Topik: Top-k Data Search in Geo-tagged Social Media

Saurabh Deochake
Rutgers University
Piscataway, NJ, USA
Email: saurabh.deochake@rutgers.edu

Saurabh Singh
Rutgers University
Piscataway, NJ, USA
Email: s.singh@rutgers.edu

Shreepad Patil
Rutgers University
Piscataway, NJ, USA
Email: shreepad.patil@rutgers.edu

***Abstract*— Increasing popularity of social media platforms and location based services has made it possible to query the data generated by social media interactions in novel ways other than traditional search. With advent of Data Science techniques, data scientists can delve deep into the data generated by social media and inspect it to find different ways the users interact on social media and in turn, Internet. In this project, we will make use of data generated by social media interaction for example, tweets, and/or geo-tagged instragram photos, and/or facebook posts on a particular topic from a specific region and bring about a top-k Social media data(User/Tweets/Posts). Given a location l, in the form of (Latitude,Longitude), distance d, and a set of keywords K, the algorithm used in our module will fetch top-k users who have posted media relevant to the desired keyword k *subset* K in a proximity to location l(Lat,Long) within distance d. We congregate, analyze and represent the data to end user. Finally, we will conduct an experimental study using real social media data sets such as Twitter to evaluate our module in development.**

I. Project Description

Social media is having more and more effect on our daily lives. Every second around 6000 tweets are recorded on twitter. In general, people use social media to keep in touch with their friends and stay updated on the social events.

Nowadays, with the widespread use of GPS-enabled mobile devices, we also get data regarding locations of the users. Geo-tagged data plays an important role in analyzing the social media data and scenarios like friend recommendation, spatial decision, etc. For example, Facebook activates the Safety Check feature during a disaster, which helps marking yourself as safe during a disaster. Also many location based services are becoming popular these days.

Traditional search, looks for the keywords in a document, twitter also uses a search and fetches the trending topics, but it lacks the geo-spatial component. With geo-spatial search it is even possible to provide novel search services.

Our approach is to recommend Twitter users by taking into account users' tweet contents, locations, and tweets' popularity that indicates the users' social influence. In particular, we study the search of top-k local users in tweets that are associated with latitude/longitude information, namely geo-tagged tweets. To that end, we devise a scalable architecture for hybrid index construction in order to handle high-volume geo-tagged tweets, and we propose functions to score tweets as well as methods to rank users that integrate social relationships, keyword relevance and spatial proximity for processing such social search.

Even though this is a vast topic of research and development, with a group of three members, this project can be built in around two months. Some of major stumbling blocks that may occur as the project advances in development and testing is a) how should we store huge geo-spatial data generated by analyzing and processing the social media data. b) representing the processed data in graphical form well enough understood by a computer science novice. As assumed above, the project shall be completed within two months (before 1 May 2016). We plan to reach requirement gathering and project design phase by end of March 2016 and we will start with infrastructure development and backend development in first week of April 2016 until 25 April 2016. The testing phase will run from 25 April 2016 to 1 May 2016 carrying out all the test cases designed.

The project has four stages: Gathering, Design, Infrastructure Implementation, and Testing.

A. Stage1 - The Requirement Gathering Stage.

This section describes in details the work done in the requirement gathering phase. We start by describing the proposed system and the types of users who will work on the system. We then move on to the real-world scenarios and the planned project phases.

- **System Description:**
  This project will fetch top-k social media users from vast data pool captured via social media platforms like Twitter. This will be a Flask-based web project which has a UI developed in Bootstrap and Javascript. The backend will be implemented in PostgreSQL and project's pivotal functionality will be implemented in Python.
- **Types of Users:**
  There are two types of users, which are described in detail as follows:
  1) End User: End user is the one who will input a keyword into the system and the desired location in the form of (lat,long). This user will have limited acess to the system which includes the input given by it to the system and output generated by the query provided.
  2) Super User: The Super User mode will be suitable for any user who wants to control the behaviour of the system. This shall include inspecting the log files of the whole system in conjunction with important data

logs such as PostgreSQL's WAL (Write Ahead Logging) XLOG records. The super user mode will give the user power to further inspect the data values generated by the system and perform data science, data mining and pattern recognition operations.

