CS 171: Project 2

Pheme: Birds of a Feather Tweet Together

Authors:
Carl Jackson and Peter Zhang

April 8, 2013

Contents

1 Introduction

1.1 Motivation

Most current visualizations of social data require viewers to already have a subject matter in mind; they allow viewers to search for and investigate specific keywords, hashtags, or locales. However, we think that social data is interesting primarily because it allows us to discover new topics for investigation - topics that are being generated in real time by the humans who are producing the social data. Accordingly, we want to build a visualization that guides a curious but unfocused viewer in identifying undiscovered topics/events that are currently occurring in the real world.

1.2 Goals

In this project, our goal is to investigate high-throughput social data streams, and answer two primary research questions:

1. Can we identify ”events” from real-time social data?
2. If we can, what can we learn about such ”events” from various social data sources?

To achieve these goals, we have built a visualization that tries to identify current events that are occurring within a user-specified geography (location + radius), using real time social data. Concretely, we pull a live stream of Twitter data being generated in the specified geography, and try to identify clusters of tweets; currently, we cluster primarily by geographical proximity, but hope to cluster using other techniques in project 3.

2 Description of Data

2.1 Source

The ultimate source of the data for our visualizations is Twitter, an online social network and microblogging service that allows users to instantaneously send and receive short messages, called tweets, on various electronic devices. Twitter is one of the most popular, if not the most popular, sites that produces real-time social data; as of 2012, it produced more than 340 million tweets per day [1]. Given our interest high-throughput social data streams,

[1] http://techcrunch.com/2012/07/30/analyst-twitter-passed-500m-users-in-june-2012-140m-of-them-in-us-jakarta-biggest-tweeting-city/

Twitter was a natural data source to focus on.

Using Twitter's API, we pulled tweets, which contained several relevant pieces of information:

1. Geographical location: Twitter users can choose to specify a geographical location to associate with their tweet. This is particularly interesting for users who tweet on their mobile phones, and choose to expose their phone's GPS data to the service.

2. Twitter Screenname: Tweets are associated with the screen name of the user publishing the tweets.

3. Tweet contents: The actual text content of the tweet. This is also available to viewers of our visualization.

4. Hashtags: Twitter users can specify hashtags, or metadata, to associate with their tweet.

5. Tweet timestamp: The time at which the tweet was published.

2.2 Scraping Method

For Carlllllll. Tell story like this:

1. Normal twitter stream is all historical.

2. Want realtime twitter data.

3. Lots of realtime twitter data doesn't have geographical data, which is essential; how did you solve?

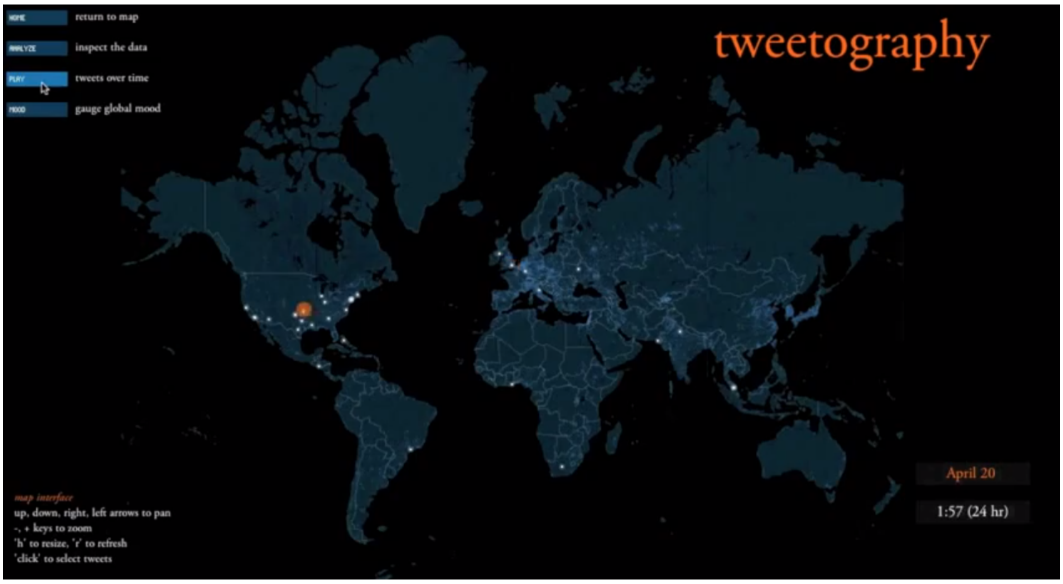
4. Challenge where you could only pull from certain locations.

5. Fix of that bug.

6. Any other challenges you can think of.

3 Related Work

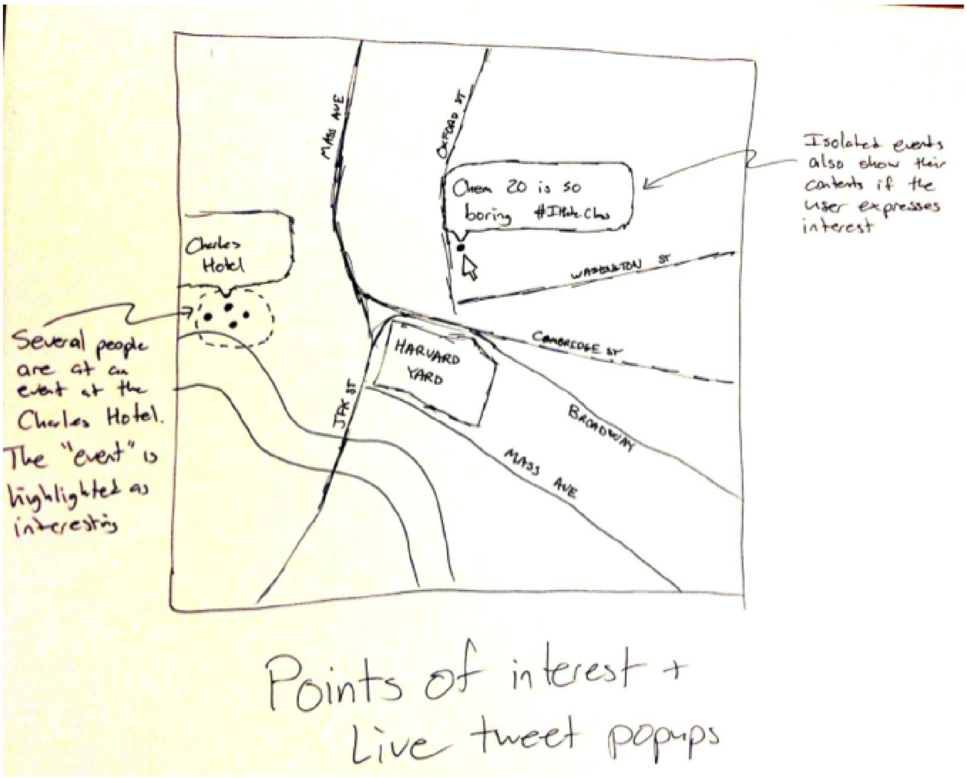
To the extent that we've used what we've learned so far in CS 171, all the readings and lecture materials are related work; when justifying our visualization choices or visualization evolution, we will refer to this material. We consulted documentation related to the Twitter API and the Google Maps API very often while working on this project. Finally, we were inspired to investigate and study tweets and their geographical data by a past CS 171 project: Tweetography.



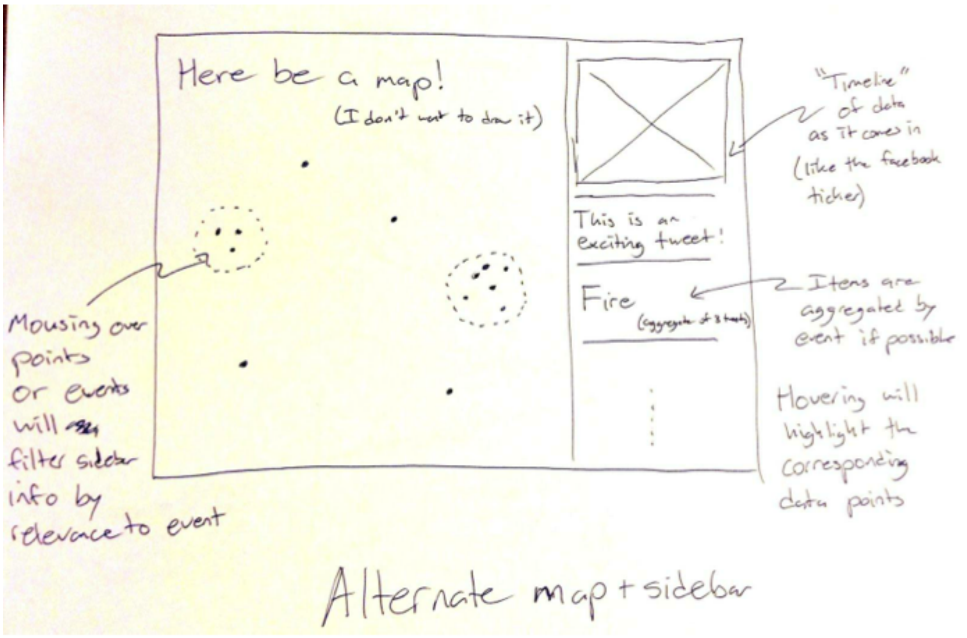
We found the concept of tweets appearing over time on a map very attractive, and based our visualization on this concept, even though the purpose and design of our visualization is very different than Tweetography.

4 Design Evolution

In our initial proposal, we envisioned a visualization where tweets, and clusters of tweets that have been identified as a cluster, are displayed on a map as they are generated; our initial sketch of how this might look is included below:


MASS AVE
OXFORD ST
Chem 20 is so boring #IHateChem
WASHINGTON ST
Charles Hotel
CAMBRIDGE ST
HARVARD YARD
JFK ST
BROADWAY
MASS AVE
Isolated events also show their contents if the user expresses interest
Several people are at an event at the Charles Hotel. The "event" is highlighted as interesting
Points of interest + Live tweet popups

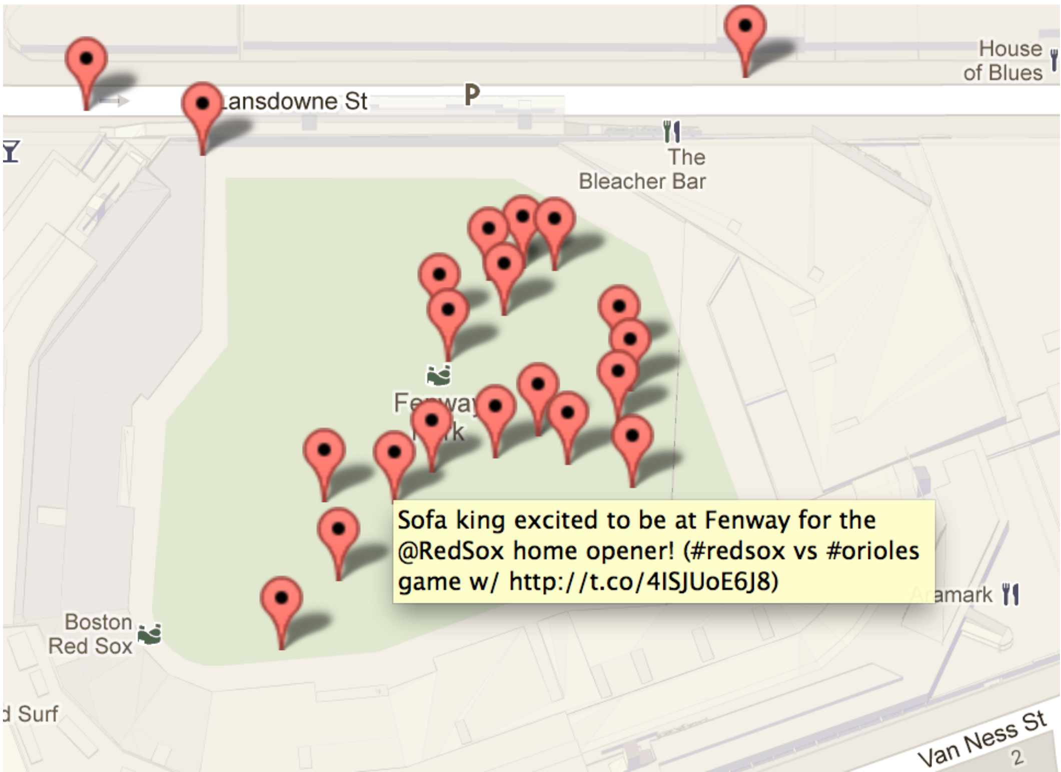
In addition, we imagined a sidebar that would display tweet and cluster information as they came in, similar to the image below:


Here be a map! (I don't want to draw it)
This is an exciting tweet!
Fire (aggregate of 3 tweets)
"Timeline" of data as it comes in (like the facebook ticker)
Items are aggregated by event if possible
Hovering will highlight the corresponding data points
Mousing over points or events will filter sidebar info by relevance to event
Alternate map + sidebar

Throught the evolution of our visualization, these fundamental design choices have not changed. However, we did have to address specific design issues during this project.

4.1 Maps

First, we had to decide on the background of our visualization, the map. As covered in class and homework, there are several ways to draw maps using D3, including an SVG map from wikipedia or the datamaps add-on. We really liked D3, especially the way it binds data directly to elements of the DOM. However, we ultimately decided that D3 was not particularly suitable for our visualization, because we needed a very finely detailed map for our visualization. Since we desired to identify events and their geographical location, it was important for users to be able to see the exact street address of the tweets and events we were visualizating, as well as the surrounding buildings and other geographical features. For example, one event we ultimately identified while testing our visualization was the opening game at Fenway Park:



Sofa king excited to be at Fenway for the @RedSox home opener! (#redsox vs #orioles game w/ http://t.co/4lSJUoE6J8)

If the map we chose to visualize tweets on did not contain detailed city, street, and building information, it would have been much harder to understand exactly what and where this event was occurring. Accordingly, we decided to build our visualization using the Google Maps API. In addition to a very detailed map, which would help viewers make a variety of useful visual queries relating to the events we identify, the Google Maps API contains several nice features, including zooming - which will allow users to drill down/filter as we explain later - and simple methods to draw basic objects on the map.

4.2 Clustering

We faced two challenges when deciding how to construct and visualize our clusters.

4.2.1 Clustering Algorithm

Our first task was to decide how to actually identify clusters of tweets; we knew that clusters would be based on geographical distance, but needed to decide on a specific clustering algorithm. Most traditional clustering algorithms, like k-means clustering, hierarchical clustering, and autoclass clustering, do not fit our visualization goal. Not only do they depend on the assumption that all the points to be clustered are available immediately - which is not true in our case, because tweets arrive in realtime in our visualization - they also try to fit every point to a cluster - which is not what we desire, because we only want to cluster points that we believe are close enough to actually be an event. Accordingly, we designed the following simple algorithm for clustering:

1.

4.2.2 Visualizing Clusters

Once we decided upon the algorithm by which we would identify clusters, we had to decide how to visualize them. Initially, we had decided

5 Visualizations

5.1 Colors

6 Conclusion

Through this process of Exploratory Data Analysis, I've identified several interesting relationships between game usage statistics and various game attributes, as well as relationships between different game attributes. That being said, if I had more time, I would be interested in further investigating this data in the following ways:

1. Examine the relationship between hourly game usage statistics and price, language count, ESRB rating, metacritic rating, etc.

2. Identify any differences between hourly game usage statistics on week and weekend days.

3. Scrape data for games other than games in the top 100 games by current user count, to get a richer picture of how various game attributes affect to each other.

Furthermore, number 3 on this list emphasizes a general weakness with the data I used: I only collected data for the top 100 games by current user count at the current date and time. This data collection method likely introduced some artifacts into the data, like the dropoff of current users playing racing games to zero at certain times of day, and thus all results presented above should be regarded with caution; further investigation is needed to confirm these trends.