Bulletin of the Technical Committee on

Data Engineering

September 2013 Vol. 36 No. 3

IEEE Computer Society

Letters

Letter from the Editor-in-Chief . . . . . *David Lomet* 1
Letter from the Special Issue Editor . . . . . *Sharad Mehrotra* 2

Special Issue on Social Media and Data Analysis

Social Media Analytics: The Kosmix Story . . . . .
X. Chai, O. Deshpande, N. Garera, A. Gattani, W. Lam, D. S. Lamba, L. Liu, M. Tiwari, M. Tourn, Z. Vacheri, STS Prasad, S. Subramaniam, V. Harinarayan, A. Rajaraman, A. Ardalan, S. Das, P. Suganthan, AH Doan 4
Architectural Implications of Social Media Analytics in Support of Crisis Informatics Research . . . . .
. . . . . *Kenneth M. Anderson, Aaron Schram, Ali Alzabarah, Leysia Palen* 13
Nature of Information, People, and Relationships in Digital Social Networks . . . . . *Rakesh Agrawal* 21
Towards Geo-Social Intelligence: Mining, Analyzing, and Leveraging Geospatial Footprints in Social Media . . . .
. . . . . *James Caverlee, Zhiyuan Cheng, Daniel Z. Sui, Krishna Y. Kamath* 33
Effective Event Identification in Social Media . . . . . *Fotis Psallidas, Hila Becker, Mor Naaman, Luis Gravano* 42
Event Detection from Social Media Data . . . . . *George Valkanas, Dimitrios Gunopulos* 51
Large Scale Tensor Decompositions: Algorithmic Developments and Applications . . . . .
. . . . . *Evangelos E. Papalexakis, U Kang, Christos Faloutsos, Nicholas D. Sidiropoulosx, Abhay Harpale* 59
Summarization via Pattern Utility and Ranking: A Novel Framework for Social Media Data Analytics . . . . .
. . . . . *Xintian Yang, Yiye Ruan, Srinivasan Parthasarathy, Amol Ghoting* 67
Some Research Opportunities on Twitter Advertising . . . . . *Milad Eftekhar, Nick Koudas* 77
Supporting Efficient Social Media Search in Cyber-Physical Web . . . . . *Lidan Shou, Sai Wu* 83
Building Social Life Networks . . . . . *Ramesh Jain, Laleh Jalali, Siripen Pongpaichet, Amarnath Gupta* 91

Conference and Journal Notices

TCDE Membership Form . . . . . back cover

The TC on Data Engineering
Membership in the TC on Data Engineering is open to all current members of the IEEE Computer Society who are interested in database systems. The TCDE web page is `http://tab.computer.org/tcde/index.html`.

The Data Engineering Bulletin
The Bulletin of the Technical Committee on Data Engineering is published quarterly and is distributed to all TC members. Its scope includes the design, implementation, modelling, theory and application of database systems and their technology.

Letters, conference information, and news should be sent to the Editor-in-Chief. Papers for each issue are solicited by and should be sent to the Associate Editor responsible for the issue.

Opinions expressed in contributions are those of the authors and do not necessarily reflect the positions of the TC on Data Engineering, the IEEE Computer Society, or the authors' organizations.

The Data Engineering Bulletin web site is at `http://tab.computer.org/tcde/bull_about.html`.

Letter from the Editor-in-Chief

TCDE Activities

Maintaining the vitality of the database community within the IEEE Computer Society is important for the long term health of the database field. In his role as TCDE Chair, Kyu-Young Whang has initiated a number of new activities designed for this purpose. Each is described below.

Awards: The TCDE now supports a number of awards for outstanding work, both of a technical nature and for professional contributions. Amr El Abadi has led this effort. For technical work, there is the Computer Science, Engineering and Education Impact Award. For young members of our community, there is the TCDE Early Career Award. And finally, there is the TCDE Service Award for contributions to the community. These awards are described in detail at `http://tab.computer.org/tcde/tcdeawards.html`.

Archive: To provide institutional memory about both the TCDE and the ICDE Conferences, a web site is being designed, the effort led by Wookey Lee, that will serve as an archive for historical information. Combined with our current web site, this should provide everyone with comprehensive information over time about the database community within the Computer Society.

Membership: Membership in the TCDE is important for the long term organizational health of the database community within the Computer Society. Xiaofang Zhou leads the effort to strengthen our community's participation in TCDE. One result of this effort was the New Members' Reception at ICDE'13 in Brisbane. Another is the membership application on the back inside cover of the current (and subsequent) issues of the Bulletin. If you are not currently a member, I urge you to join.

The Current Issue

Hundreds of millions of people the world over (perhaps billions) engage in social interaction at a growing number of web sites. To say that this has changed peoples lives and the way they interact with each other is to understate the obvious. These sites are a wonderful way to stay in touch, to follow what is happening, and who it is happening to. People who barely used computers in the past (e.g. folks of my generation or older) now participate eagerly in this new world.

Web based social services also produce useful data, ripe for analysis. Web services are usually businesses with a profit motive, and hence a need to secure revenue. Advertising is almost always an important revenue component, so social data analysis for ad placement is usually crucial for success in this space.

But direct profit enhancement is not the only function that can be served by analysis of social media data. The utility of social services can also be improved by exploiting geo-spatial and temporal information, social connections, trust relationships, etc. One can, for example, arrange to meet friends who happen to be nearby at the moment in the closest coffee shop. One might use such real time data collection to track any number of interesting social phenomena.

This area of social media data analysis is the focus of the current issue, assembed by Sharad Mehrotra. This area is truly an opportunity for *NOW*. A social media industry is in its infancy, and will surely grow to enormous size. So Sharad's focus on this area in the current issue is a great opportunity to become familiar with what is happening and has happened, as you position yourself for possible participation in a huge collaborative ("social"?) and technical effort that is changing the world. I want to thank Sharad for bringing together a great set of papers, by leading practitioners, focused on this very timely and exciting topic.

David Lomet
Microsoft Corporation

Letter from the Special Issue Editor

Over the past decade, social media has emerged as a dominant means through which people communicate. Even if we restrict ourselves to Twitter, it is estimated that people send about 400M tweets on a daily basis covering a variety of topics and opinions. While, individually, such tweets might or might not be very informative, many studies have now clearly established that collectively, such data contain a wealth of information that can be leveraged in various application contexts to bring new capabilities, use-cases, and value propositions. It is getting to be well recognized that technologies for collecting, monitoring and analyzing social media can bring transformative changes to variety of real-world domains. Already, many organizations in service-oriented industry such as hospitals and customer care monitor social media to determine public opinions about their services. Similar strategies are used by product companies (e.g., electronics, cell phones) to determine the public opinion about their products and by political parties and policy makers to assess the sentiment of the community. Department of homeland security in USA has a social media program that in addition to understanding public satisfaction with their services also uses social media as a sensor to detect emerging needs and events during crisis. Social media analysis has always been important to internet companies to understand user profiles in order to bring new customization and/or targeted advertising. Social media (specially when mixed with mobile computing and location based services) is a major driver for startup activity in the Silicon Valley and other IT hubs around the world. There are many new proposals, ideas, products that are attempting to seamlessly integrate social media with pervasive computing technologies including localization and sensing to bring an immersive experience and capabilities to users.

While social media monitoring and analysis offers numerous opportunities, it also poses a large number of technical hurdles. Challenges arise at every technological layer – consider, for instance, building a system or a capability that relies on social media. Only a very small part of the gargantuan amount of information may be relevant to the end-goal leading to the challenge of effective acquisition, filtering, and ranking of social media data. Another challenge arises due to relatively short form of the messages such as tweets and Facebook posts that are seldom well structured or grammatically correct limiting the effectiveness of standard NLP and information extraction mechanisms. Yet another layer of complexity arises due to the "big and fast" nature of such data – the amount of such data and the velocity at which it arrives (coupled with the near real-time need for analysis for certain applications) poses a significant challenge in building infrastructures that can scale. No doubt, social media monitoring and analysis is the driving force behind a large amount of research and innovations at all technology levels – infrastructure level where researchers are exploring hardware and software infrastructures that can support complex social media analysis, at the representation and analysis level, where researchers are exploring mechanisms that can provide valuable insights from social media data, and application level where researchers are exploring diverse applications and new uses of social media. This special issue consists of a set of articles from leading researchers exploring social media at different technology levels.

The special issue is roughly divided into four parts. We begin the bulletin by two articles that highlight experiences and challenges in exploiting social media analysis in the context of concrete applications from researchers who have built significant such systems.

In the article entitled "Social Media Analytics: The Kosmix Story", the authors provide a glimpse of the "insider story" in building a large real-time social knowledge base entitled Social Genome in a commercial setting as part of Kosmix which was a bay area startup that was later acquired by Walmart. In the article entitled "The Architectural Implications of Social Media Analytics in Support of Crisis Informatics Research", the authors highlight the software architecture and challenges in building large-scale systems for event monitoring on twitter to support crisis management. The article further address the key lessons learnt and the implication of social media to crisis informatics in the future.

The second set of papers describe variety of ideas and issues related to social media analytics.

"Social Media Analytics Research in MSR Search Labs" focuses on ongoing and current research within Microsoft Search Labs on modeling how information spreads and propagates through social networks and how people assimilate the information and form relationships. In the article entitled "Geospatial Footprints in Social Media: Towards GeoSocial Intelligence", the authors focus on how geotagged social media collected through smart devices opens new opportunities for developing new geo-social systems which can help uncover how ideas flow from people to people and how people organize. The next two articles in the bulletin focus on the important problem of event identification in social media which is at the heart of much of the use-cases for social media. In the article entitled "Effective and Efficient Event Identification in Social Media", the authors discuss the limitations of current solutions and describe new approaches to improving detection by increasing the set of features used for clustering as well as using a more informed event model that accounts for time decay. In the article entitled "Event Detection from Social Media Data", the authors propose using concepts from emotional theories combined with Spatio-Temporal information to build a robust and scalable event detector.

The next set of papers deal with issues related to efficient processing of large social media data. In "Large Scale Tensor Decomposition: Algorithmic Developments and Applications", the authors summarize recent algorithmic developments in scaling tensor decomposition to big data using map/reduce framework. Such tensor-based analysis is a core technique for analyzing social media data for interesting patterns and anomalies. In "Summarization via Pattern Utility and Ranking: a Novel Framework for Social Media Data Analytics", the authors describe a new dynamic pattern driven approach to summarizing social networks and topologies that enables efficient processing of user-specific and topic-specific temporal analysis.

The final set of papers in the bulletin deal with new / novel emerging applications and new research opportunities for social media analytics. In "Some Research Opportunities on Twitter Advertisement", the authors revisit the issue of social advertising which is omnipresent in social media. The authors discuss the new advertising opportunities introduced by Twitter to promote advertisements to targeted individuals and identify the research challenges/opportunities such a model promotes. "S3: A framework for Efficient Social Media Search in the Cyber Physical Systems" describes a new direction of research that seamlessly integrates sensors and cyber physical systems with social media. In particular, the authors describe a framework entitled S3 that supports social media search when queries may be a result of integrating the physical world with communication and computational devices as in a cyber physical system. In the article entitled "Building Social Life Networks", the authors discuss a novel concept of social life networks that connect people with essential life resources. They identify key challenges in building such networks (viz., algebraic framework for situation modeling and recognition, context determination) and briefly describe their experience in building such systems.

As is often the case with Data Engineering Bulletins, the range of articles vary in the level of depth and treatment of the subject – while some papers focus more on vision and challenges that lie ahead, others describe technically mature approaches based on significant prior work by the authors. Irrespective of the nature of the papers, collectively they provide a good view of the state-of-the-art thoughts and research in the area of social media analytics.

Finally, I would like to acknowledge the generous help by Mehdi Sadri in following up with the authors and compiling the papers into the bulletin. Without his help, production of the bulletin would have been significantly more difficult.

Sharad Mehrotra
University of California, Irvine

Social Media Analytics: The Kosmix Story

Xiaoyong Chai[1], Omkar Deshpande[1], Nikesh Garera[1], Abhishek Gattani[1], Wang Lam[1],
Digvijay S. Lamba[1], Lu Liu[1], Mitul Tiwari[2], Michel Tourn[3], Zoheb Vacheri[1],
STS Prasad[1], Sri Subramaniam[1], Venky Harinarayan[4], Anand Rajaraman[4],
Adel Ardalan[5], Sanjib Das[5], Paul Suganthan G.C.[5], AnHai Doan[1,5]

[1] @WalmartLabs, [2] LinkedIn, [3] Google, [4] Cambrian Ventures, [5] University of Wisconsin-Madison

1 Introduction

Kosmix was a Silicon Valley startup founded in 2005 by Anand Rajaraman and Venky Harinarayan. Initially targeting Deep Web search, in early 2010 Kosmix shifted its main focus to social media, and built a large infrastructure to perform social media analytics, for a variety of real-world applications.

In 2011 Kosmix was acquired by Walmart and converted into @WalmartLabs, the advanced research and development arm of Walmart. The goals of the acquisition were to provide a core of technical people in the Valley and attract more, to help improve traditional e-commerce for Walmart, and to explore the future of e-commerce. This future looks increasingly social, mobile, and local. Accordingly, @WalmartLabs continues to develop the social media analytics infrastructure pioneered by Kosmix, and uses it to explore a range of social e-commerce applications.

In this paper we describe social media analytics, as carried out at Kosmix. While our framework can handle many types of social media data, for concreteness we will focus mostly on tweets. Section 2 describes the analytics architecture, the applications, and the challenges. We describe in particular the Social Genome, a large real-time social knowledge base that lied at the heart of Kosmix and powered most of its applications. Section 3 describes how the Social Genome was built, using Wikipedia, a set of other data sources, and social media data. Section 4 describes how we classify and tag tweets, and extract entities from tweets and link them to a knowledge base. Section 5 describes how we detect and monitor events in the Twittersphere. Section 6 discusses how we process the high-speed Twitter stream using Muppet, a scalable distributed stream processing engine built in house [1]. Section 7 discusses lessons learned and related work, and Section 8 concludes. Parts of the work described here have been open sourced [1] and described in detail in recent papers [18, 23, 25, 32].

2 Architecture, Applications, and Challenges

The overall Kosmix architecture for social media analytics is shown in Figure 1.a. To start, we retrieve data from multiple sources, including Web sources (e.g., Wikipedia, Musicbrainz, Chrome, Yahoo Stocks) and social media sources (e.g., Twitter, Foursquare, YouTube, Facebook, Flickr). In particular, Kosmix had access to the full Twitter fire hose, which streamed in at about 3,000 tweets per second. Thus, fast and accurate processing of this fire hose became a major challenge.

In the next step, we process the retrieved data using a variety of techniques in information extraction, integration, entity matching and merging, schema matching, and event detection and monitoring, among

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

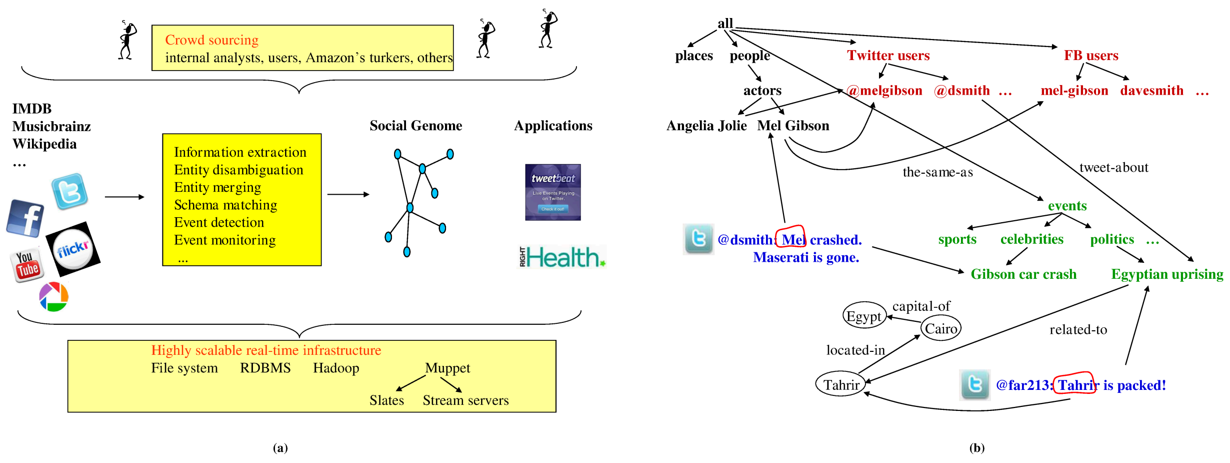


Figure 1: (a) The overall architecture for analytics, and (b) a sample fragment of the Social Genome.

others. Our goal was to build a large real-time knowledge base called *Social Genome*, which captures all important entities, relationships, and events that are happening in real time in social media. Then we leverage the Social Genome to build a variety of applications, such as TweetBeat, Firsthand, Social Cube, and RightHealth. We run the above pipeline on a highly scalable real-time data processing infrastructure, which uses the file system, RDBMSs, Hadoop, and Muppet. Muppet is a distributed stream processing engine developed in house, and was used to process the high-speed stream of tweets flowing into Kosmix. Thus, it is similar to Storm at Twitter, but with important differences (see Section 7). Throughout the entire processing pipeline, we also employed crowdsourcing (using internal analysts, Amazon Mechanical Turk's workers, end users, etc.) to improve the accuracy of the processed data.

As described, the Social Genome knowledge base lies at the heart of the Kosmix analytics infrastructure. This knowledge base consists of the following (as illustrated in Figure 1.b):

- A Freebase-like knowledge base of popular concepts and instances on the Web, such as places, people, actors, politicians, Angelina Jolie, and Mel Gibson (see the top-left corner of the figure). This Web knowledge base, called *Kosmix KB*, was constructed by integrating Wikipedia with several other databases (see Section 3).

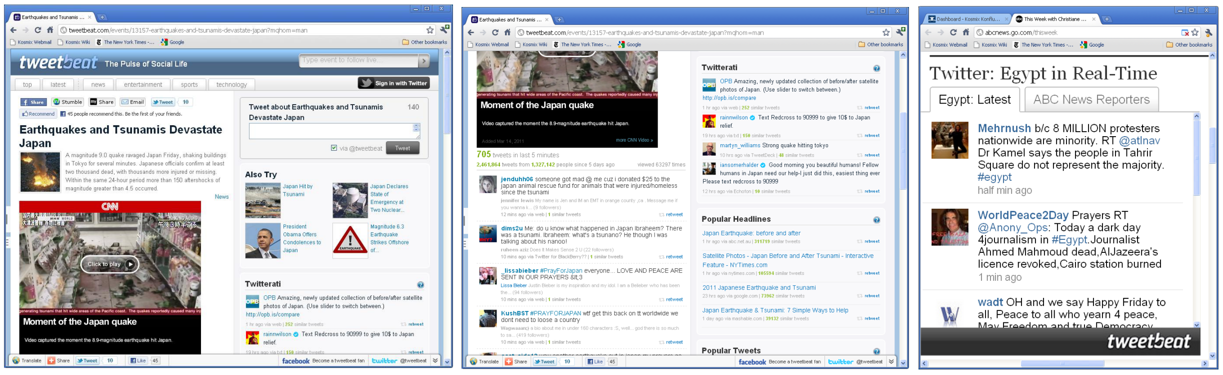
- Profiles of social media users: Twitter users such as @melgibson, @dsmith; Facebook users; Foursquare users, and so on (see the top-right corner of the figure).

- Popular events detected in the Twittersphere, such as Gibson car crash, Egyptian uprising, earthquakes (see the bottom-right corner).

- Tweets and other raw data such as Web pages (e.g., "Mel crashed. Maserati is gone." and "Tahrir is packed" in the figure).

In addition, the Social Genome also contains many relationships inferred by our algorithms. For example, we matched person instances in the Web knowledge base (i.e., the Kosmix KB) with social media user profiles, to create "the-same-as" relationship wherever appropriate. For example, we created a "the-same-as" relationship between the person instance "Mel Gibson" in the Kosmix KB and Twitter user @melgibson (see Figure 1.b), because they refer to the same real-world person.

As another example, when receiving the tweet "Mel crashed. Maserati is gone.", we established that "Mel" in the tweet is a person name, and that it refers to the person instance "Mel Gibson" in the Kosmix KB (see Figure 1.b). In other words, we perform entity extraction over tweets and link the discovered entities


tweetbeat The Pulse of Social Life
Earthquakes and Tsunamis Devastate Japan
Also Try
Twitterati
Moment of the Japan quake
Popular Headlines
Popular Tweets
(a)
(b)
(c)
Twitter: Egypt in Real-Time
Egypt: Latest
ABC News Reporters
tweetbeat

Mehrnush b/c 8 MILLION protesters nationwide are minority. RT @atlnav Dr Kamel says the people in Tahrir Square do not represent the majority. #egypt
half min ago

WorldPeace2Day Prayers RT @Anony_Ops: Today a dark day 4journalism in #Egypt.Journalist Ahmed Mahmoud dead,AlJazeera's licence revoked,Cairo station burned
1 min ago

wadt OH and we say Happy Friday to all, Peace to all who yearn 4 peace,

Figure 2: (a)-(b) Screen shots of the TweetBeat homepage showing the 2011 Japanese earthquake event, and (c) a TweetBeat widget embedded in ABC news homepage showing the 2011 Egyptian uprising event.

to the same entities in the Kosmix KB. Other examples of relationships include a Twitter user tweeting about an event (e.g., @dsmith tweeting about Egyptian uprising), and an event (e.g., Egyptian uprising) is related to an instance in the Kosmix KB (e.g., Tahrir).

We used the Social Genome to build a variety of real-world applications. Our flagship application was TweetBeat, which monitors the Twittersphere to detect important emerging events (e.g., earthquake, uprising, stock crash), then displays the most important tweets of these events in real time. For example, when TweetBeat detected that the 2011 Japanese earthquake was happening, it created an entire page for this event. The top part of this page (Figure 2.a) names the event, and gives a brief description and a picture or video (if available). The bottom part of the page (Figure 2.b) shows important tweets about this event in real time, in a continuously scrolling fashion (see the left part of this figure). Figure 2.c shows a TweetBeat widget that was embedded in the ABC news homepage back in 2011. This widget shows important tweets for the event Egyptian uprising in real time. During the period 2010-2011, TweetBeat detected and monitored hundreds of important events per day.

Firsthand is another example application that used the Social Genome[1]. When a user is reading a news article, Firsthand (installed as a browser widget) detects and highlights entities (e.g., people, organizations) that appear the article and have Twitter accounts. If the user hovers the mouse over such an entity, Firsthand will retrieve and display the latest tweets from the corresponding account. This application makes use of "the-same-as" relationships in the Social Genome. Specifically, we extract and link entities that appear in the article to instances in the Kosmix KB, then follow "the-same-as" links to access the Twitter accounts of these entities.

SocialCube is another application in which we leveraged the Social Genome to build a real-time data cube with dimensions such as location, topics, and sentiment, then used it to answer questions such as "How many are tweeting about Barack Obama in New York, by the minute for the last hour?" and "How many Twitter users in Arizona feel positive about the new Medicare plan?".

As described, developing the analytics infrastructure and the applications on top of it raises difficult challenges. First, how do we build the various knowledge bases? Second, when a tweet comes in, how do we classify and tag the tweet, and extract and link entities, such as finding out that "Mel" in the tweet is a person name and refers to the person instance Mel Gibson in the Kosmix KB? Third, how do we detect emerging important events in the Twittersphere (e.g., earthquakes), and how do we monitor tweets of these events? Finally, how do we process social media data in real time? We discuss our solutions to these challenges in the subsequent sections.

[1]appscout.pcmag.com/social-networking/269719-tweetbeat-firsthand-read-someone-s-tweets-anywhere-they-re-mentioned

3 Building the Social Genome Knowledge Base

We now describe building the Social Genome. But before that, we briefly describe the notion of a knowledge base. A knowledge base (KB) typically consists of a set of concepts organized into a taxonomy, instances for each concept, and relationships among the concepts. Figure 1.b shows a tiny KB (in the top-left corner), which illustrates the above notions. In this KB, for example, "actors" are a kind of "people", and Angelina Jolie and Mel Gibson are instances of "actors".

To build the Social Genome, we first build the Kosmix KB, which is a knowledge base of popular concepts and instances on the Web, such as places, people, actors, politicians, Angelina Jolie, Mel Gibson, etc. In this sense it is similar to Freebase and Google's Knowledge Graph. Then we augment the Kosmix KB with social-media information, such as Twitter user profiles, events, and tweets. We now describe these two steps in more details.

To build the Kosmix KB (see [18] for a detailed description), we convert Wikipedia into a KB, then integrate it with additional data sources, such as Chrome (an automobile source), Adam (health), MusicBrainz, City DB, and Yahoo Stocks. Here we highlight several interesting aspects that have not commonly been discussed in the KB construction literature. First, we found that converting Wikipedia into a taxonomy is highly non-trivial, because each node in the Wikipedia graph can have multiple paths (i.e., lineages) to the root. We developed an efficient solution to this problem. Interestingly, it turned out that different applications may benefit from different lineages of the same node, so we convert Wikipedia into a taxonomy but preserve all lineages of all Wikipedia nodes.

Second, extracting precise relationships from Wikipedia (and indeed from any non-trivial text) is notoriously difficult. We developed a solution that sidesteps this problem and extracts "fuzzy relationships" instead, in the form of a relationship graph. Later we were able to use this fuzzy relationship graph in a variety of real-world applications. Third, we extracted meta information for the nodes in the KB, focusing in particular on social information such as Wikipedia traffic statistics and social contexts. For example, given the instance "Mel Gibson", we store the number of times people click on the Wikipedia page associated with it, the most important keywords associated with it in social media in the past 1 hour (e.g., "mel", "crash", "maserati"), and so on. Such meta information turns out to be critical for many of our applications. Finally, we added new data sources to the KB constructed out of Wikipedia. In doing so, we had to match external instances with those in the KB, and heavily used taxonomy matching and entity instance matching algorithms.

Building the initial KB is difficult, but is just the very first step. In the long run, maintaining and curating the KB pose the most challenges and incur most of the workload. We developed a solution to refresh the KB every day by rerunning most of it from the scratch. We also had to address a major technical challenge: how to curate the KB and preserve the curation after refreshing the KB. Our solution is to capture most of the human curation in terms of commands, and then apply these commands again when we refresh the KB.

Once we had built the Kosmix KB, we added social media data to it. Examples include adding profiles of social media users (e.g., Twitter users), events, and annotated tweets. This process raises two key challenges. First, how do we perform entity extraction, linking, classification, and tagging for tweets? And second, how do we detect and monitor events in the Twittersphere? In the next two sections we discuss these two challenges.

4 Entity Extraction, Linking, Classification, and Tagging

To augment the Kosmix KB with social media data, we need to perform entity extraction, linking, classification, and tagging for the incoming tweets. For example, given a tweet such as "Obama gave an immigration speech while on vacation in Hawaii", *entity extraction* determines that string "Obama" is a person name, and that "Hawaii" is a location. *Entity linking* goes one step further, inferring that "Obama" actually refers to a particular entity in the Kosmix KB and that "Hawaii" refers to another entity. *Classification* assigns a set of predefined topics to the tweet, such as “politics” and “travel”. Finally, *tagging* assigns descriptive tags to the tweet, such as “politics”, “tourism”, “vacation”, “President Obama”, “immigration”, and “Hawaii”, the way a person may tag a tweet today. Our applications heavily use the results of such extraction, linking, classification, and tagging.

To solve the above problems, we proceed as follows (see [23] for more details). Given a tweet, we preprocess it, e.g., detecting the language, tokenizing. Next, we use the Kosmix KB to extract mentions from the tweet, remove certain mentions, then score the remaining ones. Here a mention refers to a pair of (string, KB node), which states that the string refers to a particular node in the Kosmix KB. So we are effectively performing entity extraction and linking at the same time. Then in the next step we use these mentions to classify and tag the tweet. Next, we go back to processing the mentions, but do so in more depth. Specifically, we extract 30+ mention features, remove certain mentions using rules involving these features, disambiguate the mentions (e.g., linking “apple” to Apple the company not Apple the fruit), then score the mentions again. Next, we use the “clean” mentions to classify and tag the tweet again. Finally we apply hand-crafted editorial rules to filter mentions and classification and tagging results. Compared to current work, our solution is distinguished in several important aspects:

- *Using a Global and “Real-Time” Knowledge Base:* The Kosmix KB (which we use to find and link to entities mentioned in tweets) is built from Wikipedia. Wikipedia is global in that it contains most concepts and instances judged important in the world. Thus, it provides a good coverage for the tasks. More importantly, it is “real time” in that contributors continuously update it with new entities that just appear in real-world events. This “real time” nature makes it especially well-suited for processing social data. In contrast, many current solutions use knowledge bases that are updated less frequently.

- *Synergistic Combination of the Tasks:* Our system interleaves the four tasks of extraction, linking, classification, and tagging in a synergistic fashion. For example, given a tweet, we begin by performing a preliminary extraction and linking of entity mentions in that tweet. Suppose many such mentions link to many nodes under the subtree “Technology” in the Kosmix KB. Then we can infer that “Technology” is a likely topic for the tweet, thereby helping classification. In return, if we have determined that “Technology” is indeed a topic for the tweet, then we can infer that string “apple” in the tweet likely refers to the node “Apple Corp.” in the KB, not the node “Apple (fruit)”, thereby helping entity linking.

- *Using Contexts and Social Information:* Given a tweet such as “go giants!”, without some context, such as knowing that this user often tweets about the New York Giants football team, it is virtually impossible to extract and link entities accurately. As another example, it is not possible to process the tweet “mel crashed, maserati gone” in isolation: we have no idea which person named Mel the user is referring to. However, if we know that in the past one hour, when people tweeted about Mel Gibson, they often mentioned the words “crash” and “maserati” (a car brand), then we can infer that “mel” likely refers to the node Mel Gibson in the KB. Our system exploits such intuitions. It collects contexts for tweets, Twitter users, hash tags, Web domains, and nodes in the Kosmix KB. It also collects a large number of social signals (e.g., traffic on Wikipedia and Pinterest pages). The system uses these contexts and signals to improve the accuracy of the tasks.

Other important features of our system include a minimal use of complex time-intensive techniques, to ensure that we can process tweets in real time (at the rate of up to 6000 tweets per second), and the use of hand-crafted rules at various places in the processing pipeline to exert fine-grained control and improve system accuracy.

5 Event Detection and Monitoring

As discussed earlier, we process the social media stream (e.g., Twitter fire hose, Foursquare checkins) to detect important emerging events, then monitor these events. Much work in academia and industry has addressed event detection. However, this work has been limited in three main ways. First, it typically exploits just one kind of heuristics, such as finding popular and strongly related keywords (e.g., Egypt, revolt). Second, it does not scale to the high volume of data streaming in, typically because the work does not exploit distributed and parallel processing on a cluster of machines. Finally, the work has not exploited crowdsourcing to improve the accuracy of event detection.

Our current event detection solution addresses these limitations. First, we employ many heuristics to detect event candidates. For example, a heuristic finds keywords that suddenly become hot and strongly related (e.g., "Haiti" suddenly became hot, and "Haiti" and "earthquake" suddenly co-occurred in many tweets). Another heuristic monitors Twitter accounts that are well known for broadcasting breaking news. Yet another heuristic checks to see if a large number of people (e.g., more than 15) check into the same location in Foursquare (potentially indicating that an event is taking place at that location), and so on. We evaluate these heuristics against the social media stream using Muppet, our in-house distributed stream processing engine, run over a cluster of machines. Finally, we employ crowdsourcing to remove false-positive events and to extract important meta data for the remaining events.

Once we have detected an event, we monitor the Twitter sphere to find tweets related to this event, then display the most important tweets. This is often called *event monitoring* or *tracking*. The simplest, and most common, solution for event monitoring is to manually write rules to match tweets to events. For example, if a tweet contains certain keywords or user IDs, then it is flagged as positive. This solution is conceptually simple, easy to implement, and often achieves high initial precision. But it suffers from three limitations. First, manually writing rules is labor intensive and does not scale to hundreds or thousands of events per day. Second, manually writing good rules can be quite hard for many events. Finally, and most importantly, rules often become invalid or inadequate over time. For example, when a shooting happened in Baltimore in 2011, initially Twitter users referred to it using the keywords "Baltimore" and "shooting". A few hours later, however, when it was clear that the shooting happened on the John Hopkins campus, most Twitter users referred to it as the "John Hopkins shooting" instead of the "Baltimore shooting", thus rendering ineffective any rules that mention "Baltimore" and "shooting". To address the above limitations, our solution uses machine learning to evolve the profile of an event over time, then uses the learned profile to find tweets related to the event.

6 Scalable Processing of Fast Data

We run most of the social media analytics pipeline on Muppet, an in-house system that processes the incoming social data streams (e.g., Twitter fire hose, Foursquare checkins) in a distributed fashion, over a cluster of machines. Muppet was motivated by the need to process such streams with minimal latency and high scalability. For example, an application that monitors the Twitter fire hose for an ongoing earthquake may want to report relevant information within a few seconds of when a tweet appears, and must handle drastic spikes in the tweet volumes. The key idea behind Muppet is to provide a MapReduce-like framework for fast data (i.e., high-speed data streams), so that developers can quickly write and execute such applications on large clusters of machines.

To realize the above idea, we first developed MapUpdate, a framework to process fast data. Like MapReduce, in MapUpdate the developer only has to write a few functions, specifically *map* and *update* ones. The system automatically executes these functions over a cluster of machines. MapUpdate however differs from MapReduce in several important aspects. First, MapUpdate operates on data streams. So map and update functions must be defined with respect to streams. For example, mappers map streams to streams, split streams, or merge streams, while updaters process events flowing in from one or multiple streams.

Second, streams may never end. So updaters use storage called *slates* to summarize the data that they have seen so far. The notion of slates does not arise in MapReduce, nor in many recently proposed stream processing systems (see the related work section). In MapUpdate, slates are in effect the "memories" of updaters, distributed across multiple map/update machines, as well as persisted in a key-value store for later processing. Making such "memory pieces" explicit and managing them as "first-class citizens", in a real-time fashion, is a key distinguishing aspect of the MapUpdate framework. Finally, a MapUpdate application often involves not just a mapper followed by an updater, but many of these, in an elaborate workflow that consume and generate data streams.

We then developed Muppet, a MapUpdate implementation. In [25] we discuss the key challenges of Muppet in terms of distributed execution, managing slates, handling failures, reading slates, and sketch our solutions. Since mid-2010, we have used Muppet extensively to develop many social media and e-commerce applications.

7 Lessons Learned & Related Work

Our work on social media analytics suggests several important lessons. First, analyzing social data is fundamentally much harder than analyzing "traditional" data, due to a lack of context, dynamic environment (concepts appear and disappear quickly), quality issues (lots of spam), quick spread of information, and fast data. Second, context is vital to analyzing social data. Given a tweet such as "go giants!", without some context, such as knowing that this user often tweets about the New York Giants football team, it is virtually impossible to extract and link entities accurately. As another example, it is not possible to process the tweet "mel crashed, maserati gone" in isolation: we have no idea which person named Mel the user is referring to. Third, it is important to use a knowledge base to help classify and tag tweets, and to extract and link entities in tweets. Finally, crowdsourcing is indispensable (e.g., in building knowledge bases, evaluating detected events), but raises many interesting challenges.

In terms of related work, in the past few years a wealth of work has addressed the problem of social media analytics, in both academia and industry (e.g., Topsy, Stocktwits, [9–11,14,16,33]). But this work has mostly analyzed the data at the keyword level, to answer questions such as "how many tweets mention the word 'Obama' today?". In contrast, Kosmix aimed to analyze at the *semantic* level, to answer questions such as "how many tweets mention President Obama today?". To do this, we need to recognize that "Obama", "the pres", "BO", and "the messiah" for example can all refer to the same person.

Regarding knowledge bases, recent work has utilized Wikipedia (and other data sources) to build global ontology-like KBs. Well-known examples include Freebase [13], DBpedia [8,12], YAGO [37,38], and WolframAlpha [41]. These works however have not described the end-to-end process of building, maintaining, and using these KBs. In particular, they have not discussed converting the Wikipedia graph into a taxonomy (as we do here and in [18]). Finally, as far as we know, no work has addressed the problem of building a large real-time social KB, such as the Social Genome.