- **User Interaction Mode:**
  The end user will be able to interact with the system via a web-based portal designed using above mentioned technologies in System Description section.
- **Real world Scenarions:**
  The real world scenarios are described below aggregated by user types:

  Type: End User

  Scenario: 1
  Scenario Description: Top-k people/social media posts in a locality about a query
  System Data Input: A query- "People Talking about Donald Trump" and lattitude, longitude of a desired area
  Input Data Type: String and a Tuple
  System Data Output: List of users and tweets/social media posts about the query in the area
  Output Data Type: List and JSON Strings

  Scenario: 2
  Scenario Description: Query-targeted advertisements in a locality
  System Data Input: A query- "Burrito" and latitude, longitude of a desired area
  Input Data Type: String and a Tuple System Data Output: List of users and tweets/social media posts about the query in the area
  Output Data Type: List and JSON Strings

  Type: Super User

  Scenario: 1
  Scenario Description: Inspecting Log file of PostgreSQL
  System Data Input: Click on the "button" input type on the form
  Input Data Type: HTML Code
  System Data Output: Opening the Log file on the console
  Output Data Type: HTML Code

  Scenario: 2
  Scenario Description: Managing the session and API keys
  System Data Input: Click on the "button" input type on the form
  Input Data Type: HTML Code
  System Data Output: List session and API keys used by the system
  Output Data Type: HTML Code

- **Project Timeline:**
  Stage 1 viz., requirement gathering shall be finished by around 21 March 2016 followed by project design which shall be completed by 1 April 2016. We will start development of infrastructure and back-end on 1 April 2016 and shall be over by 25 April 2016. From 25 April 2016 to 1 May 2016, we will thoroughly test the system against the test cases developed by one of our team members.
  The planned division of labor is as below:

  Saurabh Deochake: Requirement Gathering, Project Design, Infrastructure Implementation including front-end and back-end, Documentation.

  Shreepad Patil: Requirement Gathering, Project Design, Infrastructure Implementation including front-end and back-end, Documentation.

  Saurabh Singh: Requirement Gathering, Project Design, Front-end Infrastructure development, Testing.

B. Stage2 - The Design Stage.

This section explains the architecture of the project and talks about the algorithm and data structure to be used in the implementation of the project. We also talk about the constraints in the projects

- **Project Description:** We describe the main components of the project here. The architecture is also illustrated by the $Figure\ 1$.
  Twitter API: We use twitter streaming API to access the tweets from twitter. We use a filter on the stream to filter data relevant to our project. The pseudo code for the Filter is shown in $Algorithm\ 1$.
  Parser: Once we acquire the filtered stream, we parse the stream to get the posts from it. We use stop words and stem the words in the parser. The post is stored in the format $p(uid, t, l, W)$, where uid is User ID, t is timestamp, l is location, and W is set of words in the tweet. We store the acquired posts into a database. The pseudo code for this component is shown in $Algorithm\ 2$.
  Graph Generator: The Graph Generator then converts these posts to a graph. The graph is stored as $G(U, E)$, where U is the set of users, E is an set of edges, $(u, v) \in E$ if u has 'replied to' or 'forwarded' v's post. This component would append vertices and edges to an existing graph structure. The graph will stored in adjacency list. The pseudo code for this component is shown in $Algorithm\ 3$.
  Analyzer: This component is the core component of the architecture. The analyzer receives the query from the query processor. Internally, the analyzer finds all the posts that were posted in the query radius and assign a distance score to these posts.

