CSE 587: Data Intensive Computing

Lab 2 - Report

Data Aggregation, Big Data Analysis And Visualization

Daniel Amirtharaj

Navaneethakrishnan Ramanathan

Objective

The objective of this project is to aggregate data from multiple sources – Twitter, New York Times and the Common Crawl, apply a big data analytic method such as MapReduce to analyse unstructured text data obtained from these sources, and build a data visualization product.

Introduction

Topic selected

The topic chosen for this project was – **Music**. Music is a subject that is talked about widely in all kinds of media, and was a good candidate for "Big Data".

In order to collect as much data related to music as possible, several keywords/subtopics were used along with the word music.

The following subtopics, which are different genres of music, were used to aggregate data from all 3 data sources.

Subtopics **- Jazz, Pop, Classical, Rock, Rap and Country**.

Due to the limited news content related to music in the New York Times, additional genres, **Blues and Folk** were added, specifically to get data from this source.

Directory Structure Used

The root folder has the following folders:

1. **Tweets** – *All data and code used to collect, analyse and visualise Twitter data*.
    a) **Data** – *All data and code used to collect the data*.
        1. **Earlier Collection** – *Distinct raw tweets with truncated text*.
        2. **Final Collection** – *Distinct raw tweets with extended text*.
        3. **Pre-Processed Tweets** – *Text files after stemming, lemmatization and removing stop words.*
        4. **Tweet Collector.**ipynb – *Code to collect and pre-process tweets.*
    b) **WCooccurence_MR** – *Mapper and Reducer used for co-occurrence.*
    c) **WCooccurence_Output** – *Output after running co-occurence MR.*
    d) **WCount_MR** – *Mapper and Reducer used for co-occurrence.*
    e) **WCount_Output_small** – *Output after running count MR on small dataset.*
    f) **WCount_Output_big** – *Output after running count MR on big dataset.*

g) **Hadoop Scripts –** *Hadoop scripts used to make directories, load into and out of HDFS and run the MR program.*
h) **Twitter.**twbx – Tableau file used for visualization.

2. **NYT** – *All data and code used to collect, analyse and visualise NYT data*.
   a) **Data** – *All data and code used to collect the data*.
      1. **Extracted NYT Articles** – *Distinct raw NYT articles*.
      2. **Pre-Processed NYT Articles** – *Text files after stemming, lemmatization and removing stop words.*
      3. **nyt_client.**ipynb – *Code to collect and pre-process NYT articles.*
   b) **WCooccurence_MR** – *Mapper and Reducer used for co-occurrence.*
   c) **WCooccurence_Output** – *Output after running co-occurence MR.*
   d) **WCount_MR** – *Mapper and Reducer used for co-occurrence.*
   e) **WCount_Output_small** – *Output after running count MR on small dataset.*
   f) **WCount_Output_big** – *Output after running count MR on big dataset.*
   g) **Hadoop Scripts –** *Hadoop scripts used to make directories, load into and out of HDFS and run the MR program.*
   h) **Twitter.**twbx – Tableau file used for visualization.

3. **Common_Crawl** – *All data and code used to collect, analyse and visualise Common Crawl data*.
   a) **Data** – *All data and code used to collect the data*.
      1. **Extracted Articles** – *Distinct raw Common Crawl articles*.
      2. **Pre-Processed Articles** – *Text files after stemming, lemmatization and removing stop words.*
      3. **Common Crawl Collector.**ipynb – *Code to collect and pre-process Common Crawl articles.*
   b) **WCooccurence_MR** – *Mapper and Reducer used for co-occurrence.*
   c) **WCooccurence_Output** – *Output after running co-occurence MR.*
   d) **WCount_MR** – *Mapper and Reducer used for co-occurrence.*
   e) **WCount_Output_small** – *Output after running count MR on small dataset.*
   f) **WCount_Output_big** – *Output after running count MR on big dataset.*
   g) **Hadoop Scripts –** *Hadoop scripts used to make directories, load into and out of HDFS and run the MR program.*
   h) **Twitter.**twbx – Tableau file used for visualization.