Entity extraction and classification of formal text has been widely studied for more than two decades (e.g., [4,5,17,24,26,28]), with competitions (e.g., CoNLL [35,40], MUC [39] and ACE [20]) and off-the-shelf tools (e.g., Stanford NER, OpenNLP, GATE, LingPipe). Entity extraction and classification for tweets, on the other hand, has been a less studied problem. Liu et al. [27] extracts only person, organization and location entities, while we do it for a large number of entity types with links to a knowledge base. Finn et al. [21] use crowdsourcing to annotate a large corpus of tweets. Recently Ritter et al. [34] have developed a NLP pipeline spanning POS tagging, chunking and named entity recognition and classification for tweets. SemTag and Seeker [19] perform automatic semantic tagging of a large Web corpus using an ontology. Industrial systems for entity extraction and classification include OpenCalais [31], AlchemyAPI [6], and Semantria [36]. Semantria does not have support for linked data whereas OpenCalais and AlchemyAPI do. OpenCalais additionally extracts relations, facts and events. The paper [23] empirically shows that our system outperforms OpenCalais in many aspects. Event and trend detection in social media have been actively studied (e.g., [11, 33]). However, they typically do not use multiple heuristics, as we do, and few of them have considered scaling to the Twitter fire hose. Research has also addressed event monitoring but mostly in news stories (e.g., [7]). Our event monitoring work is perhaps most similar to Twitcident [3], but that work uses deep semantic techniques (based on named entity recognition) that tend to be error prone, and hence achieves limited accuracy.

Regarding scalable stream processing, recent works (e.g., [15, 30]) have extended MapReduce to incremental batch processing. However, they retain the MapReduce model, where a Reducer is a "blocking" operator, in that it still has to see *all* the necessary data from the Mappers before it can "reduce" and emit the final result. So this data has to be stored in the system. In contrast, MapUpdate uses slates to *summarize* the past data, so an updater can immediately process each event as the event comes in. This allows us to stream events through the system with millisecond to second latencies. Numerous stream processing systems have been developed in the database community [22] (e.g., Borealis; STREAM; Telegraph; SPADE for System S, commercialized as IBM InfoSphere Streams; Aurora, commercialized as StreamBase Systems; and Truviso). These systems often employ declarative query languages over data with schemas. In contrast, we make few assumptions about the data structure, and adopt a MapReduce style in which applications are decomposed into a procedural workflow of custom code. Second, much work has focused on optimizing query processing over data streams in an RDBMS style. In contrast, we focus on how to efficiently execute Maps and Updates over a cluster of machines, to achieve ultra-low-latency and high scalability. Our work is similar to S4 [29] and Storm [2]. These systems, however, leave it to the application to implement and manage its own state. Our experience suggests that this is highly non-trivial in many cases. In contrast, Muppet transparently manages application storage, which are slates in our case.

8 Concluding Remarks

In this paper we have described how Kosmix performed semantic analysis of social media, by extracting the most important entities, relationships, and events from the data. We believe that such semantic analysis will become increasingly important to a variety of applications, and that a key to perform them effectively is to leverage large-scale knowledge bases (such as the Kosmix KB), crowdsourcing (e.g., to clean up the knowledge bases and process the discovered events), as well as distributed stream processing infrastructure (such as Muppet).

References

[1] Muppet. Available as the open source system called Mupd8 github.com/walmartlabs/mupd8 (renamed to Mupd8 for legal reasons).

[2] Storm. `https://github.com/nathanmarz/storm`, 2011.

[3] F. Abel, C. Hauff, G. Houben, R. Stronkman, and K. Tao. Semantics + filtering + search = twitcident. exploring information in social web streams. In *HT*, 2012.

[4] E. Agichtein and V. Ganti. Mining reference tables for automatic text segmentation. In *SIGKDD*, 2004.

[5] J. S. Aitken. Learning information extraction rules: An inductive logic programming approach. In *ECAI*, 2002.

[6] AlchemyAPI. http://www.alchemyapi.com/.

[7] J. Allan, R. Papka, and V. Lavrenko. On-line new event detection and tracking. In *SIGIR*, 1998.

[8] S. Auer, C. Bizer, G. Kobilarov, J. Lehmann, R. Cyganiak, and Z. Ives. DBpedia: A nucleus for a web of open data. In *The Semantic Web*, 2007.

[9] N. Bansal, F. Chiang, N. Koudas, and F. Wm. Tompa. Seeking stable clusters in the blogosphere. In *VLDB*, 2007.

[10] N. Bansal and N. Koudas. Blogscope: A system for online analysis of high volume text streams. In *VLDB*, 2007.

[11] H. Becker, M. Naaman, and L. Gravano. Beyond trending topics: Real-world event identification on twitter. In *ICWSM*, 2011.

[12] C. Bizer, J. Lehmann, G. Kobilarov, S. Auer, C. Becker, R. Cyganiak, and S. Hellmann. DBpedia- a crystallization point for the web of data. *Web Semantics: Science, Services and Agents on the World Wide Web*, 7(3):154–165, 2009.

[13] K. Bollacker, C. Evans, P. Paritosh, T. Sturge, and J. Taylor. Freebase: a collaboratively created graph database for structuring human knowledge. In *SIGMOD*, 2008.

[14] C. Budak, D. Agrawal, and A. El Abbadi. Structural trend analysis for online social networks. *PVLDB*, 4(10):646–656, 2011.

[15] T. Condie, N. Conway, P. Alvaro, J. M. Hellerstein, K. Elmeleegy, and R. Sears. Mapreduce online. In *NSDI '10: 7th USENIX Symposium on Networked Systems Design and Implementation*, 2010.

[16] M. Das, S. Thirumuruganathan, S. Amer-Yahia, G. Das, and C. Yu. Who tags what? an analysis framework. *PVLDB*, 5(11):1567–1578, 2012.

[17] P. DeRose, W. Shen, F. Chen, A. Doan, and R. Ramakrishnan. Building structured web community portals: A top-down, compositional, and incremental approach. In *VLDB*, 2007.

[18] O. Deshpande, D. S. Lamba, M. Tourn, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan, and A. Doan. Building, maintaining, and using knowledge bases: a report from the trenches. In *SIGMOD*, 2013.

[19] S. Dill, N. Eiron, D. Gibson, D. Gruhl, R. Guha, A. Jhingran, T. Kanungo, S. Rajagopalan, A. Tomkins, J. A. Tomlin, and J. Y. Zien. SemTag and Seeker: Bootstrapping the semantic web via automated semantic annotation. In *WWW*, 2003.

[20] G. Doddington, A. Mitchell, M. Przybocki, L. Ramshaw, S. Strassel, and R. Weischedel. The automatic content extraction (ACE) program–tasks, data, and evaluation. In *LREC*, 2004.

[21] T. Finin, W. Murnane, A. Karandikar, N. Keller, J. Martineau, and M. Dredze. Annotating named entities in Twitter data with crowdsourcing. In *NAACL HLT Workshop*, 2010.

[22] M. Garofalakis, J. Gehrke, and R. Rastogi (editors). *Data Stream Management*. Springer, 2009.

[23] A. Gattani, D. S. Lamba, N. Garera, M. Tiwari, X. Chai, S. Das, S. Subramaniam, A. Rajaraman, V. Harinarayan, and A. Doan. Entity extraction, linking, classification, and tagging for social media: A wikipedia-based approach. In *VLDB*, 2013.

[24] J. Lafferty, A. McCallum, and F. CN Pereira. Conditional random fields: Probabilistic models for segmenting and labeling sequence data. In *ICML*, 2001.

[25] W. Lam, L. Liu, S. Prasad, A. Rajaraman, Z. Vacheri, and A. Doan. Muppet: Mapreduce-style processing of fast data. *PVLDB*, 5(12):1814–1825, 2012.

[26] W. Lehnert, J. McCarthy, S. Soderland, E. Riloff, C. Cardie, J. Peterson, F. Feng, C. Dolan, and S. Goldman. UMass/Hughes: Description of the CIRCUS system used for MUC-5. In *MUC-5*, 1993.

[27] X. Liu, S. Zhang, F. Wei, and M. Zhou. Recognizing named entities in tweets. In *ACL HLT*, 2011.

[28] A. McCallum, D. Freitag, and F. Pereira. Maximum entropy Markov models for information extraction and segmentation. In *ICML*, 2000.

[29] L. Neumeyer, B. Robbins, A. Nair, and A. Kesari. S4: Distributed stream computing platform. In *ICDMW 2010, The 10th IEEE Int. Conf. on Data Mining Workshops*, 2010.

[30] C. Olston, G. Chiou, L. Chitnis, F. Liu, Y. Han, M. Larsson, A. Neumann, V. B. N. Rao, V. Sankarasubramanian, S. Seth, C. Tian, T. ZiCornell, and X. Wang. Nova: Continuous pig/hadoop workflows. In *Proc. of SIGMOD '11*, 2011.

[31] OpenCalais. http://www.opencalais.com/.

[32] Y. Pavlidis, M. Mathihalli, I. Chakravarty, A. Batra, R. Benson, R. Raj, R. Yau, M. McKiernan, V. Harinarayan, and A. Rajaraman. Anatomy of a gift recommendation engine powered by social media. In *SIGMOD*, 2012.

[33] A. Popescu, M. Pennacchiotti, and D. Paranjpe. Extracting events and event descriptions from twitter. In *WWW (Companion Volume)*, 2011.

[34] A. Ritter, S. Clark, and O. Etzioni. Named entity recognition in tweets: An experimental study. In *EMNLP*, 2011.

[35] T. K. Sang and F. Erik. Introduction to the CoNLL-2002 shared task: Language-independent named entity recognition. In *NAACL*, 2002.

[36] Semantria. https://semantria.com/.

[37] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: a core of semantic knowledge. In *WWW*, 2007.

[38] F. M. Suchanek, G. Kasneci, and G. Weikum. Yago: A large ontology from Wikipedia and WordNet. *Web Semantics: Science, Services and Agents on the World Wide Web*, 6(3):203–217, 2008.

[39] B. M. Sundheim and N. A. Chinchor. Survey of the message understanding conferences. In *ACL HLT*, 1993.

[40] E. F. Tjong Kim Sang and F. De Meulder. Introduction to the CoNLL-2003 shared task: Language-independent named entity recognition. In *NAACL HLT*, 2003.

[41] S. Wolfram. Wolfram| Alpha. *http://www.wolframalpha.com*, 2009.

Architectural Implications of Social Media Analytics in Support of Crisis Informatics Research

Kenneth M. Anderson, Aaron Schram, Ali Alzabarah, Leysia Palen
Project EPIC
Department of Computer Science
University of Colorado Boulder
{ken.anderson,aaron.schram,ali.alzabarah,leysia.palen}@colorado.edu

Abstract

Crisis informatics is a field of research that investigates the use of computer-mediated communication—including social media—by members of the public and other entities during times of mass emergency. Supporting this type of research is challenging because large amounts of ephemeral event data can be generated very quickly and so must then be just as rapidly captured. Such data sets are challenging to analyze because of their heterogeneity and size. We have been designing, developing, and deploying software infrastructure to enable the large-scale collection and analysis of social media data during crisis events. We report on the challenges encountered when working in this space, the desired characteristics of such infrastructure, and the techniques, technology, and architectures that have been most useful in providing both scalability and flexibility. We also discuss the types of analytics this infrastructure supports and implications for future crisis informatics research.

1 Introduction

The field of crisis informatics has arisen as a topic of study starting in the mid 2000s in response to the pervasive access and use of technology by members of the public during times of mass emergency. First coined by Christine Hagar during the UK foot and mouth disease crisis [3,4] and then expanded by the work of Leysia Palen [6, 7, 9, 10], crisis informatics seeks to understand the range of techniques, services, and technology that can be brought to bear to better understand how the public uses technology to respond to disasters and mass emergencies. It also studies how that "informal" response interacts with and influences the "formal" response by government agencies. From that understanding, crisis informatics hopes to guide the design of future technology to better serve those needs with an eye towards making society's response to disaster more resilient and effective.

In our work as part of Project EPIC [8], we have investigated various software architectures and software components needed to enable crisis informatics research [1,2]. Our approach is to fundamentally support the larger international endeavor of crisis informatics that is geared towards developing end-user tools. These tools are typically referred to as *crisis dashboards* or *crisis mash-ups* and they attempt to display information about a specific event in various ways including reports, live streams, and maps.

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

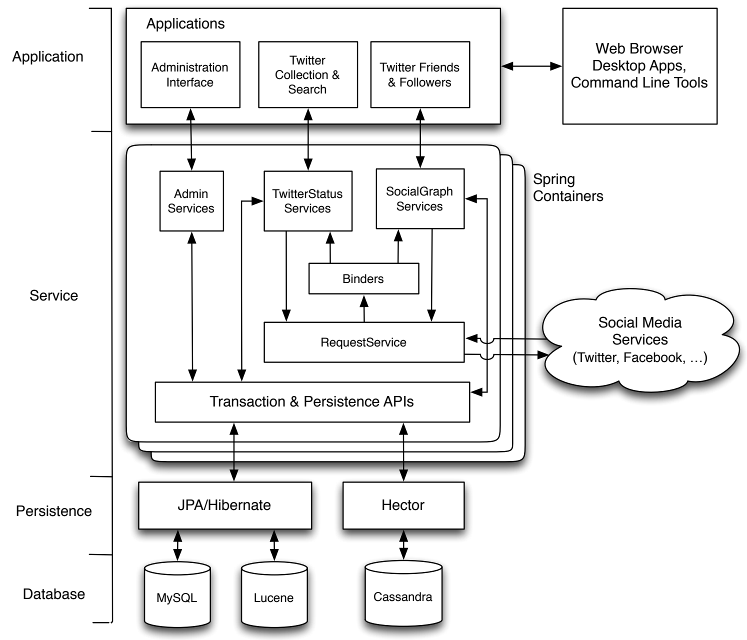


Figure 1: The Project EPIC Architecture for Scalable and Reliable Data Collection and Analysis

However, our larger goal has been to go further than individual tools and create *crisis informatics infrastructure*: a platform that can be used to develop general software tools that can be used on a variety of events and that can enable the longitudinal study of hundreds of similar crisis events over many years. With that goal in mind, we have designed and developed a layered software architecture with clear delineations between physical storage, persistence-related services, domain-specific services, and applications that can be implemented by a flexible, highly available, horizontally scalable, reliable, and robust software infrastructure [1].

Within this framework, we have investigated the capabilities of various production-class software components (such as Cassandra, Lucene, MongoDB, MySQL, and Spring) and developed the software that glues them together into a reliable and robust system that can collect large amounts of social media data during times of mass emergency while maintaining 24/7 operation with 99% system uptime [2].

Our current infrastructure adopts the logical software architecture shown in Fig. 1 and is deployed on a cluster of machines that includes one machine for web-based applications, one for analytics, one for backup, and four nodes for the storage of terabytes of collected social media data. With this infrastructure, Project EPIC has collected nearly three billion tweets across hundreds of events over the past three years.

Below, we highlight the software engineering challenges encountered when working to capture and analyze the data needed to support crisis informatics research. We present the types of questions asked of the social media data sets generated during a crisis event and the range of problems that such data sets present for collection. We also discuss the adjustments we have made to our crisis informatics infrastructure to support social media analytics and provide insight into the types of analysis enabled by our software platform.

2 Analytics for Crisis Informatics Research

The software infrastructure in Fig. 1 is able to collect millions of tweets per day for many weeks to track the social media conversations about crisis events during their immediate response and recovery period and sometimes during the warning period, depending on the nature of the hazard. Once the conversation diminishes, collection for events can end and analysis can begin. Crisis informatics researchers may then pose a wide range of queries about collected data sets [5] including identifying how many unique users contributed to the sets; the most retweeted tweets; tweets containing popular links; the most influential twitter users; the percentage of tweets that were retweets; number of geolocated tweets; a high-level description of common terms and their evolution; etc. Other questions include more prosaic metrics such as what search terms were used to generate the data set, the start and end dates of the collection, volume of tweets per day, and so on.

The analysts also like to identify a smaller representative set of tweets to study the event in more qualitative detail. They seek to understand how people use social media to cordinate or collaborate during events with each other or with emergency response organizations. The challenge is that even these data sets can consist of millions of tweets and a group of human researchers can only perform in-depth analysis on a set of a few thousand tweets. As a result, an additional form of analysis is required to identify the rules and procedures for creating a representative set of tweets. These rules and procedures can be different across event types, but in general will include the need to filter out certain types of tweets ("filter out tweets that contain the word 'pray' ") or to sample tweets based on various characteristics ("include at least one tweet from all users who contributed to the entire set") or metrics ("include only tweets that were retweeted at least 100 times") [5].

3 Challenges in Analyzing Twitter Data

From a software engineering perspective, there are numerous challenges that must be confronted when both collecting and analyzing large sets of Twitter data. On the collection side, the Twitter Streaming API can deliver large amounts of tweets in real time but cannot provide access to tweets generated in the past, while the Twitter Search API provides access to past tweets but significantly limits that access to just a small sample, sometimes going back weeks but more typically just a few hours. Furthermore, most Twitter API methods are rate limited which constrains the speed at which a data set can be generated. For instance, the API method that allows the retrieval of the last 3200 tweets generated by a user is restricted to 720 requests per hour where each request can retrieve 200 tweets of a particular user. If you assume that nothing goes wrong and all users have at least 3200 tweets, then these rates limit you to the collection of just 45 users per hour. The actual average is higher than that (since not all users have 3200 tweets) but not significantly and problems can and do occur over the long periods of time that it takes to do this type of collection. Such problems include network disconnects, hardware failures, Twitter service failure, and more. All of these problems can be handled—but not easily—and the solution requires significant engineering effort, system administation skill, and the use of (potentially unfamiliar) concurrency, distributed systems, and NoSQL techniques.

On the analysis side, further complications abound. Twitter data can be "messy" and a significant amount of time has to be invested to get it into a state where it can be effectively analyzed. In particular, each tweet can contain a different set of fields (hindering the ability to write generic processing code) and can contain text in multiple encodings (hindering the ability to parse and display them). In addition, an individual tweet does not contain all of the information that an analyst might want. For instance, the user metadata of a tweet indicates how many "followers" that user has but not who those followers are. If an analyst wants to pull follower information for the Twitter users of a data set that requires a second, significant data collection effort, subject once again to the rate limits and problems discussed above. In addition, the collection of follower graphs is a signficant engineering task itself. A collection of just "two hops" of follower/following relationships can quickly lead to millions of user objects that need to be collected, stored, and later analyzed.

Another issue that occurs with the analysis of tweets is that the values of tweet metadata (e.g., retweet count, follower count, favorite count, etc.) is dependent on when the tweet is collected. If a tweet is collected during an event, the values reflect what had happened to that particular tweet at that particular time. Thus a tweet that was just generated will have a retweet count of zero and may sit in a data set classified as "uninteresting" when it later goes on to be retweeted thousands of times. Users who had only ten followers at the time they contributed their first tweet to a data set may go on to have thousands of followers if they generate interesting content during that event. However, if that first tweet was the only one that matched the search terms being used to collect the event, subsequent analysis may classify that user as "uninteresting" since at the time his follower count was low.

One way around this problem is to collect again all tweets that make up a data set once the initial period of interest for that event is ended. This would ensure that all tweets in the data set are stored with the latest set of metadata. This approach would help reveal all tweets and users that went on to become "popular," but this approach is impractical when the size of the initial data set is large (100s of GBs). Firstly, the original metadata values would be lost or, if both versions of the tweet are kept, the size of the data set doubles. Secondly, one is required to switch from collecting tweets via the high-velocity Streaming API to the rate-limited REST API, which, for large data sets, means facing months of data collection.

Fortunately, for long-running collection events, these problems are mitigated. Interesting users gain influence by tweeting lots of useful information. Each tweet they contribute to the data set contains updated values of their followers count. A useful tweet that is retweeted multiple times is "embedded" in the retweet; the embedded tweet then contains updated values for retweet count, follower count, etc. As a result, we have developed analytical tools that track the evolution of metadata values for tweets and Twitter users over the time of a collection. Once the set of truly popular tweets and Twitter users for an event has been identified, a smaller, more focused data collection can occur via the REST API to ensure that we have the most recent metadata values for popular tweets and the complete set of tweets generated by influential users.

4 Extending the Infrastructure to Handle Analytics

The Project EPIC infrastructure, initially designed to solve the problem of collecting large amounts of social media data in a scalable and flexible fashion [1, 2], is well situated for extension to handle the challenges of social media analytics. New data stores, services, and applications can be added to the relevant layers of the architecture to provide analytical capabilities. We currently make use of four different approaches to performing analytics over our social media data sets.

Our first approach to analytics is enabled by exporting the tweets collected for an event from Cassandra to the file system of our analytics machine. Once exported, scripts written in Ruby, Python, or Java can process this data to look for the answers posed by Project EPIC analysts. These scripts are written in a highly-iterative fashion on subsets of data to demonstrate the feasiblity of answering a question or to test out the effectiveness of a particular algorithm in obtaining the answer as efficiently as possible. Once this experimental stage is complete, these algorithms are encapsulated in MapReduce jobs and applied to the entire data set.

Our second approach to analytics involves migrating event data out of Cassandra and into a NoSQL store that is suited more for analytics; while Cassandra shines as a reliable and scalable data store, it is less suited for more open-ended analytics where queries are not known ahead of time. We currently make use of MongoDB and Hadoop/HDFS for this purpose. We have written software that can export events out of Cassandra and import them into MongoDB or HDFS. We have standard queries (as described above) for these data sets written as MapReduce jobs in both MongoDB and Hadoop that will extract the information that our analysts require as a baseline. These queries currently run on a cluster of four machines generating the answers to queries on data sets with tens of millions of tweets typically in just a few hours. As both MongoDB and Hadoop are "horizontally scalable," Project EPIC is in a situation where if we need to process larger data sets (100M+ tweets) in a similar amount of time, we simply need to acquire new machines and add them to the existing cluster.

Our third approach to analytics is looking at technologies that can enable "real time analytics." For Project EPIC, real-time analytics means producing the answers to our standard queries on data sets that are still being collected. This capability would be useful both in the initial onset of an event to help our analysts determing the keywords that we should be using to collect a representative set of tweets for the event and during the event itself to highlight influential users and tweets that are worthy of additional study after the event. We have three strategies that we are pursuing in this area, all in the beginning stages of development. These approaches are 1) making use of commercial tools, 2) making use of stream-processing frameworks, and 3) making use of periodic MapReduce jobs. The first strategy is to make use of commerical software designed for the processing of time series data. We are currently using Splunk to provide a "dashboard" for monitoring the volume of tweets of an event and to make initial queries on the types of conversations that are occuring for an event.[1] We do not store all of the tweets for an event in Splunk. Rather we delete older tweets (where the definition of "old" evolves over the course of the event) to make way for more recent tweets, allowing the dashboard to present a current picture of the event to our analysts.

The second strategy is to investigate the use of stream processing frameworks, such as Storm, to generate answers to our standard questions on partially-collected data sets. Storm provides us with the ability to generate metrics as tweets arrive, before they are stored in Cassandra. These metrics (number of unique users, number of tweets per hour, etc.) can be stored in a small relational database and displayed using a web-based dashboard.

The third strategy is to make use of periodic MapReduce jobs that run on a partially collected event and report back the answers to our standard set of questions. Making use of MapReduce jobs in this fashion provides the ability to be more expressive in the types of manipulations performed on the data set, and can allow for the production of more complicated metrics, such as the number of unique users contributing tweets matching an event's keywords per hour. We are investigating whether it is more useful to run these jobs on the entire data set (which implies that performance will be less and less "real time" as the data set grows) or on a "sliding window" of tweets (e.g. the last 50,000 tweets collected) that can expand or shrink as needed to enable fast query execution. In the future, we will be looking at how technologies such as Pig and Spark enable the creation of MapReduce jobs at a higher level of abstraction. Such technologies often integrate with existing NoSQL stores allowing us to avoid migrating data and instead execute queries directly against our Cassandra data sets.

Finally, our fourth approach to analyzing our large data sets is via the use of graph databases (e.g. Neo4J, FlockDB, and Titan). While the collection of social graphs on Twitter represents a significant engineering challenge, we have software that can collect these graphs based on a given starting point (i.e. Twitter user). Now we are developing software that makes use of a graph database to read in graphs to allow our analysts to traverse the network of followers surrounding the influential contributors of a particular event with an eye towards visualizing and understanding the growth of these networks over the course of an event. We are also interested in incorporating non-Twitter data into this database where some nodes represent URLs that appear in tweets to determine if it is possible to track the flow of information about an event in and out of Twitter.

In summary, we view all four of these approaches as critical to providing a comprehensive analytics platform for crisis informatics research. To give a sense for the utility of one of these techniques, we now present a more in-depth look at the types of analysis that our work with MongoDB is providing.

5 Challenges Using MongoDB for Social Media Analysis

Project EPIC makes use of Twitter to track the social media conversation about large-scale crisis events. Members of the public increasingly make use of Twitter during mass emergencies [7]. Some of the tweets can contribute data to build "situational awareness" of the emerging event. Others have URLs to information that resides in other forums. As a result, Twitter often serves as a useful map to the range of online communications that occurred around a mass emergency event. Project EPIC makes use of Twitter's Streaming API to collect tweets; our infrastructure provides a web-based application to create/modify events, add/disable

[1]Citations to popular software tools and frameworks—e.g. Splunk, Storm, and Cassandra—are ellided as information about these projects are easily discovered via internet-based search tools.

keywords associated with an event, and manage when and for how long events are submitted to Twitter for collection. The Streaming API delivers tweets in JSON format and a varient of JSON (known as BSON) is the native format used by a popular NoSQL data store known as MongoDB. As MongoDB is designed to support arbitrary queries over objects stored in this format, MongoDB has interesting potential for supporting the analysis of large sets of tweets. MongoDB is not, however, a panacea. The use of indexes and queries on a single instance of a MongoDB server quickly runs into problems that can only be solved via the use of more complex techniques and client-server configurations.

For instance, answering the "unique number of users contributing to a data set" question identified in Sec. 2 can easily be answered in MongoDB by first importing the data set into a MongoDB collection (e.g. `tweets`) and issuing a command similar to `db.tweets.distinct("user.id_str")`.[2] This command causes the MongoDB server to look at all documents in the `tweets` collection for an attribute called "id_str" stored in an "embedded document" labelled "user." It keeps track of all the unique values found for this attribute, stores those values in an array, and, ultimately, returns that array to the caller for display and further analysis. This command executes quickly on small datasets, especially if an index has been created on the "user.id_str" attribute. However, on large data sets with many unique values, this command can encounter an internal limit of the MongoDB server which prevents the results of in-memory queries growing larger than 16MB.

For one of our smaller data sets—7.3GB of Twitter data consisting of 2.2M tweets with 932K unique users—we hit exactly this limit and were thus unable to use the `distinct()` command to calculate this particular metric. Instead, we had to make use of MongoDB's ability to run MapReduce jobs to retrieve the answer. This is straightforward to do but does represent an increase in complexity. An engineer must switch from running the simple command shown above to code similar to the following:

```
def map_command
  "function() { emit(this.user.id_str, 1); }"
end

def reduce_command
  "function(key, values) { return Array.sum(values); }"
end

mongo = MongoClient.new

db = mongo.db("norway")['tweets']

db.map_reduce(map_command,reduce_command,{:out => 'users',:verbose => true})
```

In particular, our map command is passed each input document in turn and generates an output document containing the value of the input document's `user.id_str` attribute as a key and the number one as a value, i.e., {"key" : "1234", "value" : 1}. (Note: the input document is "passed" by setting the value of `this` to point at each input document in turn.) After the map phase is complete, MongoDB combines documents that have the same key by adding their values to an array. Conceptually, this produces documents that look like this: {"key" : "1234", "value" : [1, 1, 1, 1, 1, 1]}. These combinations get passed to the reduce function which in turn produces documents that look like this: {"key" : "1234", "value" : 6}. When the reduce phase has processed all combination documents, these final output documents are stored in a `users` collection in which there is one document for each unique user in the `tweets` collection. The `value` attribute for each of these documents stores the total number of tweets that user contributed to the data set. It is still

[2]The use of `id_str` over `screen_name` is preferrable for determining unique users in a Twitter data set. Users can (and do) change the value of the `screen_name` attribute whenever they want; the `id_str` value stays constant for the lifetime of a Twitter user account.

possible to encounter out-of-memory errors using this approach if you fail to specify an output collection. In that case, MongoDB keeps all combination documents and all final output documents in memory and will return the final set of output documents to the caller when the reduce phase is done. If this in-memory set of documents consumes more than 32MB of space, the call to `map_reduce` will fail. Instead, you must tell MongoDB to store the intermediate documents in a collection (as was done above on line 13). This causes MongoDB to store the intermediate documents on disk rather than in memory and with those conditions in place it becomes possible to reliably generate the answer to the "unique users" question independent of the size of the overall data set.

However, without some additional work, the time it takes to generate the answer to this simple query can still take a long time. The default configuration for MongoDB is for it to run as a single-threaded server instance. In such a configuration, the MapReduce job above will look at all documents in a collection during the map phase sequentially and will then process all the combination documents during the reduce phase sequentially using a single thread (even on a server with multiple processors/cores). To take advantage of MapReduce's ability to run in parallel on very large data sets, MongoDB must be configured to run multiple instances of the MongoDB database on multiple servers. It must be configured to "shard" the original database across those servers. The good news here is that the programmatic interface to MongoDB does not change in this set-up, but the administration of this configuration is non-trivial and one must deal with issues such as selecting the key used to partition the database across the shards. However, the benefits of this approach is horizontal scalability in terms of performance and disk space. If one needs additional storage space or one needs queries to run faster for a given data set, adding another server to the configuration will trigger automatic repartitioning of the database allowing for increased scalability and parallelism.

Once properly configured, MongoDB can offer powerful analytical tools on large data sets via disk-based MapReduce jobs like the one above. Two analysis features that we make use of in our more complex MapReduce jobs are MongoDB's geospatial indexes and full-text indexes. For the latter, we create full-text indexes on the text of a tweet. The full-text index allow us to search data sets with more complex queries such as, "all tweets that contain 'haiti' but not 'pray'." This functionality, in turn, provides our analysts with an ability to be more expressive in their queries and to broaden their research questions. We make use of MongoDB's geospatial indexes to help narrow down tweets in a given data set to just those users who were generating tweets local to an event. For instance, our 132 GB Hurricane Sandy data set contains 26M tweets generated by 8.2M users from all over the world. However, for one of our research questions, our analysts wanted to examine only tweets generated by users who were located on the eastern seaboard of the United States right before the landfall of the hurricane. After importing the data set and ensuring that the geocoordinates of the tweets matched the format expected by MongoDB, we generated a geospatial index and wrote a MapReduce job that searched for tweets that fell within a particular bounding box. The query took 13 minutes to run on our cluster and located the 101K users who were directly in the hurricane's path. This result was immediately useful as we were able to launch a new data collection in which all of the tweets ever generated by those 101K users were collected and stored for later analysis (looking, for instance, to see if these users had adopted Twitter because of Hurricane Sandy).

6 Conclusions

In this paper, we have highlighted the challenges associated with collecting and analyzing social media data. While our experience is related to the support of crisis informatics research, many of these issues are independent of application domain and our approach, techniques, software architecture—and its implementation as Project EPIC's software infrastructure—are broadly applicable to the collection and analysis of a wide variety of application domains and types of social media. We have learned valuable lessons, including insight into the questions most important to crisis informatics researchers, the challenges associated with collecting representative Twitter data sets, and the need for multiple approaches to analysis. A single analysis technique is insufficient to answer the wide range of questions posed across the different timescales in which those answers are needed. We also learned that it is critical to develop a reasoned, modular, and extensible approach to the design and development of social media analysis tools. It is otherwise impossible to perform this analysis at scale with sufficient functionality, flexibility, and performance to make meaningful progress on societal-scale issues.

One important implication of our experience is how work in this area requires a large team with a diverse set of skills and expertise to do this type of research effectively. Project EPIC required a team with skills not only in software architecture, software design, software engineering, web engineering, distributed systems, networking, and system administration, but also with skills in human-centered computing, information visualization, statistics, and qualitative/quantitative experimental methods. The former were needed to create, deploy, and operate the scalable, flexible, reliable, and robust Project EPIC software infrastructure but the latter were needed to identify the problems it solves in the first place, what questions it must answer, how it should present those answers, and how it supports the ongoing, highly-iterative, multi-method techniques required by crisis informatics research. With such a team, our interdisciplinary approach to the collection and analysis of social media has enabled significant progress to be made in the area of crisis informatics research.

Acknowledgment

This material is based upon work sponsored by the NSF under Grant IIS-0910586.

References

[1] Anderson, K., and Schram, A. (2011). Design and Implementation of a Data Analytics Infrastructure in Support of Crisis Informatics Research: NIER Track. In *33rd International Conference on Software Engineering*, pp. 844–847.

[2] Anderson, K., and Schram, A. (2012). MySQL to NoSQL: Data Modeling Challenges in Supporting Scalability. In *ACM Conference on Systems, Programming, Languages and Applications: Software for Humanity*, pp. 191–202.

[3] Hagar, C., and C. Haythornthwaite. (2005). Crisis, Farming and Community. *Journal of Community Informatics*, 1(3):41âĂŞ-52.

[4] Hagar, C. (2009). The Information and Social Needs of Cumbrian Farmers during the UK 2001 Foot and Mouth Disease Outbreak and the Role of Information and Communication Technologies. In *The Socio-Cultural Impact of Foot and Mouth Disease in the UK in 2001: Experiences and Analyses*, eds. Döring, M. and Nerlich, B. pp. 186–199. Manchester University Press.

[5] McTaggart, C. (2012). Analysis and Implementation of Software Tools to Support Research in Crisis Informatics. MS Thesis. University of Colorado. 65 pages.

[6] Palen, L., and Liu, S. (2007). Citizen Communications in Crisis: Anticipating a Future of ICT-Supported Participation. In *ACM Conference on Human Factors in Computing Systems*, pp. 727–736.

[7] Palen, L., and Vieweg, S. (2008). The Emergence of Online Widescale Interaction in Unexpected Events: Assistance, Alliance and Retreat. In *ACM Computer Supported Cooperative Work*, pp. 117–126.

[8] Palen, L., Martin, J., Anderson, K., and Sicker, D. (2009). Widescale Computer-Mediated Communication in Crisis Response: Roles, Trust & Accuracy in the Social Distribution of Information. `http://www.nsf.gov/awardsearch/showAward.do?AwardNumber=0910586`.

[9] Palen, L., Anderson, K., Mark, G., Martin, J., Sicker, D., Palmer, M., and Grunwald, D. (2010). A Vision for Technology-Mediated Support for Public Participation & Assistance in Mass Emergencies & Disasters. In *ACM and British Computing SocietyâĂŹs 2010 Visions of Computer Science*, 10 pages.

[10] Palen, L, Vieweg, S., and Anderson, K. (2011). Supporting "Everyday Analysts" in Safety- and Time-Critical Situations. *The Information Society Journal*, 27(1): 52–62.

Nature of Information, People, and Relationships in Digital Social Networks

Rakesh Agrawal
Microsoft Research
Mountain View, CA 94043, U.S.A.
rakesha@microsoft.com

Abstract

This paper summarizes the results of our recent investigations into how information propagates, how people assimilate information, and how people form relationships to gain information in Internet-centric social settings. It includes key ideas related to the role of the nature of information items in information diffusion as well as the notion of receptivity on part of the receiver and how it affects information assimilation and opinion formation. It describes a system that incorporates availability, willingness, and knowledge in recommending friends to a person seeking advice from social network. It discusses whether having common interests makes it more likely for a pair of users to be friends and whether being friends influences the likelihood of having common interests, and quantifies the influence of various factors in an individual's continued relationship with a social group. Finally, it gives current research directions related to privacy and social analytics.

1 Introduction

The mission of Microsoft Research's Search Labs in Silicon Valley that I lead is to advance the state of art in Internet technologies and Internet-based applications. One of our focus areas is to understand how information propagates, how people assimilate information, and how people form relationships to gain information in Internet-centric social settings. This paper presents a condensed overview of some of our recent research on these topics. It includes key ideas related to the role of the nature of information items in information diffusion, presented by Agrawal, Potamias, and Terzi in [1]. It also discusses the notion of receptivity on part of the receiver and how it affects information assimilation from the same paper. Related to the same topic, it introduces the work of Bhawalkar, Gollapudi, and Munagala on opinion formation games from [2] and that of Das, Gollapudi, Panigrahy, and Salek on dynamics of opinion formation from [5]. It then reviews the system of Nandi, Paparizos, Shafer, and Agrawal that factors in availability, willingness, and knowledge to recommend friends for person to turn to for advice. Next, it recalls the work of Lauw, Shafer, Agrawal, and Ntoulas from [6] to shed light on whether having common interests makes it more likely for a pair of users to be friends, and whether being friends influences the likelihood of having common interests. Finally, it abstracts the work of Budak and Agrawal from [3] on factors that influence an individual's continued relationship with a social group. The work in [3] is based on data from thirty Twitter chat groups; algorithmic mining of chat groups from Twitter stream is described by Cook, Kenthapadi, and Mishra in [4].