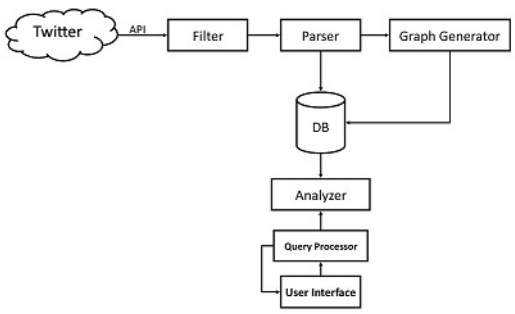

Figure 1. Proposed architecture for Topik

Another score is also calculated for the posts, we call it the keyword relevance score. This score is the function of tweet popularity and the number of matching words in the post and query. The tweet popularity is based on the number of replies and retweets of the post. The pseudo code for this component is shown in $Algorithm\ 4$.

Query Processor: We plan to use a top-k query on an aggregate score. The aggregate score is a weighted combination of distance score and keyword relevance score. As a response to the query the analyzer will respond with the top-k Users relevant to the query. The algorithm is briefly described in $Algorithm\ 5$.

User Interface: The user interface will be taking user input in the form of $q(W, l, r)$, where W is set of words, l is location and r is the distance. Once the user submits the query, it passes the query to the query processor and displays the output to the user.

The time complexity of the algorithms is linearithmic in the number of posts. However, the number of posts is expected to be very high. We use a notation $O(nlogn)$ to denote the time complexity of the algorithms, where n is number of posts.

Algorithm 1: GetStream

Data: $Filter(F)$ A broad enough filter to acquire social media feed from API to generate and store data.
Result: Subset of posts from the feed.
begin
  $Q \longleftarrow$ empty data structure to get posts to pass to parser
  $p(uid, t, l, W) \longleftarrow getCurrentTweet()$
  if *p satisfies F* **then**
    return $p(uid, t, l, W)$
  else
    return $\emptyset$
  end
end

- **Flow Diagram Major Constraints:** We describe the major constraints that are will be tackled here
  - **Twitter Rate limit:** We are limited by the Twitter API for fetching the streaming data
    Rate limits are divided into 15 minute intervals. Additionally, all endpoints require authentication, so there is no concept of unauthenticated calls and rate limits.
    Clients which do not implement backoff and attempt to reconnect as often as possible will have their connections rate limited for a small number of minutes.

Algorithm 2: ParsePost

Data: tweet t from $getStream()$ after applying a filter.
Result: Refined Tweet $p'(uid, t, l, W')$.
begin
  for $\forall i \in$ *tweet words* W **do**
    if $!IsRelevant(i)$ **then**
      $W \leftarrow W - i$
    end
  end
  return $p'(uid, t, l, W)$
end

Algorithm 3: GraphGenerator

Data: p-refined tweet post from parser, G: If the graph already exists, the algorithm will append to it.
Result: An updated graph $G(U, E)$.
begin
  for $\forall u \in U$ **do**
    if *($p.user$ replied or forwarded u's post)* **then**
      $E \leftarrow (u, v)$
    end
  end
  return G
end

Algorithm 4: Analyzer and QueryProcessor

Data: k- number of top posts needed in the result,
radius-radial distance of the posts to be analyzed from user's location,
W- set of key words provided as input by user,
thresh- Threshold score above which posts is inserted into result set.
Result: A sorted set A_k containing top k results.
begin
  $CalculateDistanceScore(radius)$
  $CalculateKeywordScore(W)$
  $L1 \leftarrow SortByDistScore()$
  $L2 \leftarrow SortByKeywordScore()$
  Run $Top - k()$
  return Top-k results
end

Algorithm 5: Top-k Query Processing using Threshold Algorithm

Data: Sorted Lists L_1 and L_2
Result: Top-k users related to the query
begin
(1) Do sorted access in parallel to each of the 2 sorted lists. As a new object o is seen under sorted access in some list, do random access (calculate the other score) to the other list to find $p_i(o)$ in other list L_i . Compute the score $F(o) = F(p_1, p_2)$ of object o. If this score is among the k highest scores seen so far, then remember object o and its score $F(o)$ (ties are broken arbitrarily, so that only k objects and their scores are remembered at any time).
(2) For each list L_i , let pi be the score of the last object seen under sorted access. Define the threshold value T to be $F(p1, p2)$. As soon as at least k objects have been seen with scores at least equal to T, halt.
(3) Let A_k be a set containing the k seen objects with the highest scores. The output is the sorted set $\{(o, F(o))|o \in A_k\}$.
end

Rate limited clients will receive HTTP 420 responses for all connection requests.
Clients which break a connection and then reconnect frequently (to change query parameters, for example) run the risk of being rate limited.