4. **demo** – *All data and code used to collect, analyse and visualise sample data*.
   a) **data** – *sample data pg345.txt*.
   b) **mapper**.py – *mapper program for word count.*
   c) **reducer**.py - *reducer program for word count.*
   d) **output** – *output of the MR word count program.*

Data Collection

Data was collected from three sources, Twitter, New York Times and the Common Crawl dataset.

1. **Twitter**

The twitter API was used along with twython, a helper library specifically to aid tweet collection for Python. Twitter was queried using the keywords - jazz, “pop music”, “classical music”, “rock music”, rap and “country music”. The word music was added while querying subtopics like pop, country etc. since irrelevant tweets which had the words that were not related to music were returned.

Data collection was done in 2 stages. Some tweets collected in the first stage had truncated text. In order to get the extended text, twitter was queried again using the ids of the tweets already collected.

2. **NYT**

The New York Times REST API provided by NYT themselves was used to retrieve news articles about music. The API was queried via HTTP requests with the fields - subsection_name and subject. The request was in the following format, “subsection_name : music OR subject : music OR subject.contains : keyword”, where keyword was a subtopic name. The response json contained only the urls of the articles, and thus the webpage from the url was retrieved and parsed using beautiful soup to get the text data.

3. **Common Crawl**

In order to collect data from domains that were related to music, the common crawl Index was used to query useful domains and identify WARC files of interest. The WARC files were requested with appropriate HTTP headers (with range as given by offset and length mentioned in the Index file) in order to retrieve the crawled webpages. Popular music websites such as rollingstone.com, jazztimes.com, loudwire.com etc. The crawled webpages were then parsed using beautiful soup to get the text data.

The text data obtained from the above data sources above were stored in .csv files in their respective folders.

Pre-Processing

The .csv files were read and pre-processed in the following manner before running the MR code.

- Irrelevant text such as URLs, @user (tweets) etc. were identified using regular expressions and removed.
- The text was tokenized using the tokenizer available in the NLTK library.
- Stop words were removed using stop words identified from NLTK library as well as irrelevant words identified manually.
- The NLTK lemmatizer was used to lemmatize the words.

- Non alphabet characters such as numbers, special characters and punctuation were removed.
- All uppercase alphabets were lowercased.

Map Reduce – Word Counts

Word counts were obtained and analysed first by feeding a small portion of the data collected (using just the main topic – Music on all 3 data sources) and then by feeding the entire data collected (using the main topic as well as sub-topics) to the MR program.

The following steps were followed to get the word counts.

- The outputs of the pre-processing stage for all 3 data sources were loaded into the Hadoop file system using the –put option.
- The word count reducer.py provided in the demo were modified to produce a sorted (descending) word count output.
- The mapper and reducer files were specified along with the Hadoop input and output directories while submitting the MR job through the Hadoop streaming .jar.
- The outputs from the MR program were then loaded to the local file system using the –get option.

Map Reduce – Word Co-occurrence

The top 10 words obtained by running the word count MR program on the smaller dataset (using just the main topic - Music), for all 3 data sources were used to obtain word co-occurrence pairs for the top words along with other words that may occur in the same tweet or paragraph.

The following steps were followed to get the word counts.

- The top 10 words for each data source was included in the word co-occurrence mapper.py file for each respective data source.
- Words in the list were used to find other words that occurred in the same tweet or paragraph and this word pair was emitted (with each word separated by "~") from the mapper.py program.
- The reducer was modified to ensure that duplicate counts were not registered, and the received word pairs were emitted with the counts computed.
- The mapper and reducer files were specified along with the Hadoop input and output directories while submitting the MR job through the Hadoop streaming .jar.

- The outputs from the MR program were then loaded to the local file system using the –get option.

Visualisation

Tableau was used to visualize the word counts using a bar graph, word cloud and the co-occurrence matrix.

