Tweets as Data: Demonstration of TweeQL and TwitInfo

Adam Marcus, Michael S. Bernstein, Osama Badar,
David R. Karger, Samuel Madden, Robert C. Miller
MIT CSAIL
{marcua, msbernst, badar, karger, madden, rcm}@csail.mit.edu

ABSTRACT

Microblogs such as Twitter are a tremendous repository of user-generated content. Increasingly, we see tweets used as data sources for novel applications such as disaster mapping, brand sentiment analysis, and real-time visualizations. In each scenario, the workflow for processing tweets is ad-hoc, and a lot of unnecessary work goes into repeating common data processing patterns. We introduce TweeQL, a stream query processing language that presents a SQL-like query interface for unstructured tweets to generate structured data for downstream applications. We have built several tools on top of TweeQL, most notably TwitInfo, an event timeline generation and exploration interface that summarizes events as they are discussed on Twitter. Our demonstration will allow the audience to interact with both TweeQL and TwitInfo to convey the value of data embedded in tweets.

Categories and Subject Descriptors

H.4 [**Information Systems Applications**]: Miscellaneous

General Terms

Databases

Keywords

Twitter, Microblog, Visualization, Stream Processing

1. INTRODUCTION

Microblogging platforms such as Twitter have seen tremendous uptake in recent time. Designed for facilitating short broadcast-oriented conversations, Twitter's stream is also a valuable source of data. For example, the tweet stream has been used to map disasters [9], predict movie success [1], and augment realtime event interfaces [4].

Previous applications built on top of the tweet stream have used ad-hoc data processing workflows over the low-level Twitter API, generally reinventing the wheel with each new application. To address this problem, we have built TweeQL, a query language and stream processor designed for tweets. TweeQL shares many properties with traditional stream processing systems, but integrates features designed to handle the unstructured nature of tweets, the social graph in which the tweets operate, and the irregularities involved in processing human-generated text.

As a proof of concept of TweeQL, and as an example of how to meaningfully aggregate data from the tweet stream, we have also built TwitInfo. TwitInfo is a web application that allows users to track a set of keywords (e.g., "obama") and gain a better understanding of the events surrounding those keywords as discussed on Twitter. The heart of TwitInfo is a timeline display that highlights peaks of high tweet activity. A novel algorithm discovers these peaks and labels them meaningfully using text from the tweets. Users can visually explore events through a variety of metadata, such as geolocation, sentiment, and popular URLs.

This demonstration will allow the audience to interact with TweeQL and TwitInfo. Users of the TweeQL interface will be able to issue SQL-like queries to TweeQL and immediately see the results of those queries on the live Twitter stream. TwitInfo users will be invited to explore annotated timelines and visual depictions of events summarized from Twitter. Users will be able to explore pre-generated queries and views in addition to exploring their own creations.

2. TWEEQL

TweeQL[1] provides a SQL-like query interface on top of the Twitter streaming API. The streaming API allows users to issue long-running HTTP requests with keyword, location, or userid filters, and receive most tweets that appear on the stream and match these filters. TweeQL provides windowed select-project-join-aggregate queries over this stream, and facilitates user-defined functions for deeper processing of tweets and tweet text. To best understand TweeQL's design, it helps to understand the challenges in processing the tweet stream:

Unstructured Records. Unlike more structured data sources, the majority of the value in Twitter's stream lies in the unstructured text being processed. TweeQL offers various facilities for extracting structure from unstructured tweets. First, it provides streaming string and regular expression matching on tweet text, to identify tweets of interest and extract fields of interest from the text. Next, it provides a classification framework, used primarily for sentiment analysis, to extract categories from tweets. Finally, it provides UDFs for common web services for deriving structure from unstructured content. As an example, TweeQL provides a UDF for geocoding addresses into latitude/longitude pairs. Another UDF takes tweet text, passes it to OpenCalais (`http://www.opencalais.com/`), and returns named entities mentioned in the text.

An example TweeQL query shows some of these features:

```
SELECT sentiment(text), latitude(loc), longitude(loc)
FROM twitter
WHERE text contains ‘obama’;
```

[1]Available as an open source distribution at `https://github.com/marcua/tweeql`

In this example, we select the classified sentiment from the user-provided tweet text, and extract geocoded latitude and longitude from a free-text user-defined location of all tweets containing the word *obama*.

