UB CSE 487 SPRING 2019

MapReduce Project

Michael Brown and Birender Singh

**MapReduce and Hadoop Infrastructure**

We used Amazon Web Services ElasticMapReduce and S3 for the cluster infrastructure. This basically uses the same approach Bina did in class, but instead of creating a jar file, we use the Streaming Program option and provide the python files for map and reduce.

**Topic**: transportation options

**Keywords** for **Twitter** and **New York Times:**

* Bike, bus, car, pedestrian, light rail, plane, train, subway, and walk

*\*We also then* ***grouped*** *some of these together, such as light rail and subway as well as pedestrian and walk*

**Keywords** for **Common Crawl:**

When it came to common crawl data, our approach was to search using domains that were roughly equivalent to some of the keywords. For example, jalopnik.com was the equivalent for the keyword car.

* *['http://trn.trains.com/','https://www.bicycling.com/','https://www.citylab.com/','https://www.amny.com/transit','https://jalopnik.com/c/news']*
* # Query on these domains, roughly equivalent to keywords *['train','bike','bus','lightrail','car']*

**Process**

1. Data Collection
   1. Twitter
      1. Modules: python-twitter, pandas, time, datetime, os.path, keyring
      2. Collect tweets based on keyword API query with a configured date range
         1. data/tw/fulltext
      3. Group any keywords together as necessary
         1. data/tw/grouped
      4. Save as CSV files and upload to S3 bucket
   2. NYT
      1. Modules: requests, pandas, time, bs4, os.path, keyring
      2. Collect articles based on keyword API query with a configured date range
         1. data/urls/ungrouped
      3. Group any keywords together as necessary
         1. data/urls/
      4. Web scrape paragraphs from NYT articles using BeautifulSoup
         1. data/articles/
      5. Save as CSV files and upload to S3 bucket
   3. Common Crawl
      1. Modules: requests, json, gzip, io, bs4, pandas
      2. Collect commoncrawl filenames based on domain search of their index, returns record list
         1. data/cc*/{domain}*Filenames.csv
      3. Collect the web pages from commoncrawl based on the record list
      4. Web scrape paragraphs from each record response using BeautifulSoup
         1. data/cc*/{domain}*.csv
      5. Save as CSV files and upload to S3 bucket
2. MapReduce
   1. Used AWS ElasticMapReduce
   2. Refer to video for more detail

**Visualizations**

1. WordCount
   1. Twitter
      1. <https://public.tableau.com/profile/birender.singh#!/vizhome/shared/9F7M2PM4M>
   2. NYT
      1. <https://public.tableau.com/views/Wordcount-Nytimes/Dashboard1?:embed=y&:display_count=yes&publish=yes>
   3. CommonCrawl
      1. <https://public.tableau.com/views/cc_wc/commoncrawl_wc?:embed=y&:display_count=yes&publish=yes>
2. Word Co-Occurance
   1. Twitter
      1. <https://public.tableau.com/profile/birender.singh#!/vizhome/shared/9F7M2PM4M>
   2. NYT
      1. <https://public.tableau.com/views/nytime_cooc/nyt_cooc?:embed=y&:display_count=yes&publish=yes>
   3. CommonCrawl
      1. <https://public.tableau.com/views/cc_cooc_15559011827380/cc_cooc?:embed=y&:display_count=yes&publish=yes>