- The outputs from the reducer i.e. the <word, count> and the <word~word, count> were used as the input for visualization. The input (data source) for the workbook was provided using those results.
- Once the input (data source) was included, columns were created. These columns are dragged and dropped to the row/columns to enable visualization.
- The bar plot is used to visualize the top 10 words based on the word count. We have added a filter in the column *word* based on the SUM(*count*) and returned the top N words. This parameter can be tuned to show the top N words.
- The word cloud was created by using the column *word* and representing the *count* in the size of the text. A parameter to determine the number of words in the word cloud was created and this can also be tuned in the user interface.
- Another input (data source) was included to form the co-occurrence matrix in the same workbook and used in the worksheet to create the matrix. The top 10 words were kept as the columns and the co-occurring words for each of the top 10 words were included as the row. This list of co-occurring words can also be tuned in the user – interface.
- The word count and word cloud were created for both the small data as well as the big data, while the word co-occurrence matrix was created for the smaller data.
- All the worksheets were saved as output images and compiled into a story.
- The same process was followed for the all the reducer outputs from Twitter, New York Times, and Common Crawl.

Results

The following pages show the visualizations obtained after the word count and word co-occurrence programs were run for the 3 data sources.

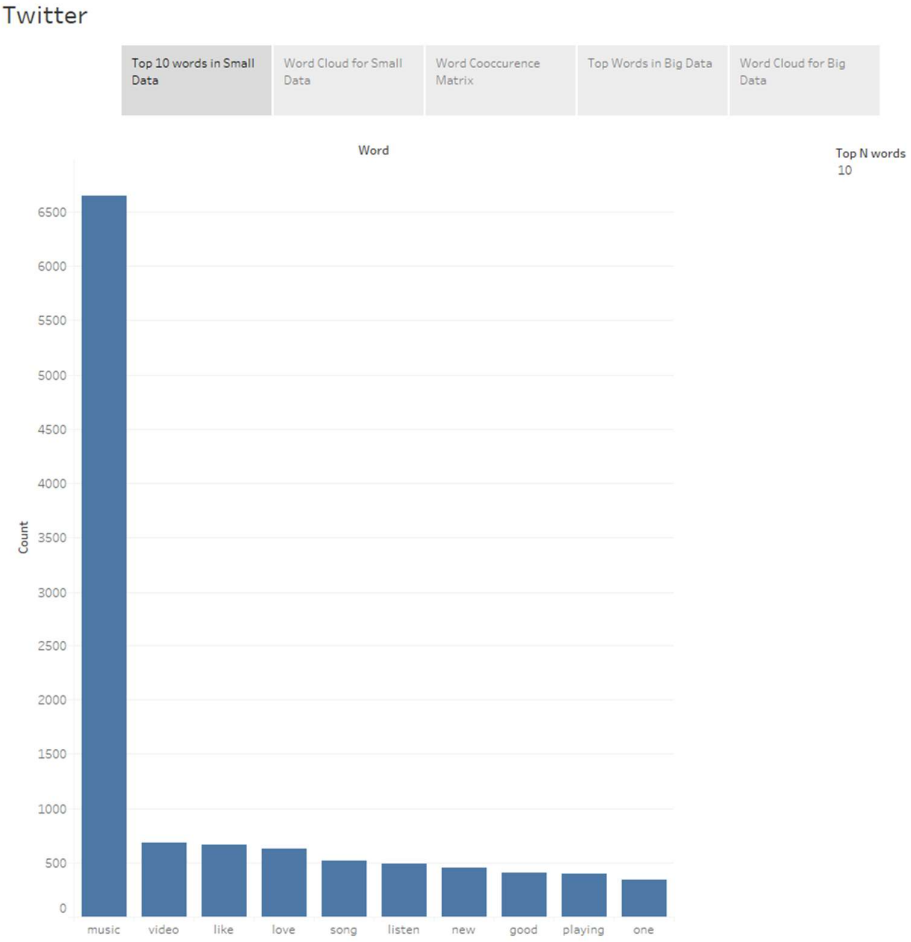
The word clouds for the big data collected converged with the increase in data collected from all 3 sources, although subtle differences can be noticed due to the nature of language used in the different media. Although data was collected only using a few keywords, many words related music that were not queried can be found on the word clouds.

The word co-occurrence matrices which were a plot of the top 10 words with their co-occurring words showed good sensible results as well, with words like ‘guitar’ co-occurring with top words like rock, music and song (common crawl data).