Uncertain Selectivities. TweeQL users might issue multiple filters that are applicable to the streaming API, but only one filter type can be submitted to the API. For example, a user issuing the query

```
SELECT text
FROM twitter
WHERE text contains 'obama'
  AND location in [bounding box for NYC];
```

wants to see all tweets containing the word *obama* that are tweeted from the New York City area. TweeQL must select between requesting all *obama* tweets, and all *NYC* tweets. TweeQL samples both streams in this case, and selects the filter with the lowest selectivity in order to require the least work in applying the second filter. We are also exploring Eddies-style [2] dynamic operator reodering to adjust to changes in operator selectivity over time.

Uneven Aggregate Groups. When aggregating over human output on a geographic region, traditional windowed result strategies are inadequate. For example, the query

```
SELECT AVG(sentiment(text)),
       floor(latitude(loc)) AS lat,
       floor(longitude(loc)) AS long
FROM twitter
WHERE text contains 'obama'
GROUP BY lat, long
WINDOW 3 hours;
```

would calculate average sentiment in 1°x 1°latitude/longitude regions of tweets containing the term *obama*. The average would be aggregated per region every three hours. This fixed time window is not flexible enough due to the uneven distribution of Twitter users across the planet. For example, Tokyo has many Twitter users, but Cape Town has far fewer, resulting in an oversampled bucket for one, and an undersampled bucket for the other. Basing the window size on tweet count rather than time is also inadequate: aggregating tweets over too long a time period may include old tweets, which are now irrelevant. Instead, we use a construct for windowing that measures *confidence* in the aggregated result, similar to what was done in the CONTROL project [6]. Once a bucket falls within a certain confidence interval for an aggregate, its record is emitted by the grouping operator.

High-latency Operators. We have used *latitude* and *longitude* operators throughout the examples above. These operators make web service API requests to some remote geocoding service, which converts a user-provided location into coordinates. Such requests optimistically take hundreds of milliseconds apiece, but incur little processing cost on behalf of the query processor. Though the operations incur little computational cost, they often bottleneck blocking operators. We employ caching to avoid requests, and batching when an API allows multiple simultaneous requests. We are exploring how to efficiently modify the query executor for necessary requests. One potential design adds asynchronous iteration to the executor as described by Goldman and Widom [5]. This, in combination with a data model that allows partial results as described by Raman and Hellerstein [8] might be a sufficient solution.

3. TWITINFO

TwitInfo [7] is an application written on top of the TweeQL stream processor. TwitInfo is a user interface that summarizes events and people in the news by following what Twitter users say about those topics over time[2]. TwitInfo offers an example of how aggregate data extracted from tweets can be used in a user interface. Other systems, such as Vox Civitas [3], allow similar exploration, but TwitInfo focuses on the streaming nature of tweet data.

3.1 Creating an Event

TwitInfo users define an *event* by specifying a Twitter keyword query. For example, for a soccer game, users might enter search keywords *soccer, football, premierleague,* and team names like *manchester* and *liverpool*. Users give the event a human-readable name like "Soccer: Manchester City vs. Liverpool" as well as an optional time window. When users are done entering the information, TwitInfo saves the event and begins logging tweets matching the query.

3.2 Timeline and Tweets

Once users have created an event, they can monitor the event in realtime by navigating to a web page that TwitInfo creates for the event. The TwitInfo interface (Figure 1) is a dashboard summarizing the event over time. The dashboard displays a timeline for this event, raw tweet text sampled from the event, an overview graph of tweet sentiment, and a map view displaying tweet sentiment and locations.

The event timeline (Figure 1.2) reports tweet activity by volume. The more tweets that match the query during a period of time, the higher the y-axis value on the timeline for that period. When many users are tweeting about a topic (for example, *Obama*), the timeline spikes. TwitInfo's peak detection algorithm is a stateful TweeQL UDF that performs streaming mean deviation detection over the aggregate tweet count. The algorithm identifies these spikes and flags them as peaks in the interface.

