

# UC Riverside

## UCR Honors Capstones 2016-2017

**Title**

GTWENDS

**Permalink**

<https://escholarship.org/uc/item/41q0w0q9>

**Author**

Asfour, Mark

**Publication Date**

2017-12-08

**Data Availability**

The data associated with this publication are within the manuscript.

GTWENDS

By

Mark Jeffrey Asfour

A capstone project submitted for  
Graduation with University Honors

October 20, 2016

University Honors  
University of California, Riverside

APPROVED

---

Dr. Evangelos Christidis  
Department of Computer Science & Engineering

---

Dr. Richard Cardullo, Howard H Hays Chair and Faculty Director, University Honors  
Associate Vice Provost, Undergraduate Education

## **Abstract**

GTWENDS is an online interactive map of the United States of America that displays the locations of trending Twitter tweets, Google Search trends, and Google Hot Trends topics. States on the map are overlaid with a blue color where Twitter trends originate and a red color where Google trends originate with respective opacity levels varying based on the levels of interest from each website. Through the use of web crawling, map-reducing, and utilizing distributed processing, this project allows visitors to have an interactive geographic visual representation of current national social media topic activities. Visitors can use GTWENDS to learn about social media trends by observing where trends originate from, comparing the contrasts and similarities between trends from Twitter and Google, understanding what types of events trigger mass social media sharing, and much more.

## **Acknowledgements**

I have worked on and developed GTWENDS with my classmates, Mehran Ghamaty (University of California, Riverside Spring 2016) and Jacob Xu (University of California, Riverside Spring 2016), during our senior design project class, CS 179G Databases Spring 2016, under the supervision of my professor and mentor, Professor Evangelos Christidis. We put hard effort into this project as a team based on the advice given by our professor and by the class's Teaching Assistants, Moloud Shahbazi, and Mohiuddin Qader. We continually worked side by side throughout the development process and are proud of the work that we have accomplished.

## Table of Contents

Abstract.....	ii
Acknowledgements.....	iii
Introduction.....	1
Methodology.....	5
Analytical Discussion.....	11
Conclusion.....	17
Bibliography.....	21

## Introduction

GTWENDS is an interactive website where visitors can view how trends originating from Twitter and Google are distributed across the United States of America. A geographic representation of the nation colored in varying shades of red for Google and blue for Twitter per state indicates different levels of interests for each trend according to each site. The name GTWENDS comes from a play on the words Google, Twitter, and “trends”. In this project, data is collected from Google’s Search Trends site, Google’s Hot Trends site, and tweets from Twitter. Thus, by replacing the first two letters from the word “trends” with a “G” from Google and the “TW” from Twitter, the easy and succinct project title GTWENDS is formed.

This project is located at <https://markasfour.github.io/SeniorDesignProject/>.

Upon loading the page, visitors are presented with the main page of the site as shown in Figure 1 which includes a large uncolored map of the continental United States of

