Identification of Localized Trends in Twitter Data

Ben Johnston

Jamison Hyman

Minhaz Mahmud

Seth Mosgin

Sothiara Em

Abstract

Our final Hadoop project accomplishes the task of identifying localized trending hashtags on Twitter. Using Twitter streaming data and a Hive analytic, we map trending tweets to their corresponding cities and display this visually using Google Maps. To accomplish this task, we used a number of technologies, including the Twitter Streaming and Google Maps Application Programming Interfaces (API), as well as the Hadoop Distributed File System (HDFS), Flume, Avro, and Hive. Our solution can be broken down into four basic steps. First, we capture raw Twitter data and format it to include only the sections in which we are interested. Next, we write this data to a Flume spooling directory, which moves the data through Avro and puts it into HDFS. After that, we perform our analysis to extract the most common hashtags for each region. Finally, we move this file to a remote server, where we display our results in Google Maps.

Problem

Our initial inspiration for this project came from the idea that while hashtags trending on Twitter are reflective of the Twitter community as a whole, they may not be relevant to a particular community. With that in mind, our main goal going into this was to identify trending data within Twitter that is specific to individual communities. This data not only allows us to visualize the "collective thinking" of specific communities, it also allows us to see the similarities between geographically separated communities. Also, we were interested in something that we could sell to Google for an amount in the Oculus to WhatsApp range.

Data Sets and Transformation

We acquired data from Twitter's public streaming API. A java class implements Twitter4J which is an open source Java library for the Twitter API. The output configurations are stored in a file called twitter_dump.properties and the OAuth configurations are stored in twitter4j.properties. The names of the property files are passed in as command line arguments to the java program along with the latitudes and longitudes of the bounding box for the region and the name of the region. The java program takes this information and connects to the Twitter Public Streaming API to download tweets that originate from the location specified. We receive the data in JavaScript Object Notation (JSON) format and use a JSON library to extract the tweet text from each JSON object that is returned and we output it in a custom format. The data is buffered until we reach a maximum file size limit (specified in properties file) and is then flushed (path in properties). That data is inserted into HDFS and a pair of Hive scripts are run that extract the most common tweets and their locations. The results of the hive script are saved onto HDFS in TSV format. The data is then transferred to a web server hosted on Digital Ocean. The data is then read and displayed on a map by utilizing the Google Maps API.

Implementation

Given the natural divisions in our task, we decided to use a divide and conquer approach to implementation. The basic breakdown of work started with Minhaz and Sothiara acquiring the data from Twitter and writing the relevant data into tab-separated data (TSV) files. They also wrote the configuration files for Flume and Avro to load these files into HDFS. At the same time, Ben was in charge of writing the Hive scripts that processed the data files and output the results in the format desired by the front end. Finally, Seth and Jamison were tasked with completing the development for the front end.

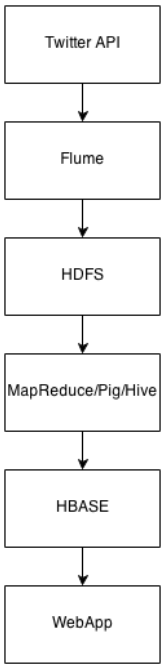
Final Flow Diagram

Initially, the bulk of our work was devoted to researching the project and approaches we could use to accomplish our task as a group. We then broke up to take care of our individual parts while providing support to each other when necessary. We found that dividing up the work in this manner allowed each member of the team to take ownership of part of the project, while also allowing us to receive any support we need from other members of the team.

For this project, we were given access to five nodes on Amazon Web Services (AWS) through the class. We also used one additional machine, hosted on Digital Ocean, which serves as a web server to host our front end. To capture the Twitter stream, each member in our team registered with Twitter and generated two sets of streaming API keys. Using a Java program that utilizes the Twitter API, we make a call to Twitter to start receiving a live-stream of tweets and their corresponding data, restricted by a specific geographic location. We then extracted the relevant information and labeled each tweet with its geographic location. This was then written into a TSV file in a Flume spooling directory, which passes it through Avro into HDFS.

Once the data was in HDFS, we used two hive scripts to process the data in three steps. First, we loaded the data into a hive table. Next, we counted the number of occurrences of each word in the text of the tweets. Finally, we found the most common hashtag in each region and wrote this out to a TSV file.

