TwitterMonitor: Trend Detection over the Twitter Stream

Michael Mathioudakis
Computer Science
University of Toronto
mathiou@cs.toronto.edu

Nick Koudas
Computer Science
University of Toronto
koudas@cs.toronto.edu

ABSTRACT

We present TwitterMonitor, a system that performs trend detection over the Twitter stream. The system identifies emerging topics (i.e. 'trends') on Twitter in real time and provides meaningful analytics that synthesize an accurate description of each topic. Users interact with the system by ordering the identified trends using different criteria and submitting their own description for each trend.

We discuss the motivation for trend detection over social media streams and the challenges that lie therein. We then describe our approach to trend detection, as well as the architecture of TwitterMonitor. Finally, we lay out our demonstration scenario.

1. INTRODUCTION

In recent years, rates of social media activity have reached unprecedented levels. Hundreds of millions of users now participate in online social networks and forums, subscribe to microblogging services or maintain web diaries (blogs).

Twitter[1], in particular, is currently the major microblogging service, with more than 11 million active subscribers. Twitter users generate short text messages — the so-called 'tweets' — to report their current thoughts and actions, comment on breaking news and engage in discussions. The product of social activity on Twitter reaches an estimated total of over 6M tweets per day. Every tweet is associated with an explicit timestamp that declares the exact time it was generated. Moreover, every user has a well-defined profile with personal information (name, location, biographical sketch).

Such a document stream contains a great wealth of information and offers significant opportunities for exploration, as well as challenges. One of the first challenges that comes to mind, and which we try to address with our system, is to automatically detect and analyze the emerging topics (i.e. the 'trends') that appear in the stream and to do so in real time. Trends are typically driven by emerging events, breaking news and general topics that attract the attention of a large fraction of Twitter users. Trend detection is thus of high value to news reporters and analysts, as they might point to fast-evolving news stories. For example, at the announcement of Michael Jackson's death on June 25, 2009, Twitter was immediately flooded with an enormous volume of related commentary. Trend detection is also important for online marketing professionals and opinion tracking companies, as trends point to topics that capture the public's attention. The requirement for real-time trend detection is only natural for a live stream where topics of discussion shift dynamically with time. Furthermore, for such a system to be scalable over massive document streams, an approach is required that makes as few passes over the data as possible.

The prevalence of social media has prompted the development of research-oriented (e.g. [8, 9, 11]) and commercial (e.g. [1, 3, 4, 5, 6, 7]) systems that aim to discover important aspects of social media activity. To the best of our knowledge, our system is the first research-oriented effort towards real time trend detection over the Twitter stream. We envision our highly scalable approach to be extended to cover even more social media streams, e.g. from blogging or social network activity.

In what follows, we describe how TwitterMonitor tackles the challenge of real-time trend detection, as well as its architecture. We conclude with a description of our demonstration scenario.

2. TREND DETECTION AND ANALYSIS

TwitterMonitor performs trend detection in two steps and analyzes trends in a third step. First, it identifies 'bursty' keywords, i.e. keywords that suddenly appear in tweets at an unusually high rate. Subsequently, it groups bursty keywords into trends based on their co-occurrences. In other words, a trend is identified as a set of bursty keywords that occur frequently together in tweets. After a trend is identified, TwitterMonitor extracts additional information from the tweets that belong to the trend, aiming to discover interesting aspects of it. Each of the three steps described above is pictured as a component of the diagram shown in figure 1 and is described in detail in the following paragraphs.

2.1 Detecting Bursty Keywords

A keyword is identified as bursty when it is encountered at an unusually high rate in the stream. For example, the keyword 'NBA' may usually appear in 5 tweets per minute, yet suddenly exhibit a rate of 100 tweets/min. Such 'bursts' in keyword frequency are typically associated with sudden popular interest in a particular topic and are often driven by

[1] http://www.twitter.com/

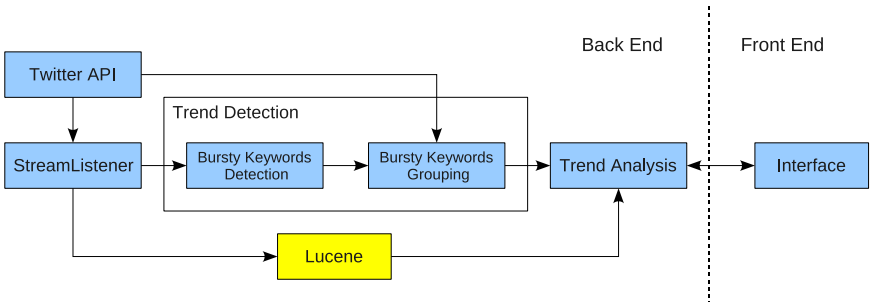

Figure 1: TwitterMonitor Architecture

emerging news or events. For example, a sudden rise in the frequency of keyword 'NBA' may be linked to an important NBA match taking place.

TwitterMonitor treats bursty keywords as 'entry points' for trend detection. In other words, whenever a keyword exhibits bursty behavior, TwitterMonitor considers this an indication that a new topic has emerged and seeks to explore it further (more on that in Sections 2.2, 2.3).