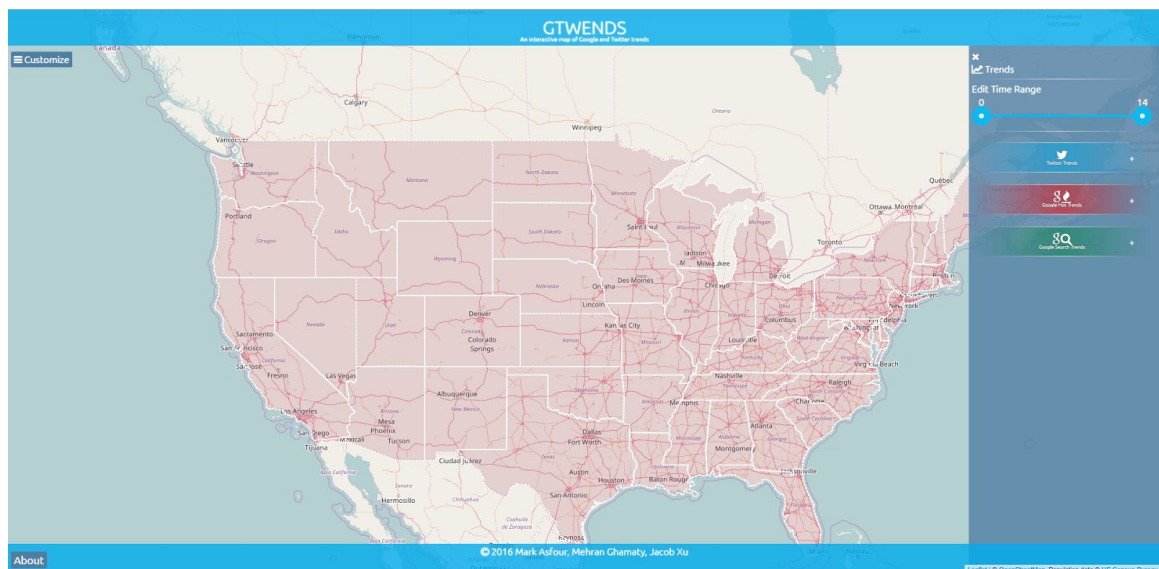


Figure 1

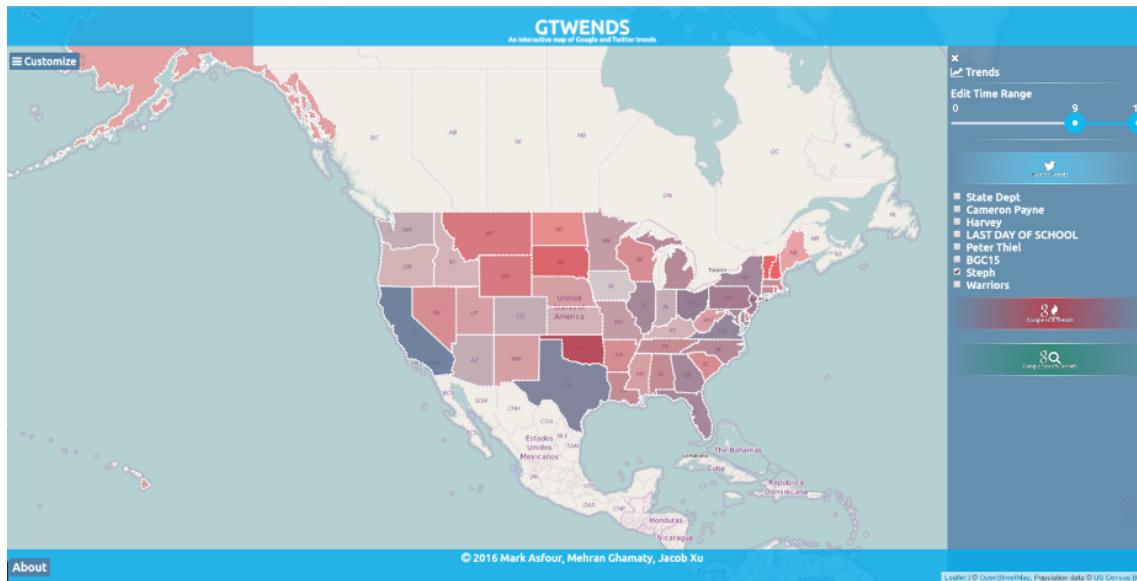
America, a trend selection panel on the right, a Customize button option at the top left, and an About button at the bottom left.

The trends panel on the right serves as an interface for visitors to explore our collected trends data and is split into two distinct parts. The first part at the top of the trends panel is the time range slider which determines how many days in the past that trends panel is the time range slider which determines how many days in the past that trends will be loaded. The days can range from current trends (zero days) to the trends of two weeks ago (fourteen days). The second section contains three expandable lists organized by their origin — Twitter Trends in blue, Google Hot Trends in red, and Google Search Trends in green. Expanding these lists displays a series of trends with checkboxes originating from their respective sites that occurred during the selected time range as shown in Figure 2.



**Figure 2**

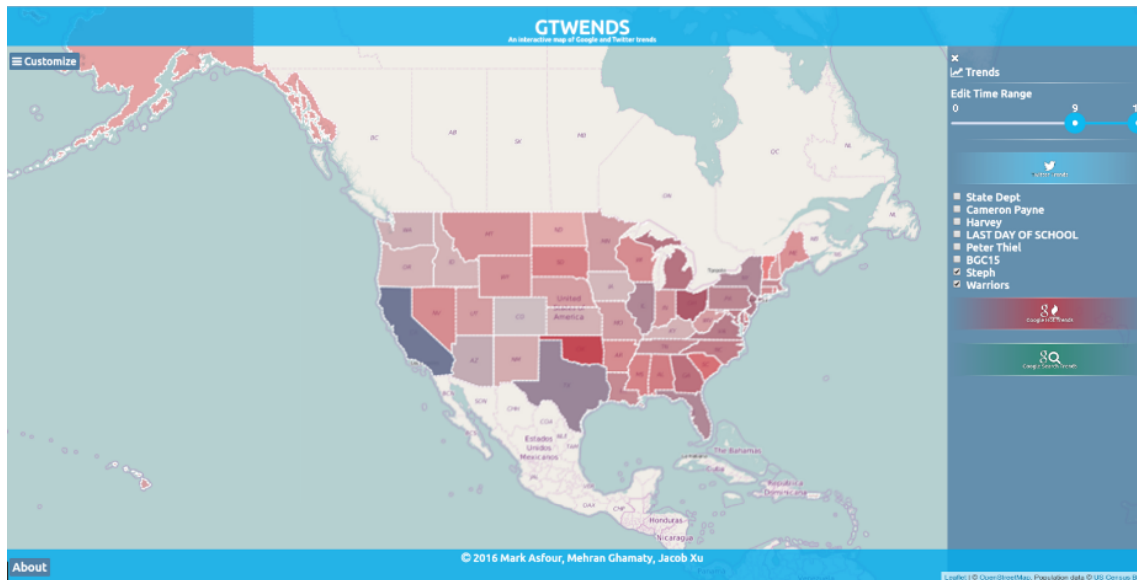
Visitors can choose what data they would like to have displayed on the map by marking the checkboxes next to the listed trends. Single trends or multiple trends from a single source or multiple sources can be selected depending on the data that is wished to be viewed. Figures 3 and 4 portray examples of how data is displayed onto the map with varying trends selected. Based on the displayed data, visitors can analyse the data and draw their own conclusions from it.