Mea Culpa: Given the space restriction, I have prioritized the presentation of Search Labs work over discussion of related research. For the latter, I refer the reader to the original papers.

2 Nature of Information

A key issue in social networks is understanding how people assimilate information in their daily lives. Recent research has focused on understanding the role that *node characteristics* (i.e., homophily) and *peer influence*, (i.e., link structure), play in explaining the appearance of information items on certain nodes of the social network. The underlying assumption is that it is the nature of the people, or the nature of the people's connections, which determines the form of information cascades.

While we recognize the importance of network structure and nodes' characteristics on information propagation, we postulate in [1] that the very nature of information items is an additional important parameter that affects the observed spread. We claim that certain information items are *endogenous* and they indeed propagate primarily through the connections between the nodes. On the other hand, some information items are *exogenous* – they will be acquired by many nodes independently of the underlying network. Given a social network and data related to the ordering of adoption of information items by nodes, our goal is to develop a framework for estimating endogeneity and exogeneity parameters.

E2 **Model:** Consider a social network $G = (V, E)$ of $|V| = n$ users, in which there is a link $(u \rightarrow u')$ between two nodes $u, u' \in V$, if node u *follows* node u'. Such a directed link suggests that there is potential of information propagation from u' to u. Assume a finite set of information items $\mathcal{I}$ with $|\mathcal{I}| = m$.

At every point in time t, every node $u \in V$ is associated with an m-dimensional vector A_u^t, whence $A_u^t(i) = 1$ if node u is *active* with respect to information item i at time t; otherwise $A_u^t(i) = 0$. If $A_u^{(t-1)}(i) = 0$ and $A_u^t(i) = 1$, then we say that an *activation* has occurred to node u with respect to item i at time t. The *observed* activation state at the end of the observation period is encoded in $\mathbf{A}$ such that $\mathbf{A}(u, i) = 1$ iff node u has, at some point, become active with respect to item i. Give the sequence of activations encoded in vectors A_u^t, one can construct the *active-neighborhood* matrix $\mathbf{\Gamma}$, such that $\mathbf{\Gamma}(u, i)$ denotes the number of neighbors of u that were active with respect to item i, the moment u became active with respect to i. If $\mathbf{A}(u, i) = 0$, then $\mathbf{\Gamma}(u, i)$ is the number of neighbors of u that were active at the end of the observation period.

Every item $i \in \mathcal{I}$ is characterized by a pair of parameters $\theta_i = (e_i, x_i)$, where $e_i \in [0, 1]$ is its *endogeneity* and $x_i \in [0, 1]$ is its *exogeneity*. Endogeneity characterizes the item's tendency to propagate through the network due to the peer effect. Exogeneity captures the item's tendency to be independently generated by nodes in the network. Parameters e_i and x_i have a probability interpretation: node u becomes active with respect to i, independently of its neighbors, with probability x_i. If u has $\mathbf{\Gamma}(u, i)$ neighbors that are already active with respect to i, then each one of them succeeds in activating u with probability e_i. At the end of the observation period, u becomes active with respect to i, with probability: $1 - (1 - x_i)(1 - e_i)^{\mathbf{\Gamma}(u,i)}$. Use $\mathbf{e}$ and $\mathbf{x}$ to represent the vectors of all items' endogeneity and exogeneity parameters, and use $\mathbf{\Theta} = \langle \mathbf{e}, \mathbf{x} \rangle$ to denote the vector of these pairs of values for all items.

Generative Process: Our model defines a generative process in which every item $i \in \mathcal{I}$ is given a set of chances to activate the nodes in $G = (V, E)$. Intuitively, for every item $i \in \mathcal{I}$, our model assumes *activation graph* $H_i = (V \cup \{s_i\}, E_i)$. The nodes of H_i consist of all the nodes in V plus an additional node s_i that corresponds to item i. The set of links E_i contains all the links in E plus n additional directed links $(u \rightarrow s_i)$. That is, in H_i every node *follows* the item-node s_i. Initially, only node s_i is active and the rest n nodes are inactive. An information item propagates from an active node only to its inactive followers. The activation proceeds in discrete steps. At each time step, activation of any node u, through links $(u \rightarrow s_i)$, succeeds with probability x_i. At the same time, activation of u through links $(u \rightarrow u')$ for $u' \in V$ succeeds with probability e_i. At most one activation attempt can be made by every link. The final activation state of all nodes with respect to all items is stored in the final activation matrix $\mathbf{A}$.

Problem Definition: Given the active-neighborhood information $\mathbf{\Gamma}$ and parameters $\mathbf{\Theta}$, the likelihood of the observed activation matrix $\mathbf{A}$ can be computed as:

$$\Pr\left(\mathbf{A} \mid \mathbf{\Gamma}, \mathbf{\Theta}\right) = \prod_{i=1}^{m}\prod_{u=1}^{n} \Pr\left(\mathbf{A}(u,i) \mid \mathbf{\Gamma}(u,i), e_i, x_i\right). \tag{1}$$

Given $\mathbf{\Gamma}$ and $\mathbf{A}$, we want to estimate vectors $\mathbf{e}$ and $\mathbf{x}$ such that the compatibility between the observed activation matrix $\mathbf{A}$ and the estimated parameters, $\mathbf{\Theta} = \langle \mathbf{e}, \mathbf{x} \rangle$, is maximized. Different definitions of compatibility lead to different problems. We focus on the parameters $\mathbf{\Theta}$ that maximize the loglikelihood of the data:

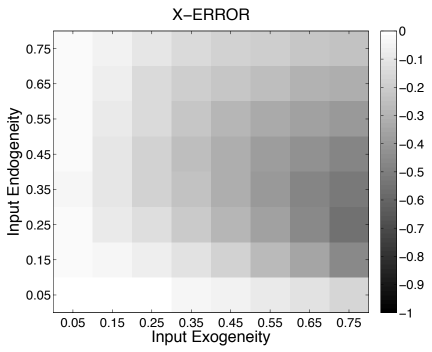
$$\mathbf{\Theta} = \arg\max_{\mathbf{\Theta}'} \mathcal{L}\left(\mathbf{A} \mid \mathbf{\Gamma}, \mathbf{\Theta}'\right) = \arg\max_{\mathbf{\Theta}'} \log \Pr\left(\mathbf{A} \mid \mathbf{\Gamma}, \mathbf{\Theta}'\right)$$

Parameter Estimation: Using Eq. (1), we rewrite the likelihood as

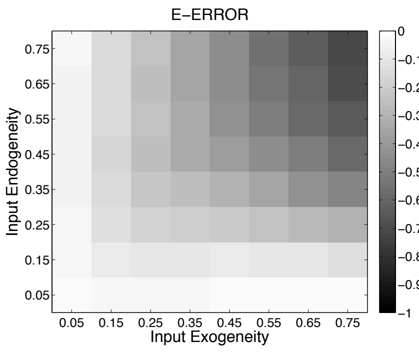
$$\mathcal{L}\left(\mathbf{A} \mid \mathbf{\Gamma}, \mathbf{\Theta}\right) = \sum_{i \in \mathcal{I}} \sum_{u \in V} \log\left(\Pr\left(\mathbf{A}(u,i) \mid \mathbf{\Gamma}(u,i), e_i, x_i\right)\right).$$

Thus, the parameters (e_i, x_i) of every item i can be computed independently by solving a two-variable optimization problem in the $[0,1] \times [0,1]$ range. Further, the independence of the items allows us to parallelize the item-parameter estimation. The function L_i is convex with respect to the item's parameters (e_i, x_i). Therefore, an off-the-shelf optimization method (e.g., Newton Raphson method) can be used to efficiently find the optimal values of the parameters.

Experiments with Synthetic Data: The goal of synthetic data experiments is to study how well the parameter estimation procedure recovers exogeneity and endogeneity values. Define the *exogeneity absolute error* for the exogeneity parameters as X-ERROR$(\mathbf{\Theta}, \widehat{\mathbf{\Theta}}) = 1/m \sum_{i \in \mathcal{I}} |x_i - \widehat{x}_i|$, where $\widehat{x}_i$ is the recovered value of the parameter x_i. The *endogeneity absolute error*, E-ERROR, is defined similarly. Figure 1 shows the results.



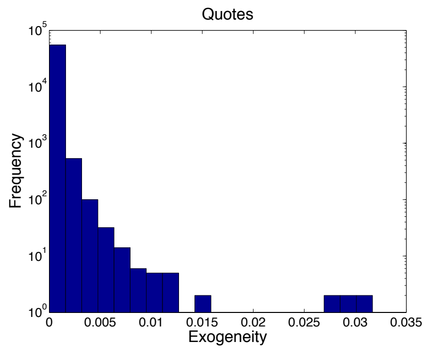
(a) X-ERROR



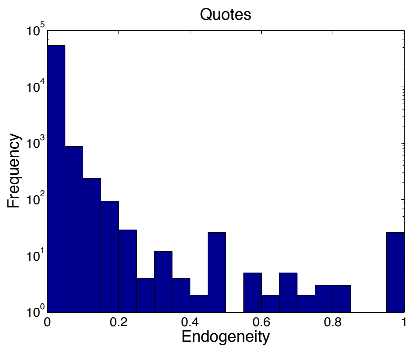
(b) E-ERROR

Figure 1: Synthetic `ScaleFree` graphs: #nodes=1000, density=1%, #items=1000, endogeneity and exogeneity $\in [0, 0.8]$ (separately picked uniformly at random).

We see that the smaller the values of the input parameters, the lower the X-ERROR and the E-ERROR. Small values of these parameters generate sparse data, i.e., data with small number of activations. Real data exhibit this behavior; the most frequent item in the dataset we consider in the next section appears in less than 10% of the nodes.



(a) Exogeneity



(b) Endogeneity

Figure 2: Histogram of exogeneity and endogeneity of quotes in `MemeTracker`.

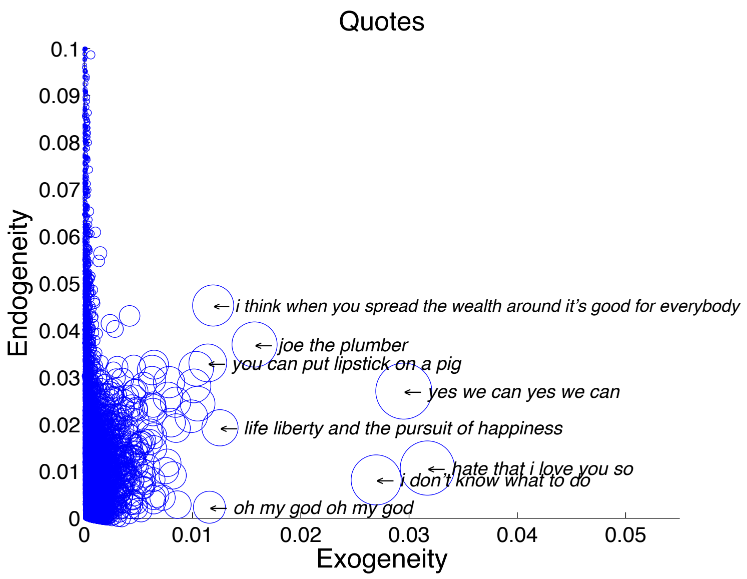


Figure 3: Scatter plot of Exogeneity and endogeneity of quotes (marker area $\propto$ frequency).

Experiments with `MemeTracker` Data: We next turn our attention to real data. We use the memetracker data available from Stanford University, which consists of quotes that have been posted on articles/blogposts from August 2008 to April 2009. Timestamps in the data capture the time that a quote was used in a post. From these data, we construct our network $G_B = (V_B, E_B)$ by selecting as nodes all the blogs hosted either by *blogspot.com* or by *wordpress.com*. For blogs $b, b' \in V_B$, there is a directed link $(b \to b')$ if there exists at least one blogpost of b linking to b'. The set of information items consists of the set of quotes that appeared in at least one blogpost of any of the blogs in V_B. We say that blog u became active with respect to quote q at time t, if t was the first timestamp that u used q in one of his blogposts.

Figure 2 plots the distribution of endogeneity and exogeneity values of the quotes. The skewed distribution of both exogeneity and endogeneity values shows that a non-negligible number of quotes are much more endogenous/ exogenous than most quotes. Figure 3 is a scatter-plot of the exogeneity and endogeneity values of the quotes. The area covered by each marker is proportional to the number of nodes it appears. For concreteness, we have also shown frequent quotes for some combinations of endogeneity and exogeneity values. Clearly, exogeneity and endogeneity are not correlated; there some quotes that have high endogeneity but low endogneity and vice versa.

Table 1: Top-5 frequent quotes.

	Exogeneity=H Endogeneity=H
1.	yes we can yes we can
2.	hate that i love you so
3.	joe the plumber
4.	i think when you spread the wealth around it's good for everybody
5.	you can put lipstick on a pig
	Exogeneity=H Endogeneity=L
1.	i don't know what to do
2.	oh my god oh my god
3.	hi how are you doing today
4.	why where are you going to john
5.	what is it
	Exogeneity=L Endogeneity=H
1.	there appears to be a sizeable number of duplicate and fraudulent applications
2.	we shouldn't let partisan politics derail what are very important things that need to get done
3.	likened zionist settlers on the west bank to osama bin laden saying both had been blinded by ideology
4.	as far as the eye can see
5.	she doesn't know yet that she has been married
	Exogeneity=L Endogeneity=L
1.	the age of turbulence adventures in a new world
2.	i've got friends in low places
3.	you shall not bear false witness against your
4.	neighbor instead of complaining about the state of the education system as we correct the same mistakes year after year i've got a better idea
5.	a woman who loves me as much as she loves anything in this world but who once confessed her...

Table 1 shows the top-5 frequent quotes for combinations of high and low exogeneity and endogeneity values. We make two observations: first, that quotes with "Exogeneity=H" exhibit shorter length than quotes with "Exogeneity=L". Second, a web search reveals that most quotes with "Endogeneity=H" were news-stories or popular quotes of the observation period. Amongst the high-exogeneity quotes, we can distinguish between those with "Endogeneity=H" and those with "Endogeneity=L". Quotes "*joe the plumber*", "*you can put lipstick on a pig*" etc. from the (H,H) bucket are front-page quotes that drew notable attention during the 2008 elections period. They are highly exogenous because they gained popularity via external media such as the television. They are also highly endogenous because they heavily propagated through the network links of the blogs. In contrast, (H,L) quotes: "*i don't know what to do*", "*oh my god*", "*hi how are you doing today*", and "*what is it*", are popular phrases that appear in various contexts ranging from casual conversations to pop songs. Such quotes are expected to be purely exogenous – they do not trigger cascades.

Amongst the low-exogeneity quotes, we can again distinguish between those for which "Endogeneity=H" and those with "Endogeneity=L". The first correspond to long phrases that were news stories during the observation period. For example, the quote "*she doesn't know yet that she has been married*", propagated in a set of connected blogs that discussed the case of the marriage of a fourth-grade girl. Similarly, the rest of the quotes in (L,H) (except for "*as far as the eye can see*") were also news stories of that period. These are highly endogenous quotes. Compare these quotes with the quotes in bucket (L,L). Neither exogenous sources nor peer influence affect the propagation of these quotes. These are all infrequently occurring phrases, e.g., lyrics from older songs and previous year book titles.

3 Nature of People

Although E2 models the observed variation between information items, it does not capture that different people may react differently to the same information item. The E2R model incorporates a *receptivity* parameter to capture this difference in the nature of people.

E2R **Model:** Associate with every node u a parameter $r_u \in [0,1]$ that quantifies the node's tendency to be *receptive* to information items coming either from u's neighbors or from sources outside the network. Same as with e_i and x_i, r_u has a probabilistic interpretation: node u accepts any candidate activation with probability r_u. Then, the probability of the observed activation matrix $\mathbf{A}$ given the item parameters $\mathbf{\Theta}$ and user receptivities $\mathbf{r}$ is:

$$\Pr(\mathbf{A} \mid \mathbf{\Gamma}, \mathbf{\Theta}, \mathbf{r}) = \prod_{i\in\mathcal{I}, u\in V} \Pr(\mathbf{A}(u,i) \mid \mathbf{\Gamma}(u,i), e_i, x_i, r_u).$$

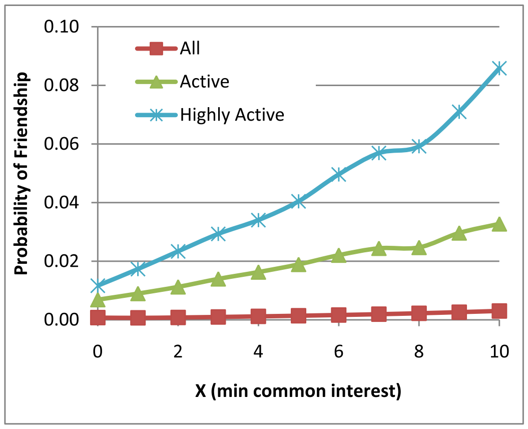
The probability of node u being active with respect to item i is computed as:

$$\Pr(\mathbf{A}(u,i) = 1 | \mathbf{\Gamma}(u,i), e_i, x_i, r_u) = 1 - (1 - r_u \cdot x_i)(1 - r_u \cdot e_i)^{\mathbf{\Gamma}(u,i)}.$$

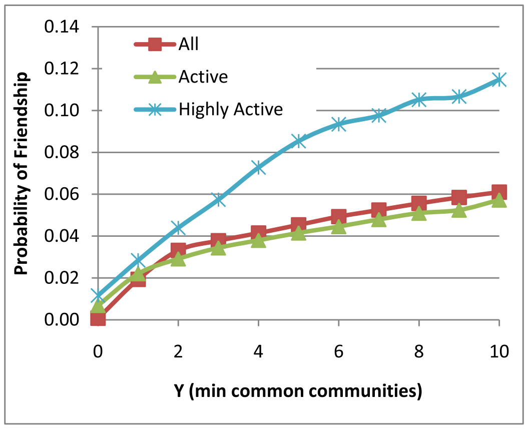
Intuitively, every time we have an endogenous or exogenous attempt to activate a user, the user also needs to accept that activation. Receptivity is both a characteristic of the nodes and a means to allow items to reveal their true nature. Consider the extreme case of a very endogenous item that all, but a small fraction of the nodes, adopt through their neighbors. In order to capture the behavior of this minority of nodes, the E2 model would assign to i endogeneity value lower than 1. On the other hand, the E2R model will capture the behavior of these nodes through receptivity and will assign to i larger endogeneity value, allowing it to reveal its true nature.

See [1] for further details, computational techniques, and experimental results.

Dynamics of Opinion Formation: In a recent work [5], we differentiate between the innate and expressed opinions and postulate that individuals update their expressed opinions in discrete time steps by taking a convex combination of their innate opinion and the expressed opinions of their social neighbors. The weights in the convex combination depends on a user's *propensity to conform*, which is reminiscent of the idea of receptivity just discussed. Through real-world experiments, they show that this value is largely specific to a given user and does not change significantly from topic to topic. In [2], we present game-theoretic models of opinion formation where opinions themselves co-evolve with friendships. In these models, nodes form their opinions by maximizing agreements with friends weighted by the strength of the relationships, which in turn depend on difference in opinion with the respective friends.



(a) Given common interests



(b) Given common communities

Figure 4: Probability of friendship

Availability, Willingness, and Knowledge: A typical person has many friends that the person can consult for opinions and advice. However, public broadcasting a question can use up social capital and the request can get lost in a myriad of status updates. Direct messaging requires manual selection and a user may have difficulty guessing which of the friends will be able to provide a quality answer in a timely manner. In [7], we describe a decision aide that provides the ranked subset of friends for a user to seek. The system mines social network data focusing on a novel set of criteria: availability, willingness and knowledge. The system response depends on (1) how likely it is that a friend is online in the near future based on past activity patterns, (2) the likelihood that a friend will respond based on the strength and nature of the interpersonal connection and past interaction behavior, and (3) a friend's knowledge and expertise on a topic and their potential for providing an informed response based on the past message content.

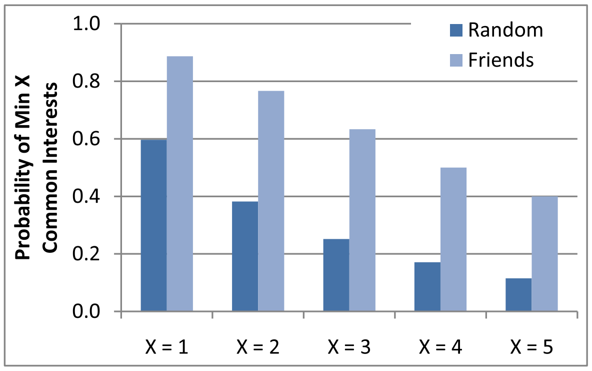
4 Nature of Relationships

Interests and Friendship: In [6], we use LiveJournal data to investigate two central questions: (1) whether having common interests makes it more likely for a pair of users to be friends, and (2) whether being friends influences the likelihood of having common interests. LiveJournal users identify each other as friends and express their interests in two ways. First, users have a list of self-proclaimed interests on their User Info page. Second, users can subscribe to communities or group blogs oriented around a given topic. We extract three binary adjacency matrices from LiveJournal data: (1) F, a user $\times$ user friendship matrix, with $F_{uu'} = 1$ iff users u and u' have friended each other, (2) I, a user $\times$ interest matrix, with $F_{ui} = 1$ iff user u specifies i as an interest, and (3) C, a user $\times$ community matrix, with $C_{uc} = 1$ iff user u watches community c.

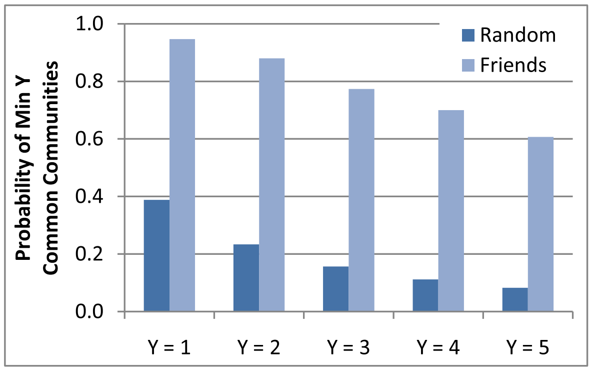
Without any prior information, the best estimate for the probability of friendship is the fraction of random pairs that turn out to be friends. Conditional on that a pair of users share a minimum number of X interests, the probability of friendship is:

$$P(friendship \mid X) = \frac{|\{(u,u') \in U \times U \mid (\mathbf{F}_{uu'} = 1) \wedge (\mathbf{I}_u \cdot \mathbf{I}_{u'} \geq X)\}|}{|\{(u,u') \in U \times U \mid (u \neq u') \wedge (\mathbf{I}_u \cdot \mathbf{I}_{u'} \geq X)\}|},$$

where U denotes the set of users in consideration. Fig. 4(a) plots $P(friendship \mid X)$ for different values of X and different subsets of users; Active (Highly Active) users have at least ten (fifty) each of friends, interests, and communities. We see that having common interests, even just one, significantly increases the probability of friendship for all data sets. This trend is also monotonic: higher X leads to higher probability.



(a) Common Interests



(b) Common Communities

Figure 5: Probability of Commonality Given Friendship

This is a surprising outcome, given that without geographic constraint, we would not expect the conditional probability to be significantly higher. It suggests that an underlying factor is at work in LiveJournal that encourages users to make friends with those having common interests. Several LiveJournal features might contribute to this. For every interest with more than one claimant, LiveJournal provides a hyperlink to the list of users who claim that interest, thus letting one user find others to connect with on the basis of interest. Blogging and commenting is another set of activities that could help users get to know others who share similar interests.

We next investigate whether a similar relationship exists between friendship and common communities. The probability of friendship given that a user pair shares a minimum of Y common communities is:

$$P(friendship \mid Y) = \frac{|\{(u,u') \in U \times U \mid (\mathbf{F}_{uu'} = 1) \wedge \ (\mathbf{C}_u \cdot \mathbf{C}_{u'} \geq Y)\}|}{|\{(u,u') \in U \times U \mid (u \neq u') \ \wedge \ (\mathbf{C}_u \cdot \mathbf{C}_{u'} \geq Y)\}|}.$$

Fig. 4(b) plots $P(friendship \mid Y)$ for different Y values and data sets. We observe similar trends as those in Fig. 4(a): a user pair is monotonically more likely to consist of friends if they share more common communities.

To study the second question raised at the beginning of this section, we write the probability that a pair of friends shares at least X common interests as:

$$P(X \mid friendship) = \frac{|\{(u,u') \in U \times U \mid (\mathbf{F}_{uu'} = 1) \wedge \ (\mathbf{I}_u \cdot \mathbf{I}_{u')} \geq X\}|}{\sum_{u \in U} \sum_{u' \in U} \ F_{uu'}}.$$

Fig. 5(a) compares $P(X \mid friendship)$ to $P(X)$ for different values of X on Highly Active subset of users. Similar trends are observed on other datasets. It shows that for every X, $P(X \mid friendship)$ is significantly higher – between 1.5 and 3.5 times higher – than $P(X)$. The likelihood of common interests conditioned on friendship is as high as $P(X = 1 \mid friendship) = 0.89$ and $P(X = 2 \mid friendship) = 0.77$. This result suggests that friendship is a potentially significant source of signals in inferring a person's interests.

We conducted a similar exercise on communities. Fig 5(b) plots $P(Y)$ and $P(Y \mid friendship)$ for various Y's and for the Highly Active dataset. We see similar trends as in Fig. 5(a), but the difference is even higher. $P(Y \mid friendship)$ is 2.4 to 7.3 times higher than $P(Y)$, suggesting that friendship is an even stronger signal in detecting common communities.

See [6] for extensions where friendship has strength associated with it.

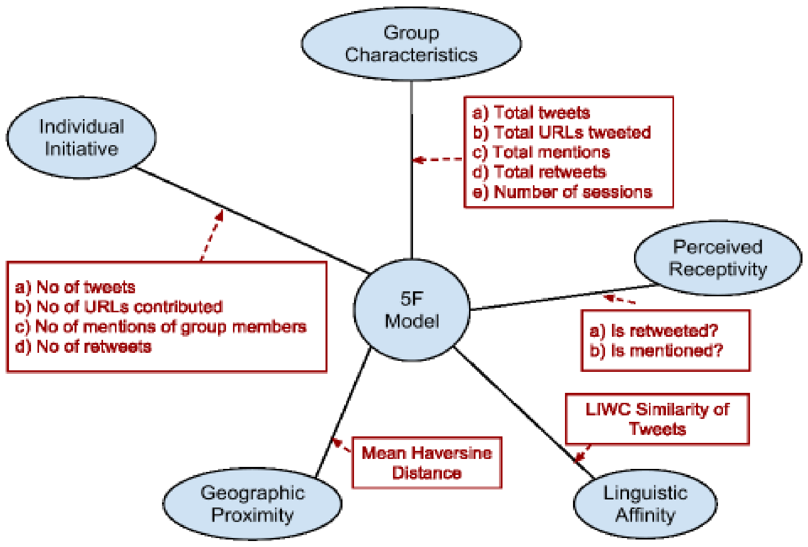


Figure 6: Overview of the *5F Model*

Group Participation: In [3], we study what makes a person become a member of a group. We addressed this question in the context of Twitter chats, which are time-bound synchronous group interactions carried out in real time on a focused topic. For instance, #engchat is a chat about English education held at 7-8pm EST on every Monday. During a chat session, the participants continuously interact on the designated topic by tweeting their opinions and marking their tweets with the hashtag of the particular chat group. While weekly groups like #engchat are the most common ones, there are others such as #mathchat that meet twice a week, #collegechat that meet bi-weekly or #edchat that are week-long conversations. Most of the chat groups also have dedicated blogs that provide various resources such as transcripts of past sessions and schedule of upcoming discussions. In a companion work [4], we describe algorithms for mining chat groups from Twitter data stream.

We developed *5F Model* that predicts whether a person attending her first chat session in a particular Twitter chat group will return to the group. This model, pictorially depicted in Figure 6, considers five different classes of factors: *individual-initiative, group characteristics, perceived receptivity, linguistic affinity* and *geographical proximity*. For example, the number of tweets, the number of URLS in the tweets, the number of mentions and retweets contributed by the person during her first session provide indication of her individual initiative. Using data from thirty education-related chat groups, we study the predictive power of these factors individually as well as collectively. We use logistic regression for statistical analysis and a *Pseudo-R* measure (Nagelkerke R^2 Index) to compare the models.

The regression results are summarized in Table 2. This table has four columns. The first column is the name of the model and corresponds to one of the five factors. The second column lists the Twitter specific variables used for each of the corresponding factors. The third column consists of two subcolumns. The first subcolumn shows the cofficients of the corresponding explanatory variables in the individual-level models, whereas the second subcolumn gives the coefficients for the unfied model. The third column gives the pseudo-R measure for the individual models. The pseudo-R value for the unfied model is 0.14 and is shown at the bottom of the table. The statistically significant variables are marked with * for p-value < 0.05, ** for p-value < 0.01 and *** for p-value < 0.001.

Individual initiative model: The results show that all the variables except for *usermentions* are statistically significant. The number of tweets are positively correlated with returning to the chat group, emphasizing the

Factors	Variables	Coefficients		Pseudo-R
		Individual Model	Unified 5F Model	
Individual Initiative	usermentions userretweets userurl usertweetcount	-0.016 -0.13*** -0.16*** 0.147***	-0.007 -0.077*** -0.092*** 0.05***	0.09
Group Characteristics	groupmentions groupretweets sessionurl sessiontweetcount groupmaturity	-0.0001 0.0014* -0.003*** -0.0005 -0.01***	-0.0004 0.002*** -0.002* -0.0008* -0.007***	0.03
Perceived Receptivity	ismentioned isretweeted	1*** 0.69***	0.445*** 0.24	0.08
Linguistic Affinity	liwccors	2.159***	1.215***	0.1
Geographical Proximity	distance	-0.00005***	-	0.01

Pseudo-R for the unified *5F Model* = 0.14
* $p < .05$, ** $p < .01$, *** $p < .001$

Table 2: Results of Statistical Analysis

predictive power of early interest exhibited by the user. The variable *userurl* is negatively correlated with returning to the group. One possible explanation for this result can be given as follows: For users that share a large number of urls, i.e. users that already acquire a certain level of knowledge, the added informational gain from chat sessions can be smaller, resulting in less incentive to attend future sessions.

The negative correlation for *userretweets* indicates that retweeting behavior can be used to distinguish *real* participants of chat groups from those that are merely retweeting the tweets of their friends who are attending a chat session. Consider the following illustrative scenario. Assume that *user1* attending *#1stchat* shares a tweet “Check out article bit.ly/342dfser #1stchat”. This tweet is seen not only by the attendees of *#1stchat* but also the followers of *user1*. One such follower, say *user2*, can find the tweet interesting and retweet it. Here, *user2* who *appears* to be attending his first *#1stchat* session may not return to this group.

Group characteristics model: Statistically significant variables are *groupretweets, sessiontweetcount, sessionurl* and *groupmaturity*. Capturing the significance of information overload, *sessionurl* and *sessiontweetcount* have negative correlation. The variable *groupmaturity* has negative correlation with the odds of come back, i.e. users that attempt to join more mature groups are less likely to return to the group. The results also indicate the significance of *informational influence* as demonstrated by the statistical significance and positive correlation of *groupretweets*. However we observe that the correlations of these factors are relatively mild. For instance, an increase of 1 retweet in group discussion decreases the log odds of come back by 0.0014. *Pseudo-R*(=0.03) values for this model are worse when compared to those of *individual initiative model*, showing that *individual initiative* factors are relatively better indicators of future participation.

Perceived receptivity model: Our results show the importance of social inclusion in ensuring continued participation. For instance, the log odds of returning to a group increases by 1 if a user is mentioned in the first session that he/she attends. Similarly, the odds of returning improves by 0.69 if the user is retweeted by others in the chat session. Both of these findings are statistically significant. This result is in agreement with relevant research in other online communities.

Linguistic affinity model: We make use of the *Linguistic Inquiry and Word Count (LIWC)* tool to compare linguistic markers between a user and a group. We consider the set of tweets a user shares in her first session as a text document and compute the value of each linguistic marker to obtain her *LIWC-vector* for that particular session. Similarly, we aggregate all the tweets from other users and compute the *LIWC vector* of the group. To compute affinity, we use Pearson correlation measure. We find that linguistic affinity is statistically significant and highly correlated with returning to a chat group, which is in line with research in social sciences, particularly the *speech codes theory*. The highest *Pseudo-R* value for this model shows that the linguistic characteristics are the best indicators of future participation.

Geographical proximity model: To study the influence of geographical proximity, we calculate the mean distance of the user to everyone else in the group using the Haversine formula. The location for each user is determined based on the location field of the user profile. We see that returning to a group is only mildly correlated with geographical proximity. An increased distance of 1km reduces the log odds of returning to the group by only 0.00005. Regression tasks performed *per-chat group* showed that geographical proximity is statistically significant for only seven educational Twitter chats. Two chats had positive correlation and five had negative correlation. For instance, #globalclassroom has positive correlation with the variable *distance*, indicating the positive effect of diverse locations in returning to the group. Such behavior is to be expected given the global goal of this particular group. Yet groups like #jedchat have negative correlation with increased distance. This group is on Jewish education and is mostly popular in Israel. Overall, the *Pseudo-R* value for this model is the worst among all models, showing that geographical characteristics are generally poor indicators of future participation.

Unified 5F Model: In this model, we consider all the explanatory variables in conjunction, except geographic proximity (*distance*). The reason for omitting the latter is that we could determine the location of only a subset of users and this factor anyway turned out to have limited fit. As expected, this model has the largest *Pseudo-R* value. Each independent variable has similar explanatory trend as we observed with individual models.

User Survey: We complemented the results from the statistical data analysis with a user survey to directly understand from users involved in Twitter chats their attitudes towards these chats. The survey had three main parts, addressing questions related to: (1) usage, advantages and disadvantages, (2) sense of community and responsibility, and (3) evolution of participation. The survey was tweeted through the hashtag of each chat group studied. Respondents of the survey were encouraged to share the survey with their Twitter followers.

The survey results highlighted various distinctions between Twitter chats and other online groups and face-to-face discussions. We found *informational* support to be more important to Twitter chat members than *emotional support*. Although prior work suggests that informational support is negatively correlated with the sense of community, we found the sense of community to be very strong in Twitter chats. In fact, its members communicate with one another outside chat sessions much more than expected from the literature. Disadvantages identified by the survey respondents also mark an interesting distinction between Twitter chats and other online groups. While for other online communities, the lack of face-to-face interactions is a main disadvantage, Twitter chat users focus on the content. More specifically, due to the synchronous and open nature of Twitter, the pace of information is the biggest challenge of Twitter chats.

The survey results reinforced most findings of the statistical analysis. Groups becoming closed to new members over time (as captured by *groupmaturity* in our model) is seen anecdotally in survey results. The importance of social inclusion is also observed in the responses of two survey participants that reduced (one ending) their participation due to the lack of receptivity. The geographical diversity listed as an advantage in the survey also indicates that geography is not a limiting factor for Twitter chats.

5 Ongoing Work

Social analytics continue to provide us opportunities for exploring critical issues and building useful systems. Some of our current research directions include:

- Many users of social media entertain an illusory sense of “privacy by hiding in the crowd”. We are interested in ascertaining if one could accurately determine a user’s attributes by building an inference system over the history of user interactions, and thus shattering this illusion. We are also interested in exploring what a user can do in order to achieve privacy (short of not participating in social media).
- A huge potential exists to leverage aggregate information from social media, news sites, and the internet as a whole for enterprise and market insights as well as enabling interesting user applications. We aim to ingest, mine, and analyze such information in order to enable a wide variety of social intelligence applications and provide useful insights by identifying interesting patterns, alerting users to unusual or trending events, allowing adhoc, “what if" analysis, and other capabilities.

References

[1] R. Agrawal, M. Potamias, and E. Terzi. Learning the nature of information in social networks. In *ICWSM*, 2012.

[2] K. Bhawalkar, S. Gollapudi, and K. Munagala. Coevolutionary opinion formation games. In *STOC*, pages 41–50, 2013.

[3] C. Budak and R. Agrawal. On participation in group chats on twitter. In *Proceedings of the 22nd international conference on World Wide Web*, pages 165–176. International World Wide Web Conferences Steering Committee, 2013.

[4] J. Cook, K. Kenthapadi, and N. Mishra. Group chats on twitter. In *Proceedings of the 22nd international conference on World Wide Web*, pages 225–236. International World Wide Web Conferences Steering Committee, 2013.