Effective and efficient detection of bursty keywords is thus crucial to TwitterMonitor's performance. To detect bursty keywords, we developed a new algorithm, `QueueBurst`, with the following characteristics: (i) One-pass. Stream data need only be read once to declare when a keyword is bursty. (ii) Real-time. Identification of bursty keywords is performed as new data arrives. No optimization over older data is involved. (iii) Adjustable against 'spurious' bursts. In some cases, a keyword may appear in many tweets over a short period of time simply by coincidence. The algorithm is tuned to avoid reporting such instances as real bursts. (iv) Adjustable against spam. Spam user groups repetitively generate large numbers of similar tweets. The algorithm is tuned to ignore such behavior. (v) Theoretically sound. `QueueBurst` is based on queuing theory results.

2.2 From Bursty Keywords to Trends

Using `QueueBurst`, TwitterMonitor computes a set of bursty keywords K_t at every moment t. Some of these keywords correspond to the same trend. For example, keywords 'NBA', 'Lakers', 'Orlando' and 'game' may be bursty at the same time, all of them occurring in tweets commenting on a Lakers vs Orlando match that is taking place. TwitterMonitor periodically groups keywords $k \in K_t$ into disjoint subsets K_t^i of K_t, so that all keywords in the same subset appear in the same topic of discussion. Given subsets $\{K_t^i\}$, a trend is identified by a single subset K_t^i.

Bursty keywords are grouped together by algorithm `GroupBurst`. To group bursty keywords, `GroupBurst` assesses their co-occurrences in recent tweets. For this purpose, a few minutes' history of tweets is retrieved for each bursty keyword and keywords that are found to co-occur in a relatively large number of recent tweets are placed in the same group. Since enumerating all possible groupings proves to be very expensive for such a real-time task, `GroupBurst` pursues a greedy strategy that produces groups in a small number of steps.

2.3 Trend Analysis

After a trend is identified as a subset K_t^i of bursty keywords, TwitterMonitor attempts to compose a more accurate description of it. The first step towards this end is to identify more keywords associated with a trend, i.e. keywords that do not exhibit bursty behavior themselves but are often encountered in the same tweets as the bursty ones. To achieve this, TwitterMonitor employs context extraction algorithms (such as PCA, SVD, etc [10]) over the recent history of the trend and reports the keywords that are most correlated with it. In the same spirit, TwitterMonitor uses Grapevine's entity extractor [8] to identify frequently mentioned entities in trends.

Moreover, a trend often consists of tweets that comment on breaking news reported on a news portal (e.g. reuters.com, nytimes.com, etc), in which case links to the related news source are cited in the tweets. TwitterMonitor identifies such frequently cited sources and adds them to the trend description. TwitterMonitor also identifies frequent geographical origins of tweets that belong to a trend. For example, when a trend is associated with a highly local event (e.g. Thanksgiving in Canada, elections in Britain), we expect that a large portion of tweets will come from the related geographical regions. Finally, a chart is produced for each trend that depicts the evolution of its popularity over time and that gets updated as long as the trend remains popular.

3. ARCHITECTURE

TwitterMonitor consists of a Back-End and a Front-End layer (Figure 1).

Back-End The `TwitterListener` module receives a nearly-uniform sample of the Twitter stream, via the Twitter API[2]. The sample currently consists of 1.2 M tweets per day, out of an estimated total of 6 M tweets per day. For every tweet received, `TwitterListener` separates tweet information into fields (tweet text, author information, timestamp, etc) and exports two feeds: one that reports tweets with all their fields to Lucene [2], where tweets are indexed accordingly, and a second one that reports only the text and timestamp of tweets to the `Bursty Keywords Detection` module.

The Twitter API is contacted again by the `Bursty Keywords Grouping` module (Section 2.2) to retrieve recent tweets (i.e. tweets generated in the last few minutes) that belong to identified trends. Due to restrictions from Twitter, for requests that date further back in time (e.g. to track the popularity of a trend in the past day during trend analysis – Section 2.3), the Lucene index is consulted instead.

Front End TwitterMonitor provides a webpage that reports recent trends in real time, as well as an interface that allows users to rank trends according to different criteria