**Figure 3**

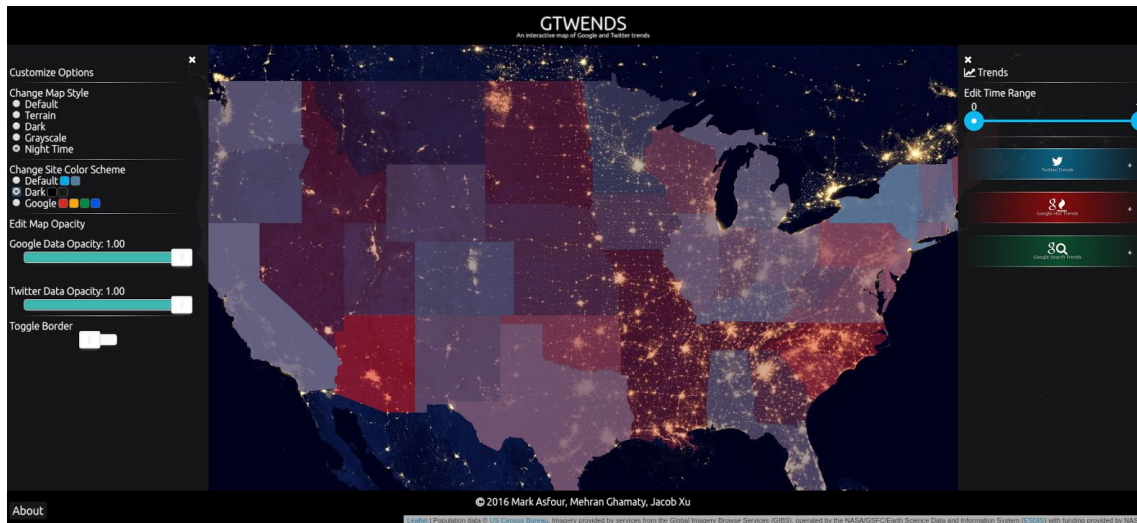
From Figure 3, it can be seen that the trend term Steph (Stephen Curry — NBA basketball player for the Golden State Warriors) has varying amounts of Twitter and Google interest levels per state across the nation. For instance, there are high levels of Twitter interest in California and Texas while there are high levels of Google interest in Oklahoma, New Hampshire, and Vermont. Furthermore, from Figure 4, it can be seen that adding the trend Warriors for data viewing has slightly altered the interest levels in some of the states. For example, the Twitter interest in Texas slightly decreased while the Google interest in Oklahoma slightly increased. Data depictions like this and much more are available for visitors to interact with and learn from.





**Figure 4**

Lastly, the appearance of the map, data, and website theme can all be customized by selecting the Customize button at the top left of the screen. Clicking it will reveal a customization options panel with four distinct sections. In the first section, the map style can be modified to display different variations of the same base map of the United States of America. The second section allows changes in cosmetic appearances to the site by toggling different general color schemes. The following area gives the option of manipulating the opacity of the color levels over all of the states coming from Google or Twitter data. The final option in the customization panel is a switch to hide or reveal the state's borders on the map display. Figure 5 is an example of a possible customization settings change.



**Figure 5**

Here, the night time map is selected along with the dark color scheme. The data opacity for Google and Twitter are both raised to their maximum levels and the state's borders are toggled off.

## Methodology

There are three main components to how GTWENDS is created: the web crawler, the back-end data processing, and the front-end web interface. The web crawler and the back-end data processing are all performed on a personal server while the front-end web interface is hosted on GitHub under my account. All three components are heavily reliant upon one another to deliver meaningful data to visitors to the site.

The purpose of the web crawler is to constantly retrieve trend data from Twitter and Google to then use for analysis. It collects the listed trends from Twitter, Google Hot Trends, and Google Search Trends, collects Twitter tweets that are streamed from the Twitter Application Program Interface (API), and collects interest levels by state for trending topics from Google. The web crawler is built as an executable script written in

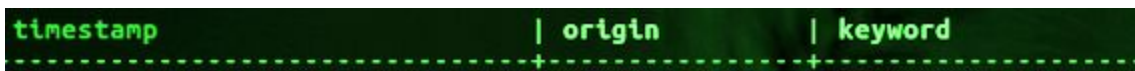
the Python 3 programming language that is continually ran on a server. This script uses several open source Python libraries to assist in retrieving the desired data including BS4, Pytrends, Tweepy, and other standard libraries such as Time, Datetime, etc.

The data extraction from the the three sites comes in two steps and in varying timing intervals. The first step is to gather all of the trends from the three sources. With this information, a general list of what is trending is created. The list is populated with trends from Twitter every fifteen minutes and from Google Hot Trends and Google Search Trends every hour. These intervals are based off of how often their respective sites update their own list of trending topics. The second step of the web crawler is to find out the level of interest for each of these trends according to Google and Twitter. A lookup is performed on Google's site by entering the trend topic and downloading a Comma-Separated Values (CSV) file containing the reported levels of interest for that trend topic. This is done for every trend in the list and is repeated hourly after the trends list is refreshed with new data. Tweets are constantly streamed in and stored in the meanwhile and will be processed for interest levels per trend topic in the processing phase of the project.