[5] A. Das, S. Gollapudi, R. Panigrahy, and M. Salek. Debiasing social wisdom. In *Proceedings of the 19th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 500–508. ACM, 2013.

[6] H. Lauw, J. C. Shafer, R. Agrawal, and A. Ntoulas. Homophily in the digital world: A livejournal case study. *Internet Computing, IEEE*, 14(2):15–23, 2010.

[7] A. Nandi, S. Paparizos, J. C. Shafer, and R. Agrawal. With a little help from my friends. In *Proceedings of the 2013 IEEE International Conference on Data Engineering (ICDE 2013)*, pages 1288–1291. IEEE Computer Society, 2013.

Towards Geo-Social Intelligence: Mining, Analyzing, and Leveraging Geospatial Footprints in Social Media

James Caverlee, Zhiyuan Cheng
Texas A&M University
{caverlee,zcheng}@cse.tamu.edu

Daniel Z. Sui
The Ohio State University
sui.10@osu.edu

Krishna Y. Kamath
Twitter, Inc.
kkamath@twitter.com

Abstract

The widespread adoption of GPS-enabled tagging of social media content via smartphones and social media services (e.g., Facebook, Twitter, Foursquare) uncovers a new window into the spatio-temporal activities of millions of people. These "footprints" open new possibilities for understanding how ideas flow across the globe, how people can organize for societal impact, and lay the foundation for new crowd-powered geosocial systems. We describe recent efforts to mine, model, and analyze large-scale geospatial footprints toward enabling new intelligent geo-social systems that leverage these footprints.

1 Introduction

The exponential growth in social media over the past decade has recently been joined by the rise of *location* as a central organizing theme of how users engage with online information services and with each other. Enabled by the widespread adoption of GPS-enabled smartphones, users are now forming a comprehensive *geo-social* overlay of the physical environment of the planet. For example, the Foursquare location sharing service has enabled over 4.5 billion "check-ins" [13], whereby users can link their presence, notes, and photographs to a particular venue. The mobile image sharing service Instagram allows users to selectively attach their latitude-longitude coordinates to each photograph; similar geo-tagged image sharing services are provided by Flickr and a host of other services. And the popular Twitter service sees 500 million Tweets per day, of which around 5 million are tagged with latitude-longitude coordinates. Confirming this trend, a recent Pew Research Center report finds that location is now an increasingly central part of the social media experience [41].

In contrast to proprietary location-based data collected by many entities – e.g., search engine query logs with an associated IP address that can be resolved to a rough location, cell-phone call records that can pinpoint a user to a particular cell tower, and point-of-sale data collected by retailers – geo-social tags and check-ins are inherently voluntary and public. As a result, they provide a rich and growing body of geo-location evidence that can potentially support basic scientific inquiry into questions that heretofore were difficult for researchers to study. These difficulties were often due to the proprietary nature of traditional location data, the cost of acquiring new data through small lab-based studies (e.g., due to navigating university IRB protocols, overcoming resistance to personal tracking devices), and the difficulty of sharing such

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

sensitive data with other researchers. Not only do voluntarily shared geo-location cues provide an alternative basis for scientific inquiry, in addition, designers of information management systems (e.g., web search systems, social media discovery, personal information management) can integrate these new public location signals into more robust user models, intelligent "location-aware" services, and so forth. Indeed, we believe that the proliferation of these fine-grained (public) spatio-temporal footprints provides an unprecedented opportunity to gain new insights into:

- The dynamics of human behavior and rhythm/pulsation of social life from local to global levels;
- The dynamics of how ideas spread and how people can organize for societal impact; and
- The development of new geo-social information systems that leverage these global-scale geospatial footprints for real-world impact.

Already, we have witnessed compelling new studies along all three of these dimensions, spanning many research communities – including the data mining and machine learning [1,4,6,11,12,28,30], geographic information systems [9,14,27,34,38], web search and information retrieval [31,33], and the emerging computational social science paradigm [15,21,23,25,29,32,36,40]. For example, the dynamics of fundamental human mobility patterns have been modeled from check-ins mined from two location sharing services – Gowalla and Brightkite – and inherent constraints on these patterns by both geographic and social factors have been discovered [6]. Facebook researchers have provided a comprehensive analysis of the distance between Facebook users, leading to new insights into how social networks are impacted by geography [1]. The LiveHoods [10] project has shown how to identify "living neighborhoods" based on the revealed locations and movements of social media users. And new geo-social information systems have been proposed based on these location cues, including earthquake detection from Twitter information flows [31], a local search system that estimates a user's location utilizing the aggregate signals from the check-ins with real-time contextual information [33], and an event discovery system that organizes spatio-temporal footprints and corresponding media to allow consumers to travel through space and time to experience the world's stories [7].

Challenges. Yet there are key challenges to delivering on this promise. First, many users in social media reveal broad, imprecise locations (e.g., at the city or state level), while others provide fine-grained latitude-longitude information. In particular, users are less likely to post precise locations such as street addresses on Twitter and related services. How can these multiple location granularities be integrated to account for uncertainty at different levels? Second, models based on users who do willingly share fine-grained location information will necessarily be biased away from the general population of social media users (and more generally, from the underlying population). How can we model and assess the impact of this bias (and its ultimate impact on applications like local information access or expert finding)? Third, personal location-revealing information may be interspersed in an inherently noisy stream of updates reflecting many daily interests (e.g., food, sports, daily chatting with friends). Are there clear location signals embedded in this mix of topics and interests that can be accurately extracted? Fourth, not all relationships in social media are the same. Some ties are stronger than others, and presumably some ties are more impactful on the development of a location-oriented user profile. How does this variable tie strength nature impact these profiles?

Even the design space of geo-social information systems is not clearly understood. For example, the ecosystem is driven both by the many users of current location sharing technologies who explicitly share location across multiple platforms (e.g., via Twitter geo-tagged posts, by Foursquare check-in) as well as the many "non-sharing" users who still leave implicit clues based on their non-geo-tagged content posts, via location-revealing images left on their social networks, and other implicit signals. How do new systems integrate these disparate signals? Do users perceive a difference in ownership of their "location" in scenarios where they explicitly reveal it versus it being inferred from large-scale data-driven approaches (e.g., by applying machine learning approaches)? And how does information access in geo-social information systems differ from traditional web search and friend finding in social networks? These and related questions lead us to believe that there is a compelling need for new techniques for mining, analyzing, and leveraging geospatial footprints in social media.

Roadmap. The rest of this paper highlights two of our parallel efforts towards the goal of enabling new geo-social intelligence. The first focuses on *the dynamics of ideas* via an exploration of fine-grained Twitter based geospatial footprints, coupled with a *predictive analytics application* that seeks to estimate the popularity of future ideas (approximated by Twitter hashtags) in particular locations. The second focuses on the *dynamics of people* via an exploration of location sharing through services like Foursquare, coupled with a prototype *location-based search* system that takes advantage of these new signals. We conclude with final thoughts on the future of geo-spatial intelligence research at the intersection of the emerging spatial computing and computational social science.

2 GeoSocial Footprints: Modeling and Predicting Spatial Diffusion

Geospatial footprints provide a new perspective on how information is shared at a global scale. Understanding how ideas are adopted, how new communities form and evolve, and how these activities shape opinions and actions are all long-standing questions, e.g. [19, 20, 22, 26, 39]. With access to fine-grained geo-spatial footprints, we face new opportunities to investigate the spatio-temporal dynamics of ideas. In this section, we highlight our research on analyzing the spatial diffusion of one type of social media – Twitter hashtags – and in building predictive models of what hashtags will ultimately be popular where. In addition to impacting these long-standing foundational questions about information diffusion, such investigations have the potential to impact the design of a variety of systems and applications, including targeted advertising, location-based services, social media search, and content delivery networks.

2.1 Dynamics of Ideas: Spatial Diffusion

We begin by describing a study of the spatio-temporal properties of social media spread through an examination of the fine-grained sharing of one type of global-scale social media – a sample of 2 billion geo-tagged Tweets with precise latitude-longitude coordinates collected over the course of 18 months. The study itself (reported more fully in [18]) focuses on the propagation of hashtags across Twitter, where a hashtag is a simple user-generated annotation prefixed with a # and serves as a simplified "semantic" marker that we can track across space and time. Questions we have explored include: (i) What role does distance play in the adoption of hashtags? Does distance between two locations influence both what users in different locations adopt and when they do so? (ii) While social media is widely reported in terms of viral and global phenomenon, to what degree are hashtags truly a global phenomenon? (iii) What are the geo-spatial properties of hashtag spread? How do local and global hashtags differ? (iv) How fast do hashtags peak after being introduced? And what are the geo-spatial factors impacting the timing of this peak? (v) How can the spatio-temporal characteristics of hashtags describe locations? Are some locations more "impactful" in terms of the hashtags that originate there?

Here we focus on two findings: the geo-spatial constraints of information sharing and peak analysis.

Geo-spatial constraints. Let's begin by modeling a location by the set of hashtags that have been observed there over a particular time window. We could define a "location" using a simple grid technique overlaid on the globe of equal-area locations, or equal-population locations, or encode some other semantic cues (e.g., by metropolitan statistical area). What then is the impact of distance between two locations on the adoption of an idea? Tobler's first law of geography [37] states that locations that are closer to each other are more alike; hence, it is reasonable to assume that nearby locations will share similar ideas (here, represented by

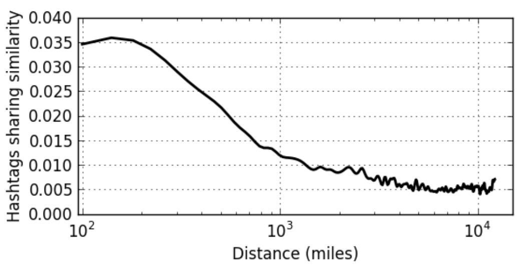


Figure 1: Hashtag sharing similarity vs Distance.

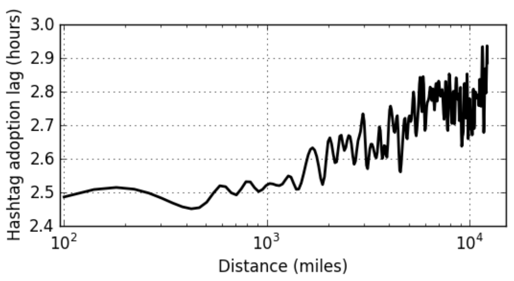


Figure 2: Hashtag adoption lag vs Distance

hashtag adoption). Alternatively, there is growing evidence of the "flattening" of the world as information spreads through Internet-enabled (geographically-neutral) media.

Given two locations, we can measure their hashtag "similarity" using the Jaccard coefficient between the sets of hashtags observed at each location: Hashtag Similarity$(l_i, l_j) = \frac{H_{l_i} \cap H_{l_j}}{H_{l_i} \cup H_{l_j}}$, where H_l is the set of unique hashtags observed in a location l. Locations that have all hashtags in common have a similarity score of 1.0, while those that share no hashtags have a score of 0.0.

Similarly, we can measure the adoption lag between two locations, to capture the time of when a new hashtag made its first appearance. Locations that adopt a common hashtag at the same time can be considered as more temporally similar than are two locations that are farther apart in time (with a greater lag). Letting t_l^h be the first time when hashtag h was observed in location l, we can define the hashtag adoption lag of two locations as: Adoption Lag$(l_i, l_j) = \frac{1}{|H_{l_i} \cap H_{l_j}|} \sum_{h \in H_{l_i} \cap H_{l_j}} |t_{l_i}^h - t_{l_j}^h|$, where the adoption lag measures the mean temporal lag between two locations for hashtags that occur in both the locations. A lower value indicates that common hashtags reach both the locations around the same time.

The relationship between hashtag similarity and distance is plotted in Figure 1. We see a strong correlation, suggesting that the closer two locations are, the more likely they are to adopt the same hashtags. As distance increases, the hashtag sharing similarity drops accordingly. Similarly, we see in Figure 2 a relatively flat temporal adoption relationship up to ~500 miles, then a generally positive correlation, suggesting that locations that are close in spatial distance tend also to be close in time (e.g., they adopt hashtags at approximately the same time). Locations that are more spatially distant tend to adopt hashtags at greater lags with respect to each other. These findings suggest the strong impact of geographic constraints (representing language commonalities, culture sharing) on meme spreading.

Peak Analysis. We next zoom in on the spatial properties of hashtag propagation during the minutes pre- and post- peak. When hashtags peak, do they peak suddenly in different locations simultaneously or do they slowly accumulate a larger spatial footprint? What are the dynamics of their spatial properties as they become popular? For every hashtag ($h \in H$) and location ($l \in L$) pair, if we let O_l^h be the set of all occurrences of h in l, then the probability of observing hashtag h in location l is defined as $P_l^h = \frac{O_l^h}{\sum_{l \in L}\{O_l^h\}}$ and the *hashtag entropy* is defined as $\mathcal{E}^h = -\sum_{l \in L} P_l^h \log_2 P_l^h$, which measures the randomness in spatial distribution of a hashtag and determines the minimum number of bits required to represent the spread. A hashtag that occurs in only a single location will have an entropy of 0.0. As a hashtag spreads to more locations, its entropy will increase, reflecting the greater randomness in the distribution.

Here we divide each hashtag's lifecycle into equal length time intervals of 10 minutes. For each time interval, we compute the hashtag entropy ($\mathcal{E}^h(t)$) over just that interval. We plot this interval-specific entropy in Figure 3. We observe that hashtags reach their lowest interval focus and highest interval entropy about 10-20 minutes after their peak. Rather than peaking with their most "global" footprint, hashtags instead reach this state *after* their peak. In effect, this single location is "championing" a hashtag. In the 10-20

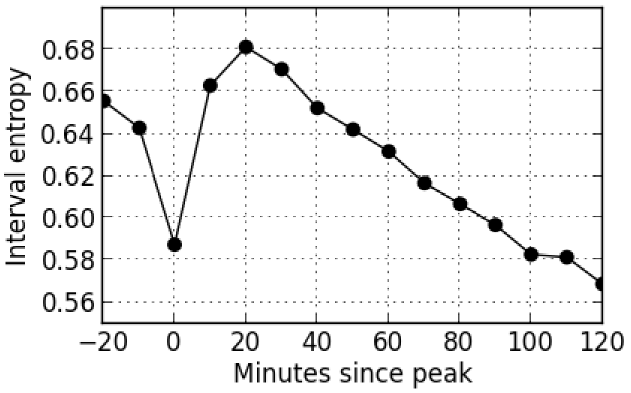


Figure 3: Hashtags peak with their most "global" footprint 10-20 minutes after their peak

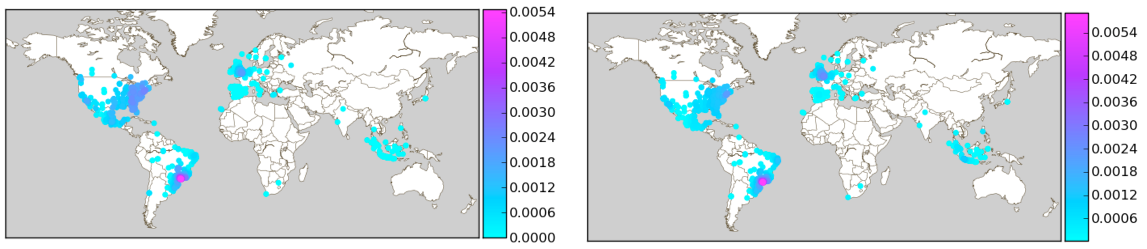
minutes after this peak period, other locations adopt the hashtag, resulting in a decrease in interval focus and an increase in entropy as the hashtags becomes more global. About 30 minutes after reaching peak, focus and entropy reverse, with focus increasing and entropy decreasing as the hashtag withdraws back to its original focus location. In essence, hashtags are spread via a single location "championing" a hashtag initially, spreading it to other locations and then continuing to propagate it after it has become popular. In [2], the authors observed a similar pattern for YouTube videos which they called the "spray-and-diffuse" pattern. Our observations over hashtags suggest that this pattern may be a fundamental property of social media spread.

2.2 Dynamics of Ideas: Predictive Analytics

Given these and related insights into the spatio-temporal spread of ideas, we can directly inform the design of distributed content delivery networks and search infrastructure for real-time Twitter-like content. For example, caching decisions to improve fast delivery of social media content to users and to support applications like real-time search can build upon the results presented here. Insights into the role distance plays and the impact locations have on hashtag spread could inform new algorithms for geo-targeted advertising. This work can also complement efforts to model network structures that support (or impede) the "viralness" of social media, measure the contagion factors that impact how users influence their neighbors, develop models of future social media adoption, and so forth.

Towards putting these results into practice, we have developed new predictive models to estimate what hashtags will eventually be popular where [17]. For example, can we accurately predict which hashtags will be popular in San Francisco over the next two hours? Can the same model also predict which hashtags will be popular in a small town like College Station, Texas? Can we identify which hashtags that have been popular in New York in the past two hours but will drop in interest? Toward answering these questions, we have develop a reinforcement learning-based approach that builds upon two competing hypotheses of information spread over geo-spatial networks we first discussed in the previous section.

- Spatial Affinity: The first hypothesis, based on Tobler's first law of geography, states that the information spread between two locations is impacted by the distance between two locations. For example, according to this hypothesis hashtags spread faster between San Francisco and Mountain View, since they are closer to each other; but slower between San Francisco and Austin.
- Community Affinity: The second hypothesis is that the "world is flat" and information spreads based on virtual communities enabled by the prevalence of the Internet. In this hypothesis, distance is less important than are the strength of these virtual ties between locations; e.g., under this hypothesis



(a) Predicted (estimated using spatial influence model after 5 minutes) (b) Actual (real distribution after 2 hours of propagation),

Figure 4: Example of using spatial influence model for the hashtag #ripstevejobs

San Francisco and Austin may be considered closer in terms of common interest (and hence, hashtags should flow more rapidly between the two), rather than Austin and its more proximate neighbor Houston.

We have investigated a series of features inspired by these two hypotheses for predicting which hashtags will be popular in a specific location at a specific time. An example of modeling propagations using the spatial influence model for the hashtag #ripstevejobs is shown in Figure 4. We predicted the future distribution of this hashtag using the spatial influence model based solely on its initial (first 5 minutes) distribution. The comparison between the predicted and actual distribution is shown in Figure 4(a) and Figure 4(b) respectively. We observe that the relative distribution (indicated by color) and its values (indicated by scale) are very close to each other. In our experimental evaluation over 755 million geo-tagged tweets, we find the best approach is able to predict close to 70% of future hashtags occurrences accurately. Interestingly, both the spatial affinity and community affinity models are valid for *particular hashtags* and for *particular locations*, suggesting the interplay of both geographic constraints and community homophily across large distances impacting what ideas flow where.

3 GeoSocial Footprints: Activity-Based Information Access

While geosocial footprints provide evidence of idea spread, they also can provide an insight into actual human movements. This human mobility and the interaction of human movement with space provides a fascinating and unique opportunity. What do large-scale voluntarily contributed human mobility data reveal? And how can these insights be incorporated into the design of new mobile+location-based services, traffic forecasting, urban planning, and models of disease spread?

3.1 From Dynamics of People to Location-Based Information Search

Toward better understanding the spatial, temporal, and social characteristics of how people use these services, we have engaged in a large-scale study of location sharing services in [5] that focuses on the wheres and whens of over 22 million check-ins across the globe. Specifically, we have studied human mobility patterns revealed by these check-ins and explored factors that influence this mobility, including social status, sentiment, and geographic constraints. We have found (i) that locations can be modeled by the activity patterns of people; and (ii) that people follow simple, reproducible patterns – motivating us to explore whether activity patterns (i.e., the temporal dynamics of check-ins) can be used to augment traditional information access (see, e.g., [3]) through a prototype location-based search system. In many ways analogous to how clickstreams have been successfully incorporated into traditional search systems based on content similarity and link analysis by connecting real-world user actions (clicks) to relevance, this prototype framework mines the spatio-temporal activity patterns of location-based crowds for augmenting traditional location-based search and for supporting enhanced location recommendations. In particular, we have developed an activity pattern-driven approach for supervised location categorization, wherein activity patterns can be used to accurately predict the semantic category of uncategorized locations. We augment this approach with an activity-driven location clustering algorithm that can group semantically related locations, and we show how activity-driven semantic organization of locations may be naturally incorporated into location-based web search. We conclude with an activity-driven location recommendation system that incorporates multiple factors, including physical distance, semantic correlation, social desires, interest-based reputation, and temporal dynamics.

Enhancing Local Search. Activity patterns and category information for venues can be easily incorporated into traditional location-based search to answer the information need for traffic. One scenario for answering the activity-driven query is: Karen is searching for a restaurant which is off-peak during dinner time between 5 - 7 PM, so that she can enjoy the quiet environment talking with her friends. Knowing the activity patterns and category for venues, the system could easily retrieve the venues nearby in the category of food, and rank the results by the descending order of busyness.

Location Recommendation. Another use case is recommendation of venues having similar activity patterns. For example, Jerry plays a lot of basketball, and tennis. He usually goes to the Williams Park during Wednesday early evening, and Saturday afternoon, which are both free time for him and peak times for guys to get-together and play basketball and tennis. Recently, he moves to a new neighborhood, and wants to find places nearby that have similar activity patterns, so that he can meet new friends there and play some basketball or tennis. A activity-driven location-based search can also easily handle this kind of queries. Given the name of the venue, the system calculates temporal similarity between activity patterns of the venue and other venues in the same category (or in other categories as well), and return the locations with the highest temporal similarities.

4 Conclusion

We believe that the increasing ubiquity of location-based social media has the potential to fundamentally disrupt basic scientific inquiry into questions that heretofore were difficult to study and to provide the basis for new "intelligent" geo-social information systems. Accomplishing this will require new methods, new algorithms, and new frameworks for mining and analyzing vast fine-grained (public) spatio-temporal footprints, as well as new systems and techniques to leverage these footprints. We have outlined some of the challenges facing this opportunity and highlighted two of our related efforts toward informing this emerging research area. Moving forward, we believe that geo-social intelligence research is poised to make major breakthroughs in the years to come due to the growing interests of social scientists in computational/data-intensive approaches and the 4th paradigm [16, 24] and computer scientists in spatial computing [8]. We also believe that transformative research in geo-social system can be accelerated along multiple fronts if we continue to embrace and fine-tune the emerging open science paradigm [35] to promote interdisciplinary collaboration and improve the infrastructure for geo-social intelligence research.

References

[1] L. Backstrom, E. Sun, and C. Marlow. Find me if you can: improving geographical prediction with social and spatial proximity. In *WWW*, 2010.

[2] A. Brodersen, S. Scellato, and M. Wattenhofer. Youtube around the world: geographic popularity of videos. In *WWW*, 2012.

[3] Z. Cheng, J. Caverlee, K. Y. Kamath, and K. Lee. Toward traffic-driven location-based web search. In *CIKM*, pages 805–814, 2011.

[4] Z. Cheng, J. Caverlee, and K. Lee. You are where you tweet: a content-based approach to geo-locating twitter users. In *CIKM*, 2010.

[5] Z. Cheng, J. Caverlee, K. Lee, and D. Z. Sui. Exploring millions of footprints in location sharing services. In *ICWSM*, 2011.

[6] E. Cho, S. A. Myers, and J. Leskovec. Friendship and mobility: user movement in location-based social networks. In *Proceedings of the 17th ACM SIGKDD international conference on Knowledge discovery and data mining*, pages 1082–1090. ACM, 2011.

[7] T. Coletta. Seen, by mahaya: A tool to sort the mess of social media. http://www.popularmechanics.com/technology/gadgets/news/seen-by-mahaya-a-tool-to-sort-the-mess-of-social-media-15510890/, 2013. [Online; accessed Sep 24, 2013].

[8] C. C. Consortium. From gps and virtual globes to spatial computing - 2020: The next transformative technology. Technical report, 2013.

[9] J. W. Crampton, M. Graham, A. Poorthuis, T. Shelton, M. Stephens, M. W. Wilson, and M. Zook. Beyond the geotag: situating Ôbig dataÕ and leveraging the potential of the geoweb. *Cartography and Geographic Information Science*, 40(2):130–139, 2013.

[10] J. Cranshaw, R. Schwartz, J. I. Hong, and N. M. Sadeh. The livehoods project: Utilizing social media to understand the dynamics of a city. In J. G. Breslin, N. B. Ellison, J. G. Shanahan, and Z. Tufekci, editors, *ICWSM*. The AAAI Press, 2012.

[11] C. Davis, G. Pappa, D. de Oliveira, and F. de L Arcanjo. Inferring the location of twitter messages based on user relationships. In *Transactions in GIS*, 2011.

[12] J. Eisenstein, B. O'Connor, N. Smith, and E. Xing. A latent variable model for geographic lexical variation. In *EMNLP*, 2010.

[13] Foursquare. About foursquare, April 2012.

[14] D. D. Ghosh and R. Guha. What are we ÔtweetingÕ about obesity? mapping tweets with topic modeling and geographic information system. *Cartography and Geographic Information Science*, 40(2):90–102, 2013.

[15] B. Hecht, L. Hong, B. Suh, and E. Chi. Tweets from justin bieber's heart: the dynamics of the location field in user profiles. In *SIGCHI*, 2011.

[16] T. Hey, S. Tansley, and K. M. Tolle, editors. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, 2009.

[17] K. Y. Kamath and J. Caverlee. Spatio-temporal meme prediction: Learning what hashtags will be popular where. In *CIKM*, 2013.

[18] K. Y. Kamath, J. Caverlee, K. Lee, and Z. Cheng. Spatio-temporal dynamics of online memes: A study of geo-tagged tweets. In *WWW*, 2013.

[19] D. Kempe, J. Kleinberg, and E. Tardos. Maximizing the spread of influence through a social network. In *KDD*, 2003.

[20] D. Kempe, J. Kleinberg, and É. Tardos. Influential nodes in a diffusion model for social networks. In *IN ICALP*, 2005.

[21] G. King. Ensuring the data-rich future of the social sciences. *Science*, 331(6018):719–721, 2011.

[22] G. Kossinets, J. Kleinberg, and D. Watts. The structure of information pathways in a social communication network. In *KDD*, 2008.

[23] J. Kulshrestha, F. Kooti, A. Nikravesh, and K. Gummadi. Geographic dissection of the twitter network. In *International AAAI Conference on Weblogs and Social Media*, 2012.

[24] D. Lazer, A. Pentland, L. Adamic, S. Aral, A.-L. BarabĞsi, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann, T. Jebara, G. King, M. Macy, D. Roy, and M. Van Alstyne. Computational social science. *Science*, 323(5915):721–723, 2009.

[25] K. Leetaru, S. Wang, G. Cao, A. Padmanabhan, and E. Shook. Mapping the global twitter heartbeat: The geography of twitter. *First Monday*, 18(5), 2013.

[26] K. Lerman and R. Ghosh. Information contagion: an empirical study of the spread of news on digg and twitter social networks. *CoRR*, 2010.

[27] L. Li, M. F. Goodchild, and B. Xu. Spatial, temporal, and socioeconomic patterns in the use of twitter and flickr. *Cartography and Geographic Information Science*, 40(2):61–77, 2013.

[28] R. Li, S. Wang, H. Deng, R. Wang, and K. Chang. Towards social user profiling: unified and discriminative influence model for inferring home locations. In *KDD*, 2012.

[29] J. Lindqvist, J. Cranshaw, J. Wiese, J. Hong, and J. Zimmerman. I'm the mayor of my house: Examining why people use foursquare - a social-driven location sharing application. In *SIGCHI*, 2011.

[30] A. Sadilek, H. Kautz, and J. Bigham. Finding your friends and following them to where you are. In *WSDM*, 2012.

[31] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *WWW*, 2010.

[32] S. Scellato, A. Noulas, R. Lambiotte, and C. Mascolo. Socio-spatial properties of online location-based social networks. In *ICWSM*, 2011.

[33] B. Shaw, J. Shea, S. Sinha, and A. Hogue. Learning to rank for spatiotemporal search. In *WSDM*, 2013.

[34] A. Stefanidis, A. Cotnoir, A. Croitoru, A. Crooks, M. Rice, and J. Radzikowski. Demarcating new boundaries: mapping virtual polycentric communities through social media content. *Cartography and Geographic Information Science*, 40(2):116–129, 2013.

[35] D. Z. Sui. Opportunities and impediments of open gis. *Transactions in GIS (in press)*, 2014.

[36] Y. Takhteyev, A. Gruzd, and B. Wellman. Geography of twitter networks. *Social Networks*, 34(1):73 – 81, 2012.

[37] W. Tobler. A computer movie simulating urban growth in the detroit region. *Economic Geography*, 46(2), 1970.

[38] C. Xu, D. W. Wong, and C. Yang. Evaluating the Ògeographical awarenessÓ of individuals: an exploratory analysis of twitter data. *Cartography and Geographic Information Science*, 40(2):103–115, 2013.

[39] J. Yang and J. Leskovec. Modeling information diffusion in implicit networks. In *ICDM*, 2010.

[40] S. Yardi and D. Boyd. Tweeting from the town square: Measuring geographic local networks. In *ICWSM*, 2010.

[41] K. Zickuhr. Location-based services. *Pew Research Center*, September 2013.

Effective Event Identification in Social Media

Fotis Psallidas
fotis@cs.columbia.edu
Columbia University

Hila Becker
hila@google.com
Google, Inc.

Mor Naaman
mor.naaman@cornell.edu
Cornell Tech

Luis Gravano
gravano@cs.columbia.edu
Columbia University

Abstract

Online social media sites are extensively used by individuals to produce and distribute content related to real-world events. Unfortunately, this social media content associated with an event is generally not provided in any structured and readily available form. Thus, identifying the event-related content on social media sites is a challenging task. Prior work has addressed the event identification task under two different scenarios, namely, when the events are known ahead of time, as is sometimes the case for planned events, and when the events are unknown, as is the case for spontaneous, unplanned events. In this article, we discuss both the unknown- and known-event identification scenarios, and attempt to characterize the key factors in the identification process, including the nature of social media content as well as the behavior and characteristics of event content over time. Furthermore, we propose enhancements to our earlier techniques that consider these factors and improve the state-of-the-art unknown-event identification strategies. Specifically, we propose novel features of the social media content that we can exploit, as well as the modeling of the typical time decay of event-related content. Large-scale experiments show that our approach exhibits improved effectiveness relative to the state-of-the-art approaches.

1 Introduction

Online social media sites (e.g., Flickr, YouTube, Twitter) serve as the main outlet for individuals to distribute and receive meaningful content about real-world events. This content may appear in various forms, including status updates, photos, and videos, that can be created or posted before, during, and after an event. Furthermore, for known and planned events, structured information (e.g., title, time, location) might be available through event-aggregation social media sites (e.g., Facebook Events, Meetup, EventBrite). Such prior knowledge, however, is not available for unknown or spontaneous events (e.g., natural disasters). By automatically identifying the social media content related to either known or unknown events, which is the focus of this article, we can enhance powerful event browsing and search.

In the known-event identification scenario, information is available explicitly online. For instance, consider the 2013 NBA All-Star game in Houston, Texas. Structured information about the event (e.g., time and location) may be available on a related Facebook Event page. Because this information may be limited, though, users may seek to obtain additional content for the event from other social media sites (e.g., check what Twitter users discuss about the event or watch the game again on YouTube after the event). Thus, automatically identifying social media content related to known events enhances the event-based experience a user may seek by providing meaningful information before, during, and after an event.

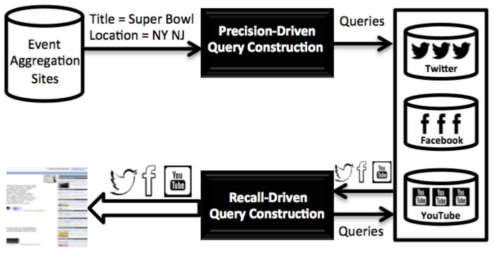


Figure 1: Known-event identification: Retrieving relevant content for the Super Bowl event, starting with its (known) title and location.

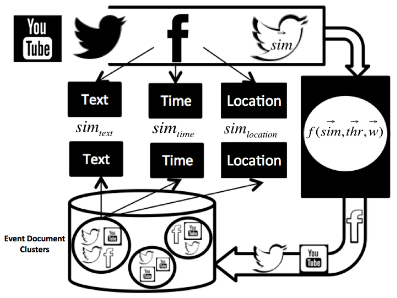


Figure 2: Unknown-event identification: Online clustering over a stream of social media content using time, text, and location as weak indicators.

In contrast, in the unknown-event identification scenario there is no a priori knowledge of the events. For instance, consider an unplanned presidential announcement on a political issue. Such an event, although it is not known beforehand, prompts individuals to react and discuss, often extensively (e.g., through Twitter messages discussing the event or YouTube videos of the announcement). Just as in the known-event scenario, users also turn to social media in search of content for the event, both during the event and after it has ended.

Attempting to automatically identify event-related social media content in both scenarios is a challenging task. First, social media content is noisy and heterogeneous. For instance, Twitter messages are short, informal, and often grammatically incorrect. The event identification process should then account for these characteristics of social media content. Furthermore, social media content is not always related to an event. For instance, a Facebook post that states "@alice I am impressed by your new song!!" corresponds to chatter or social activity, and is not event-related. The event identification process should then discard such non-event content. Finally, the manner in which event-related discussions evolve in social media affects the event identification process. For example, the fact that an event has been inactive for a long time might serve as a strong indicator that new social media content is unlikely to correspond to the event, as we argue in Section 3.

Most related research identifies event-related social media content by considering only a subset of these factors. For instance, Shakaki et al. [7] identify event-related social media content by using a predefined set of terms (e.g., "earthquake") as keywords that social media documents should contain. However, social media documents are noisy and heterogeneous, and might not contain a specific keyword or have arbitrary variations thereof (e.g., "eaaarthquaaake"). Improving on this point, Sankaranarayanan et al. [13] use both textual and temporal features to reduce the impact of the noisy data, but they ignore other features, such as location. Also, to further distinguish between event and non-event documents their methodology assumes the existence of "seeders," which are treated as always producing event-related content. However, these seeders might distribute documents unrelated to events that the identification process should differentiate from the event-related ones.

In this article, we first summarize our earlier work on known- and unknown-event identification [2–4], which addresses many of the factors discussed above (Section 2). Furthermore, we also propose novel techniques to better address these key factors in the context of the unknown-event identification scenario (Section 3). More precisely, we introduce novel features to address the noise and heterogeneity of the social media content, as well as a time decay function that fits the behavior of event-related discussions over time. Finally, we evaluate our techniques using a large real-world dataset compiled from photo-sharing social media site Flickr (Section 4). Our experiments show that our proposed techniques substantially improve the effectiveness for the unknown-event identification scenario compared to our baselines.

2 Event Content Identification in Social Media

Major social media sites are popular venues for publishing rich and diverse information about a variety of real-world events. As an example of such an event, consider the 2014 NFL's Super Bowl in New York and New Jersey. Event-aggregation platforms (e.g., Facebook Events, Meetup, EventBrite) might list a description of this event, including some of its prominent features (e.g., title, time, location). Also, individuals share their reactions and discuss the event on different social media sites (e.g., Flickr, YouTube, Twitter). Such content is highly valuable, because it offers a user perspective of events that would otherwise not be found on the Web. Unfortunately, this meaningful user-contributed information is not readily available in any structured form, so it is generally unclear what social media content refers to which event. Overall, automatically identifying event content poses interesting challenges, because the social media content is noisy and highly heterogeneous.

To address the event identification task, we consider two substantially different scenarios. The first scenario corresponds to events that are known ahead of time, as is the case for planned events announced, say, on Facebook Events or Meetup. In the second scenario, we do not have have any a priori knowledge of an event, either because the event occurred unexpectedly (e.g., an earthquake or an automobile accident) or because the event was not announced online (e.g., a local street fair or a birthday celebration). Both scenarios of the event identification task, namely, *known-event identification* and *unknown-event identification*, present distinctive challenges and opportunities, which we discuss next:

Known-Event Identification: In this scenario, key properties of the events (e.g., title, time, location) are known ahead of time, usually posted on event-aggregation social media sites. Social media content related to these known events could reside in multiple social media sites, each contributing different information about the event. For instance, YouTube might contain videos for the Super Bowl event, whereas Twitter users might discuss the event by sharing short text messages, or *tweets*. To retrieve cross-site social media documents associated with the Super Bowl event, we could then use the APIs provided by major social media sites, such as the Twitter or YouTube APIs, to formulate high-precision queries using the event's known features (e.g., a [2014 NFL Super Bowl] query using the event title "2014 NFL Super Bowl") and extract the related tweets and YouTube videos. However, such highly specific queries tend to retrieve event-related documents with high precision but with low recall, meaning that they miss many relevant event documents.