- **Size constraints:** The data gathered from the social media is humongous. But we will be working on local machines with limited storage capacities, so we would be storing limited data and applying constraints on the streaming data fetched.

C. Stage3 - The Implementation Stage.

Specify the language and programming environemnt you used for your implementation. The deliverables for this stage include the following items:

- Sample small data snippet.
- Sample small output
- Working code
- Demo and sample findings
  - Data size: In terms of RAM size; Disk Resident?; Streaming ?;
  - List the most interestng findings in the data if it is a Data Exploration Project. For other project types consult with your project supervisor what the corresponding outcomes shall be. Concentrate on demonstrating the Usefuness and Novelty of your application.

D. Stage4 - User Interface.

Describe a User Interface (UI) to your application along with the related information that will be shown on each interface view (How users will query or navigate the data and view the query or navigation results). The emphasis should be placed on the process a user needs to follow in order to meet a particular information need in a user-friendly manner. The deliverables for this stage include the following items :

- The modes of user interaction with the data (text queries, mouse hovering, and/or mouse clicks ?).
- The error messages that will pop-up when users access and/or updates are denied
- The information messages or results that wil pop-up in response to user interface events.
- The error messages in response to data range constraints violations.
- The interface mechanisms that activate different views in order to facilitate data accesses, according to users' needs.
- Each view created must be justified. Any triggers built upon those views should be explained and justified as well. At least one project view should be created with a justification for its use.

Please insert your deliverables for Stage4 as follows:

- The initial statement to activate your application with the corresponding initial UI screenshot
- Two different sample navigation user paths through the data exemplifying the different modes of interaction and the corresponding screenshots.
- The error messages popping-up when users access and/or updates are denied (along with explanations and examples):
  - The error message:
  - The error message explanation (upon which violation it takes place): Please insert the error message explanation in here.
  - The error message example according to user(s) scenario(s): Please insert the error message example in here.
- The information messages or results that pop-up in response to user interface events.
  - The information message: Please insert the error message in here.
  - The information message explanation and the corresponding event trigger
  - The error message example in response to data range constraints and the coresponding user's scenario Please insert the error message example in here.
- The interface mechanisms that activate different views.
  - The interface mechanism: Please insert the interface mechanism here.

II. Project Highlights.

- Only working applications will be acceptable at project completion. A running demo shoul be presented to your project advisor at a date to be specified after the second midterm. A version of

your application shall be installed in a machine to be specifed later during the semester. Your final submissiom package will also include a final LaTeX report modeled after this document, as well as a Power Point Presentation.

- The presentation (7 to 8 minutes) should include at least the following items (The order of the slides is important):
  1) Title: Project Names (authors and affiliations)
  2) Project Goal
  3) Outline of the presentation
  4) Description
  5) Pictures are essential. Please include Interface snapshots exemplyfing tthe different modes of users's interaction.
  6) Project Stumbling Blocks
  7) Data collection, Flow Diagram, Integrity Constraints
  8) Sample Findings
  9) Future Extensions
  10) Acknowledgements
  11) References and Resources used(libraries, languages, web resources)
  12) Demo(3 minutes)

  Please follow the sample presentation mock up that is posted on Sakai.
- By May 1 your group should have completed the final submission. This includes a presentation (7 to 8 minutes) to your project advisor as well as a convincing demo of your project functionalities (3 minutes): every group member should attend the demo (and presentation) indicating clearly and specifically his/her contribution to the project. This wil allow us to evaluate all students in a consistent and fair manner.
- Thank you, and best of luck!