Extracting the data from the websites requires unique approaches for each of the three sites. Twitter provides a free API that makes collecting their current trends and grabbing streamed tweets fairly simple and straightforward. To do this, the Tweepy Python library came into use to connect to our Twitter Developer's account, giving us access to their data. The data collected per tweet are the message, location (country, state, latitude, longitude), number of favorites, and number of retweets. An unofficial

API for Google trends created by General Mills is used to collect the Google Search trends data and interest levels per state for trending topics. This API alongside a valid Google account automates the process of downloading CSV files containing data about trends and their interest levels in each state from the site. Lastly, the Google Hot Trends' site does not provide an official API to use. Thus, retrieving the data comes down to extracting the list of top trends straight from the webpage's HTML code. The Python library BS4 and its BeautifulSoup HTML parser tool are used to parse through the online code and store the listed trends from the site.

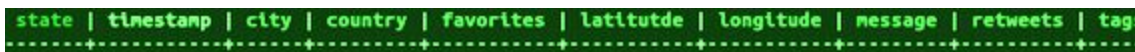
As all of this data is collected, it is moved to a Cassandra NoSQL database on the server where it will be stored and processed in the second phase of the project. The Cassandra Python library is used to connect and push the data collected from the web crawler to the database into their respective tables. The table schemas setup in Cassandra to store the collected data are as shown in Figure 6 and Figure 7.



timestamp	origin	keyword
-----------	--------	---------

**Figure 6**

Figure 6 shows the keywords table which contains the list of trends along with their respective origins and timestamp of when they were retrieved.

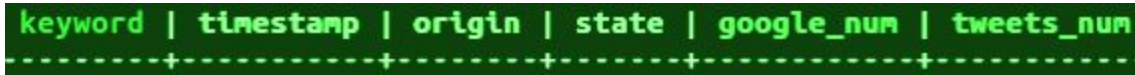


state	timestamp	city	country	favorites	latitude	longitude	message	retweets	tag
-------	-----------	------	---------	-----------	----------	-----------	---------	----------	-----

**Figure 7**

Figure 7 shows the Twitter tweets table which is organized by the state of the tweets origin, the timestamp, city, country, number of favorites, latitude, longitude, message,

number of retweets, and tags. Another table is setup in Cassandra to store the processed data from the processing phase of the project with the schema as shown in Figure 8.



keyword	timestamp	origin	state	google_num	tweets_num
---------	-----------	--------	-------	------------	------------

**Figure 8**

Figure 8 shows how the data collected from the Google CSV files and the data post-processing from the Twitter tweets are stored, organized by the trend, timestamp, origin, state, Google interest level, and Twitter interest level.

The Twitter tweets data processing is a separate message parser program that looks for a trend word to be mentioned within the tweet's message. It counts and keeps track of the number of times a trend is mentioned within all of the gathered tweets. It also adds the number of times a tweet that mentions the trend has been favorited and retweeted to the total to account for the audience reached by the tweet. The program then stores this interest level number into the database table shown previously in Figure 8. This number is placed under the column "tweets\_num" and into the row corresponding to the trend being analysed.

This second component of GTWENDS that processes the Twitter tweets is done by using the MapReduce programming model ran across several nodes set up on our server through the Apache Hadoop MapReduce implementation. MapReduce is a programming model used for processing and generating large data sets on a parallelized computer system. It runs two main processes, mapping and reducing. The MapReduce would first perform the mapping procedure that filters and sorts the data passed into the program. Then, it runs the reducing procedure that performs the desired data analysis and

summarizing of the mapped data. Apache Hadoop is the open source implementation of MapReduce that we used for GTWENDS. With it, the mapping and reducing of the our data is completed in the Hadoop Distributed File System (HDFS). This file system allows the data and processing to be distributed across multiple computers, or nodes. With the power of HDFS, computation time is greatly enhanced because of the parallel processing and a form of redundancy is created to keep the system running despite any issues that may arise on a single node.

In GTWENDS, the Apache Hadoop MapReduce processes are written in the Python programming language and are ran across the HDFS installed on our server split into a cluster of six nodes, each on one of the six cores of the server. The mapping process first filters all of the gathered tweets for tweets that only originate in the United States of America. Then, it organizes these tweets by state before it parses each of the tweet's message. The mapper tallies the number of times a trend word is mentioned and the number of times the tweet has been favorited and retweeted as it loops through all of the tweets passed to it. Special cases such as trend words typed out with incorrect capitalization is handled by our code. The tallied number, the timestamp of the tweet, the state of origin, and the trend word matched is passed per iteration of the loop to the reducing process where it analyses the given data. The reducer then formats this data so that it is ready to be stored to the Cassandra database table. Communication between the mapper and reducer is completed through standard console input and output writing.