From HDFS, the TSV file is uploaded to a web server hosted by Digital Ocean. When the user opens up the web page, aptly named index.php, there is a combination of HTML, JavaScript, and PHP code that populates pins on a Google map. After the TSV file is on the server, helper.php, which is called in index.php, parses through the data in the file. This data is then sent over to index.php, which then loads the values into the Google map. This is done by reading in the latitude, longitude, name, and most popular hashtag of each location. A new marker is made at each location with the corresponding data.



Milestone 2 Flow Diagram

Challenges

We had a variety of struggles in our implementation. First, Twitter rate limits our data stream, which limited the amount of data we could collect. On top of that, most of the tweets did not contain both location data and hashtags. The lack of this

critical information renders the tweet useless for our purposes, so we were forced to ignore a decent amount of the data we were able to collect. Furthermore, the text of the tweets occasionally contained non-ASCII character sets, which we then filtered out.

We also ran into issues with respect to some of the enhancements we wanted to make to the system. One such feature we originally wanted to include was a sentiment analysis of the tweets. However, due to the particular vernacular used on Twitter, we were not confident in the ability of a sentiment analysis program to accurately categorize Tweets. Ultimately, we ended up implementing sentiment analysis, but decided not to include it in our final project because we were not able to address these concerns. The code is included in our submission.

Additionally, we encountered difficulties with using HBase. Our initial plan was to use HBase to store our results as key-value pairs that map cities and their coordinates to trending hashtags. However, we ran into our most difficult obstacle when we tried to connect our web interface to HBase. There are two technologies that HBase implements for communication if the client isn't Java or the HBase shell. These are the Thrift and Representational State Transfer (REST) libraries. We could not get the Thrift library to start properly, and the REST API, called Stargate, did not produce responses in the manner that is demonstrated in the documentation. In the end, we decided to not use HBase. Instead, we decided to have Apache Hive create a TSV file on HDFS which we then moved to the web server.