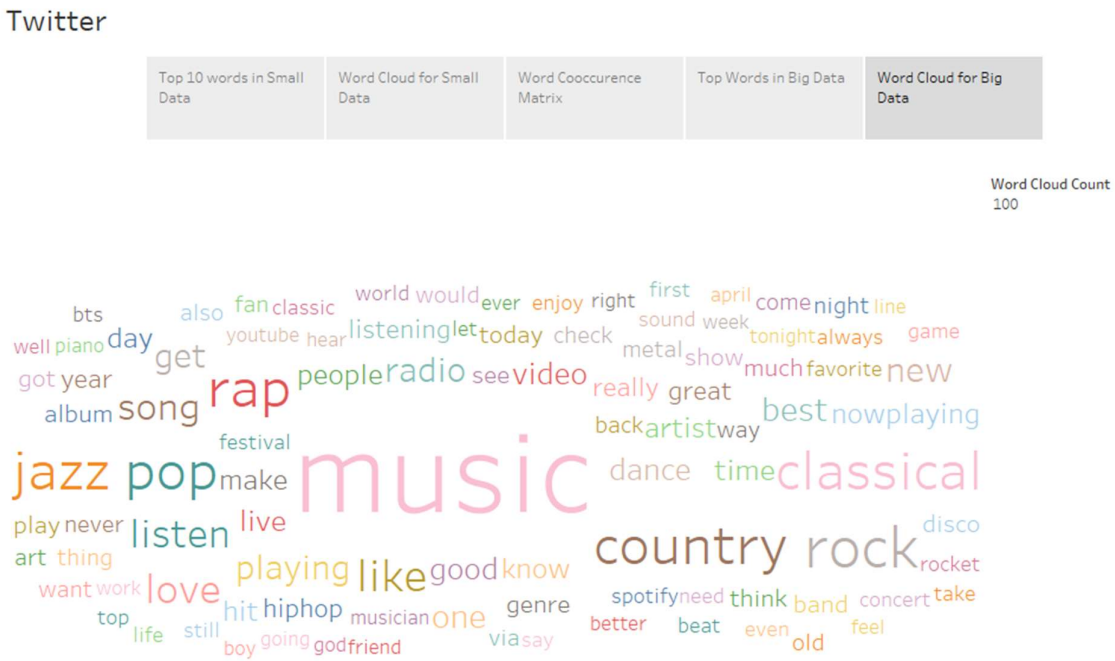
Note: The top 10 small data words for twitter are a bit skewed with music occupying a large portion of the graph. This may be due to the fact that tweets were extracted using the keyword “Music” and all tweets will contain the word.

Twitter

- Word Count – (small data with keyword Music alone)