Finally, after all of the data is collected and processed, GTWENDS' web interface is ready to display the results. This third component of the project is composed of two

unique functionalities for getting the processed data available online, and a separate method to display the data interactively. The GTWENDS website is hosted on my GitHub account and serves as a web interface to the data collected and processed on our backend server. GitHub provides free website hosting for account members, removing the need for having a dedicated, secure server to host the site for a varying visitor count.

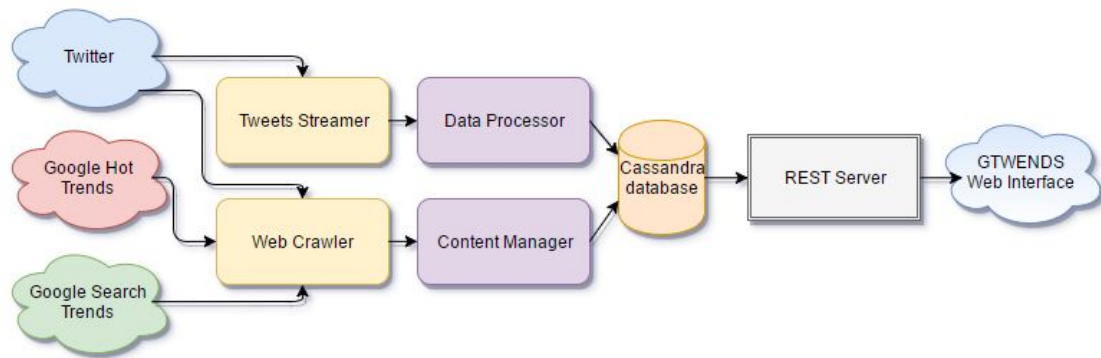
The first step in moving the data online is the back-end Representational State Transfer (REST) server. It simply hosts the data online to be readily available for the website to retrieve in Javascript Object Notation (JSON) format. The REST server is built with the Flask microframework for Python. It collects the processed data from the Cassandra database, transforms it to a JSON format, and posts it online for the GTWENDS website to retrieve.

The second step in getting the data to the web interface is reading and storing the information hosted by the REST server automatically as a visitor loads the webpage and makes trend viewing selections. Google's AngularJS Javascript library does the trick by connecting to our REST server as the site loads. It first populates the expandable trends lists in the trends panel with the collected trends when the site first loads. When a visitor clicks a trend to view, AngularJS makes a request to our REST server to deliver the data corresponding to the selected trend. After this data is received, AngularJS stores this data locally for the interface to then handle for displaying. These requests between AngularJS and the REST server are completed via Uniform Resource Locator (URL) communication. The URL is read and interpreted by the REST server as a query for data and makes that requested data available online for AngularJS to retrieve and store.

The final element is the User Interface (UI) of the GTWENDS website. The UI is built with standard HTML5, CSS3, and Javascript along with the JQuery Javascript library, the LeafletJS Javascript map handler package, the noUiSlider Javascript slider package, and the Slick Javascript carousel package. Independent of AngularJS's operations, this set of code displays the fundamental structure of the interactive website. The structural skeleton of the site is written in the HTML5 code and is designed and stylized with CSS3. Javascript and JQuery are used to provide the transitional animations, LeafletJS is used to display the map along with a choropleth heat map overlay over the states to represent the data, noUiSlider is used for the opacity sliders and border toggle in the customization panel, and Slick is used for the About slides animation and functionality. The maps displayed by LeafletJS are provided by OpenStreetMap for the default, grayscale, and dark maps, MapQuest for the terrain map, and NASA for the night time map. The data for the choropleth heat map is dependant upon AngularJS retrieving the data from the REST server. Once it does so, it uses the data stored by AngularJS to respectively recolor each state based off of their Twitter and Google interest levels.

Figure 10 summarizes the entire methodology used for GTWENDS in a concise flow diagram.





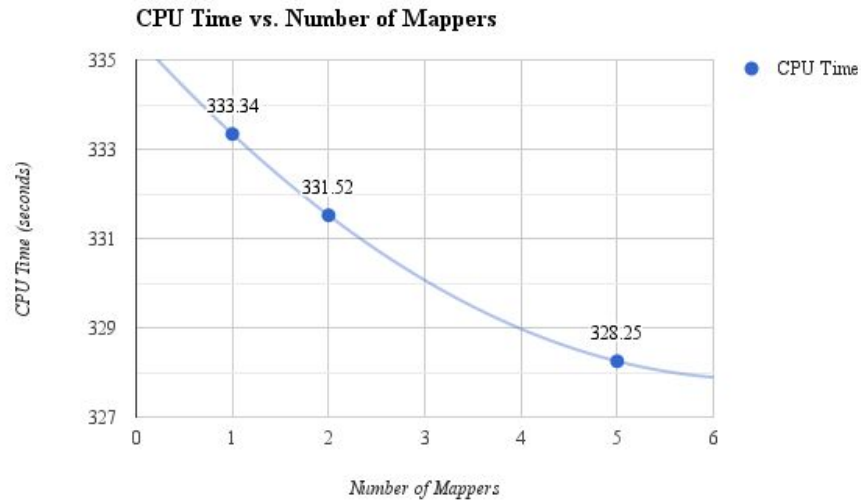
**Figure 10**

## **Analytical Discussion**

GTWENDS is a project that is dealing with big data and is presented as an online web interface. This means that efficiency is a crucial point in how this project runs in gathering its data and in how it is presented to the end users. Through the proper usage of software services and testing with various efficiency options, we have built a program that can function smoothly and effectively.

