CSE 587 Data Intensive Computing

LAB2: DATA AGGREGATION, BIG DATA ANALYSIS AND VISUALIZATION

Team Members:

Raman Keswani
Abhishek Krishna

Video URL: https://drive.google.com/open?id=1ww4TULYvsJ7Qv3dOy-MQwX7UiUTeFRFV

Steps:

Data Collection:

Tweets: Execute Tweets_Collection.ipynb(in R) file with desired keyword and proper names for output files. Save the files in TweetData folder.

NYTimes Article: NYT_Article_Search.ipynb(in Python) file with desired keyword and proper names for output files. Save the files in NewsData folder.

Pass these folders to Hadoop File System. (We collected data in Windows so we transferred these files to VM Hadoop via our google drive.)

MapReduce:

Load VMWare with Hadoop ova and execute the following steps:

```
$ start-hadoop.sh
$ hdfs dfs -put /<path_of_data>/ tweet_input
$ hdfs dfs -put /<path_of_data>/ news_input
$ hadoop jar /home/hadoop/hadoop/share/hadoop/tools/lib/hadoop-streaming-
2.6.4.jar -mapper /home/hadoop/Downloads/LAB_2_DOCS/Python_Code/mapper.py -
reducer /home/hadoop/Downloads/LAB_2_DOCS/Python_Code/reducer.py -input
tweet_input -output tweet_output
$ hadoop jar /home/hadoop/hadoop/share/hadoop/tools/lib/hadoop-streaming-
2.6.4.jar -mapper /home/hadoop/Downloads/LAB_2_DOCS/Python_Code/mapper.py -
reducer /home/hadoop/Downloads/LAB_2_DOCS/Python_Code/reducer.py -input
news_input -output news_output
$ hdfs dfs -get tweet_output /<desired_path>/
$ hdfs dfs -get tweet_output /<desired_path>/
```

We then transferred the output files back to Windows.

Json for word cloud:

In windows, convert the output text files to required json form by executing convertToJson.ipynb file.

The json files are saved in the folder where we have created web page files.

For word cloud, there are three different html files one each for one-day data word cloud, one-week word cloud and word co-occurrence word cloud. Execute each for required output.

Hosting Web page:

We had downloaded a dev server to host the pages. Simply paste the folder containing all the files and start the server. Select one file at a time to run(shown in video).

The further details of the project are explained below.

Part 1:
For this part we complete the python code expositions discussed in Chapter 3-5 of our textbook.
We have used Pycharm and Jupyter Python notebook for the exercise. We also found a few bugs in the code that are commented in the notebooks/.py files that we have submitted.

Part 2:

Topic Selection:
For the Part 2 we had to do data analytics by collecting data from Social Media, like Twitter and, News Media, like New York Times.
The first and foremost task was to select a current and trending. So after some brainstorming we came to a conclusion that what could be more trending than Artificial Intelligence or Self Driving Cars. We narrowed our focus to Self Driving Cars and AI.
Finally we had:

1. Topic Chosen: Self Driving car
   Keywords: "self driving car", "driverless car", "autonomous cars"

2. Topic Chosen: Artificial Intelligence
   Keywords: "artificial intelligence", "deep learning", "machine learning"

Using the topics and other keywords relevant to the topic, we gathered data from Twitter and New York Times.

Data Collection & Storage:

Tweets Collection -
For collecting tweets we used what we had learnt in Lab 1. We used R and Twitter API to collect data with various selected keywords.
The output was saved in folder "TwitterData".

Articles Collection -
Just like we collected tweets, we collected articles from NY Times using New York Times' API. We used Python for collecting articles. We used various inbuilt libraries like nytimesarticle, urllib2, bs4 etc for getting articles in desired form.

We saved the articles in separate folder, "NewsData".

At the end of data collection we had two separate folders, TwitterData and NewsData, with relevant files.

VMWare & Hadoop Setup:

Using the information provided in the Lab 2 we first installed Oracle VMWare in our Windows laptops. Then imported the provided Hadoop ova file to the VMWare and completed the Hadoop setup.
After setup, we executed the basic MapReduce examples that were explained in the class to check whether the setup is working fine or not. The examples were executed successfully and we got output for the MapReduce.