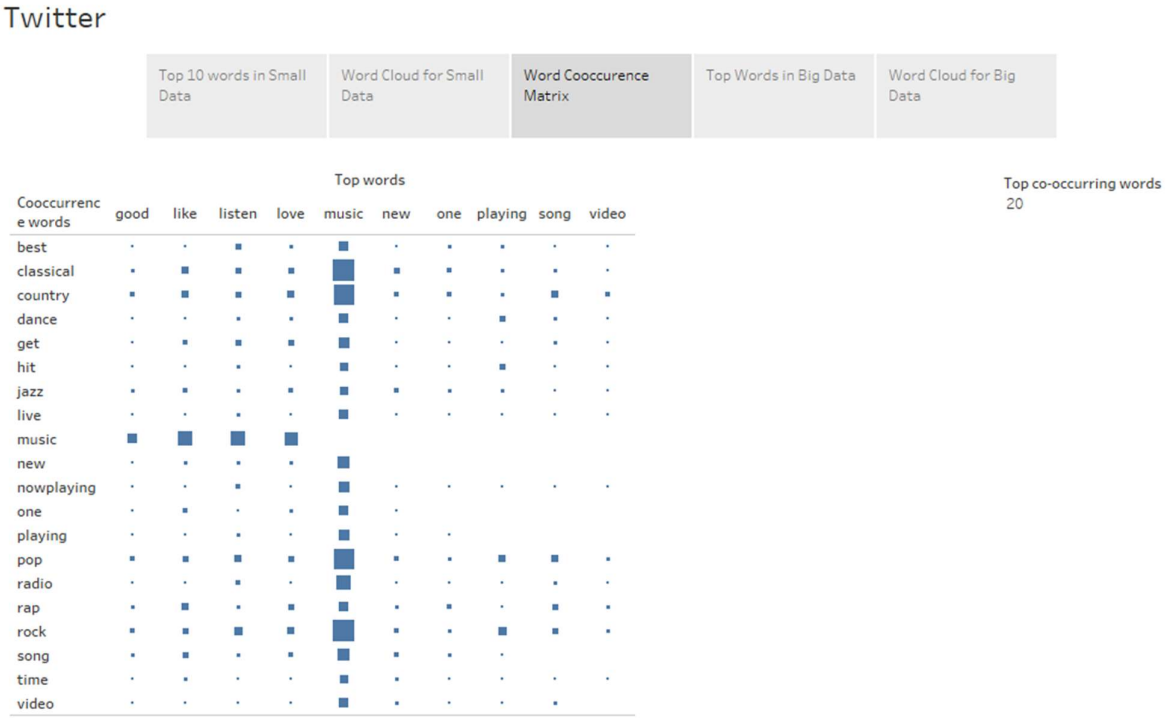


- Word Cloud – (on all data collected so far)



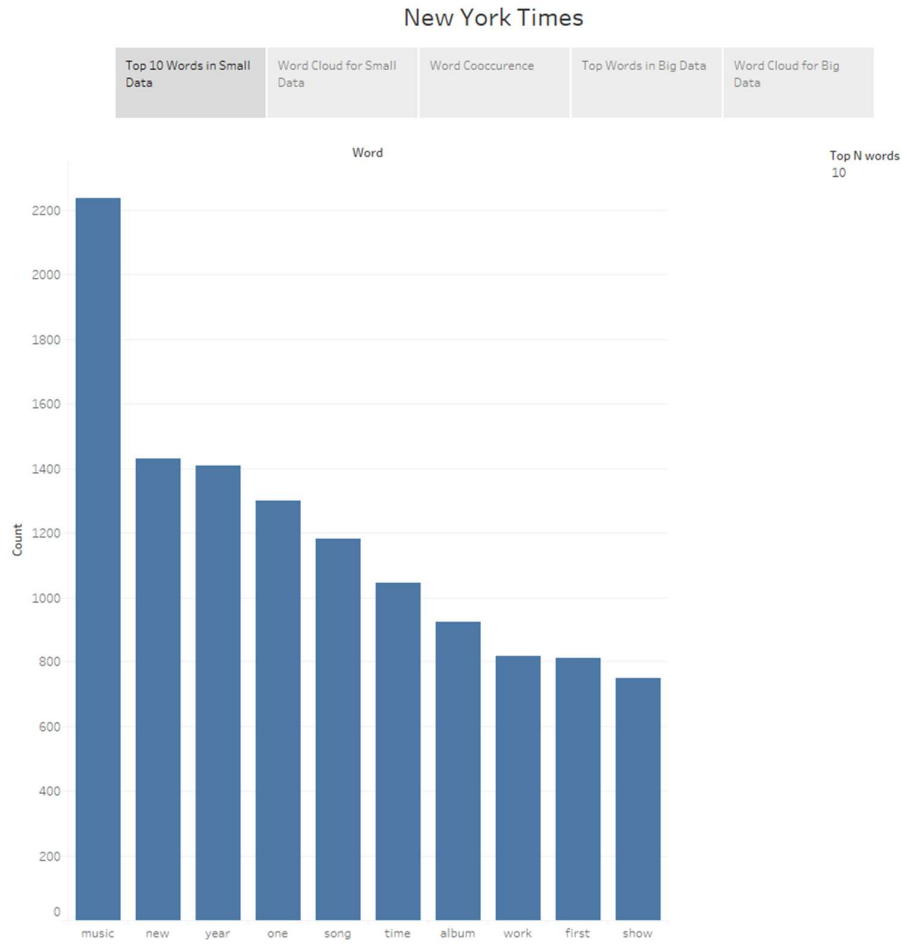
- Word co-occurrence Matrix – (On top 10 words from small data)

  Top 10 words from small data are represented by “Top words” and co-occuring words by “Cooccurrence words” labels.