In our previous work [2], we have addressed these challenges by proposing a two-step query formulation approach. Figure 1 illustrates this process for our example. In the first step, we use highly specific queries, using the known event properties of an event, to achieve high-precision results. For instance, in Figure 1 we use the title and the location of the Super Bowl to construct queries that retrieve related YouTube videos, Facebook posts, and tweets. As we mentioned, however, these high-precision queries result in low recall. Thus, the second step builds on these high-precision cross-site results, using term extraction and frequency analysis, aiming to improve recall and contribute to the high quality and diversity of the identified information. Interestingly, we also proposed ways to leverage the event-related social media content retrieved from one social media site, retrieved using high-precision queries, to obtain additional content from other social media sites [2]. This task is important in the case where the high-precision queries do not return sufficiently many results from some of the available social media sites (e.g., querying YouTube using the title of a local event might not yield any results). Thus, we proposed using the content from one social media site (e.g., Twitter) to build recall-oriented queries to be used on other social media sites (e.g., YouTube).

Unknown-Event Identification: In contrast to the known-event identification scenario, in the unknown-event identification scenario we do not have any a priori information about the events and their properties. Therefore, we are presented with a stream of event-related social media documents without any knowledge of the events that may be reflected in the stream. For example, a Twitter stream (see Table 3) may contain many tweets related to an event (e.g., a high-profile announcement of U.S. President Obama about the

Publisher	*Text*	*Time*
New York Times	Apple Shows Off 2 New iPhones, One a Lower-Cost Model http://nyti.ms/19EP2DI	10:06 PM - 10 Sep 13
iPhone News	Apple changes up colors: ‘space gray’ comes to iPhone 5s & iPods http://dlvr.it/3xdkm5 #iPhone	03:44 AM - 11 Sep 13
Wall Street Journal	President Obama addresses the nation on #Syria. Follow our live blog: http://on.wsj.com/1aoGPV0	04:05 AM - 11 Sep 13
Bob	@alice He is my favorite illusionist davidcopperfield.com …	04:08 AM - 11 Sep 13
Huffington Post	Obama now: “I will not put American boots on the ground in Syria." http://wapo.st/1aoHmX8	04:10 AM - 11 Sep 13
IGN	Apple iPhone 5s vs. iPhone 5c: Which phone should you buy? - MobileïijŽThe iPhone 5c is $100 cheaper than the iPhone	04:14 AM - 11 Sep 13
New York Times	Breaking News: Obama Asks Congress to Postpone a Vote on Syria Action	04:22 AM - 11 Sep 13

Table 3: Twitter messages including both event and non-event content.

Middle-East) interspersed with messages related to other events (e.g., Apple’s announcement of new iPhone models) as well as messages unrelated to events (e.g., Bob mentioning to Alice his favorite illusionist). Automatically identifying unknown-event content in this scenario poses some interesting challenges both due to the high rate of the social streams and the noisy nature of the data.

To address these challenges, we proposed an online clustering framework (see Figure 2) that leverages the multiple features associated with each social media document (e.g., publisher, text, and time for the tweets in Table 3) [3]. These features help define weak indicators of event-related content and collectively produce stronger document-similarity judgments, to decide when two social media documents correspond to the same event, than when used individually. To see why, consider the first tweet in Table 3, from The New York Times, discussing the iPhone 5c release. Judging solely based on the text, the tweet might (erroneously) be linked to earlier Apple announcements involving iPhones. To link the tweet to the correct event, however, we can exploit the fact that the publication time of the New York Times tweet (first row of Table 3) is close to the publication time of other tweets that explicitly discuss the iPhone 5c release. In contrast, considering the publisher and text together does not assist in correctly determining the correct event content in the example above.

Different features of social media documents thus have significantly varying impact on the final clustering —and hence unknown-event identification— decision. Based on this observation, we have proposed [3] an ensemble learning methodology that decides for each feature (e.g., text, time, and location in Figure 2) (a) how more indicative of event-related content it can be compared to the rest of the features (e.g., the time feature is considered more revealing than the publisher one in our example), and (b) under what circumstances its judgment on the similarity of two social media documents (e.g., $sim_{text}(d, d')$, $sim_{time}(d, d')$, $sim_{location}(d, d')$ in Figure 2, where d, d' are social media documents) is considered trustworthy. More precisely, we deployed ensemble learning methods to learn and associate each feature with a weight and a threshold that capture the importance of the features. Furthermore, based on the set of weights $\vec{w}$, thresholds $\vec{thr}$, and individual feature similarity functions, we constructed a “consensus function" $f(\vec{sim}(d, C), \vec{w}, \vec{thr})$ that plays the role of the final similarity function between a document d and a cluster C in the document clustering process (see Figure 2). Cluster C can be alternatively represented as its centroid vector c (i.e., $\vec{sim}(d, C) = \vec{sim}(d, c)$) or by all the cluster documents (i.e., $\vec{sim}(d, C) = \frac{1}{|C|} \cdot \sum_{d' \in C} \vec{sim}(d, d')$).

Social media documents, however, are not always event-related. For instance, the tweet by Bob (see Table 3) does not relate to any event. Therefore, after clustering the social media documents, we need to identify which clusters correspond to events and which ones do not. For this, we deployed event classification techniques that operate on the output of the clustering process. Specifically, they help distinguish between event-related clusters and non-event ones (e.g., clusters that contain event-related social media documents, such as those related to the iPhone 5c release, and clusters that contain chatter and social activity, like Bob’s message). Technically, our event classification techniques rely on a rich family of aggregate cluster statistics, including temporal (e.g., frequency of the terms associated with a cluster), social (e.g., proportion of retweets and mentions of a tweet within a cluster), topical (e.g., topical coherence in a cluster), and platform-centric (e.g., tags and presence of multi-word hashtags), as best suitable indicators for event and non-event content separation.

The overall process above yields high-quality results [3, 4] on the unknown-event identification task. Nonetheless, its effectiveness can be further improved. For instance, the text feature of Figure 2 would treat the terms Obama and Syria as equal to the terms follow and blog of the Wall Street Journal tweet in Table 3.

Feature Type	*Example*	Jaccard Similarity
Whole URL	mashable.com/2013/09/08/bruno-mars-super-bowl-halftime-show-confirmed rollingstone.com/music/news/bruno-mars-will-perform-at-super-bowl-20130908	0.0
Parsed URL	2013, 09, 08, bruno, mars, super, bowl, halftime, show, confirmed music, news, bruno, mars, will, perform, at, super, bowl, 20130908	0.25
Parsed Query Part of URL	bruno, mars, super, bowl, halftime, show, confirmed, bruno, mars, will, perform, at, super, bowl, 20130908	0.36

Table 4: Extracted URL features for two URLs and the Jaccard similarity for each feature.

The former play an important role with respect to the corresponding event, while the latter only add noise. Furthermore, events evolve in certain ways over time. For instance, by the time tweets were discussing the iPhone 5c model release (see Table 3), discussions about previous iPhone releases were limited. If the first social document about the iPhone 5c release does not explicitly mention the 5c model (e.g., as is the case with the first tweet by The New York Times in Table 3), the clustering procedure may erroneously relate the new event to the iPhone 4 release event it has already identified, hence compromising the effectiveness of the clustering procedure. Next, we introduce a variety of refinements to our techniques to improve the effectiveness of the unknown-event identification task. (Improving the known-event identification task remains the subject of our future work.)

3 Improving Unknown-Event Identification Effectiveness

The textual features of the social media documents (e.g., text, publisher of the tweets in Table 3) and how events behave over time have a significant impact on the effectiveness of the document clustering procedure. How to refine the clustering procedure to benefit from these factors is a challenging task that we discuss next. Specifically, we discuss new features for the unknown-event identification process, namely, "URLs" and "bursty vocabularies." Then we show how to leverage the typical temporal characteristics of event content.

URLs: URLs in event-related social streams are ubiquitous. Individuals use them to share meaningful event-related external content. Furthermore, using URLs assists users to adhere to the limited-length text that characterizes the social media documents. For instance, we can directly discuss the confirmation of the Super Bowl's halftime performer using an appropriate URL instead of explicitly describing the choice of performer. To capitalize on this behavior, we propose the modeling of three new textual features as event indicators, namely, *whole URL*, *parsed URL*, and *parsed query part of URL* (see Table 4). Intuitively, the *whole URL* feature captures the similarity of social media documents that use the same URL. However, multiple sources might discuss the same event-related content, a case where the *whole URL* feature fails to identify their similarity (see Table 4). In such cases, the *parsed URL* feature appropriately tokenizes the URL, as illustrated in Table 4, and uses the extracted tokens to identify the underlying similarity. However, different sources use different URL patterns, so the *parsed URL* might introduce noise into the underlying similarity metric. For instance, the *parsed URL* for the Mashable URL in Table 4 contains the terms 2013, 09, and 08 while the Rolling Stones *parsed URL* contains the terms music and news, which are both not descriptive and add noise to the similarity computation. To address this challenge, we finally introduce the *parsed query part of URL* as a highly indicative feature of event content. We note, however, that there are cases (e.g., http://tinyurl.com/nove79c[1]) where the *parsed query part of URL* feature fails to recognize the underlying similarity and the *parsed URL* feature performs substantially better.

Bursty Vocabulary: The social media content related to an event tends to revolve around a central topic. In social media documents related to an event, this central topic is expressed by a set of terms that is significantly more frequent than the rest of the terms (e.g., NFL, Super, and Bowl for the 2014 NFL Super Bowl event). However, events that span a wide time range typically exhibit a different set of these *bursty* terms at different points of their lifetime. Figure 4 shows three different time windows for the Super Bowl event. Initially, terms like Super, Bowl, and NFL are the most bursty. Then, the announcement of the

[1]The url http://tinyurl.com/nove79c resolves to http://www.usatoday.com/story/sports/nfl/2013/09/07/bruno-mars-super-bowl-halftime-show-metlife-stadium-february-2014/2779895.

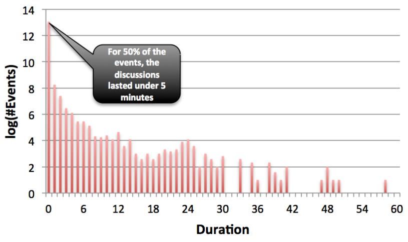


Figure 3: Histogram of the log-scaled number of events as a function of their discussion time (step $\Delta t = 15$ days).

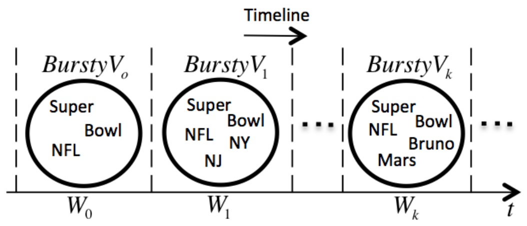
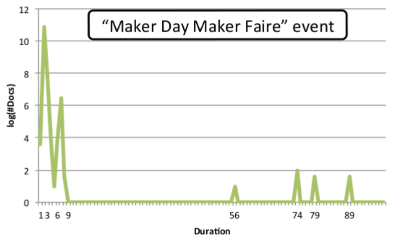
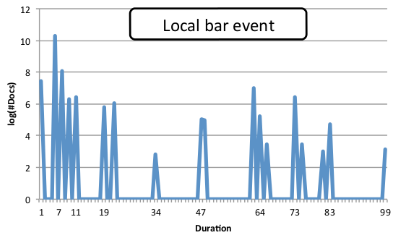


Figure 4: Bursty vocabulary of the Super Bowl event in various time windows (W_i and $BurstyV_i$ denote the i-th time window and its bursty vocabulary).



(a)



(b)

Figure 5: Example of two event discussions, over duration time, expressed as the log-scaled number of social media documents discussing the event (step $\Delta t \approx 4$ and 7 days for (a) and (b), respectively).

location of the Super Bowl (i.e., New York and New Jersey) triggers terms like NY and NJ to exhibit bursty behavior. Finally, the announcement of Bruno Mars as the Super Bowl halftime performer causes terms such as Bruno and Mars to exhibit bursty behavior. To capture the terms associated with an event that exhibit a bursty behavior within a given time window, we introduce the notion of *bursty vocabulary* per time window.

Technically, we tailor the notion of bursty vocabulary within the clustering framework [3] using the following methodology: For each cluster, at the end of a time window W_i, we extract the bursty vocabulary $BurstyV_i$, which we model as a weak indicator for the next time window W_{i+1}. To determine if a term is about to be bursty in the next time window, we follow a technique similar to the one proposed in [5]. More precisely, we say that a term *t* is bursty in W_{i+1} if the expected number of occurrences of *t* in W_{i+1} is higher than the average number of occurrences that the term had in the previous time windows. To compute the expected occurrences of a term *t* in the time window W_{i+1} we can model the probability of the occurrences of the term *t* by a hyper-geometric distribution [5]. Here, however, we use the less computationally expensive binomial distribution, which in fact coincides with the hyper-geometric distribution for large volumes of data [5]. Moreover, the time windows are determined by partitioning the incoming document space into chunks of B documents, for a value of B that we can determine experimentally.

Clustering with Time Decay: Beyond developing new features for clustering, we can improve the effectiveness of the event identification process by exploiting the typical temporal behavior of event-related content. Figure 3 shows that the number of events as a function of the duration of their discussions in social media tends to follow a power law distribution with a noisy tail (Figure 3 was derived from the *Upcoming* dataset that we describe in Section 4). Consequently, small-scale events (e.g., street fairs and birthdays) tend to dominate the event space while they are discussed significantly less than large-scale events. As a result, the clustering procedure considers many events that have ended as alive, thus adding noise to the identification process of active events. To alleviate this problem, we could attempt to identify the end of the discussions for an event. However, event-related discussions might not exhibit a clear, or any, ending time. Figure 5 shows two events from the *Upcoming* dataset that last more than one year and are discussed at different points in time at different rates.

To address these challenges, we leverage an interesting observation of event-related discussions in social media: these discussions generally attract massive participation initially, followed by a decline in attention.

This decline might be permanent or be followed by a massive restart, as shown in the examples of Figure 5. As an illustration of the latter, consider our Super Bowl example. The announcement of the location of the Super Bowl (i.e., New York and New Jersey) triggers a discussion of the event that is followed by a decay. Then, the announcement of the halftime performer again triggers the discussion, followed by another decay. Based on this observation, we introduce a time decay function to the clustering framework that weighs the consensus function $f(\vec{sim}(d,C),\vec{thr},\vec{w})$ described in Section 2, to best capture the behavior of discussions over time. Technically, the new similarity score between an incoming document d and a cluster C is computed as: $sim_{decay}(d,C) = f(\vec{sim}(d,C),\vec{thr},\vec{w}) \cdot e^{-a \cdot \frac{|T_d - T_{e_C}|}{|T_{e_C} - T_{s_C}|}}$, where T_d is the time of creation of document d, T_{e_C} is the maximum time of document creation in the cluster C (i.e., $T_{e_C} = max(\{T_{d'} : d' \in C\})$), T_{s_C} is the minimum time of document creation in the cluster C (i.e., $T_{s_C} = min(\{T_{d'} : d' \in C\})$), and a is the decay constant that we can decide experimentally. Using this time-decay function, the clustering process (a) penalizes clusters that have been inactive for a long time (e.g., as is likely the case for small-scale events) and (b) re-triggers events that have been inactive for some time if the similarity score without the time-decay factor is strong enough (e.g., as is often the case for large-scale events). Both of these properties capture the observations above and adequately improve the clustering procedure, as we show next in our experimental evaluation.

4 Experimental Evaluation

In [3], we reported extensive experiments for the unknown-event identification task, showing that modeling multiple features as weak indicators of event related content, and using them collectively, can produce stronger judgments compared to using them individually. The similarity learning techniques described in Section 2 yielded better performance than the baselines on which we built, including traditional approaches that use text-based similarity. In this section, we report additional experiments to evaluate the contribution of the URL and bursty vocabulary features, as well as the time decay function, which we introduced in Section 3. Specifically, we summarize our experimental settings in Section 4.1 and report the experimental results in Section 4.2.

4.1 Experimental Settings

Data: We use the *Upcoming* dataset presented in [3]. This dataset includes 273,842 multi-featured Flickr photos that correspond to 9,613 real-world events from the Upcoming event catalog[2]. The features associated with each photo that we use as baseline indicators include the title, description, time shot, upload time, and location in longitude-latitude format. (See Section 4.2 for a discussion on other social media sites.)

Methodology: Our evaluation methodology mirrors that of [3]. Specifically, to initiate the clustering procedure we are first required to learn, for each feature, an associated threshold and weight, used in the construction of the consensus function (i.e., $f(\vec{sim}(d,C),\vec{w},\vec{thr})$ in Section 2). To this end, we first sort the *Upcoming* dataset (descending order of upload time to imitate a real-world streaming scenario) and divide it into three equal parts. Then, we use the earliest two parts to learn the set of weights and thresholds. The features that we use include all the features of the Flickr photos as well as the bursty vocabulary and URL features. Consequently, we construct the final similarity function, using the centroid vector for the cluster representation, and we run our experiments on the last part of the *Upcoming* dataset, on which we report our results. To quantify the quality of our results, we use the well known NMI [10] and $B\text{–}Cubed$ [1] quality metrics.

Implementation: For the indexing of the textual features we deployed the Oracle Berkeley DB version 6.0.20[3], which assists on the construction of the tf-idf vectors, used to represent textual features. As similarity functions, we use the cosine similarity for textual features, sim_{title}, $sim_{description}$; the Haversine

[2]This web site, now defunct, was available at http://upcoming.org.

[3]http://www.oracle.com/us/products/database/berkeley-db/overview/index.htm

	Baseline	*Parsed Urls*	*BurstyV*	*TimeDec*	*BurstyV+TimeDec*
NMI	0.89703	0.90328	0.92192	0.92933	**0.9414**
B-Cubed	0.80345	0.7897	0.82095	0.81768	**0.83919**

Table 5: Effectiveness of proposed techniques over the *Upcoming* test dataset.

	Baseline	*Whole URL*	*Parsed URL*	*Parsed Query Part*
NMI	0.93517	0.91567	**0.95971**	0.92634
B-Cubed	0.82759	0.81333	**0.87162**	0.82592

Table 6: Effectiveness of URL features for the events of the *Upcoming* dataset associated with at least one URL.

distance [9] for the location feature, $sim_{location}$; and the ℓ_1 norm for the time feature, $sim_{time\text{-}shot}$.
Techniques for Comparison: As a baseline approach, we consider the clustering procedure that models all the individual features (i.e., title, description, time shot, location features) as weak indicators. We evaluate four options: (a) *ParURLs*: Baseline + Parsed URLs, (b) *BurstyV*: Baseline + Bursty Vocabulary, (c) *TimeDec*: Baseline + Time Decay, and (d) *BurstyV* + *TimeDec*: Baseline + Bursty Vocabulary + Time Decay.

4.2 Experimental Results

As we show in Table 5, the BurstyV + TimeDec technique obtained the highest quality results. This technique combines the bursty vocabulary, which reduces the noise of the textual features, and the time decay function, which fits the typical behavior of events over time. In contrast, the BurstyV technique improves over the baseline, but is problematic in two scenarios. One scenario corresponds to large-scale event discussions that were inactive for a long time (e.g., as in the Figure 5(a) example) but do not necessarily exhibit a similar bursty vocabulary in the next active time window. Another problematic scenario corresponds to discussions that are highly active but tend to change their bursty vocabulary frequently. Both scenarios emphasize the importance of automatically adjusting the B parameter, which regulates the number of documents that have to be appended in a cluster in order to recompute its bursty vocabulary, as described in Section 3. Similarly, the TimeDec technique yields better results than the baseline, but suffers from the noise from unimportant terms, which is handled properly by the BurstyV technique.

In contrast, the ParUrls technique, which uses the *parsed URLs* feature, does not appear to further improve the baseline in this benchmark (the same behavior applies for the *whole URLs* and the *parsed query part of URLs*): Fewer than 11% of the *Upcoming* documents contain URLs, an expected behavior for photographs. In the absence of a URL feature, the corresponding indicator returns a zero similarity score, translating to a failure to detect the true document similarity in most cases. Fortunately, the learning procedure identifies this behavior and assigns a close–to–zero weight to the URL features. Thus the judgments from URL features tend to have no impact on the final decision. If we limit the benchmark to the set of events whose associated documents contain URLs, we observe an improvement over the baseline, as seen in Table 6. This suggests that the proposed URL features may be beneficial for social media sites with a more substantial presence of URLs in their documents (e.g., tweets tend to include URLs frequently, and the presence of URLs could be indicative of event-related content [6]). We now turn to the performance of the alternate URL features (see Table 6). The Parsed Query Part technique, which uses the *parsed query part of URL* feature, performs worse than the Parsed URL technique, which uses the *parsed URL* feature, because most of the URLs in this corpus do not actually have a query part.

5 Conclusions

Social media captures our shared experiences with increasing comprehensiveness. Social media thus serves as an important record of our culture and our society. Moreover, by making new types of information easily accessible on an unprecedented scale, social media has triggered an information revolution perhaps only comparable with the advent of the Web itself in the early 1990s. Still, the methods to retrieve and organize social media content are in their infancy. In our work, we have focused on an important slice of social media content, namely, the content that is associated with real-world events. Specifically, in this article we discussed the event identification task under two substantially different scenarios, known- and unknown-event identification. We showed how we can exploit rich features of the social media documents, as well as revealing temporal patterns of the relevant content, to identify event content effectively. Many open challenges remain for the problems of detection of events in social media and identification of event content, as well as for the presentation and organization of this information for a growing variety of tasks and stakeholders. Beyond events, we hope that our research will help understand, organize, and retrieve social media content around topics, people, places, and more from these new shared records.

Acknowledgments: This material is based on work supported by NSF Grants IIS-0811038, IIS-1017845, and IIS-1017389, and by two Google Research Awards. In accordance with Columbia University reporting requirements, Professor Gravano acknowledges ownership of Google stock as of the writing of this paper.

References

[1] E. Amigó, J. Gonzalo, J. Artiles, and F. Verdejo. A comparison of extrinsic clustering evaluation metrics based on formal constraints. *Information Retrieval*, 12:461–486, 2009.

[2] H. Becker, D. Iter, M. Naaman, and L. Gravano. Identifying content for planned events across social media sites. In *Proceedings of the 5th ACM International Conference on Web Search and Data Mining (WSDM '12)*, 2012.

[3] H. Becker, M. Naaman, and L. Gravano. Learning similarity metrics for event identification in social media. In *Proceedings of the 3rd ACM International Conference on Web Search and Data Mining (WSDM '10)*, 2010.

[4] H. Becker, M. Naaman, and L. Gravano. Beyond trending topics: Real-world event identification on Twitter. In *Proceedings of the 5th International AAAI Conference on Weblogs and Social Media (ICWSM '11)*, 2011.

[5] G. Fung, J. Yu, P. Yu, and H. Lu. Parameter free bursty events detection in text streams. In *Proceedings of the 31st ACM International Conference on Very Large Databases (VLDB '05)*, 2005.

[6] M. Naaman, H. Becker, and L. Gravano. Hip and Trendy: Characterizing emerging trends on Twitter. *Journal of the American Society for Information Science and Technology*, 62(5):902–918, 2011.

[7] T. Sakaki, M. Okazaki, and Y. Matsuo. Earthquake shakes Twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th ACM International Conference on World Wide Web (WWW '10)*, 2010.

[8] J. Sankaranarayanan, H. Samet, B. Teitler, M. Lieberman, and J. Sperling. Twitterstand: News in tweets. In *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems (GIS '09)*, 2009.

[9] R.W. Sinnott. Virtues of the Haversine. *Sky and Telescope*, 68(2):159, 1984.

[10] A. Strehl and J. Ghosh. Cluster ensembles - A knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, 3:583–617, 2002.

Event Detection from Social Media Data

George Valkanas [1], Dimitrios Gunopulos [2]
Department of Informatics & Telecommunications, University of Athens
Athens, Greece
{[1]gvalk, [2]dg}@di.uoa.gr

Abstract

Microblogging platforms, such as Twitter, Tumblr etc., have been established as key components in the contemporary Web ecosystem. Users constantly post snippets of information regarding their actions, interests or perception of their surroundings, which is why they have been attributed the term Live Web. Nevertheless, research on such platforms has been quite limited when it comes to identifying events, but is rapidly gaining ground. Event identification is a key step to news reporting, proactive or reactive crisis management at multiple scales, efficient resource allocation, etc. In this paper, we focus on the problem of automatically identifying events as they occur, in such a user-driven, fast paced and voluminous setting. We propose a novel and natural way to address the issue using notions from emotional theories, combined with spatiotemporal information and employ online event detection mechanisms to solve it at large scale in a distributed fashion. We present a modular framework that incorporates these ideas and allows monitoring of the Twitter stream in real time.

1 Introduction

The web ecosystem has changed dramatically over the last decade, with the users becoming its driving force. A major shift has been that users are no longer passive observers but actively engage in online activities and experiences. Social media sites, such as Facebook, Twitter, Flickr, etc, have been at the forefront of this change, providing the necessary platforms for users to share aspects of their everyday lives online. For instance, Twitter now counts more than 200 million active users, with an approximate 400 million "tweets" on a daily basis [1]. Users can post short messages, up to 140 characters, mimicking a web-based version of the cell-phone SMS technology. The result is a constant flow of user generated content, arriving at varying rates depending on various factors, and is usually referred to as the Twitter stream.

Social media are complementary to online blogs (web logs), where the former contain snippets of more up-to-date information, while the latter are used for expressing ones thoughts, ideas, beliefs and are the result of a more thought-through process. The speedy nature of social media sites has earned them the name "*Live web*" or "*Now web*". In that respect, these platforms may serve as real-time news reporting and / or crisis-management services, as exemplified with the recent political termoil in the Middle East, with Japanese earthquakes [1], or the 2007 Southern California wildfires [2]. Given their prominent role in

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

[1]https://business.twitter.com/audiences-twitter, access Aug 2013

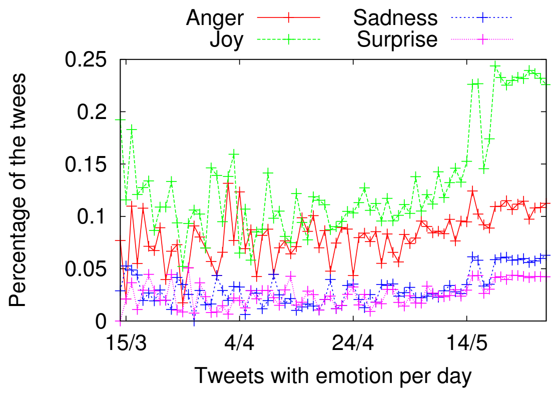


Figure 1: A timeseries of daily emotions from Twitter, between March 15 and May 24 2012

disseminating information today, it comes as no surprise that social media sites and their properties have come under considerable attention by both the academia and the industry.

One of the basic applications for analysing social media, is the problem of identifying real-life events as they happen or short after from their impact in social media. Generally we take *event* to mean an important phenomenon with a local extend and a temporal dimension in the physical world. Despite the obvious advantages in being able to do so, automatically identifying real-life events from social media data is not easy. Some of the challenges are: i) The large adoption means that we must process in *real time* voluminous amounts of data. ii) The content is usually *short*, noisy, and diverse in terms of location, languages and topics. Finally, iii) user location is also a scarce commodity leading to several techniques for location extraction [1, 3–5].

Taking into account these impediments, it is no surprise that most existing works that deal with event detection in Twitter simplify the problem by focusing on detecting events of specific event type, monitoring the stream for specific terms, or #hashtags (i.e., user generated topic labels). Clearly such approaches are useful but limited to work only when the event can be described by a small set of terms, e.g., âĂIJ[..] now shaking [..]âĂİ for earthquakes. Detecting new events by such means is difficult as the descriptive terms have to be known a priori.

Motivated by these shortcomings of existing work, we address the problem of detecting events in a stream of short-form messages, focusing on Twitter. The main goal is to devise techniques that work regardless of the category the events belong to. We take a novel approach and employ techniques grounded on influential theories of emotions, such as *Cognitive* and *Affective* [6]. According to these theories, users feel a need to express themselves as a result of an event.

Our goal is to use the Twitter stream to access such reactions. Moreover, we argue that such tweets will not be a flat description of the event, but will also convey the user's *emotional state*, partially disclosing how it affected them. An event can then be modeled as a *time*- and *place*- related phenomenon, which triggered a significant change in the emotional state of a (potentially large) group of people and our goal is to automatically capture such sudden changes. Figure 1 validates our claim: We plot the relative occurrence of the 4 most prominent emotions, from a sample of the Twitter stream, between May and March 2012. We ommit neutral tweets, which we assert to be non-informative. Surges in *anger* in early April are related with the Syrian uprising, whereas the high values of *joy* towards the end are due to the Champions League final, and the Eurovision song contest.

The rest of the paper is organized as follows: Section 3.1 discusses our event detection model and algorithmic approach, followed by Section 3 which describes the system that realises our approach. Section 2 presents related work on the event detection problem. Section 4 concludes our work and presents future directions.

2 Event Modeling And Detection

We begin by formalising the event detection problem. Following [7], *An event* $\mathbf{e}$ *is a real-world phenomenon, that occurred at some specific time* $\mathbf{t}$ *and is usually tied to a location* $\mathbf{l}$. However, using social media data, we can mainly monitor the aftermath of the event, i.e., its effects on actual people and how these are reflected in people's reactions in social media. According to influential theories of emotions [6], events will impact the users that experienced it, who will be urged to externalize their reactions, e.g., tweet about it. We expect such spontaneous reactions to convey a user's emotional state, i.e., *how* the event affected them. Making this motivation more concrete, we state our problem as follows:

Problem Statement 1: [Event Detection] Given a time ordered stream of tweets as input, identify those messages which i) alter significantly and abruptly the emotional state of a (potentially) large group of users, and ii) can be traced back to event e.

This definition fits well with an outlier detection formalisation, whereby we observe a sudden and significant change in the emotions of users, with respect to the recent history, as a result of an event taking place. However, in our definition we do not monitor individual users, using aggregate counts instead. Monitoring the reactions of individual users is very inefficient in terms of resources; however more a important problem is the ethical questions raised regarding a user's privacy as well.

To address these limitations, we use *aggregate* information from large, geographically associated groups user. Users are clustered together according to their geographical location, which we extract from available information. We then monitor the emotional state of each geographically distributed group, independently of the others and report an event when the group's *cumulative* emotional state changes suddenly. Note that this approach covers inherently the part of the definition that wants the event to affect large groups of users.

Instead of putting all users to a single group, which has no local coherency, we decompose $\mathcal{G}$ into smaller groups $\mathcal{G}_i$ and organize them hierarchically. We denote $\mathcal{G}_i^j$ as group i at level j, assuming leaf nodes at $j = 0$. The hierarchy can be administrative (e.g., country, state, etc.), or constructed algorithmically, e.g., via hierarhical clustering. For a fixed level j in the hierarchy, it holds that $\cup\mathcal{G}_i^j = \mathcal{G}$ and $\cap\mathcal{G}_i^j = \emptyset$, and $\mathcal{G}_i^j = \cup\mathcal{G}_k^{j-1}$. This decomposition offers a trade-off of high-level granularity versus a higher need in resources.

Each group $\mathcal{G}_i^0$ is then monitored by a virtual sensor s_i. Each s_i processes all of the tweets from that group. Upon arrival, each tweet is classified to one of 7 emotions: the 6 basic emotions suggested by Paul Ekman [8]: *anger, fear, disgust, happiness, sadness, surprise*, plus a *none* state. Tweets of the *none* state are not considered further, on the grounds that they are uninteresting, e.g., they reflect a mundane task. Sensors aggregate the rest of the incoming tweets along the temporal dimension, for each emotion separately. Using an *aggregation interval* a (e.g., a=1min), each s_i produces a single value for each emotion, which is the respective *count* of tweets for that emotion during a. The aggregation interval acts as a discretization unit, to cope with the streaming nature of the medium. The sensor operates over the w most recent points with a sliding window. The combination of a and w define the history that the sensor keeps track of.

Example: Assume, for instance, a sensor s_i, with a = 5 minutes and w = 12. The sensor maintains a history of the past 5 $\times$ 12 = 60 minutes. Every 5 minutes, s_i will process a single value for each emotion, extracted from the tweets received during that interval from the group of users that it monitors. The oldest point will be discarded and the new one will take its place.

2.1 Approximating the Emotional State Distribution

Given that a user's emotional state is a result of several factors, it would be unfounded to assume that it will follow a predefined distribution, much less a static one. To approximate it efficiently in an online fashion we estimate the Probability Density Function (*PDF*) of the emotional distribution of each group $\mathcal{G}_i$, and we do that through kernel estimators. According to kernel estimation, each point distributes its weight in around it, and the *kernel function* $k(x)$ describes how this is done. The distribution we want to approximate is given by $f(x)$

$$f(x) = \frac{1}{|\mathcal{T}|} \sum_{r \in \mathcal{R}} k(\overline{r} - \overline{x})$$

Here, $\mathcal{T}$ is the actual set of values that we want to approximate, $\mathcal{R}$ is a data sample maintained online by each s_i, and $k(x)$ is the kernel function. We opt for the Epanechnikov kernel function, which has a closed form integral, and can thus be computed very efficiently, given by:

$$k(x) = \begin{cases} (\frac{3}{4})^d \frac{1}{B_1 B_2..B_d} \prod_{1 \leq i \leq d}(1 - (\frac{x_i}{B_i})^2) \\ \qquad\qquad\qquad \text{if } \forall i, 1 \leq i \leq d, |\frac{x_i}{B_i}| < 1 \\ \\ 0, otherwise \end{cases}$$

where B_i is the kernel's bandwidth, computed with Scott's rule [9], $B_i = \sqrt{5}\sigma_i|\mathcal{R}|^{-\frac{1}{d+4}}$, and σ_i is the standard deviation for the i-th dimension (i.e., emotion). For simplicity, we ignore the interplay of emotions, and set $d = 1$. Although values need to be normalized in the $[0, 1]^d$ space, we do not find this really restrictive: A straightforward solution is to normalize with the maximum value allowed by the system's architecture (e.g., $2^{32} - 1$ for int). Alternatively, system specification requirements can indicate the load it must sustain, which will also be an upper bound (within constant factor) on the values it can process.

To approximate the data distribution, we need i) a random sample over the data that fall within the window w, and ii) the standard deviation σ of the values in w, both of which can be easily maintained online. We use "chain sampling" [10] to produce the random sample. Chain sampling selects a point s from the sample to evict and replaces it with the new point p, regardless of s being expired or not. Sampling and smoothing using a kernel function can be seen as an indirect way for filtering out spurious bursts while improving the scalability of the system.

2.2 Event Detection

We can now use the kernel density estimator to identify changes in the data distribution. The rationale is to identify events on the basis that the most recent aggregate emotional state observed by sensor s_i was not "as expected", according to s_i's history. Therefore, if a sudden change was observed, this could be caused by an external phenomenon. To characterize a new point p as a significant deviation, we first compute its probability mass over the sample $\mathcal{R}$, by evaluating the quantity

$$P(p, r) = \frac{1}{|\mathcal{R}|} \int_{[p-r, p+r]} \sum_{t_i \in \mathcal{R}} k(x - t_i) dx$$

The value r defines a neighborhood range, within which to search for points from $\mathcal{R}$. From the definition of the Epanechnikov kernel, the values need to be in the $(p_i - r - B_i, p_i + r + B_i)$ range, to contribute to the integral. If $P(p, r)$ is below a certain threshold, we say that p is an outlier, i.e. a significant change was detected in the emotional state of the observed population. Since this could be the result of an occurring event, we should trigger additional mechanisms to describe it. Therefore, event detection is decoupled from event description.

3 The TwInsight System

From the description of an event e and our event detection mechanism, it should be clear by now that we need the following information: a location l, the time of occurrence t, a set of keywords to describe it, and the emotions that were elicited as a result of the event. Figure 2 shows a schematic view of the components required to extract that information and their interaction[2].

[2]Storage image by Barry Mieny, under CC BY-NC-SA license.

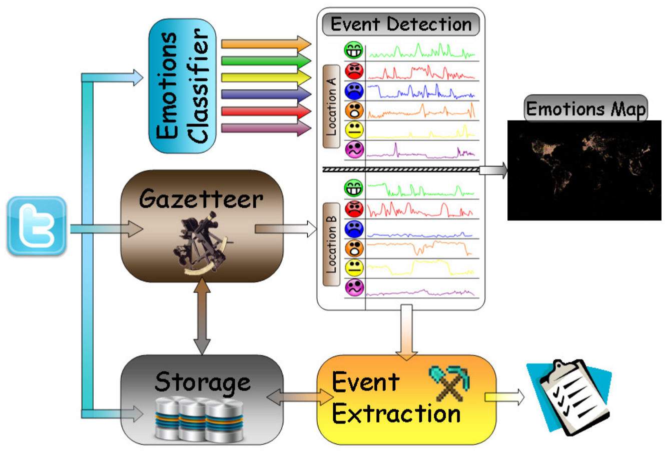


Figure 2: Schematic interaction of our system's components

The Twitter stream is our system's input, feeding two components, namely the *emotions classifier* and the *location extraction* subsystem. Locations are extracted through a custom built component [5], whereby each incoming tweet is mapped to a location. Tweets are then forwarded to the virtual sensor s_i responsible for the location it was mapped to.