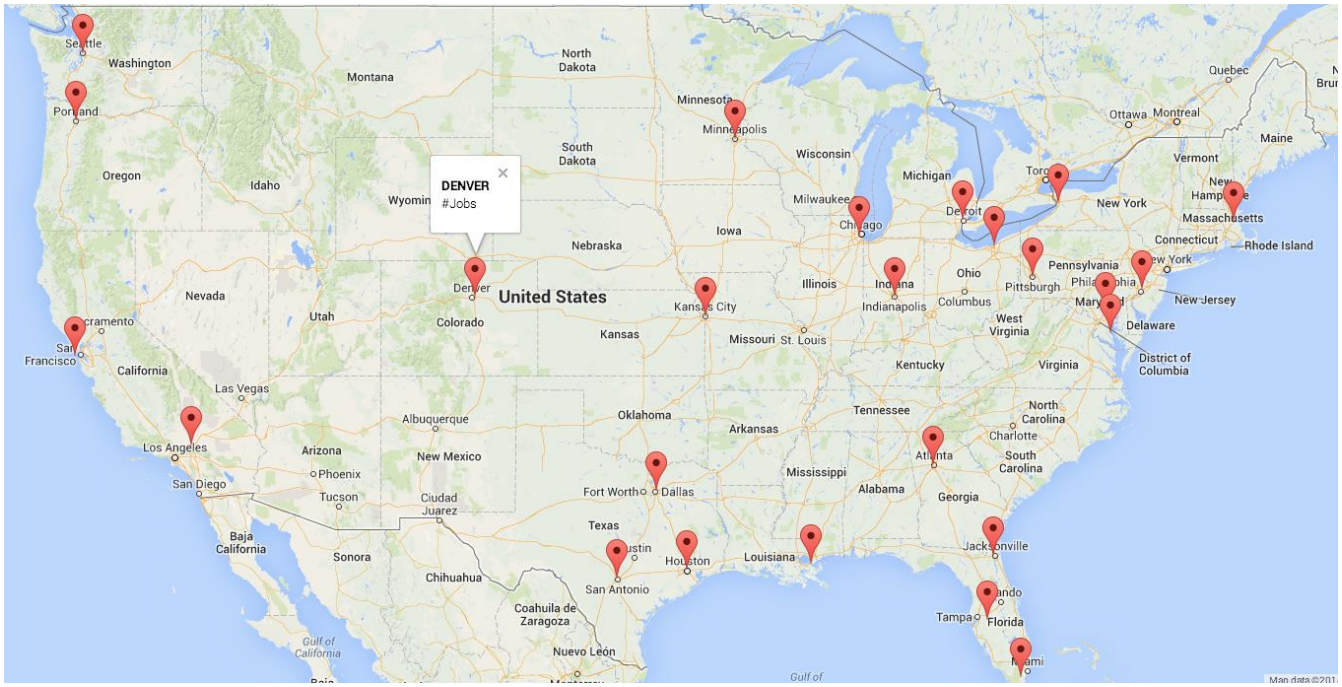
Results

We downloaded tweets from various major cities around the United States. We discarded some parts of the tweet that were unnecessary so the analysis would not take as long. The Hive scripts compute which hashtags are most common, as well as their regions, and exports them in a TSV file that is later sent to the web server. We were able to deduce which hashtags are trending at any location given by a latitude/longitude bounding box. For example, Denver's most common hashtag was #jobs, while San Francisco's most popular hashtag was #coachella. Some of the hashtags were completely expected based on current events. However, one odd one was #NashTo2Mil, which baffled us. Who was this Nash? Why was he trying to get 2Mil? After some research, we discovered that it was a teenager that makes Vine videos who was trying to get to two million Twitter followers. It was trending in Los Angeles and the UMBC area. Our system is set up in a manner such that a bounding box can be specified for any region to record the tweets in that area to see what's trending. Having information about what people are talking about on Twitter in specific areas could be useful for marketing, campaigning and other concepts that may make efficient use of this type of information. Marketing strategists could look at different locations during an event to see how people are reacting to it in real time to then determine where to focus their attention. A political campaign could also use this data to see how specific regions are reacting to a debate or advertisement. They can then use this data to optimize their campaign efforts by focusing on areas that don't yet have a positive view on the candidates. Another possible use for this would be for someone that is thinking about moving to an area but is unsure about the people in said area. It is easy to find information about neighborhoods on the internet, as well as crime

statistics, but it is often extremely difficult, if not impossible, to find information about the attitudes and opinions of people in an area.

The results are displayed in the map below. The map has markers for each geographic location that we have collected tweets for. Mousing over a marker will display the name of the region and the hashtag that was most popular at that time.

You can see it at: http://www.minhazm.com/hadoop/index.php



Example Localized Trends Map Centered On United States

Future work and improvements

Due to the time constraint and the scope of the project, we did not implement several features that utilize more of the tweets' metadata. As mentioned earlier, if we can accurately measure the sentiment of the tweets and also determine the subject of the statements, we could measure relative "happiness" or "mood" of a geographic location. For instance, with sufficient data, we can color the U.S. states or region of the states with a color scheme that rates the population's "happiness". This data can then be used to draw correlations to the region's socio-economic health by looking at additional data from the U.S. Census Bureau. This feature would be useful in determining property investment, vacation spots, or primary targeting points for businesses.

In addition to sentiment analysis, we can also perform graph analysis on the hashtag(s) of tweets. Given that tweets can contain multiple hashtags, we can use Map-Reduce and Mahout to

construct a relationship graph linking the hashtags. Doing so will show the correlation of trending hashtags with other hashtags, which will give us a better insight on how trends evolve. This data could be used in advertising in order to directly target customers that share similar interests based on the products they have previously purchased.

Additionally, we can collect more tweet data from more or larger geographic areas. Clearly, having more data would provide a more accurate sample of the population. This would allow us to make better predictions and draw better connections. Furthermore, we can try to find a better sentiment analysis API that specifically trains toward twitter data or a similar source.

Bibliography

Twitter Java Library:

http://twitter4j.org/en/index.html

JSON Library:
https://code.google.com/p/json-simple/

Apache Thrift:

https://thrift.apache.org/

HBase Rest API:

http://hbase.apache.org/book/rest.html

Flume:

http://flume.apache.org/

Avro:

http://avro.apache.org/

Hive:

http://hive.apache.org/

PHP:

http://www.php.net/