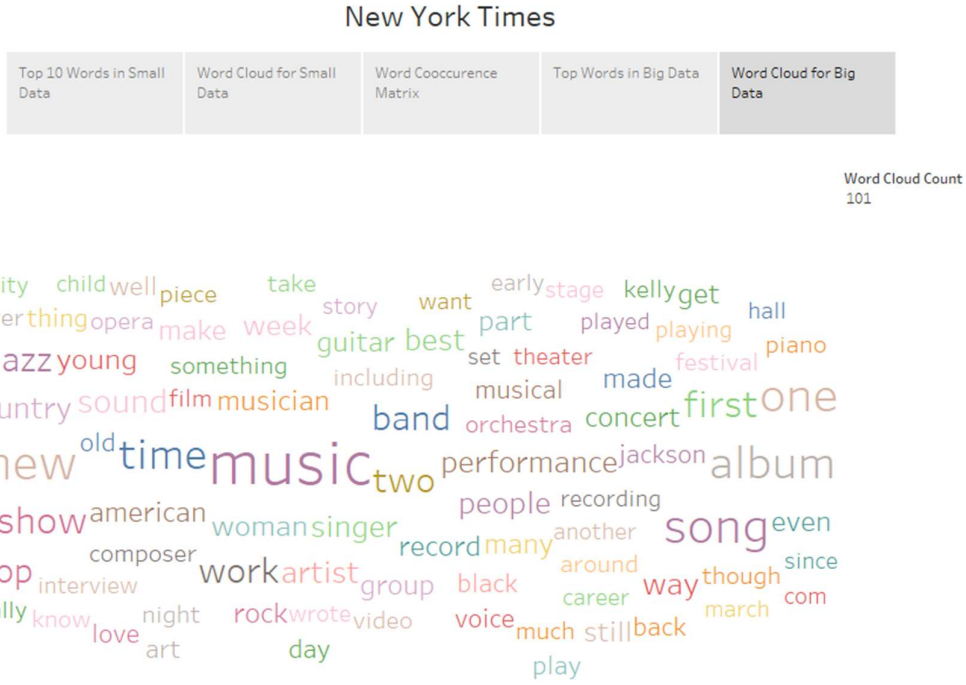


New York Times

- Word Count – (small data with keyword Music alone)



- Word Cloud – (on all data collected so far)