Meanwhile, the tweet has been classified to one of the 7 emotions that we use. Neutral tweets are not considered for further processing, but are stored nonetheless. It is worth noting that this approach allows for an elegant integration with spam detection mechanisms: spam tweets can be cast to the *neutral* class, thereby preventing them from any subsequent processing.

When a sensor receives a tweet for further processing, we know its location and which of the 6 basic emotions it has been cast to. For each emotion, separately from the rest, the sensor counts how many tweets it has received during the last aggregation interval a. Each count is the input to a separate instance of our event detection mechanism, one for each emotion. Therefore, on each sensor, there are 6 instances of event detection mechanisms, executing simultaneously. Each event detection module updates its values and identifies whether a surge, i.e., an event, in any emotional state has occurred.

If an emotional surge was identified (i.e., an event), we report the end time of the aggregation interval as the event's time of occurrence t. Additionally, the tweet ids that caused the peak for that emotion are passed to the event extraction mechanism, which is responsible for summarizing the event. This step will provide the descriptive keywords of the event, and its operation is subject to the detection of an event. Since event detection and description are decoupled, several techniques can be used to describe the event: term frequency or TF-IDF score, summaries, etc. In any case, the user will be presented all of the necessary information: *location, timestamp, emotion* and *description*.

Table 7: Average Component Processing Time (ms)

	Location Extraction	**Classification**	**Event Detection**	**Total**
TIME:	3.36	0.35	0.001	3.72

Table 7 illustrates the efficiency of our system, **TwInsight** [11], where we show the average time taken by each component to apply its functionality on a newly received tuple. Table 8 provides some examples of events extracted by applying our method to a stream of tweets obtained between April and May 2012. A contextual user interface can also facilitate the presentation of this information, as described in [12].

Event 1 is related to the goal by Bayern's football player, Thomas Müller, in the Champions League

Table 8: Sample Summary of 15 Prominent Events Identified By *TwInsight*

ID	Emotion	Where	When (GMT)	Description
1	Joy	Germany	19/05, 20:23	thomas bayern championsleague cfc mueller muller müller
2	Joy	UK	19/05, 20:29	didier drogba f... beauty enjoying fair gal gaz goal great
3	Sadness	Canada	20/05, 22:42	died b... breaking mio robin singer @rodneyedwards gib gibb opa
4	Anger	Canada	20/5, 15:19	@ctvcalgary aime ambition chacun earthquake femme frais http://t.co/0hJEez9Q italy kills
5	Anger	US	20/5, 11:23	@Mou2amara alive assad onus prove regime shawkat showusshaukat syria

(CL) 2012 final, that took place on May 19. The goal was scored in the 83rd minute of the match, i.e. on 22:23 CEST (20:23 GMT). This places our finding the event the moment that it actually occurred and was posted. We identify similar tweets in Canada and Spain, at the *exact* same timestamp. Clearly, the event is related with Joy.

Event 2 is about the equilizer goal by Didier Drogba in the CL finals. The goal was scored in the 88th minute, i.e. on 22:28 CEST (20:28 GMT), and we identify several joyous tweets on 20:29, right after the goal.

Event 3 is about the death of Bee Gee's singer Robin Gibb. He was pronounced dead at 23:30 BST (22:30 GMT) on May 20th [3], and a surge in sad tweets is seen at 22:42, only 10' after his death.

Event 4 is about the earthquake in Italy, on May 20, that resulted in the death of six people.

Event 5 refers to Assef Shawkat, deputy Minister of Defense of Syria. On May 20, 2012, there was a claim he had been murdered [4], and tweets requesting proof were posted. We have also found tweets on 26th and 27th of May regarding the Houla Massacre of the Syrian civic war which occurred on May 25. We ommit such tweets, as they contain URLs to pictures of immense brutality.

From the list of events presented above, there are two things we would like to point out:

- Our approach is able to identify events of various types. We see events related to sports, earthquakes, popular personalities, and politics.
- We are able to identify such events promptly, as indicated by the first three events. This means that such a method is not only useful as a news reporting tool, but could be crucial in dealing with emergency and disaster management situations.

4 Related Work

Event identification from Twitter is gaining attention. Early works focus on events of specific types, e.g. earthquakes [1] or news [13]. The idea is to whitelist specific keywords and phrases, but such approaches are destined to fail when the event type is not known in advance. The technique we present here was introduced in [14].

A closely related concept is *trending topics*, i.e., terms which gain in popularity over a period of time. However, trends are not necessarily indicative of events; rather the contrary, since they are *always* present.

[3]http://www.bbc.co.uk/news/entertainment-arts-18140862

[4]http://newsfromsyria.com/2012/05/20/asef-shawkat-assassinated/

They are also prone to topical groups, e.g., a big fan base, and could be the result of recurring phenomena, such as TV shows, or *memes*, e.g., the “Follow Friday” (#FF) hashtag.

Type-independent techniques typically employ online clustering methods. Though such methods may be appropriate for some slow-paced settings [15], the data volume makes them unfit for microblogs [16], They are also sensitive to popular terms or large groups of users with similar interests, and can be easily gamed by spammers [17]. Finally, as shown in [11], these techniques require extremely careful data cleaning and preprocessing to be able to work in practice.

Taking into account these impediments, early techniques simplified the problem by focusing on a specific event type [1,18,19]. They then monitor online data for specific terms that can be used to describe the event. However, this can only work when the event can be described by a handful of terms, e.g., “[..] *now shaking* [..]” for earthquakes. Clearly such approaches cannot detect new events for which the descriptive terms are unknown a priori. Recent approaches also correlate information coming from other sources with data from the twitter stream to understand events [20].

Online clustering solutions [15,16,21] are used to identify events and trends in Twitter. Such approaches generally suffer from scalability issues [16, 22], and coping with the increasing volume of the data is a research issue itself.

5 Conclusion

In this paper, we focused on the problem of automatically identifying events from the *Live Web as they occur*. We combined notions from emotional theories with spatiotemporal information, and tackled the problem using online event detection techniques. We integrated our ideas in a modular framework and experimentally demonstrated the validity and scalability of the method.

In future work we will work to develop the system along several thrusts: i) improve the performance of location extraction method, by applying online location clustering, using GPS signals, and by using information about the Twitter graph to estimate the location of a tweet from the location of related Twitter users, ii) improve the event description by incorporating novel summarization techniques, iii) improve the classification accuracy to filter uninformative points.

Acknowledgements: The authors would like to thank the data annotators. This work has been co-financed by EU and Greek National funds through the Operational Program "Education and Lifelong Learning" of the National Strategic Reference Framework (NSRF) - Research Funding Programs: Heraclitus II fellowship, THALIS - GeomComp, THALIS - DISFER, ARISTEIA - MMD" and the EU funded project INSIGHT[5].

References

[1] T. Sakaki, M. Okazaki, and Y. Matsuo, “Earthquake shakes twitter users: real-time event detection by social sensors,” in *WWW*, 2010.

[2] J. Sutton, L. Palen, and I. Shlovski, “Back-channels on the front lines: Emerging use of social media in the 2007 southern california wildfires,” 2008.

[3] J. Eisenstein, B. O’Connor, N. A. Smith, and E. P. Xing, “A latent variable model for geographic lexical variation,” in *EMNLP*, 2010.

[4] L. Hong, A. Ahmed, S. Gurumurthy, A. J. Smola, and K. Tsioutsiouliklis, “Discovering geographical topics in the twitter stream,” ser. WWW, 2012.

[5] G. Valkanas and D. Gunopulos, “Location extraction from social networks with commodity software and online data,” in *ICDM Workshops (SSTDM)*, 2012.

[6] M. Mikolajczak, V. Tran, C. Brotheridge, and J. J. Gross, *Using an emotion regulation framework to predict the outcomes of emotional labour*. Bingley, UK: Emerald, 2009.

[5]www.insight-ict.eu

[7] H. Becker, F. Chen, D. Iter, M. Naaman, and L. Gravano, "Automatic identification and presentation of twitter content for planned events," in *ICWSM*, 2011.

[8] P. Ekman, W. Friesen, and P. Ellsworth, *Emotion in the human face: guide-lines for research and an integration of findings*. Pergamon Press, 1972.

[9] D. Scott, *Multivariate Density Estimation: Theory, Practice, and Visualization*, ser. Wiley series in probability and mathematical statistics: Applied probability and statistics. Wiley, 1992.

[10] B. Babcock, M. Datar, and R. Motwani, "Sampling from a moving window over streaming data," in *SODA*, 2002.

[11] G. Valkanas and D. Gunopulos, "How the live web feels about events," in *CIKM*, 2013 (to appear).

[12] ——, "A ui prototype for emotion-based event detection in the live web," in *CHI-KDD*, 2013, pp. 89–100.

[13] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling, "Twitterstand: news in tweets," in *SIGSPATIAL-GIS*, 2009.

[14] G. Valkanas and D. Gunopulos, "How the live web feels about events," in *ACM CIKM 2013*, 2013.

[15] H. Becker, M. Naaman, and L. Gravano, "Learning similarity metrics for event identification in social media," ser. WSDM, 2010.

[16] F. Alvanaki, S. Michel, K. Ramamritham, and G. Weikum, "See what's enblogue: real-time emergent topic identification in social media," in *EDBT*, 2012.

[17] C. Grier, K. Thomas, V. Paxson, and M. Zhang, "@spam: the underground on 140 characters or less," in *CCS*, 2010.

[18] E. Benson, A. Haghighi, and R. Barzilay, "Event discovery in social media feeds," in *ACL-HLT*, 2011.

[19] D. A. Shamma, L. Kennedy, and E. F. Churchill, "Tweet the debates: understanding community annotation of uncollected sources," in *WSM*, 2009.

[20] E. M. Daly, F. Lecue, and V. Bicer, "Westland row why so slow?: fusing social media and linked data sources for understanding real-time traffic conditions."

[21] M. Mathioudakis and N. Koudas, "Twittermonitor: trend detection over the twitter stream," in *SIGMOD*, 2010.

[22] T. Lappas, M. R. Vieira, D. Gunopulos, and V. J. Tsotras, "On the spatiotemporal burstiness of terms," *PVLDB*, vol. 5, no. 9, 2012.

Large Scale Tensor Decompositions: Algorithmic Developments and Applications

Evangelos Papalexakis[⋆∗], U Kang[†] , Christos Faloutsos[⋆], Nicholas Sidiropoulos[§], Abhay Harpale[⋆]
[⋆] Carnegie Mellon University, [†] KAIST, [§] University of Minnesota

Abstract

Tensor decompositions are increasingly gaining popularity in data science applications. Albeit extremely powerful tools, scalability to truly large datasets for such decomposition algorithms is still a challenging problem. In this paper, we provide an overview of recent algorithmic developments towards the direction of scaling tensor decompositions to big data. We present an exact Map/Reduce based algorithm, as well as an approximate, fully parallelizable algorithm that is sparsity promoting. In both cases, careful design and implementation is key, so that we achieve scalability and efficiency. We showcase the effectiveness of our methods, by providing a variety of real world applications - whose volume previously rendered their analysis very hard, if not impossible- where our algorithms were able to discover interesting patterns and anomalies.

1 Introduction

Tensors and tensor decompositions are powerful tools, and are increasingly gaining popularity in data analytics and mining. Despite their power and popularity, tensor decompositions prove very challenging when it comes to scalability towards big data. Tensors are, essentially, multidimensional generalizations of matrices; for instance, a two dimensional tensor is a plain matrix, and a three dimensional tensor is a cubic structure.

As an example, consider a knowledge base, such as the "Read the Web" project [1] at Carnegie Mellon University, which consists of (noun phrase, context, noun phrase) triplets, such as ("Obama", "is the president of", "USA"). Figure 1 demonstrates how we can formulate this data as a three mode tensor and how we may analyze it in latent concepts, each one representing an entire cluster of noun phrases and contexts.

Alternatively, consider a social network, such as Facebook, where users interact with each other, and post on each others' "Walls". Given this posting activity over time, we can formulate a tensor of who interacted with whom and when; subsequently, by decomposing the tensor into concepts, as in Fig. 1, we are able to identify cliques of friends, as well as anomalies.

In this paper, we present a brief overview of tensor decompositions and their applications in the social media context, geared towards scalability.

[∗]Funding was provided to the first three authors by the U.S. ARO and DARPA under Contract Number W911NF-11-C-0088, by DTRA under contract No. HDTRA1-10-1-0120, by ARL under Cooperative Agreement Number W911NF-09-2-0053, and by grant NSF IIS-1247489. N. Sidiropoulos was partially supported by ARO contract W911NF-11-1-0500 and by NSF IIS-1247632 Any opinions, findings, and conclusions or recommendations expressed in this material are those of the author(s) and do not necessarily reflect the views of the funding parties

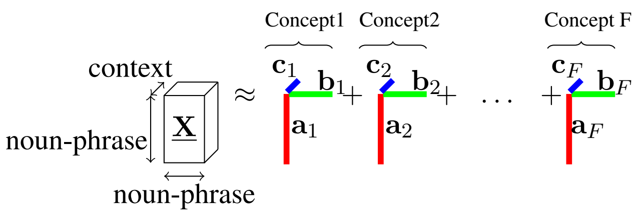


Figure 1: PARAFAC decomposition of three-way tensor as sum of F outer products (rank-one tensors), reminiscent of the rank-F Singular Value Decomposition of a matrix.

1.1 A note on notation

Tensors are denoted as $\mathcal{X}$. Matrices are denoted as $\mathbf{X}$. Vectors are denoted as $\mathbf{x}$. We use Matlab notation for matrix indexing, e.g. $\mathbf{A}(1,2)$ refers to the (1,2) element of $\mathbf{A}$, and $\mathbf{A}(:,1)$, refers to the entire first column of $\mathbf{A}$. The rest of the symbols used are defined throughout the text.

1.2 Tensors and the PARAFAC decomposition

Tensors are multidimensional matrices; in tensor terminology, each dimension is called a 'mode'. The most popular tensors are three mode ones, however, there exist applications that analyze tensors of higher dimensions. There is a rich literature on tensor decompositions; we refer the interested reader to [12] for a comprehensive overview thereof. This work focuses on the PARAFAC decomposition [8] (which is the one shown in Fig. 1).

The PARAFAC decomposition can be seen as a generalization of matrix factorizations, such as the Singular Value Decomposition, in higher dimensions, or as they are referred to in tensor literature, *modes*.

The PARAFAC [7, 8] (also known as CANDECOMP/PARAFAC or Canonical Polyadic Decomposition) tensor decomposition of $\mathcal{X}$ in F components is $\mathcal{X} \approx \sum_{f=1}^{F} \mathbf{a}_f \circ \mathbf{b}_f \circ \mathbf{c}_f$, where $[\mathbf{a} \circ \mathbf{b} \circ \mathbf{c}](i,j,k) = \mathbf{a}(i)\mathbf{b}(j)\mathbf{c}(k)$ and denotes the three mode outer product.

Often, we represent the PARAFAC decomposition as a triplet of matrices $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$, i.e. the f-th column of which contains $\mathbf{a}_f, \mathbf{b}_f$ and $\mathbf{c}_f$, respectively.

Definition 1 (Tensor Matricization): We may matricize a tensor $\mathcal{X} \in R^{I \times J \times K}$ in the following three ways: $\mathbf{X}_{(\mathbf{1})}$ of size $(I \times JK)$, $\mathbf{X}_{(\mathbf{2})}$ of size $(J \times IK)$ and $\mathbf{X}_{(\mathbf{3})}$ of size $(K \times IJ)$. We refer the interested reader to [10] for details.

Definition 2 (Kronecker product): The Kronecker product of $\mathbf{A}$ and $\mathbf{B}$ is:

$$\mathbf{A} \otimes \mathbf{B} := \begin{bmatrix} \mathbf{B}\mathbf{A}(1,1) & \cdots & \mathbf{B}\mathbf{A}(1,J_1) \\ \vdots & \ddots & \vdots \\ \mathbf{B}\mathbf{A}(I_1,1) & \cdots & \mathbf{B}\mathbf{A}(I_1,J_1) \end{bmatrix}$$

If $\mathbf{A}$ is of size $I_1 \times J_1$ and $\mathbf{B}$ of size $I_2 \times J_2$, then $\mathbf{A} \otimes \mathbf{B}$ is of size $I_1 I_2 \times J_1 J_2$.

Definition 3 (Khatri-Rao product): The Khatri-Rao product (or column-wise Kronecker product) $(\mathbf{A} \odot \mathbf{B})$, where $\mathbf{A}, \mathbf{B}$ have the same number of columns, say F, is defined as:

$$\mathbf{A} \odot \mathbf{B} = \left[\mathbf{A}(:,1) \otimes \mathbf{B}(:,1) \cdots \mathbf{A}(:,F) \otimes \mathbf{B}(:,F)\right]$$

If $\mathbf{A}$ is of size $I \times F$ and $\mathbf{B}$ is of size $J \times F$ then $(\mathbf{A} \odot \mathbf{B})$ is of size $IJ \times F$.

The Alternating Least Squares Algorithm for PARAFAC. The most popular algorithm for fitting the PARAFAC decomposition is the Alternating Least Squares (ALS). The ALS algorithm consists of three steps, each one being a conditional update of one of the three factor matrices, given the other two. Without delving into details, the update of, e.g., the factor matrix $\mathbf{A}$, keeping $\mathbf{B}, \mathbf{C}$ fixed, involves the computation and pseudoinversion of $(\mathbf{C} \odot \mathbf{B})$ (and accordingly for the updates of $\mathbf{B}, \mathbf{C}$). For a detailed overview of the ALS algorithm, see [7, 8, 12].

2 Related Work

2.1 Applications

In [11], the authors incorporate contextual information to the traditional HITS algorithm, formulating it as a tensor decomposition. In [5] the authors analyze the ENRON email social network, formulating it as a tensor. In [2] the authors introduce a tensor-based framework in order to identify epileptic seizures. In [17], the authors use tensors in order to incorporate user click information and improve web search. The list continues, including applications such as [13], [16], and [2].

2.2 State of the art

The standard framework for working with tensors is Matlab; there exist two well known toolboxes, both of very high quality: The Tensor Toolbox for Matlab [4, 6] (specializing in sparse tensors) and the N-Way Toolbox for Matlab [3] (specializing in dense tensors).

In [15], the authors propose a partition-and-merge scheme for the PARAFAC decomposition which, however, does not offer factor sparsity. In terms of parallel algorithms, [19] introduces parallelization strategies for speeding up each factor matrix update step in the context of alternating optimization. Finally, [16, 18] propose randomized, sampling based tensor decompositions (however, the focus is on a different tensor model, the so called Tucker3).

The latest developments on scalable tensor decompositions are the works summarized in this paper: in [9], a massively distributed Map/Reduce version of PARAFAC is proposed, where, after careful design, issues fatal to scalability are effectively alleviated. In [14], a sampling based, parallel and sparsity promoting, approximate PARAFAC decomposition is proposed.

3 Scaling Tensor Decompositions Up

3.1 Main Challenge

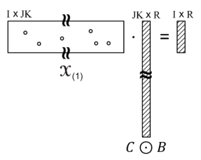
Previously, when describing the ALS algorithm, we mentioned that the update of $\mathbf{A}$ involves manipulation of $(\mathbf{C} \odot \mathbf{B})$. This is the very weakness of the traditional ALS algorithm, a naive implementation thereof will have to materialize matrices $(\mathbf{C} \odot \mathbf{B})$, $(\mathbf{C} \odot \mathbf{A})$, and $(\mathbf{B} \odot \mathbf{A})$, for the respective updates of $\mathbf{A}, \mathbf{B}$, and $\mathbf{C}$.

Problem 1 (Intermediate Data Explosion): The problem of having to materialize $(\mathbf{C} \odot \mathbf{B})$, $(\mathbf{C} \odot \mathbf{A})$, and $(\mathbf{B} \odot \mathbf{A})$ is defined as the intermediate data explosion.

In order to give an idea of how devastating this intermediate data explosion problem is, consider the knowledge base dataset, such as the one referenced in the Introduction. The version of the data that we analyzed, consists of about $26 \cdot 10^6$ noun-phrases. Consequently, a naive implementation of ALS would generate and store a matrix of $\approx 7 \cdot 10^{14}$. As an indication of how devastating this choice is, we would probably need a data center's worth of storage, just to store this matrix, let alone manipulate it.

In [4], Bader et al. introduce a way to alleviate the above problem, when the tensor is stored in Matlab sparse format. However, this implementation is bound by Matlab's memory limitations.

In the following subsections we provide an overview of our two recent works, which both achieve scalability, pursuing two different directions.


(a)

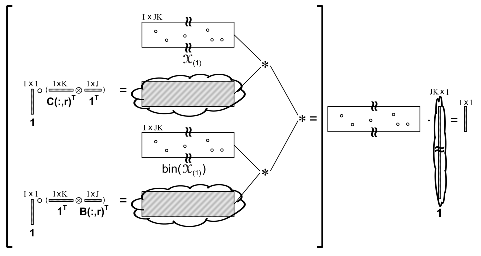

(b)

Figure 2: Subfigure (a): The intermediate data explosion problem in computing $\mathbf{X_{(1)}}(\mathbf{C} \odot \mathbf{B})$. Although $\mathbf{X_{(1)}}$ is sparse, the matrix $\mathbf{C} \odot \mathbf{B}$ is very dense and long. Materializing $\mathbf{C} \odot \mathbf{B}$ requires too much storage: e.g., for $J = K \approx 26$ million as the main dataset of [9], $\mathbf{C} \odot \mathbf{B}$ explodes to *676 trillion* rows. Subfigure (b): Our solution to avoid the intermediate data explosion. The main idea is to decouple the two terms in the Khatri-Rao product, and perform algebraic operations using $\mathbf{X_{(1)}}$ and $\mathbf{C}$, and then $\mathbf{X_{(1)}}$ with $\mathbf{B}$, and combine the result. The symbols $\circ, \otimes, *$, and $\cdot$ represents the outer, Kronecker, Hadamard (element-wise), and the standard product, respectively. Shaded matrices are dense, and empty matrices with several circles are sparse. The clouds surrounding matrices represent that the matrices are *not* materialized. Note that the matrix $\mathbf{C} \odot \mathbf{B}$ is never constructed, and the largest dense matrix is either the $\mathbf{B}$ or the $\mathbf{C}$ matrix. Both figures are taken from [9].

3.2 GigaTensor

GigaTensor, which was introduced in [9], is a highly scalable, distributed implementation of the PARAFAC decomposition. Scalability was achieved through a crucial simplification of the algorithm, a series of careful design choices and optimizations. Here, we provide an overview of our most important contributions in [9].

We observed that $\mathbf{X_{(1)}}(\mathbf{C} \odot \mathbf{B})$ can be computed without explicitly constructing $\mathbf{C} \odot \mathbf{B}$. The theorem below solidifies our observation:

Theorem

Computing $\mathbf{X_{(1)}}(\mathbf{C} \odot \mathbf{B})$ *is equivalent to computing* $(\mathbf{N_1} * \mathbf{N_2}) \cdot \mathbf{1}_{JK}$*, where* $\mathbf{N}_1 = \mathbf{X_{(1)}} * (\mathbf{1}_I \circ (\mathbf{C}(:,f)^T \otimes \mathbf{1}_J^T))$*,* $\mathbf{N}_2 = bin(\mathbf{X_{(1)}}) * (\mathbf{1}_I \circ (\mathbf{1}_K^T \otimes \mathbf{B}(:,f)^T))$*, and* $\mathbf{1_{JK}}$ *is an all-1 vector of size* JK*, and* $f = 1 \cdots F$.

The $bin()$ function converts any nonzero value into 1. As a result, we can simplify every step of the ALS algorithm *without sacrificing accuracy*, since the above theorem states that both operations are *equivalent*. Computing $\mathbf{X_{(1)}}(\mathbf{C} \odot \mathbf{B})$ naively would require a total of $JKF + 2\text{nnz}(\mathcal{X})F$ flops, and $JKF + \text{nnz}(\mathcal{X})$ intermediate data size, where $\text{nnz}(\mathcal{X})$ denotes the number of non-zeros in $\mathcal{X}$. On the other hand, the method GigaTensor requires $5\text{nnz}(\mathcal{X})F$ flops, and $max(J + \text{nnz}(\mathcal{X}), K + \text{nnz}(\mathcal{X}))$ intermediate data size. Figure 2(b) illustrates our approach.

Other contributions in [9] include:

Order of computations: By leveraging associativity properties of the algebraic operations of ALS, we chose the ordering of operations that yields the smallest number of *flops*. As an indication of how crucial this optimization is, for the knowledge base dataset that we analyze in [9], a naive ordering would incur 2.5×10^{17} *flops*, whereas the ordering that we select results in 8×10^9 *flops*.

Parallel Outer Products: Throughout the algorithm, we need to compute products of the form $\mathbf{A}^T\mathbf{A}$. We leverage row-wise matrix partitioning; we store each row of a matrix separately in HDFS, thus enabling efficient matrix self join. Using column-wise storage would render this prohibitively expensive.

Distributed Cache Multiplication: We broadcast small matrices to all reducers, thus eliminating unnecessary loading steps. This improves both the latency, as well as the size of the intermediate results produced and stored by the algorithm.

3.3 PARCUBE

On a different note, in [14], we introduce PARCUBE, an approximate, parallelizable algorithm for PARAFAC; on top of parallelizability, PARCUBE is sparsity promoting: starting from a sparse tensor, the algorithm operates, through its entire lifetime, on sparse data, and it finally produces a sparse output. In this way, we both alleviate the intermediate data explosion problem, as well as producing sparse factors, an attribute which is vital for interpretation purposes.

The algorithm, roughly, consists of the following steps:

Biased sampling: We use biased sampling to select indices from all three modes of the tensor, creating a significantly smaller tensor. The sample bias is proportional to the marginal sum for each of the three modes. We may draw multiple such samples, and indeed, we show in [14] that this improves accuracy.

Parallel decomposition on the samples: The second step includes fitting of the PARAFAC decomposition to the sample tensors, obtained from the previous step. This step can be performed entirely in parallel, offering great speedup gains.

Merging of partial results: The final step is merging the intermediate decomposition results into a full sized set of decomposition factors. In [14], we introduce the FACTORMERGE, which is pictorially represented in Fig. 3(b). The original paper contains theoretical justification of the algorithm's correctness.

Figure 3 contains a pictorial description of PARCUBE.

3.4 Results & Discoveries

Comparison against the state of the art

At the time when [9] and [14] were written, Tensor Toolbox [6] was the state of the art (and excluding the work we showcase here, still is the state of the art), hence we chose it as our baseline. Comparison of GigaTensor against the Tensor Toolbox was made, merely, to show that it was able to handle data at least two orders of magnitude larger than what the state of the art was able to.

Figure 4 illustrates the comparison of both methods with the state of the art, in terms of scalability, as well as output sparsity (for PARCUBE). Detailed comparisons can be found in the respective original papers.

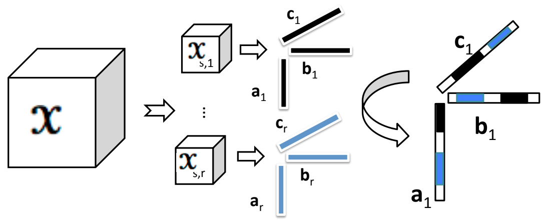
Contextual Synonym Detection.

In [9], we analyzed a knowledge base dataset, coming from the Read the Web project [1]; this dataset recorded (noun-phrase, noun-phrase, context) relationships (such as the example of Figure 1); the size of the tensor was $26M \times 26M \times 48M$, which made it prohibitive to analyze, for any existing tool. After obtaining the PARAFAC decomposition, we were able to perform contextual synonym detection, i.e. detect noun-phrases that may be used in similar contexts. Using cosine similarity, we took the low dimensional representation of each noun-phrase, as expressed by matrix $\mathbf{A}$, and we calculated the similarity of each noun-phrase to the rest. In this way, we were able to obtain noun-phrases that are contextually similar, albeit not synonyms in the traditional sense. Figure 5 contains the most notable ones.

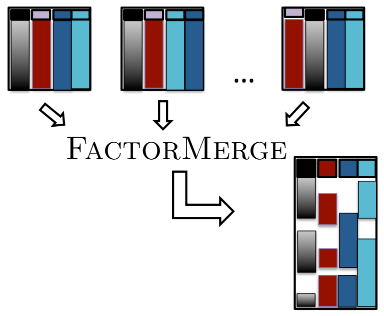
Facebook Wall posts

In [14], we analyze a Facebook wall posts dataset [1]. More specifically, the dataset we analyzed consists of triplets of the form (Wall owner, Poster, day), where the Poster created a post on the Wall owner's Wall on the specified timestamp, resulting in a $63891 \times 63890 \times 1847$ tensor. After running PARCUBE, we stumbled

[1]Download the Facebook dataset from `http://socialnetworks.mpi-sws.org/data-wosn2009.html`

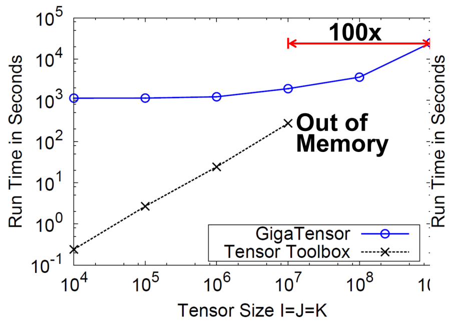


(a)

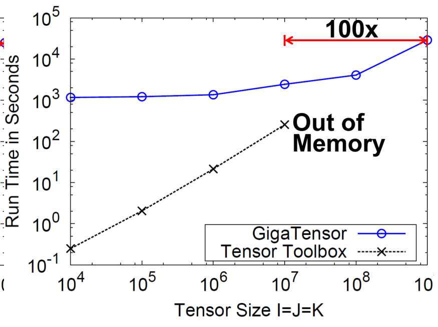


(b)

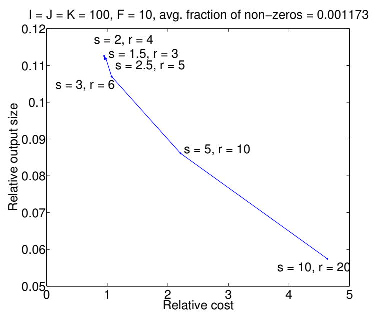
Figure 3: Subfigure (a): From the original tensor, draw r different samples, by sampling indices from each mode. It is crucial for this step, that a small subset of the drawn indices is common across different samples. Without delving into the details, this plays a major role in the third step of the algorithm, which merges partial results. For each sample tensor, compute its PARAFAC decomposition and merge the partial results. Here, for simplicity, we show a simple, rank one, case. This figure comes from our recent paper [14]. Subfigure (b): Here, we describe the merging procedure, where the rank is larger than one. From each sample tensor, we obtain a set of vectors for each mode. Let each one of the small matrices on the top represent the "sample" factor matrices. Our goal is to merge all partial components into a full-sized component, as shown at the bottom. Each color corresponds to a distinct latent component, and the upper part is marked as common across samples; notice that there are component permutations across matrices which need to be resolved in order to merge the components correctly. In [14] we provide the FACTORMERGE algorithm, as well as theoretical analysis of correctness.



(a)



(b)



(c)

Figure 4: Subfigures (a), (b): The scalability of GigaTensor compared to the Tensor Toolbox [6], for synthetic tensors of size $I \times I \times$, for two different scenarios. (a) Increasing mode dimensions, and number of nonzeros fixed and set to 10^4. (b) For a tensor of size $I \times I \times I$, increasing both the mode dimensions, and the number of nonzeros, which is set to $I/50$. In both cases, GigaTensor solves at least $100\times$ larger problem than the Tensor Toolbox which runs out of memory, for tensors of sizes beyond 10^7. We should note that in cases where the tensor fits in main memory, Tensor Toolbox is faster than GigatTensor, since it does not need to load anything from the disk (like MapReduce does). However, GigaTensor's strength is prominent when the tensor does not fit in memory, where the state of the art is unable to operate. These two subfigures are taken from our recent paper [9]. Subfigure (c): PARCUBE outputs sparse factors: Relative Output size (PARCUBE/ ALS-PARAFAC) vs Relative cost (PARCUBE PARAFAC objective function / ALS PARAFAC objective function). We see that the results of PARCUBE are more than 90% sparser than the ones from Tensor Toolbox [6], while maintaining the same approximation error. Parameters s and r are the sampling factor and the number of sampling repetitions for PARCUBE. This figure is taken from our paper [14].

(Given) Noun Phrase	(Discovered) Potential Synonyms
pollutants	dioxin, sulfur dioxide, greenhouse gases, particulates, nitrogen oxide, air pollutants, cholesterol
vodafone	verizon, comcast
Christian history	European history, American history, Islamic history, history
disbelief	dismay, disgust, astonishment

Figure 5: By decomposing a tensor of text corpus statistics, which consists of noun-phrase, context, noun-phrase triplets, we are able to identify near-synonyms of given noun-phrases. We propose to scale up this process, to form the basis for automatic discovery of new semantic categories and relations in the "Read the Web" project. This table comes from our recent paper [9].

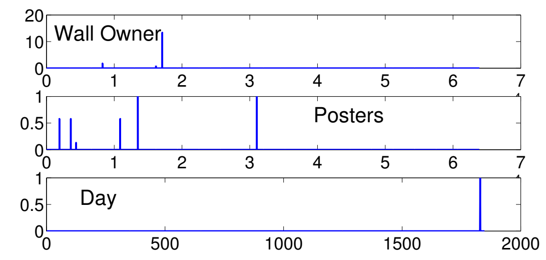


Figure 6: Facebook "anomaly": One Wall, many posters and only one day. This possibly indicates the birthday of the Wall owner. This figure is taken from our recent paper [14].

upon a series of surprising findings, an example of which we demonstrate in Figure 6: this Figure shows what appears to be the Wall owner's birthday, since many posters posted on a single day on this person's Wall; this event constitutes an "anomaly", since it deviates from normal behaviour, both intuitively, and by inspecting the majority of the decomposition components, which model the "normal" behaviour. If it hadn't been for PARCUBE's sparsity, we wouldn't be able to spot this type of anomaly without post-processing the results.

4 Insights and Conclusions

In this paper, we provided an overview of two different, successful means of scaling up tensor decompositions to big data:

GigaTensor: Scalability is achieved through simplifications of the most costly operations involved in the PARAFAC decomposition. In particular, we show how a prohibitively expensive operation can be alleviated, without sacrificing accuracy. Furthermore, we introduce a series of optimizations, which, in combination with the Map/Reduce environment, lead to a highly scalable PARAFAC decomposition algorithm.

PARCUBE: Scalability is achieved through *sketching* of the tensor, using biased sampling, parallelization of the decomposition on a number of sketches, and careful merging of the intermediate results, which, provably, produces correct PARAFAC components. Sketching might be a familiar concept in databases and data stream processing, however, in the context of large scale tensor decompositions, it has been fairly under-utilized, thus offering ample room for improvement.

The aforementioned approaches constitute, by no means, an exhaustive list of approaches one may envision in order to scale tensor decompositions up, however, we hope that they will be able to spark new research challenges and ideas, towards faster and more scalable tensor decompositions.

References

[1] Read the web. `http://rtw.ml.cmu.edu/rtw/`.

[2] E. Acar, C. Aykut-Bingol, H. Bingol, R. Bro, and B. Yener. Multiway analysis of epilepsy tensors. *Bioinformatics*, 23(13):i10–i18, 2007.

[3] C.A. Andersson and R. Bro. The n-way toolbox for matlab. *Chemometrics and Intelligent Laboratory Systems*, 52(1):1–4, 2000.

[4] Brett W. Bader and Tamara G. Kolda. Efficient MATLAB computations with sparse and factored tensors. *SIAM Journal on Scientific Computing*, 30(1):205–231, December 2007.

[5] B.W. Bader, R.A. Harshman, and T.G. Kolda. Temporal analysis of social networks using three-way dedicom. *Sandia National Laboratories TR SAND2006-2161*, 2006.

[6] B.W. Bader and T.G. Kolda. Matlab tensor toolbox version 2.2. *Albuquerque, NM, USA: Sandia National Laboratories*, 2007.