The data collected for analysis from Twitter, Google Hot Trends, and Google Search Trends total in the gigabytes (GB). Over a one week period, the web crawler and Twitter tweets streamer receives approximately 5.7 GB of data, roughly half a megabyte of data a minute. A large majority of this data comes from the collected tweets followed by the CSV files downloaded from Google Search Trends.

Processing the Twitter tweets data is managed by the Apache Hadoop MapReduce that parallelizes the process across multiple nodes set up on our server. We tested the different run times of analysing the Twitter tweets with different node configurations on our server and had the following results shown in Figure 11.

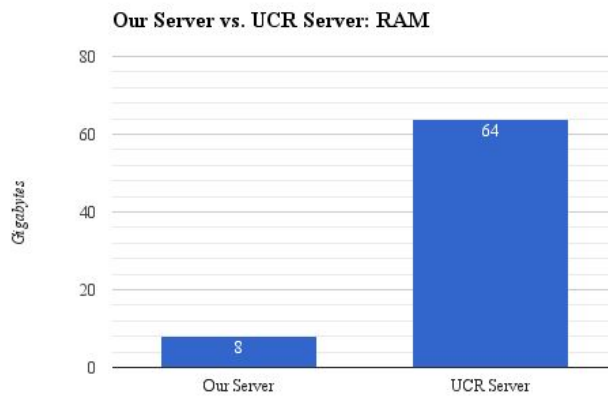


**Figure 11**

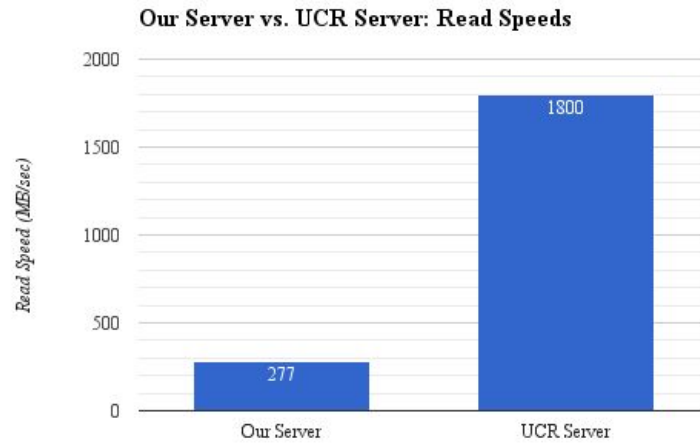
Three tests were conducted on our server with one, two, and five nodes configured respectively for each test. A mapper was ran in each case and the results shown in the graph in Figure 11 indeed indicates that the additional nodes with parallelized computing enhances the run time performance at a noticeable level. There is a difference in seconds between running the data processing on a single node compared to running on a cluster of five nodes. It should be noted that an improvement in seconds may not seem like much of an improvement, but its beneficial impact computing-wise is shown profoundly as the data sets processed grow larger the longer the project is ran. As the trendline indicates in the graph, adding more nodes is not a guarantee that the desired process will complete at a quicker pace. It will come to a point where the tasks are so evenly distributed across a selection of nodes such that adding additional nodes cannot improve the system performance because there is no more work to be shared. This worked ideally with our

system since it has six processing cores. Having five nodes in our cluster as opposed to six has very minimal impact on the possible performance improvement. This last core is conveniently used for hosting the REST server connecting our data with the internet.

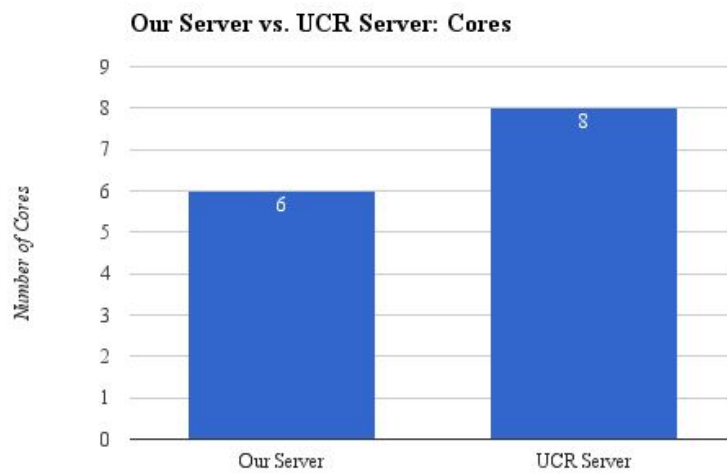
We also tested running our data processing programs on the server clusters provided by the University of California, Riverside (UCR) Computer Science department given to our class for the Spring 2016 quarter. This server offers significantly more Random Access Memory (RAM), faster read speeds, more cores, and faster Central Processing Unit (CPU) clock speeds compared to our system as shown in the charts in Figures 12 through 15.



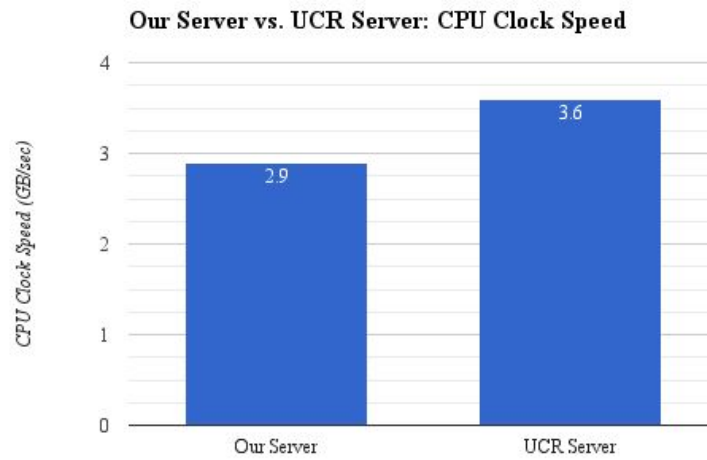
**Figure 12**



**Figure 13**

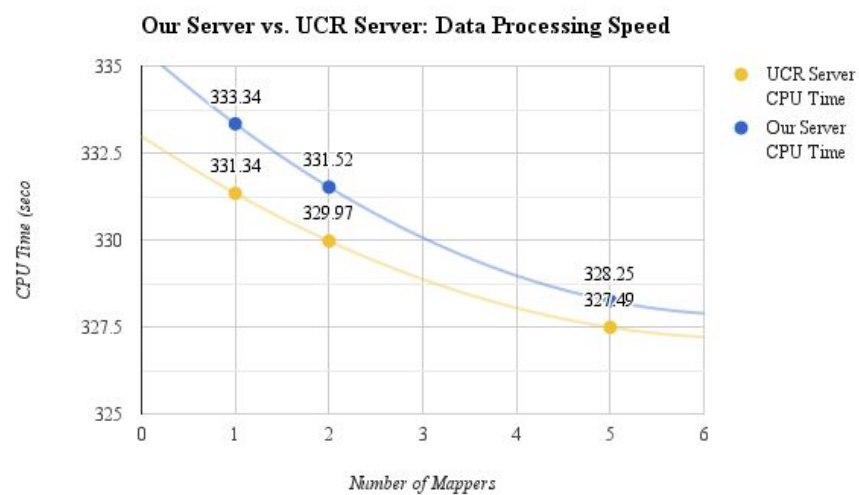


**Figure 14**



**Figure 15**

Despite these large system improvements, when running our Twitter tweets data processing across the UCR server cluster, the performance enhancements were underwhelming with only only a couple of seconds of improvement. As shown in the graph in Figure 16.



**Figure 16**

Given the large system improvements that the UCR server system provides, there is a visible margin of improvement over our server. However, this margin is not the giant performance leap that one may expect from a massive performance increase. This is certainly due to the fact that our data processing requirements are fairly simplistic. A string parser of only a few gigabytes worth of data is a relatively simple task that does not necessarily require the industrial level system that UCR provides. The server cluster that UCR offers would truly make a substantial difference had there been more processing required by GTWENDS, but for the purposes of this project, our server does suffice.

Connection speed from the GTWENDS website to the REST server for data retrieval to the front-end depends greatly on the end user's internet connection. The initial loading time of the website can be as quick as five seconds after landing on the site or as long as one minute depending on the local internet speed. Filtering the time range of trends to be shown with the time slider in the trends panel will again require some download time. This can take between five seconds or up to a minute based on the local internet speed. Displaying the trends data after clicking the checkboxes under the expandable lists in the trends panel requires some processing time as well. This means that after retrieving the data from the REST server, the local system additionally has to perform some minor processing to color the map accordingly, ranging in processing time anywhere between two seconds to a minute depending on the local system. For instance, viewing a trend's data on a mobile device will generally take longer than viewing the same trend data on a desktop device due to the relatively greater performance on the desktop (assuming that the compared desktop device does have greater processing

capabilities than the mobile device). Customizing the GTWENDS site layout is close to instantaneous because it is only minor web display alterations.

## **Conclusion**

GTWENDS utilizes big data processing tools to provide data for an informative interactive website. Through web crawling and mass data collection, social media data is extracted and processing to reveal new illuminating data. With the ability to manipulate how the extracted data is displayed on the website, GTWENDS provides a unique experience of learning about the social media trends in the United States of America.

There is plenty of room for expansion upon this project such as upgrading the services used throughout the process, improving the server setup, or adding additional social media sources for data collection. The API currently used for retrieving the Twitter tweets is a free version of Twitter's full developer API product. Since it is a free version, the current API used does not receive all of the tweets that the full version is capable of receiving. Thus, one area that can further improve the quality of Twitter data for GTWENDS is to use the full version of Twitter's API. Furthermore, a licensed version of a MapReduce system can be used as opposed to the open source Apache Hadoop implementation could possibly yield quicker overall execution times. Such a product could assist the performance on the cluster with more efficient parallelization algorithms or even more efficient mapping and reducing processes possibly.

Upgrading the server to something of the likes of the UCR server cluster offered during the Spring 2016 quarter can surely improve the overall running time. Additional storage for the Cassandra database can also factor in well if additional data is to be

retrieved while growing the project. Whether the improvement remains to only make several seconds of a difference, it does provide the power and capability of supporting an expanded GTWENDS containing more data to retrieve, process, and deliver.

Adding additional social media sources for data collection and interpretation is another way GTWENDS can be expanded. It can include trend information from other websites such as Facebook, Instagram, Foursquare, or Pinterest to name a few. If alternate website have their trend data readily available to be used by developers, it can certainly be a candidate to be added on top of the existing project. Such an addition can provide visitors with additional insight into how trends can vary between different sites and how the sites' data represents regions of the nation's interest levels in the topics. These additions will require more programs for web crawling and API usage to retrieve the data, more database storage, and more processing capabilities. The system expansions mentioned previously can definitely asses these new needs, removing the boundaries set by computational limitations. Whether or not the name GTWENDS will remain as the title of this project as additional social media sites are added can be up for debate.

It should be noted that changes made by Google or Twitter on their trend information, changes on their websites, or changes on their services could severely affect the functionality of GTWENDS. For example, if Twitter removes its free tweet streaming API service, the project can no longer retrieve tweets from Twitter without needing to upgrade to their paid API version. Also, since Google Search Trends does not provide an official API for collecting their data, there is a great possibility that Google



can modify their site while the third party open source Pytrends API by General Mills can lag behind the new update. Thus, GTWENDS becomes dependent upon the consistency of the Google Search Trends site, the update delivery time of the Pytrends API, and the level of sameness in the methodology of retrieving the data post upgrade. Issues like these reveal the possible downsides that can arise by further expanding a multi-dependent site like GTWENDS. However, if successful, the results pay off with plenty of informative data from a variety of sites.

GTWENDS provides an online interactive service for exploring the nature of social media trends in the United States of America. Gigabytes of trend data retrieved from Google and Twitter are transformed into geographical based data and are then presented online in a friendly and simple to use manner. Nontechnical visitors can enjoy and learn from the trend data that they wish to view and can modify the website design to suit their visual liking. Through back-end big data processing, GTWENDS is made possible for all to gain a greater understanding of trending social media.

## **Bibliography**

Agafonkin, Vladimir. "Leaflet." Leaflet Dev Blog Atom. N.p., 27 Sept. 2016. Web. 21

Oct. 2016. <<http://leafletjs.com/>>.

"AngularJS." AngularJS. Google, Web. 20 Oct. 2016. <<https://angularjs.org/>>.

"Apache Cassanrda." Apache Cassandra. The Apache Software Foundation, 29 Sept.

2016. Web. 20 Oct. 2016. <<http://cassandra.apache.org/>>.

"Earth at Night." NASA. NASA, 2 Apr. 2009. Web. 20 Oct. 2016.

<[http://www.nasa.gov/topics/earth/earthday/gall\\_earth\\_night.html](http://www.nasa.gov/topics/earth/earthday/gall_earth_night.html)>.

"Google Trends." Google Trends. Google, 20 Oct. 2016. Web. 20 Oct. 2016.

<<https://www.google.com/trends/>>.

"Google Hot Trends." Google Hot Trends. Google, n.d. Web. 20 Oct. 2016.

<<https://www.google.com/trends/hottrends/atom/hourly>>.

Hill, Aron, and Joshua Roesslein. "Tweepy/tweepy." GitHub, 17 Oct. 2016. Web. 20

Oct. 2016. <<https://github.com/tweepy/tweepy>>.

Gersen, Léon. "NoUiSlider." NoUiSlider - JavaScript Range Slider. N.p., 11 Oct. 2016.

Web. 21 Oct. 2016. <<https://refreshless.com/nouislider/>>.

"JQuery." JQuery. The JQuery Foundation, n.d. Web. 20 Oct. 2016.

<<https://jquery.com/>>.

"Maps ~ A Great Canvas for Any Geospatial Application." MapQuest Developer

Network. N.p., 17 Aug. 2016. Web. 20 Oct. 2016.

<<https://developer.mapquest.com/products/maps>>

OpenStreetMap Contributors. "OpenStreetMap." OpenStreetMap. UCL VR Centre,

Imperial College London and Bytemark Hosting, n.d. Web. 20 Oct. 2016.

<<http://www.openstreetmap.org/about>>

"Pytrends." GitHub. General Mills, 19 Sept. 2016. Web. 20 Oct. 2016.

<<https://github.com/GeneralMills/pytrends>>.

"Streaming APIs." Public Streams. Twitter, Inc., Web. 20 Oct. 2016.

<<https://dev.twitter.com/streaming/public>>.

"Welcome to Apache™ Hadoop®!" Apache Hadoop. The Apache Software Foundation,

11 Oct. 2016. Web. 20 Oct. 2016. <<http://hadoop.apache.org/>>.

Wheeler, Ken. "Slick." Slick. N.p., 19 Oct. 2016. Web. 20 Oct. 2016.

<<http://kenwheeler.github.io/slick/>>.