Peaks appear as flags in the timeline and appear to the right of the timeline along with automatically-generated key terms that appear frequently in tweets during the peak. For example, in Figure 1.2, TwitInfo automatically tags one of the goals in the soccer game as peak "F" and annotates it on the right with representative terms in the tweets like '3-0' (the new score) and 'Tevez' (the soccer player who scored). Users can perform text search on this list of key terms to locate a specific peak.

The timeline is a way to filter the tweets in the rest of the interface: when the user clicks on a peak, the other interface elements (map, links, tweet list, and sentiment graph) refresh to show only tweets in the time period of that peak.

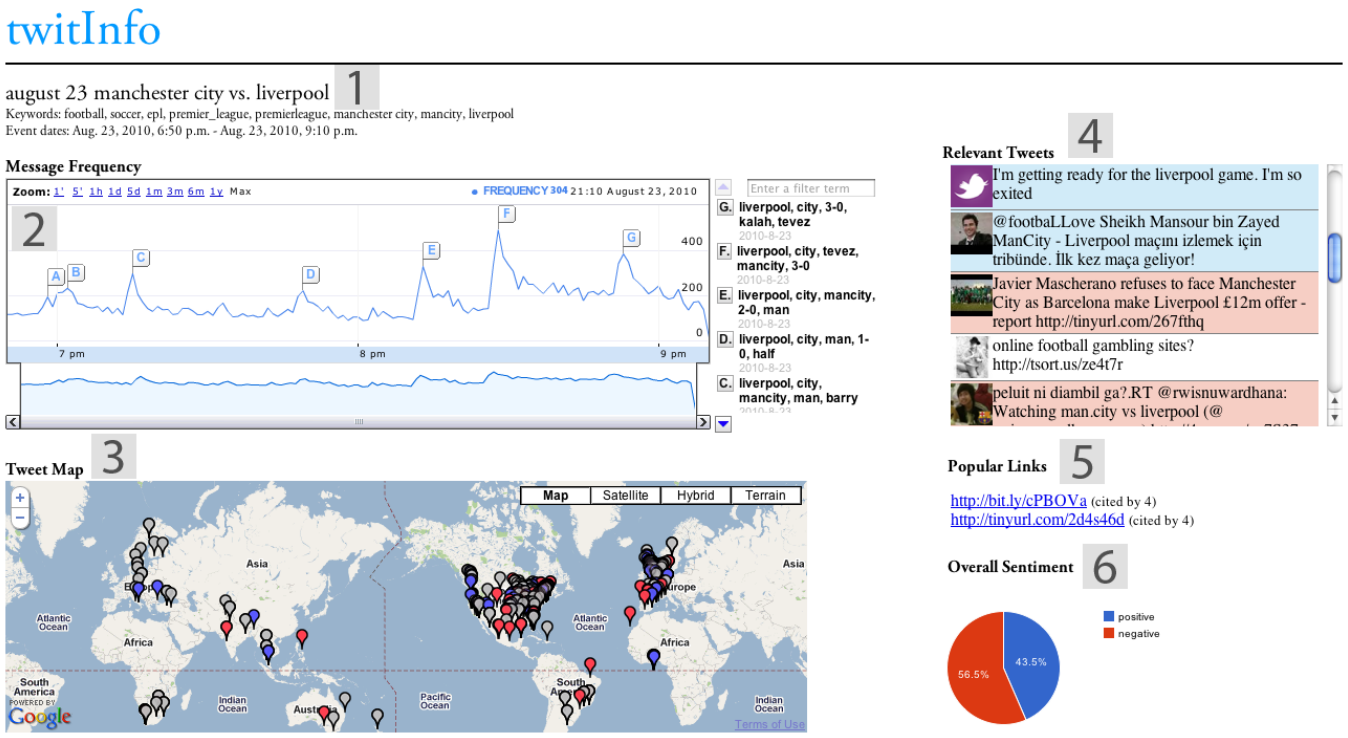
The Relevant Tweets panel (Figure 1.4) lists tweets that fall within the event's time window. These tweets are sorted by similarity to the event or peak keywords, so that tweets near the top are most representative of the selected event. Tweets are colored blue, red, or white depending on whether their detected sentiment is positive, negative, or neutral.