[7] R. Bro. Parafac. tutorial and applications. *Chemometrics and intelligent laboratory systems*, 38(2):149–171, 1997.

[8] R.A. Harshman. Foundations of the parafac procedure: Models and conditions for an" explanatory" multimodal factor analysis. 1970.

[9] U. Kang, E. Papalexakis, A. Harpale, and C. Faloutsos. Gigatensor: scaling tensor analysis up by 100 times-algorithms and discoveries. In *SIGKDD*, pages 316–324. ACM, 2012.

[10] H.A.L. Kiers. Towards a standardized notation and terminology in multiway analysis. *Journal of Chemometrics*, 14(3):105–122, 2000.

[11] T.G. Kolda and B.W. Bader. The tophits model for higher-order web link analysis. In *Workshop on Link Analysis, Counterterrorism and Security*, volume 7, pages 26–29, 2006.

[12] T.G. Kolda and B.W. Bader. Tensor decompositions and applications. *SIAM review*, 51(3), 2009.

[13] K. Maruhashi, F. Guo, and C. Faloutsos. Multiaspectforensics: Pattern mining on large-scale heterogeneous networks with tensor analysis. In *Proceedings of the Third International Conference on Advances in Social Network Analysis and Mining*, 2011.

[14] E. Papalexakis, C. Faloutsos, and N. Sidiropoulos. Parcube: Sparse parallelizable tensor decompositions. *Machine Learning and Knowledge Discovery in Databases*, pages 521–536, 2012.

[15] A.H. Phan and A. Cichocki. Block decomposition for very large-scale nonnegative tensor factorization. In *Computational Advances in Multi-Sensor Adaptive Processing (CAMSAP), 2009 3rd IEEE International Workshop on*, pages 316–319. IEEE, 2009.

[16] J. Sun, S. Papadimitriou, C.Y. Lin, N. Cao, S. Liu, and W. Qian. Multivis: Content-based social network exploration through multi-way visual analysis. In *Proc. SDM*, volume 9, pages 1063–1074, 2009.

[17] J.T. Sun, H.J. Zeng, H. Liu, Y. Lu, and Z. Chen. Cubesvd: a novel approach to personalized web search. In *Proceedings of the 14th international conference on World Wide Web*, pages 382–390. ACM, 2005.

[18] C.E. Tsourakakis. Mach: Fast randomized tensor decompositions. *Arxiv preprint arXiv:0909.4969*, 2009.

[19] Q. Zhang, M. Berry, B. Lamb, and T. Samuel. A parallel nonnegative tensor factorization algorithm for mining global climate data. *Computational Science–ICCS 2009*, pages 405–415, 2009.

Summarization via Pattern Utility and Ranking: A Novel Framework for Social Media Data Analytics

Xintian Yang
Google Inc.
xyang@google.com

Yiye Ruan, Srinivasan Parthasarathy
Dept. of Computer Science and Engineering
The Ohio State University
{ruan,srini}@cse.ohio-state.edu

Amol Ghoting
IBM T. J. Watson
Research Center
aghoting@us.ibm.com

Abstract

The firehose of data generated by users on social networking and microblogging sites such as Facebook and Twitter is enormous. The data can be classified into two categories: the textual content written by the users and the topological structure of the connections among users. Real-time analytics on such data is challenging with most current efforts largely focusing on the efficient querying and retrieval of data produced recently. In this article, we present a dynamic pattern driven approach to summarize social network content and topology. The resulting family of algorithms relies on the common principles of summarization via pattern utilities and ranking (SPUR). SPUR and its dynamic variant (D-SPUR) relies on an in-memory summary while retaining sufficient information to facilitate a range of user-specific and topic-specific temporal analytics. We then follow up by describing variants that take the implicit graph of connections into account to realize the Graph-based SPUR variant (G-SPUR). Finally we describe scalable algorithms for implementing these ideas on a commercial GPU-based systems. We examine the effectiveness of the summarization approaches along the axes of storage cost, query accuracy, and efficiency using real data from Twitter.

1 Introduction

Social networking and microblogging sites are ubiquitous nowadays, and an increasing number of organizations and agencies are turning to extract and analyze useful nuggets of information from such services to aid in various functions. However, a fundamental challenge for effectively analyzing social network data is its sheer scale. Twitter, for instance, has over 200 million users and several hundred million tweets per day. Supporting interactive querying and analytics requires novel approaches for summarizing and storing such data.

The data generated by social networking services can be classified into two categories: the textual content written by the users (e.g. tweets in Twitter) and the link structure of user connections (e.g. follower – followee relationship in Twitter). The textual content carries the information that people want to share with their friends. It is large-scale and streaming in nature. The link structure captures how the textual content will spread through the social network of users. While user connections are relatively stable compared with the high speed content stream, it is also large-scale because of the enormous size of user base.

We recently proposed the SPUR (Summarization via Pattern Utility and Ranking) family of algorithms to build a queryable summary of social network content stream [10]. Here we briefly overview some of the main features of SPUR and its variant Dynamic-SPUR (D-SPUR) and show how the summarization created by these algorithms can fit in a limited memory budget, and can help answer complex queries. In this work we describe extensions to SPUR wherein effective compression and efficient querying on the link structure of social network can also be supported called Graph-based SPUR (or G-SPUR). Furthermore, with the advent of general-purpose computing using Graphical Processing Units (GPUs) , we discuss strategies for leveraging such technology in the context of the SPUR family of algorithms.

We begin by briefly describing SPUR and D-SPUR as preliminaries in Section 2. Then we present G-SPUR (including GPU speed-up) in Section 3. In Section 4, we discuss how D-SPUR is adapted to work with G-SPUR for supporting content and time aware network queries Two particular query tasks, PageRank and clustering, will be described. Finally, we present experiment results in Section 5 and conclude.

2 SPUR: Summarization via Pattern Utility and Ranking

In our previous work [10], we described a method of summarizing the user generated content from a social network. The network content is in the form of a high speed message stream fluxing into the data processing system. We propose a novel stream processing framework to summarize the input stream with efficient, incremental summary construction and budgeted memory footprint. Given the input message stream with proper word stemming and stop-word removal performed, we divide it into approximately equal-sized batches, e.g. one hour per batch (the first arrow in Figure 1).

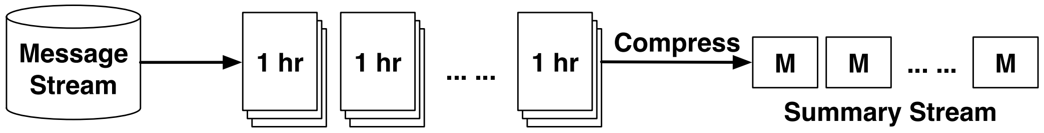
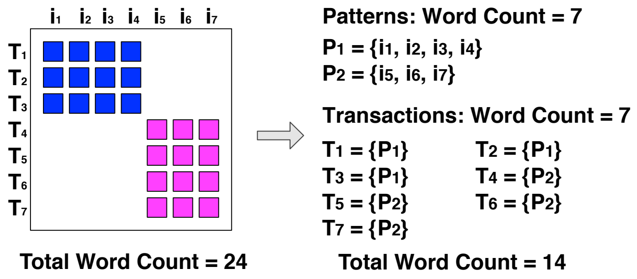


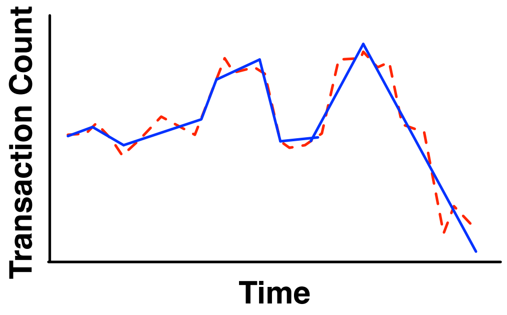
Figure 1: Division and compression of message stream

To compress each batch of messages into a summary object which can fit in a constant memory budget M (the second arrow in Figure 1), we replace individual words with frequently used phrases that can cover the original content of incoming messages (Figure 2(a)). Our approach represents each message as a transaction of words and a batch as a set of transactions. Therefore the challenge of finding frequent phrases becomes a frequent itemset mining problem. The utility of each pattern is represented by the reduction of storage cost if it would have been used. We also take into account the impact of using one pattern on other patterns, and perform dynamic ranking adjustment to reflect such changes. Patterns are selected in a greedy fashion, based on their utility values, until the size after compression satisfies the memory budget.

To guarantee the summary size grows logarithmically with time instead of linearly, we enhance and modify the pyramidal time window suggested by Aggarwal *et al.* [1] for clustering in data streams. The key operations of the system, D(ynamic)-SPUR, are merging two time windows, and managing time information. When merging time windows, patterns from both time windows are ranked together by their utility values, and ones with lower values are dropped until the merged summary can fit in the memory budget. The purpose of maintaining time information for the transactions in a summary is to be able to effectively answer a query about an arbitrary time interval. To this end, we store distinct transactions in the summary and associate a count with each transaction to indicate how many times a transaction appeared in a batch. When merging two time windows, if two transactions contain the same set of patterns, they must be from two different batches, because within a batch, we only keep distinct transactions. Instead of summing the count of these two transactions, we could concatenate their counts in time order and form a time series with two points. As D-SPUR combines more summaries, we concatenate more points to each transaction. A time series that spans batches (i.e. the red dashed line in Figure 2(b)) is therefore formed for each transaction, enabling



(a) A batch of messages compressed to a summary



(b) An example of the time series of a transaction

Figure 2: Illustration of message compression and time information management

reconstruction of the exact count in any time interval.

3 Compression of Network Topology with G-SPUR

While SPUR and D-SPUR algorithms are designed to compress social content streams, we are also interested in compressing the link structure of social networks themselves. We assume that the network topology is relative stable, and can therefore take snapshots of it at different times. The complete topological information of a social network snapshot can be modeled as a directed graph $G = (V, E)$, where each node $v \in V$ represents a user and a directed edge $(v, u) \in E$ indicates user u is a follower of v in the social network (i.e. the direction reflects the information flow). The storage space of such a graph is proportional to the number of edges, and can easily become overwhelming for large networks. To serve real-time queries related to user link structures, it is desirable to have an in-memory compact summary of them.

Our solution to this problem is to represent the adjacency list of a user as a transaction of items, where each item is a follower of this user. Then the entire social graph G can be seen as a batch of user-follower transactions. We would like to apply our SPUR algorithm to a batch of such transactions and compress its storage space. However, the SPUR algorithm is not directly applicable to the graph summarization problem for two reasons. First, SPUR produces a lossy compression by dropping infrequent items (that are, edges in the graph data), which can possibly disconnect a graph and introduce errors to various mining algorithms. Second, the frequent itemset mining stage of SPUR would not be scalable to graphs with hundreds of millions of nodes and billions of edges, if the whole graph would to be processed at once.

To address those issues, we propose the G-SPUR algorithm with two modifications of SPUR to enable lossless and fast summarization of large-scale graph data. A graph G is represented by a database of transactions, where each transaction is the adjacency list of a vertex and items in the transaction are the neighbors connected to the vertex. G-SPUR drop nodes whose numbers of adjacent edges are below a support threshold σ, and use a separate graph G_{infreq} to preserve those infrequent edges. The frequent edges will be stored in another graph G_{freq} for further processing. If G, G_{infreq} and G_{freq} are represented as adjacency matrices, we can see the above process decomposes G as the sum of G_{infreq} and G_{freq} (Equation 2). Because

G_{infreq} only contains the edges connected to vertex with in-degree below the support threshold, we directly store it as a sparse matrix.

$$\begin{aligned} G &= G_{infreq} + G_{freq} \quad (2) \\ &= G_{infreq} + T \times P \quad (3) \end{aligned}$$

After the above separation of infrequent and frequent edges, we run SPUR on G_{freq} without loss of information because all edges in G_{freq} are frequent. The SPUR algorithm will compress G_{freq} to a pattern set P and a transaction set T. Each pattern $p \in P$ contains a set of vertices from G_{freq}. Each transaction $t \in T$ contains a set of patterns from P, corresponding to the compressed representation for the original adjacency list. To reconstruct the adjacency list of a vertex from G_{freq}, we can take the union of patterns in a transaction. In a binary sparse matrices representation of the pattern set P and transaction set T, the SPUR algorithm essentially decomposes the frequent graph G_{freq} to the product of transaction set T and pattern set P (shown in Equation 3). By using G_{infreq}, T and P, we are able to store the original graph G with smaller storage size and reconstruct G without information loss.

Note that G_{freq} contains most of the edges in the original graph G. Therefore, the input to the SPUR algorithm will be as large as hundreds millions of transactions and billions of items. To maintain a scalable solution, we use minwise independent hashing [3] to partition the data into small samples. The SPUR algorithm will operate on each partition independently and produce a summary for each partition. We can generate the final solution by merging the pattern sets and transaction sets from the partitions. This method has been used by Buehrer *et al.* [4] to improve the scalability of large-scale web graph compression problems.

3.1 Speeding up PageRank with G-SPUR

The PageRank algorithm models the link structure of web graphs by the random walk behavior of a *random surfer* [2,6]. The web graph can be represented by a directed graph $G = (V, E)$, and its adjacency matrix A is defined as $A(u, v) = 1$ if edge $(u, v) \in E$; otherwise, $A(u, v) = 0$. Matrix W denotes the row normalized matrix of A. The PageRank vector p is computed iteratively using the following equation until convergence:

$$p^{(k+1)} = cW^T p^{(k)} + (1 - c)p^{(0)} \quad (4)$$

where c is a damping factor (set to 0.85 in our experiment), and $p^{(0)}$ is initialized as a n by 1 vector with all elements set to $1/n$. The major computational cost of Equation 4 is to compute the product of sparse matrix W^T and vector $p^{(k)}$. Previous work [5] shows that graph mining algorithms such as PageRank and SALSA can be directly computed from compressed graphs and the performance can be improved because the total number of computations can be reduced due to compression. From Equation 4, we can see that the PageRank algorithm can be implemented by iteratively calling the sparse matrix and vector multiplication (SPMV) kernel on a graph G. Since G-SPUR decomposes a graph G into G_{infreq}, T, and P, all of which can be stored as sparse matrices, we can directly implement the PageRank algorithm as iterative SPMV on $G_{infreq} + T \times P$.

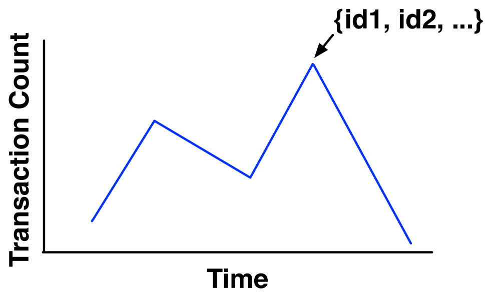
4 Content and Time Aware Network Topology Queries

In the previous sections, we introduce methods to create summarization of social network content stream and link structure. Besides those topics, another important analytical task is to investigate the network topology of the users who have written or read messages about a topic. Given a topic or keyword, example queries can be as simple as finding users who wrote or read this topic. More insights into the network topology can be obtained if we can find the social connections among these users. Furthermore, with these user connections, we can find which users are influential writers about this topic, whether there is any community structures among the users.

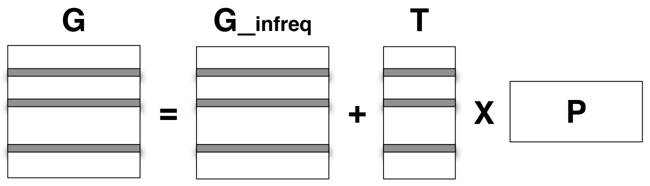
We can use the compact storage of the network content (SPUR, D-SPUR) and topology (G-SPUR) to answer the above queries in two major steps. In the first step, given a query keyword and time interval, we extract a subgraph of the entire network topology which contains all the users who either wrote or read a message about this keyword during the query time interval. In the second step, we run various graph mining algorithms on the extracted subgraphs to find influential users, and community structures in the network topology.

4.1 Content and Time Aware Subgraph Construction

First, we present our method of constructing a user subgraph given a query keyword and time interval. We achieve this by querying on the summaries built by the D-SPUR and G-SPUR algorithms.



(a) Modification to the D-SPUR algorithm



(b) Extract subgraph with G-SPUR

Figure 3: Construction of Content and Time Aware Subgraph

4.1.1 Incorporate Author Information into D-SPUR

Given a content keyword, we first find the list of users who have written messages about this keyword during the query time interval. We can slightly modify our D-SPUR algorithm to fulfill this query requirement as follow. First, we query the summary objects within the query time interval from the pyramidal time window; Second, in each summary object, we retrieve the patterns that contain the query keyword, and the transactions that include these patterns. These transactions represent all the messages containing the query keyword in the query time interval. Third, we need to find the writers of these messages. These users are who have written messages about the query keyword during the query time interval. Here we need to slightly change the D-SPUR algorithm to retrieve these users. In the original D-SPUR algorithm, IDs of users in each transaction is dropped. To preserve user information, we modify it by adding the list of user IDs at each point in the time series. Figure 3(a) illustrates this modification. Therefore, at a given time and a specific transaction, we can know which users wrote the transaction. With the above steps, we can extract the complete list of the writers of messages with a query keyword during a query time interval.

4.1.2 Extract Subgraph from Compressed Social Network

With the list of users U who have written messages about a query keyword during the query time interval, we then find their social connections by extracting a subgraph of them and their followers from the network topology. We will use the compressed network topology built by G-SPUR to find these social connections.

Suppose a graph G represents the original network topology which contains the social connections among all users. Equations 2 and 3 in Section 3 show that our G-SPUR algorithm decomposes the adjacency matrix of G into the adjacency matrix of a graph G_{infreq} and the product of a *transaction-pattern* matrix T and a *pattern-item* matrix P. In the original graph G, this subgraph of users U corresponds to a subset of the rows $G(U,:)$ in the adjacency matrix of G, where each row represents the followers of a user in U. For example, the three highlighted rows in the matrix G in Figure 3(b) contains the social connections of three users and their followers. Because of the G-SPUR decomposition $G = G_{infreq} + T \times P$, we can extract a subset of the rows $G(U,:)$ as $G(U,:) = G_{infreq}(U,:) + T(U,:) \times P$. This means we can get the rows of users in U from matrices G_{infreq} and T first, then use the rows from T to multiply with the pattern matrix P and then add to the rows from G_{infreq}. For instance, to get the three highlighted rows from G in Figure 3(b), we first get three rows from G_{infreq} and T respectively, then multiply the three rows from T with P, and then add the result to the three rows from G_{infreq}, the final results will be equivalent to the three highlighted rows in the original G. Therefore, we can efficiently extract a subgraph of the network topology from the D-SPUR and G-SPUR summaries by querying a content keyword and a time interval.

4.2 Mining Algorithms on Subgraphs of Network Topology

With a subgraph of a network topology conditioned on a query keyword in a query time interval, we can perform static analysis such as finding relevant users or communities of users on a topic during the query time interval. Here, we introduce two example mining queries to perform these tasks under our proposed framework.

4.2.1 Content and Time Aware PageRank

To rank users' importance regarding a topic keyword w during a time interval t, we can extract the subgraph $G(w,t)$ from G. This directed subgraph captures the social connections among all users who wrote messages about w during t. We then iteratively run PageRank (Equation 4) on the adjacency matrix of $G(w,t)$. The computation can be accelerated by the GPU based high performance computing platform introduced in Section 3.1.

4.2.2 Clustering

Given a subgraph $G(w,t)$ of the entire network topology G, we would like to find communities in such subgraphs to capture the topological relationships among the users who write or read the content keyword w in time interval t. This is meaningful because users have different interests in different topics and form different community structures. In this section, we introduce a new type of complex query to find such communities. Given a content keyword w and a time interval t, the query will return a clustering result C of the *active* users who write or read messages about the keyword w during time interval t in the social network. The general idea of answering such query is to first extract the subgraph $G(w,t)$, then apply graph clustering algorithms on the subgraph to find user communities. However, there are several challenges to execute the above process:

- **Scalability**: The scale of the subgraph $G(w,t)$ varies, and can become very large if a popular keyword, a long query time interval or a high-degree user is involved. A scalable clustering algorithm is thus needed to answer large number of queries efficiently.
- **Noise**: Previous work [9] has shown that there are two types of users in modern social network such as Twitter: a small fraction of influential users (e.g. celebrities, organizations), and a large number of auxiliary users (mostly followers of the influential ones). While community kernels [9] formed by influential users are more related to the network content, auxiliary users also form clusters and create noise in finding community kernels. The presence of auxiliary users also slow down the graph clustering algorithms.

- **Connectivity**: The influential users usually have a lot of followers but they rarely follow back. The connections among the influential users are weak as they rarely follow back, and it becomes hard to find dense communities if not including the auxiliary users. The auxiliary users who follow different influential users are essential to improve the connectivity of our subgraph $G(w,t)$.
- **Directionality**: The follower relationships among users are directed, so is the subgraph $G(w,t)$. Satuluri *et al* [8] show that it is non-trivial to cluster directed graphs by using graph clustering algorithms designed for undirected graphs.

To overcome the above challenges, we use a simple but effective preprocessing approach to symmetrize the directed graph $G(w,t)$ and adjust the edge weight to improve connectivity among influential users and reduce the noise from auxiliary users.

1. Symmetrization: We use the bibliometric symmetrization [8] method to transform our asymmetric subgraph $G(w,t)$ to a symmetric graph $SG(w,t)$. Given the adjacency matrix $A(w,t)$ of $G(w,t)$, bibliometric symmetrization essentially calculates the adjacency matrix of $SG(w,t)$ as $A(w,t)A(w,t)^T + A(w,t)^T A(w,t)$. This transformation not only removes the directionality of edges in $G(w,t)$ but also adds edges to vertices sharing similar set of in- or out- links. In $SG(w,t)$, edges can exist between two influential users who are not directly connected, but share common followers. This improves the connectivity among influential users.

2. Edge Re-weight: Edge weights in $SG(w,t)$ are only based on the topological information. We would like to incorporate the network content information to down-weigh the connections among auxiliary users who rarely contribute content to the network, and to reduce the noise. Given the keyword w and a node i, we calculate the weight $W(i)$ for node i by counting the number of times user i mentioned the keyword w in the time interval t. Suppose we have an edge from node i to node j, and the edge weight in the symmetrized graph $SG(w,t)$ is $SG_{w,t}[i][j]$. We adjust the weight of edge $< i,j >$ by multiplying $SG_{w,t}[i][j]$ with the sum of $W(i)$ and $W(j)$. In this way, we can construct a new weighted symmetric graph $WSG(w,t)$ with edge weight $WSG_{w,t}[i][j] = SG_{w,t}[i][j] \times (W(i) + W(j))$. The intuition of this edge re-weight method is that the node weight $W(i)$ captures how much interest user i has on the keyword w. Therefore, we want to boost up the weight of edges connected to nodes with strong interest on the query keyword w, especially for the edges that both the follower and the followee express strong interests.

After the above preprocessing on our subgraph $G(w,t)$, we have a weighted undirected graph $WSG(w,t)$ where the nodes of similar influential users are connected together and the links to noisy auxiliary nodes are down-weighted. We then run scalable graph clustering algorithms such as MLR-MCL [7] to efficiently find dense communities of influential users.

5 Experimental Results

In this section, we present results for an extensive set of experiments we conducted to evaluate the G-SPUR algorithm. We discuss the compression performance of several large web graphs and the follower-followee graph of twitter. To show the benefits of graph compression in speeding up graph mining kernels, we also implement the PageRank algorithm using the summarized graph with CPUs and GPUs.

We gathered 2100 hours of Twitter message streams from June to September in 2010 [1], and crawled the follower lists of all the users in the above message stream [2]. We construct the follower-followee graph of Twitter from this dataset. There are about 131 million vertices and 3.8 billion directed edges. We also use four other web graphs, shown in Table 9. All algorithms were implemented in C++.

[1]As provided by Twitter, it is a 15% random sample of all messages.

[2]Some of the users' follower information is not available because of their privacy settings

Graph	Nodes	Edges	Edges/Node	Density	Power-law?
it-2004	41,291,594	1,150,725,436	27.9	6.75×10^{-7}	Yes
sk-2005	50,636,154	1,949,412,601	38.5	7.60×10^{-7}	Yes
uk-union	133,633,040	5,507,679,822	41.2	3.08×10^{-7}	Yes
web-2001	118,142,155	1,019,903,190	8.6	7.31×10^{-8}	Yes

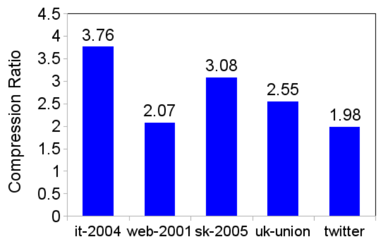
Table 9: Web Graph Datasets

5.1 Graph Compression with G-SPUR

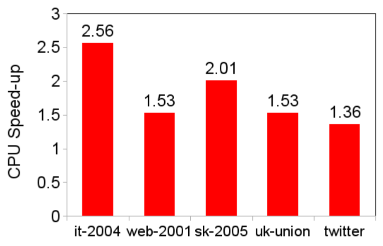
We use min-wise hashing to cluster the graph into partitions with partition size less than 1000 and we ran G-SPUR algorithm on each partition with absolute support value at 5. Figure 4(a) shows the compression ratio of the G-SPUR algorithm on these graphs. We can see that the G-SPUR algorithm can compress the storage size of large-scale web graphs to as low as 4 times smaller than the original graphs. In the Twitter follower-followee graph, our G-SPUR algorithm can still reduce the storage size by half.

5.2 Graph Mining Speed-up on CPU and GPU

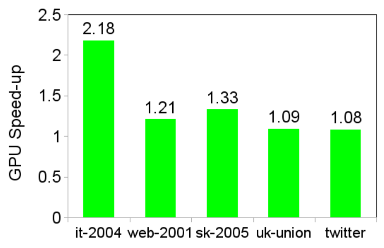
Next, we present some experimental results of speeding up the PageRank algorithm with G-SPUR. On CPU, we store the three sparse matrices G_{infreq}, T and P in CSR format and perform matrix and vector multiplication. Figure 4(b) plots the speed-up numbers of compressed graph over uncompressed on the four large web graphs and the twitter social graph. We can achieve from 1.4x to 2.6x speed-ups on these dataset. On GPU, we store the three sparse matrices in our optimized composite storage format. To compare with the performance of our multi-GPU SPMV kernel, we distribute matrices G_{infreq} and T to multiple GPUs, and each GPU will keep a copy of P because it is needed by all nodes of the GPU cluster. Figure 4(c) plots the speed-up numbers of the five datasets on GPUs. The PageRank algorithm can achieve from 1.1x to 2.2x speed-ups on the compressed graph over the original graphs. The major computational cost of the PageRank algorithm is the iterative SPMV kernel whose running time is proportional to the storage size of the graph. Our G-SPUR algorithm can effectively compress the size of the graph by decomposing it into three smaller matrices. Therefore, we can conclude that the speed-ups of the PageRank algorithm come from the compression of the graphs by the G-SPUR algorithm.



(a) Compression ratio of graphs



(b) CPU speed-up of PageRank



(c) GPU speed-up of PageRank

Figure 4: Compression ratio and speed-up of G-SPUR

Rank	Screen Name	Truncated Account Profile Description
1	Lonely Planet	Tweeting (& retweeting) the best in travel
2	GWPStudio	Photography, Socialmedia & sharing ... Love to travel & connect with people
3	American Airlines	The official channel of American Airlines
4	Sean Gardner	Digital Media Consultant
5	Gary Arndt	Traveler, blogger and photographer. A one man National Geographic. Been to over 100 countries ...
6	Tavelzoo	Travelzoo deal experts research and evaluate thousands of deals each week, selecting only the best ...
7	SmarterTravel	SmarterTravel.com is the largest online travel resource for unbiased travel news, deals, and timely ...
8	WhereIveBeen	Travel industry's leading social networking travel platform
9	TravelDeals	Use Twitter to save on travel in popular locations. Get a customized feed of travel deals near you
10	USATodayTravel	USA TODAY Travel offers the latest travel news, deals and consumer features about flights, hotels, ...
11	Andreas Susana	A guy from Austria, who writes about his trips and his website concerning books, castles, ...
12	Melvin	Love to travel, to discover the world, to travel free & untroubled & still be informed like an insider! ...
13	JD Andrews	World Traveler, Dad, 3xEmmy winner, Video, Adventure, Photographer, love dogs, Sharing & Caring ...
14	British Airways	Official British Airways global account
15	Get a Travel Deal	I find the best travel deals so you don't have to. Life's Short Travel Often!
16	Eagles Nest Wine	San Diego's Medal winning-ist Boutique Winery! Share an Authentic Wine Lifestyle with us! ...
17	Chicago Events	Real-time local buzz for live music, parties, shows and more local events happening right now in Chicago!
18	travelblggr	TV Host. Writer. Videographer. Travelista.
19	TravelGreen	Tips for sustainable travel and green living. Exploring the world, trying new foods & being green.
20	Tourism Malaysia	The official Tourism Malaysia Twitter account.

Table 10: Top 20 ranked users about the keyword "travel"

5.3 PageRank on Subgraph

Next, we show experimental results of content aware PageRank queries on the Twitter social network data we crawled from June 2010 to September 2010. We extracted the subgraph of users who mentioned the Twitter hashtag "#travel". We run the PageRank algorithm on this subgraph and rank accounts by their PageRank value from high to low. Table 10 lists the top 20 accounts, which can be classified into the following categories:

- Free information sources that people follow to find and share travel information, such as #1 Lonely Planet, #8 WhereIveBeen, #10 USATodayTravel and #19 TravelGreen.
- Travel deal websites, including #6 Travelzoo, #7 SmarterTravel, #9 TravelDeals and #15 Get a Travel Deal. These results are from a subgraph queried during the time of summer 2010. We know that people often have their vacation trips in summer and they want to reduce their travel expenses. Therefore, it is expected that those travel deal websites are active and popular on Twitter during the summer.
- Airline companies such as #3 American Airlines and #14 British Airways, because information of air transportation is a huge factor for travelers to plan their itinerary.
- Interesting travel destinations including #16 Eagles Nest Winery, #17 Chicago Events and #20 Tourism Malaysia to promote their travel packages. Since our dataset is only a random sample from all tweets, we did not find any tweets written by accounts representing famous places of interest in our dataset.
- Famous individual bloggers to share their experiences. This category includes #2 GWPStudio, #4 Sean Gardner, #5 Gary Arndt, #11 Andreas Susana, #12 Melvin and #13 JD Andrews. Some commonalities among these accounts are that they have large number of followers, they almost always follow back to their followers, they also write a lot of tweets and post photos to share their own traveling experiences.

From the above example, we can see that by running the PageRank algorithm on the subgraphs of a social network, we can find popular and influential account representing organizations, companies or individuals related to a content keyword in a time interval.

Cluster 1		Cluster 2			
PrizeDrawsUK	PiggyCodeUK	coupons2grab	CouponsInfo	SavvyPCDeals	internetshopper
Deals4UK	CodesUK	slickwallet	CouponNet	Spaffin_ebay	CouponCodeFeed
TopUKDeals		CouponSpy	Deals_Vista	DirectCoupons	redtagdeals

Table 11: Clustering results for the keyword “coupons”

5.4 Clustering on Subgraph

Here, we present experimental results on graph clustering queries. Table 11 shows the influential users from two clusters we obtained when querying the subgraph for the keyword “coupons” in the time interval from July 1st, 2010 to July 31st 2010. We can see from the screen names of these users that they are all related to coupons and deals in online shopping. Furthermore, we can see that the user names listed in cluster 1 are the Twitter accounts for online shopping websites in UK whereas the user names in cluster 2 are mostly in US with some global online shopping websites. Both cluster 1 and cluster 2 can be considered as community kernels because the user accounts have a lot of followers who are their customers. Also this cluster arrangement is reasonable because the accounts in different clusters have different follower populations. The followers of cluster 1 are mostly customers from UK whereas the followers of cluster 2 are mostly in US. Such clustering analysis is useful for online marketing with a targeted customer population.

6 Conclusions

We proposed G-SPUR, a novel algorithm to compress social network topology with low compression ratio, high quality and fast running time. The compressed link structures require less storage space, and can be directly used to speed up a series of graph mining kernels. It can also be used together with compressed social content stream to answer content and time aware network queries.

References

[1] C. C. Aggarwal, J. Han, J. Wang, and P. S. Yu. A framework for clustering evolving data streams. *VLDB 2003*.

[2] S. Brin and L. Page. The anatomy of a large-scale hypertextual web search engine. *Comput. Netw. ISDN Syst.*, 30(1-7):107–117, 1998.

[3] A. Z. Broder, M. Charikar, A. M. Frieze, and M. Mitzenmacher. Min-wise independent permutations. *Journal of Computer and System Sciences*, 60(3):630–659, 2000.

[4] G. Buehrer and K. Chellapilla. A scalable pattern mining approach to web graph compression with communities. *WSDM 2008*.

[5] C. Karande, K. Chellapilla, and R. Andersen. Speeding up algorithms on compressed web graphs. *WSDM 2009*.

[6] L. Page, S. Brin, R. Motwani, and T. Winograd. The pagerank citation ranking: Bringing order to the web. Technical report, 1999.

[7] V. Satuluri and S. Parthasarathy. Scalable graph clustering using stochastic flows: applications to community discovery. *KDD 2009*.

[8] V. Satuluri and S. Parthasarathy. Symmetrizations for clustering directed graphs. *EDBT 2011*.

[9] L. Wang, T. Lou, J. Tang, and J. E. Hopcroft. Detecting community kernels in large social networks. *ICDM 2011*.

[10] X. Yang, A. Ghoting, Y. Ruan, and S. Parthasarathy. A framework for summarizing and analyzing twitter feeds. In *SIGKDD 2012*. ACM.

Some Research Opportunities on Twitter Advertising

Milad Eftekhar
Department of Computer Science
University of Toronto
Toronto, ON, Canada
milad@cs.toronto.edu

Nick Koudas
Department of Computer Science
University of Toronto
Toronto, ON, Canada
koudas@cs.toronto.edu

Abstract

Social media have enjoyed a rapidly increasing adoption among users in recent years. Millions of users execute billions of actions (in form of tweets, messages, replies, likes, etc.) every day. The massive amount of social interactions online has contributed to the proliferation of social advertising.

Alongside the interest in online and social advertising, Twitter has introduced several advertising opportunities to aid advertisers to promote their products/services to the "right" audiences. This includes providing advertisers with (1) different advertising options of "Promoted Tweets", "Promoted Accounts", and "Promoted Trends", and (2) different user targeting options based on keywords, interests, location, etc. In this paper, first we briefly discuss the Twitter advertising platform, and then we introduce some research problems in this domain.

1 Introduction

Social media is used daily by millions of people (including but not limited to, journalists, celebrities, business owners, charities, etc.). Services such as Twitter, Facebook, LinkedIn, and Pinterest allow millions of users worldwide to interact with each other, share and consume content.

Micro-blogging platforms such as Twitter have experienced significant growth in user acquisition and participation in recent years. Twitter enjoys worldwide adoption with 500M registered users who generate over 400M tweets and 1.6B search queries every day. Social connections are established by "following" users. When a user u follows a user v, u sees all tweets posted by v in its timeline. The timeline of a user (say u) is the set of tweets generated by any user that u follows.

Given the wide user success Twitter enjoys, the focus on monetization for the platform has been on advertising and marketing. Users utilize Twitter as a marketing tool to broadcast messages to their followers. Companies, organizations, celebrities, etc. take advantage of this opportunity to target their followers for different purposes (e.g., brand/product awareness, sales leads, or general information dissemination, etc.) by broadcasting messages that will appear in followers' timelines.

Several companies including Google enjoy wide success based on a keyword based advertising model. Users utilize the search engine for search and business bid on the search keywords, taking the opportunity to showcase ads in suitable places of the screen along with the search results. In a social setting however, as is the case of Twitter, keyword search is only one functionality aspect. Twitter users follow other users

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

consuming content. The opportunity to mix content consumption and advertising is unique in a social setting. Typically the content consumed by an individual is highly specific; understanding the type of content each user consumes offer great opportunities for highly tailored advertising.

Recently Twitter introduced several advertising models ranging from the typical keyword search model (that is very common in many online advertising platforms such as Google AdWords) to advertising models based on interest targeting. In Section 2, we take a quick look into the different advertising models introduced by Twitter. Considering the interest-based targeting models, there exists a clear need to identify the interests and expertise of users in a social platform. Section 3 details how we can locate different topics of expertise and interests for each user utilizing the Peckalytics system [1], followed by Section 4 introducing research problems in this domain.