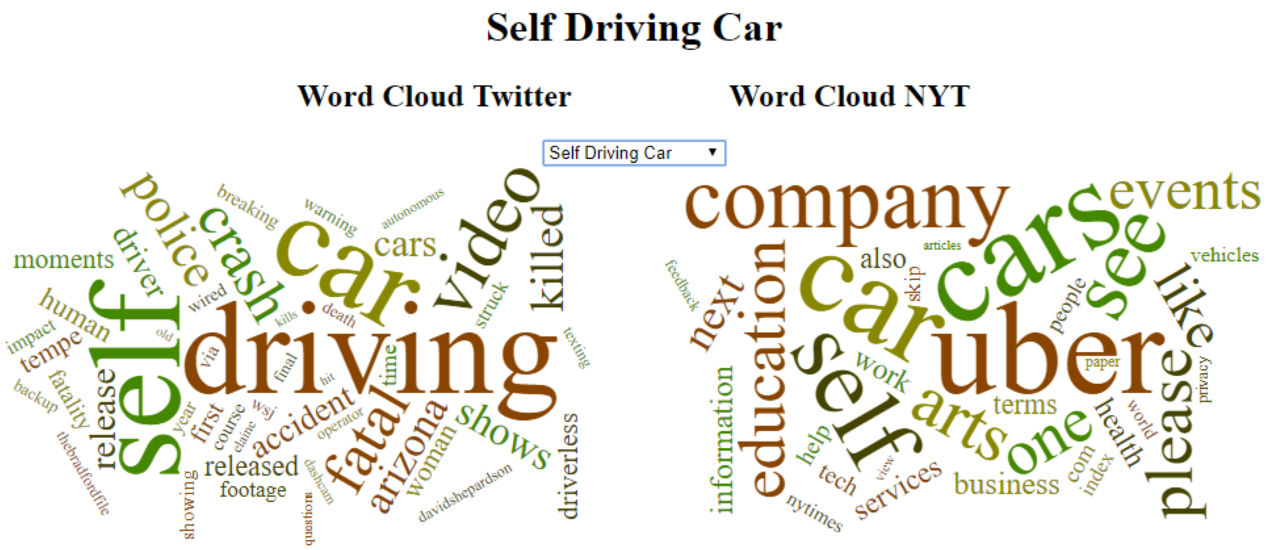
Coding:

The next big task was to code the Mapper and Reducer for our data. Since both, twitter and NYTimes were not in plain simple format, we had to code our Mapper to filter the junk and operate on appropriate data only. Also, we had to remove 'Stopwords' from the data and do analysis on main data.
We created our mapper and reducers using Python as it has various inbuilt libraries that supports data cleaning and formatting. To remove 'Stopwords', we used 'NLTK' library's 'stopwords' and for removing punctuations we used Regular Expressions. Rest of the working of mapper and reducer were same as normal MapReduce function.

We created two versions of MapReduce function, one for simple word-pair count and the other for co-occurrence word-pair count. The details of word co-occurrence MapReduce is given further in the report.

Visualization:

For visualization we have created Word-Cloud. We created json 'key:value' pair from the out of the reducer. These pairs were stored in json files that were used in building final word cloud. We created web page for dynamically creating word-cloud based on selected word. For hosting the webpage and dynamically executing the javascript, we installed a php devserver. The interactive web page shows the words clouds for TweetData and NewsData. We have created different word-clouds for one-day data, one-week data and top-ten co-occurring word data. These are as shown below

