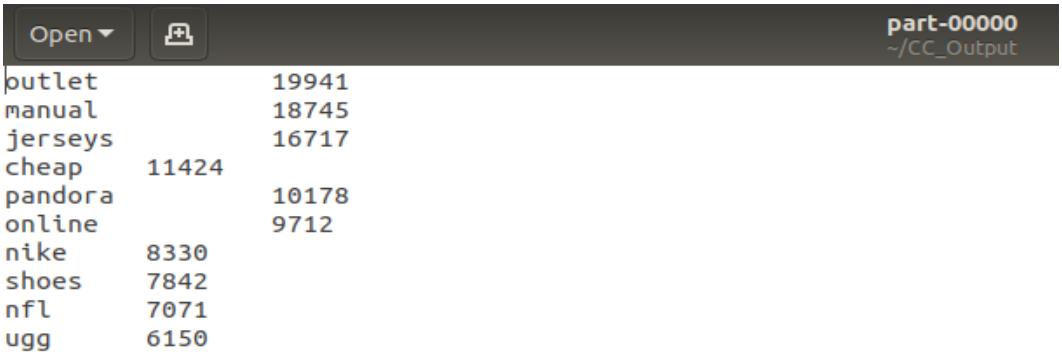


NYT – Word Co-occurrence

The output for the NYT word Co-occurrence is as above.

5.3 Common Crawl

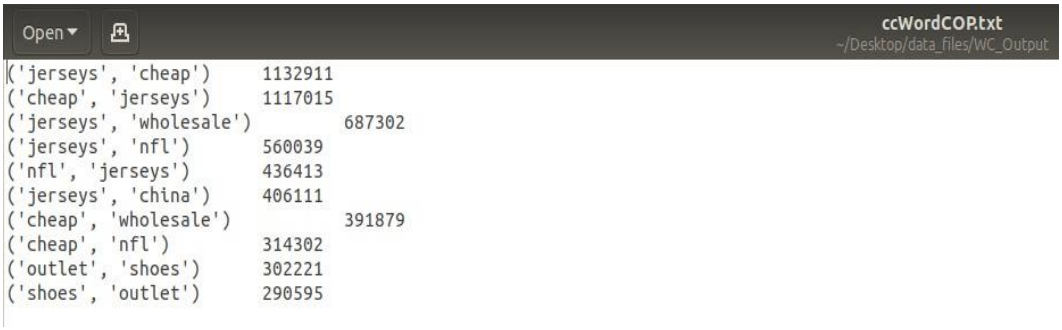
Word Count:



CC Data – Top 10 words

From the above figure, you can see that the word count output is not very well aligned with our topic (Sports). This is due to the fact that the data in Common Crawl is from various sources and many of the articles are vague in terms of the information present in them. Though, we pre-processed the raw data which was gathered from the Common Crawl. Due to the pre-processing we are able to see some words which are, “nfl’, “nike”, “shoes”, “manual”, “jerseys”, “outlet” etc. These words are related to our topic Sports. Also, out of our 5 subtopics, only one subtopic which is “nfl”, showed up in our top 10 list of word count. Since the data in CC is scattered, this uncertainty is expected.

Word Co-occurrence:



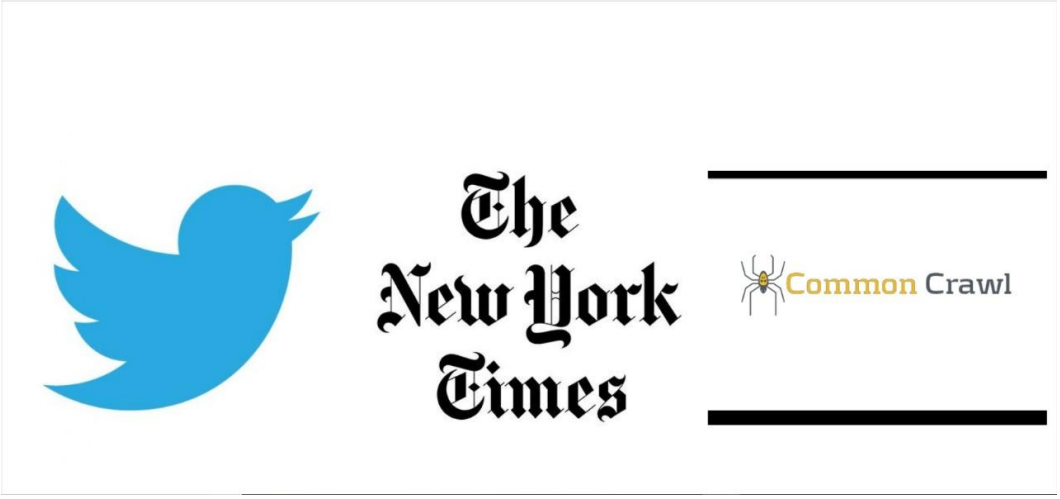
CC – Word Co-occurrence

From the above figure, we can see the vagueness in the data from the CC. Though, all the CC articles are extracted based on the relevant keywords, there’s still plenty of data which is remotely related to our subtopics. This uncertainty is reflected in the word co-occurrence for CC.

6 Visualization

Visualization was carried out by using Tableau. (View Twitter Dashboard, NYTimes Dashboard, Commoncrawl Dashboard on opening the .twb file)

Following are the Word Clouds:



6.1 Twitter

Word Count and Word Co-occurrence:

6.2 NYT

Word Count and Word Co-occurrence:

6.3 Common Crawl

Word Count and Word Co-occurrence: