Twitter trends similarity analysis based on users

Fahad Shaon (fxs121530)
CS, UTD
Email: fahad.shaon@utdallas.edu

Harichandan Roy (hxr121830)
CS, UTD
Email: harichandan.roy@utdallas.edu

***Abstract*—Now a days, people are sharing their views on different topics in social medias like Twitter, Facebook, Google+ etc. In this project, we are trying to find out the similarity of users set based on GeoLocation, who twitted trending topics. The question we tried to analyze - "Based on GeoLocation is it the same user set, who are making a topic trending or not?" The user interface for this project- http://fahad-sed.utdallas.edu:25001/**

I. Introduction

Social medias e.g. Twitter, Facebook, Google+ are getting popular day by day. People are sharing their views, making comments on different topics. In our project, we worked on Twitter's public API. The main idea is to find out the similarity of users set those are tweeting on various trending topics. High similarity between two topics could mean a lot of things. Firstly it could mean topics are originated from single source. Secondly, it also could mean same type of users will like these two topics, this can directly be used for recommendation. For simplicity in this project we only focus on finding the similarity matrix.

II. Theory

A. User set

User set of a topic means sets of users who twitted a trending topic.

B. Similarity Measure

For similarity measure of two topic's user set, we use **Jaccard coefficient**. Jaccard coefficient $J(A, B)$ of two sets A and B is computed by

$$J(A, B) = \frac{|A \cap B|}{|A \cup B|} \tag{1}$$

III. System Architecture

System has two main components

- Data Fetching and Computation Module - Figure **??** summarizes this module
- User Interface

IV. Work Flow

The steps are following-
1. Fetching the locations that Twitter has trending topic information.
2. Fetching the trending topics from those locations.
3. Finding the similarity of the users set for the trending topics.

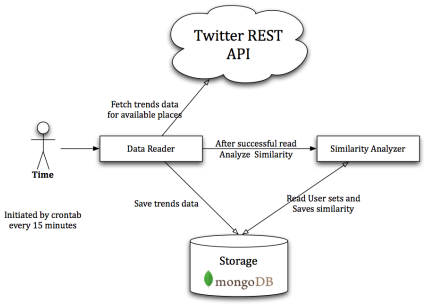

Fig. 1. Data Fetching and Computation Module

A. Step 1: Fetching the locations that Twitter has trending topic information

In this step, we fetched the locations. Twitter API provides the location information in $JSON$ format. Figure 3 shows the $JSON$ object for the locations. Locations from United States are only fetched. Name along with other useful fields are retrieved for next steps.


```
{
  "country": "United States",
  "countryCode": "US",
  "name": "New Haven",
  "parentid": 23424977,
  "placeType": {
    "code": 7,
    "name": "Town"
  },
  "url": "http://where.yahooapis.com/v1/place/2458410",
  "woeid": 2458410
},
{
  "country": "Japan",
  "countryCode": "JP",
  "name": "Sapporo",
  "parentid": 23424856,
  "placeType": {
    "code": 7,
    "name": "Town"
  },
  "url": "http://where.yahooapis.com/v1/place/1118108",
  "woeid": 1118108
},
```


Fig. 2. JSON object for location information

Figure 4 shows the list of location fetched in this step.

B. Step 2 : Fetching the trending topics from those locations

In this step, we fetched the trending topics trending on the locations. Similar to step 1, Twitter API provides the trend topics' information in $JSON$ format. Figure 5 shows the $JSON$ object for the trending topics.

Figure 6 shows the list of trending topics found in the step 2.

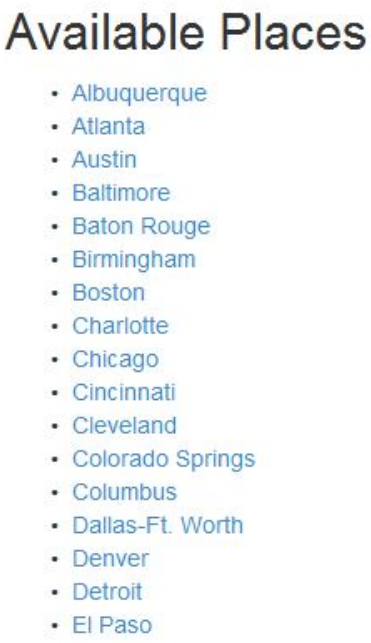

Fig. 3. Locations

Fig. 4. json object for trending topics

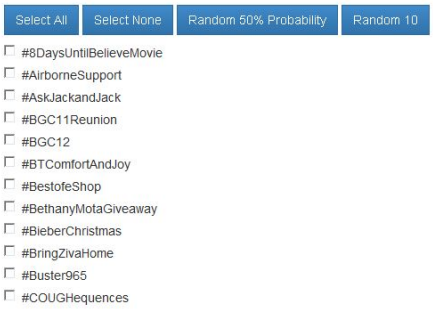

Fig. 5. Trending topics

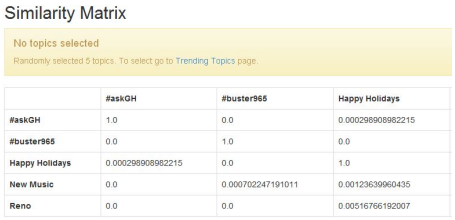

Fig. 6. Similarity of users set on trending topics

C. Step 3 : Finding the similarity of the users set for the trending topics

In this step, given a set of trending topics we calculated the similarity of the users set and displayed in a matrix form. Figure 7 shows the result after this step.

V. Data Model

We used MongoDB [1] for storing our data. We saved *places*, *trends*, *topic_users*, and *similarity* informations for our analysis. Following are the MongoDB schemes for these collections.

place collection - each document represents an available place for trend query.

```
{
    "latitude" : ..,
    "longitude" : ..,
    "name" : "...",
    "woeid" : 2391279
    ...
}
```

trends collection - each document represents a specific place's trending topics at a particular time.

```
{
    "woeid" : 2391279,
    "created_at" : "2013-12-17T19:03:34Z",
    "locations" : [
        {
            "woeid" : 2391279,
            "name" : "Denver"
        }
    ],
    "trends" : [
        {
            ...
            "query" : "Santa",
            "name" : "Santa"
            ...
        },

        ... // more trending topics \\

    ]
}
```

topic_users collection - each document represents a set of users who have twitted about a topic.

```
{
    "topic" : "#HamptonHoliday",
    "users" : [
        {
            "id" : 16753440,
            "name" : "Natalie",
            "screen_name" : "natalie_e_s"
        },
        ... // more user information \\
    ]
}
```

similarity collection - each document represents the Jaccard similarity in user set of two topics. This is the pre computed table for fast response to user.

```
{
    "topic1" : "#AirborneSupport",
    "topic2" : "#HolidayReady",
    "jaccard" : 0.02040816326530612
}
```

VI. Technology Used

Table I summarizes used technologies.

TABLE I. Used Technologies

Technology or Libraries	Utility
MongoDB [1]	Store Data
Python [2]	Analyze and user interface logic
OAuth2 [3]	Authentication before fetching Twitter REST API
urllib [4]	Fetch Twitter and Yahoo GeoPlanet REST API
PyMongo [5]	MongoDB python driver, to query MongoDB
Flask [6]	Simple web framework for building web based interface

VII. Conclusion

In our project, we observed that some trending topics are tweeted by same common users. In that case we got the similarity non-zero. These similarity may have different implications. High similarity might simply be the case that two distinct topics are really same topic or have some origin. It also can be used for recommendations.

References

[1] *http://www.mongodb.org/*

[2] *http://www.python.org/*

[3] *https://pypi.python.org/pypi/python-oauth2/*

[4] *http://docs.python.org/2/library/urllib.html*

[5] *http://api.mongodb.org/python/current/*

[6] *http://flask.pocoo.org/*