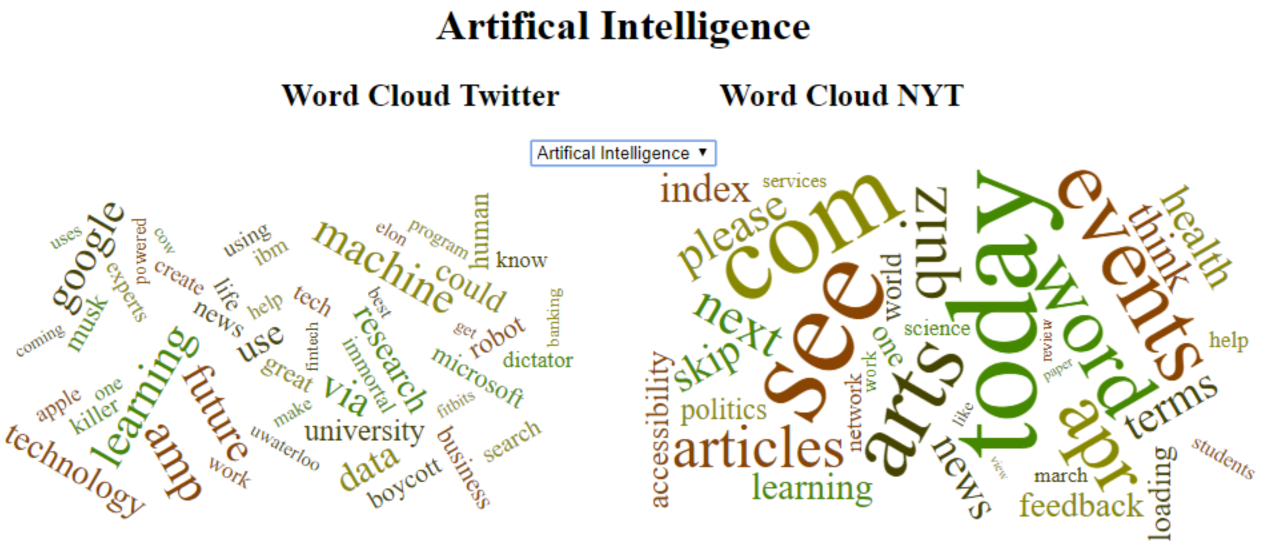


Self Driving Car

Word Cloud Twitter

Word Cloud NYT

Self Driving Car ▾

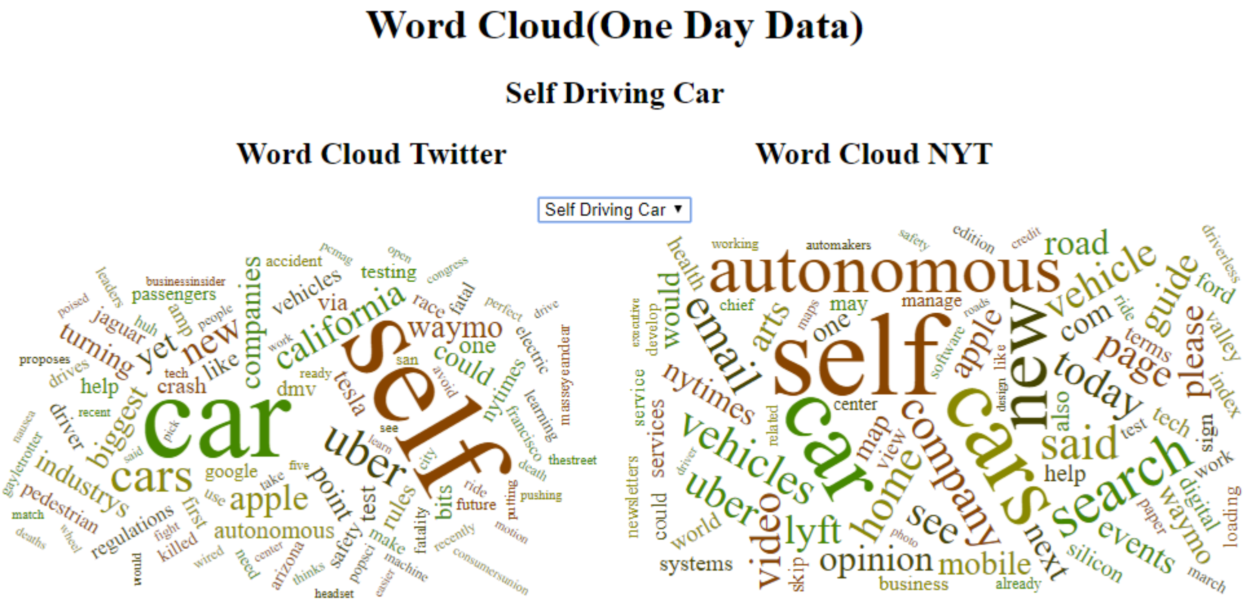


Artifical Intelligence

Word Cloud Twitter

Word Cloud NYT

Artifical Intelligence ▼



Word Cloud(One Day Data)