[2]`http://apiwiki.twitter.com/Twitter-API-Documentation`

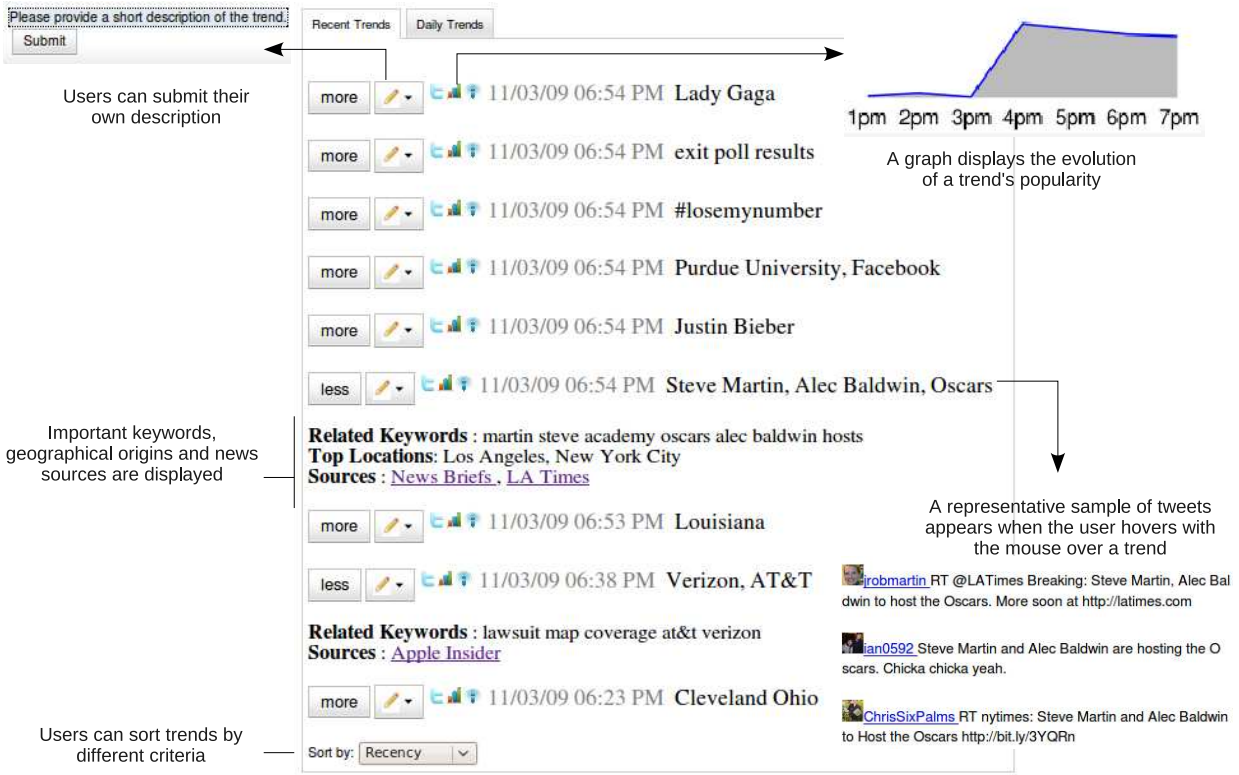

Figure 2: TwitterMonitor snapshot

and submit their own short description of a trend (Figure 2). Specifically, trends can be ranked by volume, recency or a combined score of the two. Moreover, user-created descriptions are stored on TwitterMonitor and displayed on the website as long as there are enough of them that exhibit significant overlap. Reported trends are also accompanied by a small sample of representative tweets and a link to Twitter's live stream for the trend[3].

Finally, TwitterMonitor uses an additional tab to display daily trends, i.e. trends that have emerged within the last day, ranked by aggregate volume of tweets.

4. DEMONSTRATION

In our demonstration, we will display the online version of TwitterMonitor, with all features described in this proposal and give the audience the chance to perform in-depth inspection of recent Twitter trends. Every trend will be represented by the entities involved or, in absence of identified entities, by the related bursty keywords. The audience will have the option to use the interface in order to acquire more information about trends they deem interesting. In particular, they will be shown additional keywords that are correlated with a trend and skim through representative tweets in order to obtain a better understanding of the related discussion. Moreover, they will be able to track a trend's popularity over time and spot the origin of geographically focused trends. Finally, the audience will interact with the system by ranking the displayed trends according to different criteria and submitting their own descriptions to the system. Similar functionality will be provided for daily trends.

In parallel, we will describe TwitterMonitor's functionality and architecture, as well as the algorithms that are employed in different components of the system. In general, we will share our experience from building a dynamic monitoring system, the design choices we made and the challenges we faced during development ('spurious' bursts, spam, etc).

In case of low connectivity, a back-up scenario will be in place. The back-up scenario will include a simulated run of the system over older, locally stored data.

[3] `http://search.twitter.com/`

5. REFERENCES

[1] Alltop, `http://alltop.com/`.
[2] Lucene, `http://lucene.apache.org/`.
[3] Radian6, `http://www.radian6.com/`.
[4] Scoutlabs, `http://scoutlabs.com/`.
[5] Sysomos, `http://www.sysomos.com/`.
[6] Thoora, `http://www.thoora.com/`.
[7] Twitscoop, `http://www.twitscoop.com/`.
[8] A. Angel, N. Koudas, N. Sarkas, and D. Srivastava. What's on the grapevine? In *SIGMOD*, 2009.
[9] N. Bansal and N. Koudas. Blogscope: A system for online analysis of high volume text streams. In *WebDb*, 2007.
[10] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, and R. A. Harshman. Indexing by latent semantic analysis. *JASIS*, 41(6):391–407, 1990.
[11] J. Leskovec, L. Backstrom, and J. M. Kleinberg. Meme-tracking and the dynamics of the news cycle. In *KDD*, 2009.