3.3 Aggregate Metadata Views

In addition to skimming sentiment for individual tweets, a user may wish to see the general sentiment on Twitter about a given topic. The Overall Sentiment panel (Figure 1.6) displays a piechart representing the total proportion of positive and negative tweets during the event.

Twitter users share links as a story unfolds. The Popular Links panel (Figure 1.5) aggregates the top three URLs extracted from tweets in the timeframe being explored.

[2]The TwitInfo website with interactive visualizations is accessible at `http://twitinfo.csail.mit.edu/`



august 23 manchester city vs. liverpool

Keywords: football, soccer, epl, premier_league, premierleague, manchester city, mancity, liverpool

Event dates: Aug. 23, 2010, 6:50 p.m. - Aug. 23, 2010, 9:10 p.m.

Message Frequency

G. liverpool, city, 3-0, kalah, tevez
2010-8-23

F. liverpool, city, tevez, mancity, 3-0
2010-8-23

E. liverpool, city, mancity, 2-0, man
2010-8-23

D. liverpool, city, man, 1-0, half
2010-8-23

C. liverpool, city, mancity, man, barry

Tweet Map

Relevant Tweets

I'm getting ready for the liverpool game. I'm so exited

@footbaLLove Sheikh Mansour bin Zayed ManCity - Liverpool maçını izlemek için tribünde. İlk kez maça geliyor!

Javier Mascherano refuses to face Manchester City as Barcelona make Liverpool £12m offer - report http://tinyurl.com/267fthq

online football gambling sites? http://tsort.us/ze4t7r

peluit ni diambil ga?.RT @rwisnuwardhana: Watching man.city vs liverpool (@

Popular Links

http://bit.ly/cPBOVa (cited by 4)
http://tinyurl.com/2d4s46d (cited by 4)

Overall Sentiment

Figure 1: The TwitInfo user interface.

Often, opinion on an event differs by geographic region. A user should be able to quickly zoom in on clusters of activity around New York and Boston during a Red Sox-Yankees baseball game, with sentiment toward a given peak (e.g., a home run) varying by region. The Tweet Map (Figure 1.3) displays tweets that provide geolocation metadata. The marker for each tweet is colored according to its sentiment, and clicking on a pin reveals the associated tweet.

4. DEMONSTRATION

Our demonstration will invite the audience to interact with both TweeQL and TwitInfo. Users will be able to view canned examples or generate their own.

The TweeQL demo will feature a command line query interface that is familiar to most database users. We will offer the audience a selection of pre-built queries, which they can copy and paste into the command line to view live streaming results on their screen. Once audience members are familiar with the SQL-like language, they will be able to generate their own queries of interest, and build their own UDFs for more advanced processing.

The TwitInfo demo will focus on the user interface described above. We will provide three canned examples: a soccer match, a timeline of earthquakes, and a summary of a month in Barack Obama's life. The audience will be invited to explore these pre-built experiences, or track new terms of interest.

5. REFERENCES

[1] S. Asur and B. A. Huberman. Predicting the future with social media. *CoRR*, abs/1003.5699, 2010.

[2] R. Avnur and J. M. Hellerstein. Eddies: Continuously adaptive query processing. In *In SIGMOD 2000*.

[3] N. Diakopoulos et al. Diamonds in the rough: Social media visual analytics for journalistic inquiry. In *VAST 2010*.

[4] N. Diakopoulos and D. A. Shamma. Characterizing debate performance via aggregated twitter sentiment. In *CHI '10*. ACM Press, 2010.

[5] R. Goldman and J. Widom. WSQ/DSQ: a practical approach for combined querying of databases and the web. *SIGMOD Rec.*, 29(2):285–296, 2000.

[6] P. J. Haas and J. M. Hellerstein. Online query processing: a tutorial. In *SIGMOD*, 2001.

[7] A. Marcus et al. Twitinfo: Aggregating and visualizing microblogs for event exploration. In *CHI 2011*.

[8] V. Raman and J. M. Hellerstein. Partial results for online query processing. In *SIGMOD 2002*.

[9] S. Vieweg et al. Microblogging during two natural hazards events: What twitter may contribute to situational awareness. In *CHI 2010*.