Self Driving Car

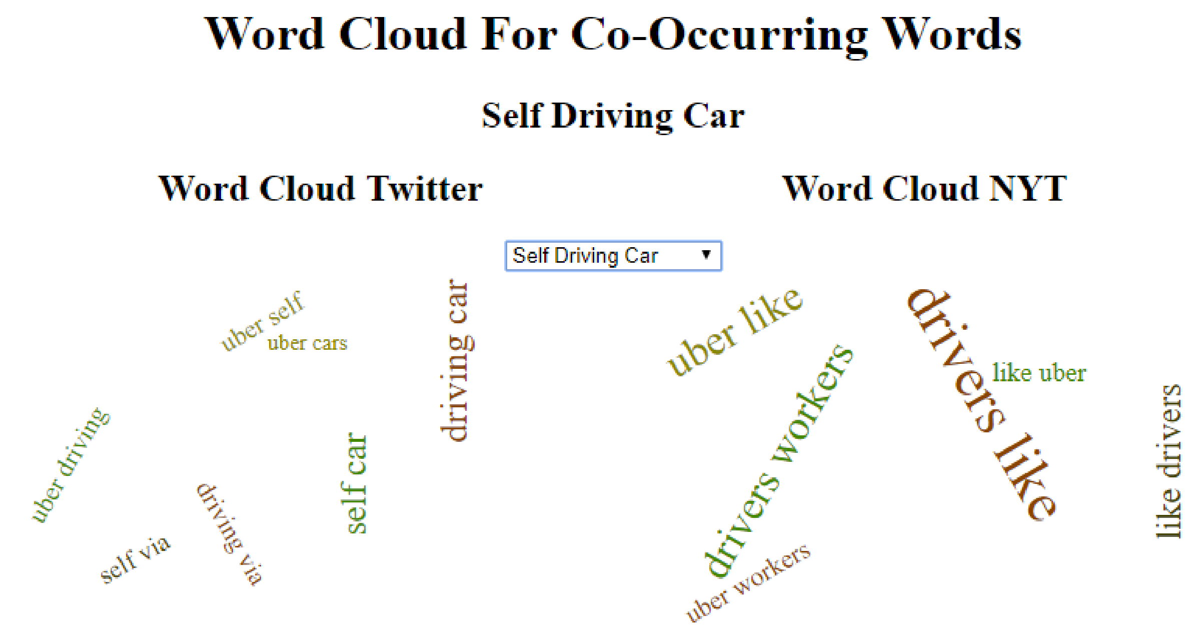
Word Cloud Twitter

Word Cloud NYT

Self Driving Car ▼

Word Co-occurrence:

The next important task was to create MapReduce function for word co-occurrence. We had to analyze each set (Twitter and News) word co-occurrence for only the top ten words. The context for co-occurrence is the “tweet” in the case of TwitterData, and the paragraph of the news article in the NewsData.
For this requirement we tweaked our original mapper and reducer to get word co-occurrence pairs. The filtering and formatting was same as original functions. Only we created and extra loop for emitting the current word with all the words in its paragraph.
From this data also we created word cloud for the top-ten co-occuring words.