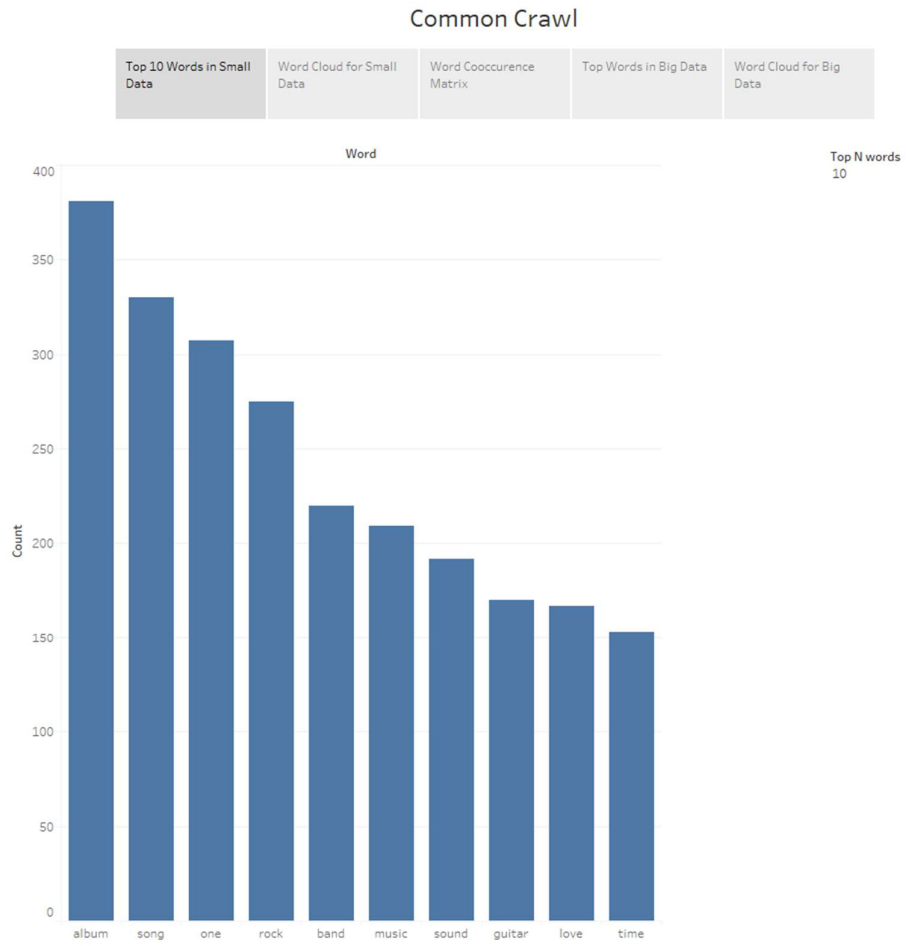


- Word co-occurrence Matrix – (On top 10 words from small data)

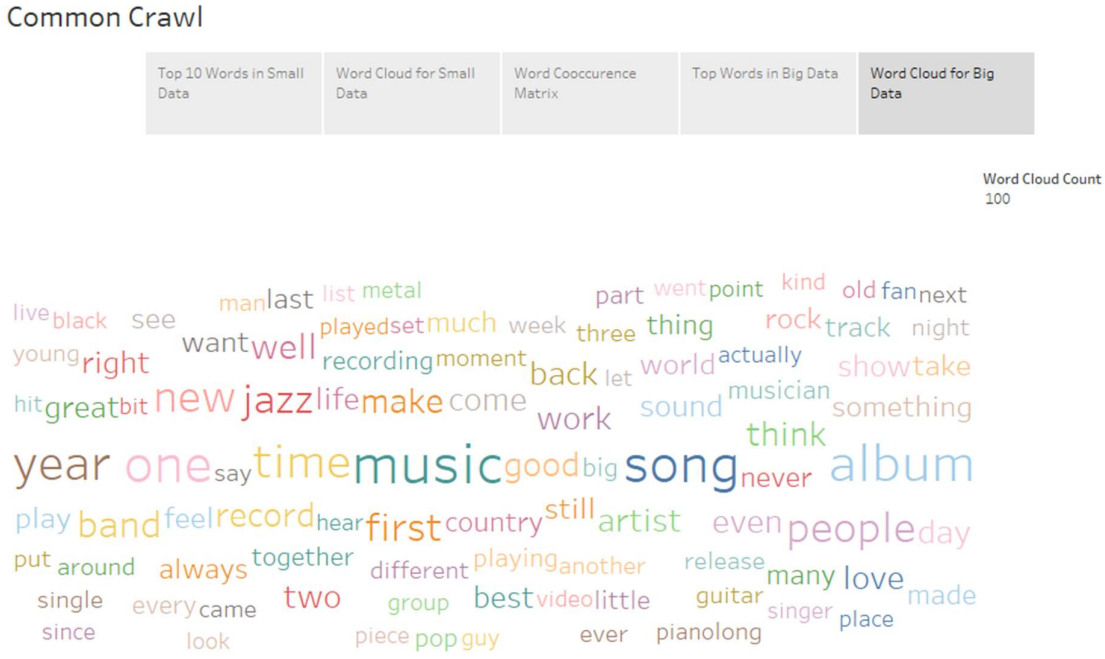
  Top 10 words from small data are represented by “Top words” and co-occuring words by “Cooccurrence words” labels.

Common Crawl

- Word Count – (small data with keyword Music alone)



- Word Cloud – (on all data collected so far)



- Word co-occurrence Matrix – (On top 10 words from small data)

  Top 10 words from small data are represented by “Top words” and co-occuring words by “Cooccurrence words” labels.