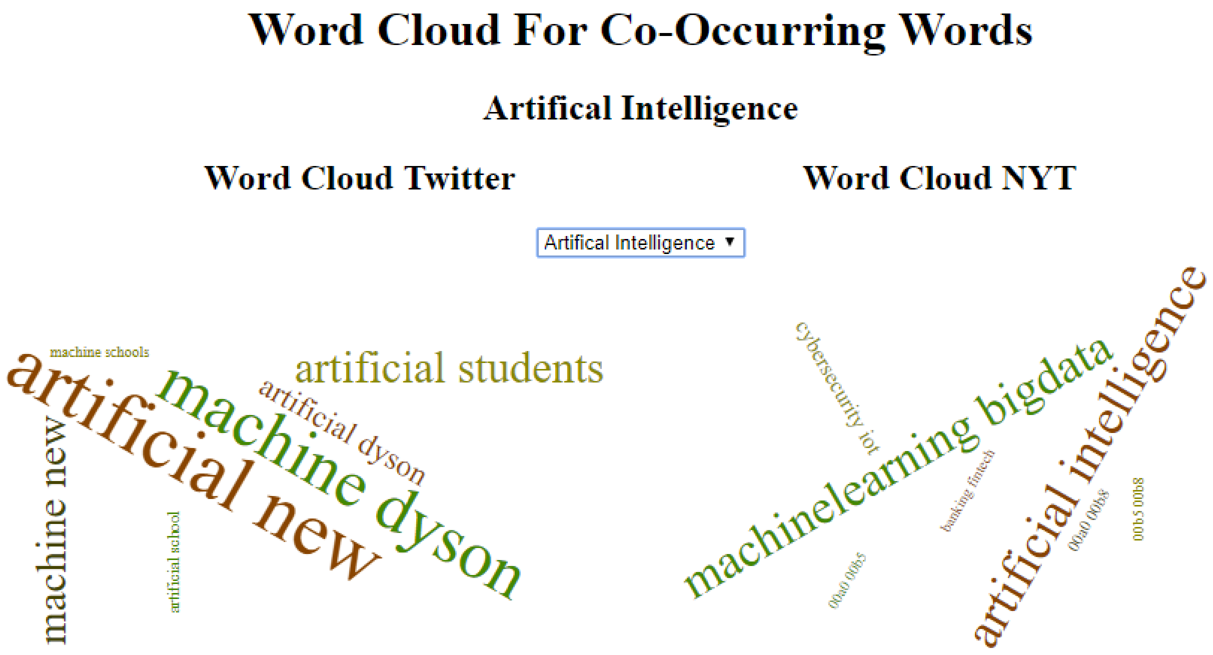
Word Cloud For Co-Occurring Words

Self Driving Car

Word Cloud Twitter

Word Cloud NYT

Self Driving Car ▾



Word Cloud For Co-Occurring Words

Artifical Intelligence

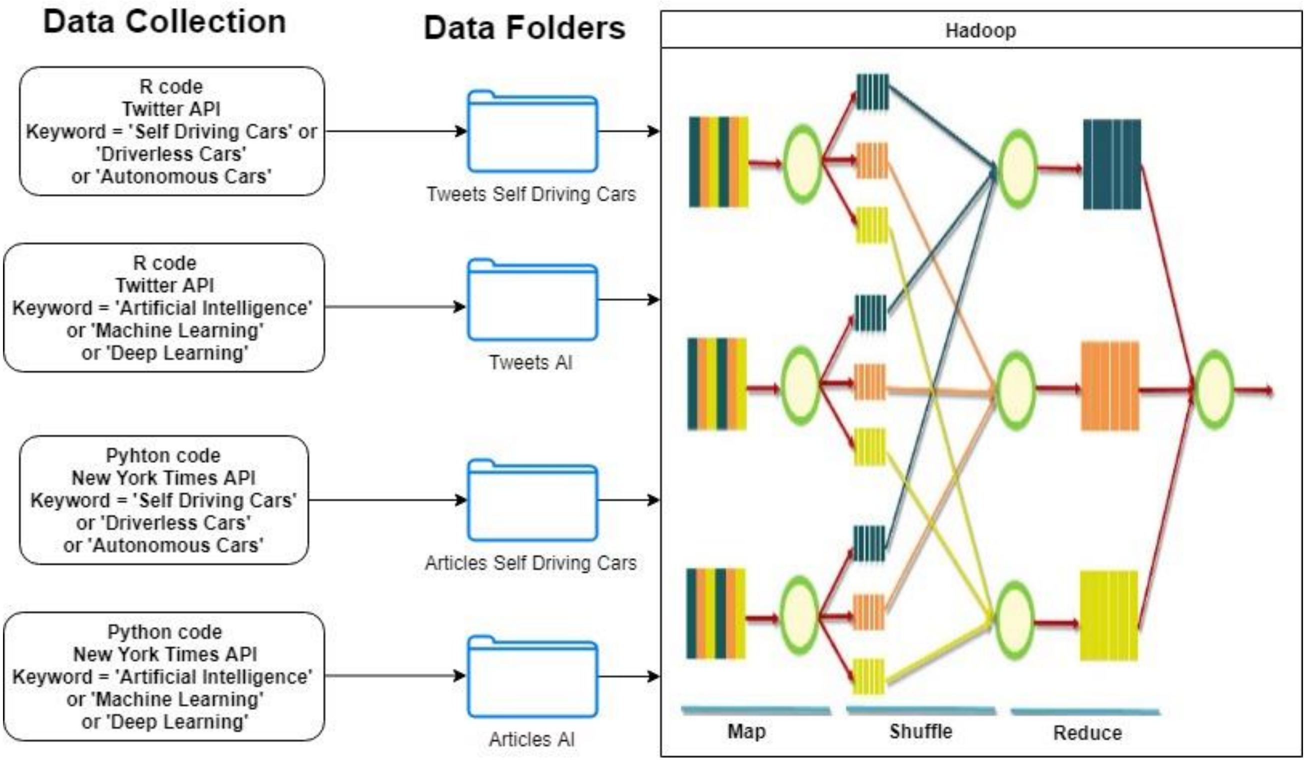
Word Cloud Twitter

Word Cloud NYT

Artifical Intelligence ▾

Block Diagram:

The block diagram representing the overall flow of the process is shown below:



References:

https://github.com/wvengen/d3-wordcloud
http://www.michael-noll.com/tutorials/writing-an-hadoop-mapreduce-program-in-python/
http://codingjunkie.net/cooccurrence/
https://github.com/monisjaved/Data-Processing-With-Hadoop