2 The Twitter Advertising Platform

Twitter recently introduced advertising models aiming to fulfill diverse marketing and advertising functions on the platform. These models can be categorized into two main groups: (1) Models that facilitate advertisers to better promote their products/services, (2) Models that aid advertisers target specific users more effectively.

Advertisers typically promote three types of products: tweets, accounts, and trends. With the “promoted tweets" option, a tweet is provided by the advertiser and it is inserted in the timelines or search results of all users who are targeted. The “promoted accounts" option enables advertisers to acquire more followers and build a larger community of advocates. The promoted account (provided by advertisers) is shown in the search results and in the “Who to follow" section of targeted users’ profiles. “Who to follow" is a section inserted next to any user u’s timeline suggesting what users u is likely to be interested in following. The results in the “Who to follow" section are generated by Twitter’s recommendation engine [4]. Finally, advertisers may sponsor trends utilizing the “promoted trends" Twitter advertising option. Trends are hot topics of the day and are placed next to the timeline of each user. Trends can be promoted globally or regionally. Consequently promoted trends offer wide exposure.

Twitter also introduced several advertising options to target users. Besides the keyword targeting model that is popular in search engine advertising (e.g., in Google AdWords), Twitter introduced a new advertising model that allows advertisers to target users by their interests, geography, and gender. Moreover, followers of a set of Twitter accounts can be targeted as well as Twitter users who are similar (have similar interests) to these followers (or to any specified set of users) [6].

When targeting users based on interests, an advertiser is able to provide a promoted tweet or a promoted account and target it to a set of users who have specific interests. Interests are explicitly chosen by the advertiser. In particular, an advertiser chooses a topic (or set of topics) t and provides a promoted tweet or a promoted account. Based on proprietary algorithms, the Twitter advertising platform identifies all the users who are *interested* in the topic t; the promoted tweet is inserted in the timeline or search results of these users (explicitly identifying it as a promoted tweet); or suggests these users to follow the promoted account. For example, one can conduct an advertising campaign on a wine festival providing a tweet and selecting topics such as “wine", “tourism", etc. The provided tweet will be inserted in the timelines and search results of users who are interested in “wine" and/or “tourism".

Users can also be targeted by geography. A promoted tweet or account is promoted to a group of people who live in a specific location (country or metropolitan area). These options aid advertisers to target regional audiences. Naturally regional targeting is preferable when the event or product is of regional interest only.

Advertisers also have the option (a) to provide Twitter accounts and target a promoted tweet to the followers of these accounts and (b) to target users who are similar to the followers of some accounts. Assume one aims to advertise a new independent movie. Naturally it makes sense to target those Twitter users that are interested in independent movies. Notice that those interested in independent movies may or may not tweet about such movies. Primarily they will be consuming content from Twitter users that are experts in independent movies and produce content primarily about such movies. Utilizing option (a), one can identify the "*experts*" on " independent movies" and target the followers of these experts. By following accounts who have expertise on a topic such as independent movies, these users show interest on the topic; hence, they are good targets for promoting a tweet on the new movie. Utilizing option (b), we look for users who are similar to (have similar interests) the followers of experts on independent movies. These include all users interested in independent movies.

Central to these approaches is the ability to identify "expert" users on particular topics on Twitter. In Section 3, we explain some approaches including that of Peckalytics that utilizes crowd-sourced information and social connections to identify the expertise and interest of users.

3 Identifying Expertise and Interests

It is expected that users generate content on topics they know well or care about. The expertise area of user v is the set of topics that v knows well, or is known for in the community. For example, a company shares tweets related to its products whereas a soccer player post tweets about the team, the upcoming events, some personal life information, etc. Technically, by sharing information that is related to one's expertise area, one aims to become more popular to users who are interested in that field. This leads to gain more followers, attention, and observability. On the other hand, one can understand what a user is interested in, by examining who they follow. When user u follows user v, u expresses interest in the content v generates.

Let $U = \{u_1, u_2, \cdots, u_n\}$ be the set of users and $T = \{t_1, t_2, \cdots, t_m\}$ be the set of topics. For a user u, let T_u be the set of topics associated with u. It is feasible to obtain the set of topics mostly associated with a Twitter account. These are usually the topics of expertise for the account holder. Several approaches could be applied to assign an account to a list of topics declaring its expertise. Machine learning technology is mature enough to classify the tweets one generates based on topics. Alternatively, the approach taken by the Peckalytics project [1] provides results taking a crowd sourcing approach.

Definition 1: A user $u \in U$ is an *expert* on topic $t \in T$, iff $t \in T_u$. This means that (for our specific way of extracting topics) other Twitter users recognize u as an expert on topic t. We call topic t, a topic of expertise for u.

Assume $C = \{c_1, c_2, \cdots, c_r\}$ is the set of contents generated by users in U. In the Twitter setting, a content can be a tweet, a shared video, a posted link, etc. Moreover, assume there exist a mapping $M : C \to 2^T$ that maps each content in C to a subset of topics T (topics that are associated with this content). For example, $M(c_1) = \{t_1, t_3, t_5\}$ translates to "content c_1 is associated with topics t_1, t_3, and t_5" where content c_1 can be the following tweet "The first lady Michelle Obama makes surprises by reading the best picture in Oscars 2013" and topics t_1, t_3, and t_5 can be Politics, Hollywood, and Art.

Definition 2: A user $u \in U$ is *interested* in topic $t \in T$ iff the probability that u follows (reads) any content c that is associated with topic t ($t \in M(c)$) is higher than a given threshold $\theta \in [0, 1]$.

The approach taken by the Peckalytics project is to utilize Twitter lists to extract expertise and interests for Twitter users. Twitter introduced the concept of *Twitter lists*. A Twitter list is a collection of Twitter accounts (users). Typically, users create lists annotated with a descriptive name and place their favorite accounts who are perceived by the creator of the list to be experts on a particular topic (typically the list name) into the list. For example, a user may create a list with the name "politics" that includes Twitter accounts @BarakObama, @AngelaMerkel, @HillaryClinton, @JohnKerry, and @DavidCameron. Lists facilitate content filtering by topic. In other words, by creating a list on "politics", a user can filter the tweets in the timeline to see just the tweets generated by accounts in that list. Creating lists on different topics by different users is a typical experience in Twitter. A user can create multiple lists and a Twitter account can belong to any number of lists.

Recent works have utilized Twitter lists to address some problems such as identifying users' topics of expertise [1, 2, 5], separating elite users (e.g., celebrities) from ordinary users [7], and sampling user-generated data in social networks utilizing an expert based method [3].

Peckalytics utilizes the Twitter lists to identify topics of expertise and interest for different users. It crawls the lists and utilizes Apache Lucene to store and index the lists and the users in the lists. This process is executed constantly as new lists are regularly added and existing lists are dynamically modified by the creators. Peckalytics associates with each Twitter account, a set of topics extracted from the names of the lists containing that account. The process of extraction includes tokenization of the name, common word (stop word) and spam filtering, entity extraction, and related word grouping via Wikipedia and WordNet. An index mapping each Twitter account to the set of topics associated with the account is constructed and managed by Lucene. Thus, for each Twitter account u, a set of topics that best describe the topics associated (by other Twitter users) with u is identified. We refer to this set of topics as the *expertise vector* of user u.

The Lucene index supports all kinds of typical search query syntax including regular match (match in any order, e.g., *social media*), phrase match (e.g., *"social media"*), boolean expressions (e.g., *social AND media*), negative match (e.g., *-social media*), etc. These matching types are also supported by Twitter advertising platform.

To identify the set of interested users for any topic t, Peckalytics utilizes the experts and the social connections between Twitter users. Peckalytics reports the set of followers of experts on topic t as the set of users who are interested in t. This is due to the assumption that a user u follows another user v provided that u has an interest on the contents v generates. Therefore for each user u, we can identify the set of topics, u has an interest (referred to as the *interest vector* of u) as the union of the expertise vectors of the users u follows.

Peckalytics provides the following main functionality:

1. Identifies the expert Twitter users for any topic t. This functionality is helpful when an advertiser aims to provide a set of Twitter accounts and promote a tweet to the set of followers of these accounts. The experts on t are the most relevant accounts to provide while initiating an advertising campaign on t.

2. It offers analytical functions on the set of experts for any topic t. These functions include the identification of other topics of expertise for a user, their conversations (e.g., the frequent keywords, keyword pairs, and hashtags in their tweets), and the most popular sites they share content from. These analytics aid to assist the selection of topics or keywords an advertiser could choose for campaigns.

4 Some Research Problems

The Twitter advertising platform offers new research problems that could be of interest to business and advertisers. This section introduces three research problems in this domain.

Alternative topics: When targeting a specific set of users, it is natural to ask how can we target them with the lowest cost possible. In particular, assume one aims to advertise on topic t. Clearly, different topics have different costs based on the popularity of the topic, number of advertisers that target the topic, etc. The question is whether we can target the same or approximately the same set of users by advertising on a cheaper topic t' instead of t. We call t' an *alternative* topic.

For example, suppose we want to conduct an advertising campaign on *wine*. Let's say, we understand that most of the users who are interested in *wine* are also interested in *tourism*. Moreover, suppose that the cost of advertising on *wine* is higher than *tourism*. Instead of advertising on *wine*, we can advertise on *tourism*, pay less, and still target a similar audience. In this example, *tourism* is an *alternative* topic for *wine*. Identifying the alternative topics, therefore, is the first problem of interest. Moreover being able to quantify precisely the similarity between the audiences of two alternative topics is of interest as well. It creates an interesting optimization trade-off between similarity of audiences and cost for alternative topics.

This problem can be studied in two scenarios: *(1)* The set of audiences for each topic is unknown, or *(2)* The audience sets are given or can be computed based on the existing information. In both, the problem requires solutions that (approximately) measure the similarity of different topics (based on given or calculated audience sets) for a specific input topic and identify topics with lowest cost.

Combining interests: A second problem is to identify the different interests of a set of users. A set of users interested in topic t (I_t), can be more useful for advertising purposes if we are able to understand other topics users in I_u are interested in. For example, while conducting an advertising campaign on topic *social media*, if we know that users who are interested in *social media* are also interested in *seo* (search engine optimization), we can design a campaign that combines *social media* and *seo* (for example promoting the benefits of social media for SEO campaigns). This could serve us better and attract more attention as the campaign combines different interests of the audience.

One issue in this approach is the cardinality of I_t; potentially the set interests can be large. Conducting campaigns that cover all the topics of interest is clearly impossible. For example assume that users interested in *movies* are also interested in *food, soccer, hockey*. We aim to organize these topics into high-level categories (e.g., by merging *hockey* and *soccer* into a bigger category *sports*) and partition the users in I_t based on these categories. Executing this organization, we can target users in I_{movies} (users who are interested in *movies*) who are interested in *food* with a campaign combining *movies* and *food*; and target users in I_{movies} who are interested in *hockey* and/or *soccer* with a campaign combining *movies* and *sports*.

The goal of the second problem is, therefore, to create a set of users in I_t as input, organize their other interests into high-level categories (e.g., sports), and partition the users in I_t based on these categories. Thus algorithms to partition interests and create high-level categories are required.

Expert refinement: Experts on a topic t may have expertise on other topics too. One problem of interest would be to categorize the experts based on their different topics of expertise. For example, starting with the experts on *cloud computing* we may be able to categorize them into a partition representing the experts on *cloud computing* and *virtualization* and another partition representing *cloud computing* and *data centers*. This organization aids us to have a clearer understanding of different experts. Why is this important? One goal in advertising campaigns is to engage experts into promoting products. Consequently the followers of these experts become aware and possibly adopt the product; subsequently this effect propagates in the network. This behavior is a result of word of mouth in social media. One crucial step to instigate a word of mouth activity is to identify the “right” initial experts to convince (the seeders). Identifying the right initial experts is a challenging task as experts may vary in the presence of different objectives. For example, the set of experts on the partition *cloud computing* and *virtualization* and those in the partition *cloud computing* and *data centers* may each be more suitable while conducting different campaigns. The objective will dictate which one is more suitable in each case.

Computationally, partitioning the experts based on the different topics of their expertise can be a hard task as the “right" number of partitions is not clear. Moreover, it is not clear how topics and experts should be grouped together. This points to the need for soft clustering approaches that optimize criteria specific to the campaign under consideration.

5 Conclusion

The growth in the penetration of social networks in everyday life, opened new opportunities in advertising. Deeper understanding of users based on their actions and disclosed information in social networks, offer opportunities to target users based on their specific interests. In this paper, motivated by the Twitter advertising platform we detailed techniques to identify expert users and users with specific interests on Twitter. Subsequently we presented an initial set of problems that could be of interest to solve in the social (micro-blog) advertising context. Social advertising is still in its infancy and research in this area could result in real impact.

References

[1] A. Cheng, N. Bansal, and N. Koudas. Peckalytics: Analyzing experts and interests on twitter. SIGMOD Demo Track, pages 973–976, 2013.

[2] S. Ghosh, N. Sharma, F. Benevenuto, N. Ganguly, and K. Gummadi. Cognos: crowdsourcing search for topic experts in microblogs. In *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, pages 575–590, 2012.

[3] S. Ghosh, M. B. Zafar, P. Bhattacharya, N. Sharma, N. Ganguly, and K. P. Gummadi. On sampling the wisdom of crowds: Random vs. expert sampling of the twitter stream.

[4] P. Gupta, A. Goel, J. Lin, A. Sharma, D. Wang, and R. Zadeh. WTF: the who to follow service at twitter. In *Proceedings of the 22nd international conference on World Wide Web*, WWW '13, pages 505–514, 2013.

[5] N. K. Sharma, S. Ghosh, F. Benevenuto, N. Ganguly, and K. Gummadi. Inferring who-is-who in the twitter social network. In *Proceedings of the 2012 ACM workshop on Workshop on online social networks*, pages 55–60, 2012.

[6] Twitter. Start Advertising | Twitter for Business. `https://business.twitter.com/start-advertising`.

[7] S. Wu, J. M. Hofman, W. A. Mason, and D. J. Watts. Who says what to whom on twitter. In *Proceedings of the 20th international conference on World wide web*, pages 705–714, 2011.

Supporting Efficient Social Media Search in Cyber-Physical Web

Lidan Shou   Sai Wu [†]
College of Computer Science and Technology, Zhejiang University, China
{should,wusai}@zju.edu.cn
[†] Corresponding Author

Abstract

The cyber-physical systems (CPS) are envisioned as a class of real-time systems integrating the computing, communication, and storage facilities with monitoring and control of the physical world. With the proliferation of social media contents in the Web, a novel type of CPS applications allow users to link interesting objects in the physical world to the corresponding social media contents in the virtual world. Such new CPS applications require novel techniques to find the semantic relationships between the two worlds, and provide efficient search tools in the integrated cyber-physical world. In this paper, we propose a general framework for supporting social media search in the cyber-physical Web, or CPSS. This framework addresses the key problems identified on different schematic levels of CPSS. We report our studies on these problems and present our solutions to four different categories of searches/recommendations in CPSS, namely the model-based search, the non-model-based search, the real-time media search, and the geo-social analytics.

1 Overview

The cyber-physical systems (CPS) are envisioned as a class of real-time systems integrating the computing, communication, and storage facilities with monitoring and control of the physical world. One interesting CPS application in the mobile Internet is to provide the social media search "on the spot" with regard to the physical world that a user sees, or literally WYRIWYS (What-You-Retrieve-Is-What-You-See). For instance, a user viewing the Statue of Liberty through her smartphone camera can search for either the latest tweets about the statue or its history from Wikipedia.

Social media search in CPS, or Cyber-Physical Social-media Search (CPSS), poses new technical challenges against the database and information retrieval community. First, various social media data, in the form of textual documents, tweets, images, videos etc., have to be captured and organized by a unified framework, which enables efficient access by CPSS applications. As an example, in [1] [6], new retrieval models are proposed to extract structured data, such as names and addresses from the social archives, to rebuild the social relationship and support efficient search. Second, some key techniques are required to support WYRIWYS, such as retrieving objects by visibility [10] [7], searching with spatial information [15] [12], and linking objects to their semantic tags etc. Third, different applications adopt different search philosophy. Thus, the ranking and recommendation algorithms should be redesigned as well [2] [8] [13].

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

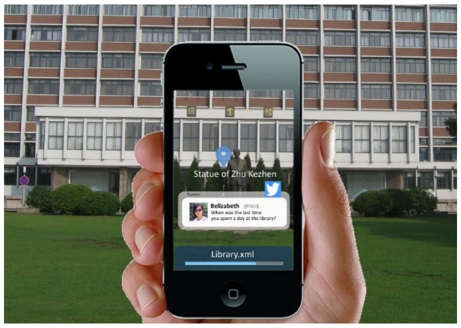


Figure 1: The S4 document browser showing an object and a tweet box

Some people consider CPSS as Search with Augmented Reality (AR) interface. However, this opinion is only partly right, as CPSS is much more than Augmented Reality Search. We believe that CPSS is uniquely identified by the following properties:

- CPSS is tightly coupled with physical attributes of objects.
- CPSS requires real-time search over multi-media data.
- CPSS is highly relevant to social networks.

In this paper, we introduce a unified framework called S4 (literally for Sensor-enhanced Social media Search System) for supporting CPSS. This framework eases the development of various applications using a generic hyper-media markup language which we call Cyber-Physical Markup Language (CPML). Users can navigate in the physical world, which is integrated with documents written in CPML, using an Augmented-Reality browser. S4 has an infrastructure supporting the basic navigation and interaction with CPML documents which are bound to physical locations or objects. Figure 1 illustrates the mobile browser screen of document "Library" containing a place-of-interest (POI) and a real-time tweet box.

As a part of the framework, we have implemented in S4 some of the key modules to support WYRIWYS, such as searching by visibility and real-time tweet searching/summarization [5] [9] [11] [3]. Applications in our framework can leverage these modules to enhance their own functionalities. We will report our design philosophy and techniques used in developing these modules. We will also give our vision and suggestions on some relevant research problems.

2 Overview of the S4 Framework

Cyber-Physical Social-media Search (CPSS) differs from conventional search in many aspects. In what follows, we discuss these differences at several schematic levels.

(1) At the *Query level*, traditional search accepts keywords or images as the query input, while CPSS is required to support more complex query types with novel search semantics. For instance, one distinguishing feature of CPSS applications is the need for Augmented Reality (AR) user interface as an attempt to support WYRIWYS.

(2) At the *Data Access/Processing level*, CPSS searches virtual "documents" which combine the contents of the static HTML documents, the data attributes collected from physical objects, and the social media associated with the corresponding objects. This requirement poses technical challenges in data access and processing.

(3) *Infrastructure level* Apart from the internet, CPSS also requires certain infrastructure support, such as indoor-space localization and sensor network communication.

In the remainder of this section, we propose a general framework called S4, literally for (Sensor-enhanced Social media Search System) as our solution to CPSS. We shall focus on the *first two levels*

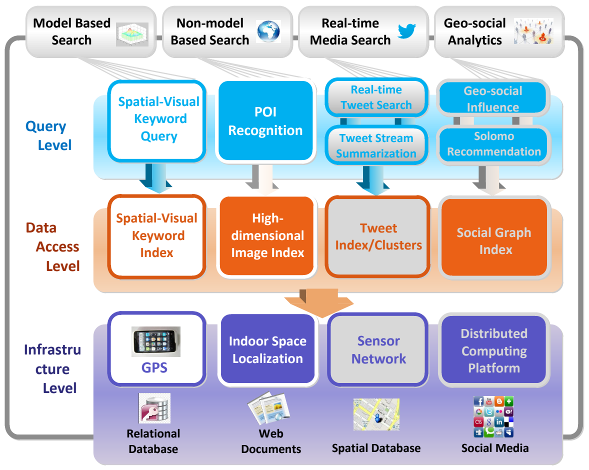


Figure 2: Schematic View of S4 Framework

and report our recent studies on these topics. The third level is necessary for the implementation of the S4 framework. However, detailed discussion of CPSS infrastructure is out of the scope of this paper. Apart from the mentioned work, we believe there are still countless open questions to be answered and problems to be solved.

Figure 2 shows a schematic view of the S4 search engine. To help understand the functionalities of proprietary components in S4, we shall introduce these components in different vertical (application) categories, depicted in four columns in the diagram.

Category 1 (Model-Based Search) One prominent requirement in CPSS is the user's ability to search for documents associated with the physical objects that are captured by the phone's camera. To support such query, namely*WYRIWYS*, we need a special index structure which can retrieve objects not only by textual relevancy but also by user's physical visibility to these objects.

Category 2 (Non-Model-Based Search) S4 provides multi-modal POI recognition functionality at the query level. This functionality relies on the support from both spatial indexing and object matching with computer vision techniques. The latter is built on top of a *High-Dimensional Image Index*.

Category 3 (Real-Time Media Search And Analysis) The social media generated from mobile devices can be linked to the physical objects, due to the availability of various device sensors. Compared to the conventional Web search, social media search poses new challenges as we need to merge the results from the physical world and the virtual world in real-time.

Category 4 (Geo-Social Analytics) Search and recommendation in CPSS should consider the users' social relationship and geo-locations. S4 includes a number of modules supporting geo-social analysis. The *Geo-Social Influence* module computes at scale the Geo-Social Influence of historical events or places recorded in large geo-social networks. Specifically, the module can tell which events (or places) are of global attraction, while the others are more suitable only for the local people.

3 Modules of S4

3.1 Model-Based Search

Model-based search assumes that the physical objects have their virtual geometric models stored in the database of the CPSS engine. S4 implements the *spatial-visual keyword* (SVK) query for processing model-based search [14]. Specifically, given a collection C of spatial keyword objects and a query q, a SVK query returns a list of k objects in C with the highest scores, where the score of each object is computed by combining its physical visibility and semantic relevancy with respect to q. The SVK query is different from simply imposing visible judgment on distance-based spatial keyword queries, since the visual conspicuousness of objects should be quantitatively measured, based on human visual perception. For this purpose, we used a novel visibility metric, which measures the visible parts of an object in a cumulative way.

The SVK query is processed using a *Complete Occlusion-Map Retrieval* (COR) method. This method employs a hybrid index of spatial keyword objects called IR-tree [4] and works in a two-step manner.

- In the first step, a dynamic structure called *Complete Occlusion-Map* (COM) is built. This structure partitions the surrounding space of the query point into a number of angular ranges and maintains the visibility information for each range.
- In the second step, COR computes the concrete visibility for objects, as well as the tightest visibility upper bounds for IR-tree nodes. Owing to the best-first search paradigm, the search space can be effectively reduced and the top-k relevant objects are returned in an incremental manner.

The SVK query can be used to provide reality-augmented Web search for mobile users. The work presented in [14] is a preliminary study which considers the simplest form of textual relevancy and the so-called 2.5 dimensional visibility. It can be extended to more complicated metrics such as semantic relevancy and 3D space visibility.

3.2 Non-Model-Based Search

Non-model-based search refers to the case when the geometric model of objects is not available and that the sensor errors (as for location, orientation, viewing angles etc.) are fully considered for those stored in the database. One essential problem for non-model-based search is to recognize a POI being captured from a mobile phone camera.

The POI recognition module of S4 employs a *two-phase approach* to recognize the place of interest from a large, impure set of images downloaded from online photo sharing services [10]. (1) During the spatial phase, we use a *probabilistic field-of-view(pFOV)* model which captures the uncertainty in camera sensor data. Based on this model, all POIs relevant to the FOV are given a *likelihood* of being captured by the camera. (2) During the visual phase, we put forward the *SC-similarity* relying on the Sparse Coding, a technique originated from the signal processing domain. The final ranking combines an *uncertain geometric relevance* with the *visual similarity*. The most distinguishing feature of our approach is its ability to perform well in contaminated, real-world online image database. Besides, our approach is highly scalable as its implementation does not require any complex data structure.

Given the current camera parameters, the pFOV culling algorithm can quickly determine the candidate POIs which may possibly appear in the image. As a result, the cost of subsequent evaluation of geo-relevance and SC-similarity can be reduced significantly.

To compute the visual similarity, each candidate image is represented as a bag-of-visual-words column vector. Let D be the matrix where each column is the vector for a candidate image. Then the problem can be described as: Given a query image $\mathbf{x}$, can we represent it as a linear combination of other candidate images(columns in D)? This is a typical problem of sparse coding, where we try to find an optimal weight vector aiming at reproducing the query image from the candiates. The output of the solution to this problem (the weight values) are defined as our SC-similarity.

Finally, for each candidate POI, we compute a voting score as a linear combination of the geo-relevance and SC-similarity of each candidate POI. The POI with the top ranking score is taken as the final result. Experiments on densely populated areas report recognition accuracy of up to 92%.

3.3 Real-Time Media Search

The need for real-time media search comes from the explosive growth of user generated contents on the Web. In S4, we provide the real-time search and summarization modules for tweets (short texts in microblogs).

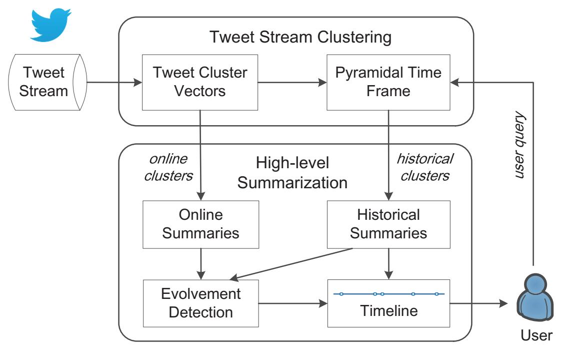


Figure 3: The Tweet Stream Summarization Module of S4

3.3.1 Real-Time Tweet Indexing and Search

In order to provide real-time search on fast arriving tweets, S4 employs the *TI*, an efficient indexing scheme which relies on inverted index [2]. For each inserted microblog, it is identified either as an important blog or a noise. Important blogs are indexed in real-time, while noisy blogs are indexed in a batch manner. To help rank the microblogs, some statistics are maintained in memory.

TI adopts the bag-of-words model, which splits each microblog into a set of keywords. The inverted index of the keywords are maintained in *TI*'s database. For a keyword, its index entry consists of a set of microblog IDs, which are sorted by their timestamps (the time when a tweet is inserted into the system). We maintain some social information in the index to support complex ranking functions.

1. In *TI*, the users of microblog are ranked based on their proprietary PageRank values in the user graph of microblog system. The PageRank value of the publisher is appended to the records of the corresponding microblog.

2. *TI* organizes the microblogs as a topic tree based on the reply/comment relationship. So we can rank a microblog based on the popularity of the whole topic. The tree ID is then maintained by the index as well.

3. Finally, we maintain the TF (Term Frequency) value for the keywords of each microblog as well. The basic TF/IDF ranking function can be applied.

4. The ranking function in *TI* combines the timestamps, the TF/IDF value, the PageRank and the topic popularity. Based on the above information, we can quickly classify a microblog as an important or noisy microblog and adopt different indexing scheme accordingly.

3.3.2 Real-Time Tweet Summarization

To support real-time summarization of fast arriving tweet streams, S4 employs a scheme called *continuous tweet summarization* [11]. The advantage of the scheme is that, given a topic-related tweet stream, one can (i) continuously monitor the stream and produce a continuous timeline which grows by time. (ii) A range timeline can also be provided to illustrate the big picture of topic evolution during some period. (iii) Users can ask for summaries of arbitrary time durations (like drill-down or roll-up summary). Such functionalities would not only facilitate easy navigation in topic-relevant tweets, but also support a range of data analysis tasks such as instant reports or historical survey.

The structure of the tweet stream timeline module is depicted in Figure 3. The module consists of two main components, namely a *Tweet Stream Clustering* component and a *High-level Summarization* component. In the *Tweet Stream Clustering* component, we design an efficient *tweet stream clustering algorithm*, an online algorithm allowing for effective clustering of tweets with only one pass over the data. This algorithm uses two data structures to keep important tweet information in clusters. One is a compressed structure called *Tweet Cluster Vector (TCV)*. TCVs are considered as potential sub-topic delegates and maintained dynamically in memory during stream processing. The other is a *Pyramidal Time Frame (PTF)*, which is used to store and organize cluster snapshots at different moments. The PTF allows historical tweet data to be retrieved by any arbitrary time durations.

In *High-level Summarization*, we generate two kinds of summaries: online summaries and historical ones. The summarization module also contains a topic evolving detection algorithm, which consumes online/historical summaries to produce continuous/range timelines.

3.4 Geo-Social Analytics

The S4 engine also provides functionalities for geo-social analytics. We present the techniques used in two modules for geo-social analytics in S4. One is for analyzing the geo-social influence of users and events. The other is for recommending items in SoLoMo application.

3.4.1 Geo-Social Influence

This module conducts an in-depth analysis on the geographical and social correlations among SoLoMo users for different events. Assuming that each user in the geo-social network is associated with a number of events, the basic queries to be answered are:

- *User Influence* In a geo-social network, how can we measure the geo-social influence of one user to the others?
- *Influential Events Discovery* Given a set of events in the network, which ones are the influential events?

In our solution, we provide a unified user influence metric which combines social proximity and geographical mobility features of geo-social network users [15]. On the social side, we use a modified version of the hitting time measure, named *penalized hitting time* (PHT), to quantify the social proximity between LBSN users. Hitting time is a random-walk-based graph proximity measure which has been shown to be effective for link prediction, query suggestion, graph clustering, and so on. However, it is sensitive to long paths and tends to benefit popular entities. Our PHT measure intrinsically avoids this drawback. On the geographical side, we model the geographical influence with regard to distance by the *power law distribution*.

For efficient computation of PHT, our solution uses two approximate algorithms, namely the *global iteration* (GI) and the *dynamic neighborhood expansion* (DNE) algorithms. Both algorithms work efficiently when computing PHT, and meanwhile ensure tight theoretical error bounds. In particular, the DNE algorithm can compute PHT in a constant time regardless of the network size.

Relying on the user influence metric, the influence of an event can be measured by aggregating the influences of different users, with two specific aggregate functions, namely MAX and AVERAGE. We employ the sampling technique to avoid computing geo-social influence for each user when estimating event score, and adopt the *threshold algorithm* to efficiently retrieve the top-K influential events.

3.4.2 SoLoMo Recommendation

In S4, to measure the similarity between users, we use a new metric called *co-space distance* which considers both the user distances in the real world (physical distance) and the virtual world (social distance). The challenge of introducing such a hybrid distance is two-fold:

First, computing the social distance between all pairs of users is costly, even in an offline manner. In our module, we use an efficient social distance computation algorithm based on the parallel processing framework of MapReduce. The results of the MapReduce jobs are indices for the social distances between users (social index), which are inserted into a key-value store, such as Hbase[1]. To reduce the index lookup overhead, top social distances are kept in the adaptive cache.

Second, if user u_0 issues a friend recommendation query, the query engine needs to look up two indices, the location index (e.g., the R-tree index) and the social index. Because the users update their geo-locations continuously, we have to dynamically compute the co-space distances between u_0 and other users at query time. Such query processing incurs high I/O access costs. Therefore, we adopt two techniques, a progressive query processing approach and an adaptive caching approach, to optimize the kNN recommendation algorithm.

The recommendation process involves the following steps: Given a query from the mobile user, the query engine exploits the R-tree index and the social index to retrieve the distances between users. Before the two types of distances are returned to the query engine, they are merged into the co-space distances by SVM model, which is trained from the human workers in the Amazon Mechanical Turk [2]. The engine will rank the users by their co-space distances and generate the top-K users as the query result.

4 Conclusions

In summary, we presented in this paper a general CPSS framework called S4. We focused our discussions on the design and implementation of the key modules which support four different categories of queries and recommendations in CPSS, namely *model-based search*, *non-model-based search*, *real-time media search*, and *geo-social analytics*. There certainly remain vast amount of space to be explored in the mission to build up a CPSS system. We briefly discuss the challenges and open questions to face when developing CPSS applications, and our views of possible research directions.

(1) For model-based search, an essential problem is the sensor inaccuracy. For outdoor-space objects, the geometric models acquired from sensors or other sources are almost always error-prone. As for indoor-space objects, the localization of objects could be a major problem which is still being heavily studied. We do expect disruptive technology on the infrastructure-level to address this problem.

(2) The non-model-based search produces poor performance when the visual database contains impure images which include noises irrelevant to the POI being recognized. The proposed sparse-coding technique only provides a limited solution to this issue. Therefore, we expect stronger computer vision techniques to play an important role in this regard. For example, due to the large number of images available for each POI, it is possible to detect for each image the salient object of interest. As a result, the object recognition can expectedly achieve higher accuracy.

(3) For real-time media search/recommendation, the run-time performance is still a major challenge as the "big-fast data" grows further in both scale and speed. The recent developments in big data processing on new software and hardware platforms (such as GPU clusters) may shed some lights on this issue.

Apart from the above challenges, there are also needs for novel queries and recommendation algorithms from new CPSS applications. We hope this paper would solicit attention from the research community to CPSS which we envision as a future paradigm for mobile Web search.

References

[1] Eugene Agichtein, Carlos Castillo, Debora Donato, Aristides Gionis, and Gilad Mishne. Finding high-quality content in social media. In *Proceedings of the 2008 International Conference on Web Search and Data Mining*, WSDM '08, pages 183–194, 2008.

[2] Chun Chen, Feng Li, Beng Chin Ooi, and Sai Wu. Ti: an efficient indexing mechanism for real-time search on tweets. In *Proceedings of the 2011 ACM SIGMOD International Conference on Management of data*, SIGMOD '11, pages 649–660, 2011.

[1]http://hbase.apache.org/

[2]http://aws.amazon.com/mturk/

[3] Xing Chen, Lin Li, Guandong Xu, Zhenglu Yang, and Masaru Kitsuregawa. Recommending related microblogs: A comparison between topic and wordnet based approaches. In *AAAI*, 2012.

[4] Gao Cong, Christian S. Jensen, and Dingming Wu. Efficient retrieval of the top-k most relevant spatial web objects. *PVLDB*, 2(1):337–348, 2009.

[5] Sanda M. Harabagiu and Andrew Hickl. Relevance modeling for microblog summarization. In *ICWSM*, 2011.

[6] Chia-Jung Lee, W. Bruce Croft, and Jin Young Kim. Evaluating search in personal social media collections. In *Proceedings of the fifth ACM international conference on Web search and data mining*, WSDM '12, pages 683–692, 2012.

[7] Chen Liu, Sai Wu, Shouxu Jiang, and Anthony K. H. Tung. Cross domain search by exploiting wikipedia. In *ICDE*, pages 546–557, 2012.

[8] Rinkesh Nagmoti, Ankur Teredesai, and Martine De Cock. Ranking approaches for microblog search. In *Web Intelligence*, pages 153–157, 2010.

[9] Brendan O'Connor, Michel Krieger, and David Ahn. Tweetmotif: Exploratory search and topic summarization for twitter. In *ICWSM*, 2010.

[10] Pai Peng, Lidan Shou, Ke Chen, Gang Chen, and Sai Wu. The knowing camera: recognizing places-of-interest in smartphone photos. In *SIGIR*, pages 969–972, 2013.

[11] Lidan Shou, Zhenhua Wang, Ke Chen, and Gang Chen. Sumblr: continuous summarization of evolving tweet streams. In *SIGIR*, pages 533–542, 2013.

[12] Shuangyong Song, Qiudan Li, and Hongyun Bao. Detecting dynamic association among twitter topics. In *Proceedings of the 21st international conference companion on World Wide Web*, WWW '12 Companion, pages 605–606, 2012.

[13] Jaime Teevan, Daniel Ramage, and Merredith Ringel Morris. #twittersearch: a comparison of microblog search and web search. In *Proceedings of the fourth ACM international conference on Web search and data mining*, WSDM '11, pages 35–44, 2011.

[14] Chao Zhang, Lidan Shou, Ke Chen, and Gang Chen. See-to-retrieve: efficient processing of spatio-visual keyword queries. In *SIGIR*, pages 681–690, 2012.

[15] Chao Zhang, Lidan Shou, Ke Chen, Gang Chen, and Yijun Bei. Evaluating geo-social influence in location-based social networks. In *CIKM*, pages 1442–1451, 2012.

Building Social Life Networks

Ramesh Jain* Laleh Jalali* Siripen Pongpaichet*

Amarnath Gupta†

* Department of Computer Science, University of California Irvine

{jain, lalehj, spongpai}@ics.uci.edu

† San Diego Supercomputer Center, University of California San Diego

a1gupta@ucsd.edu

Abstract

The availability of enormous volumes of heterogeneous Cyber-Physical-Social (CPS) data streams allow design and implementation of networks to connect people with essential life resources. We call these networks Social Life Networks (SLNs). We are developing concepts, technology, and infrastructure to design and build these networks. SLNs will be helpful in addressing several essential societal problems in everyday life as well as during abnormal situations. A person needs to be connected to appropriate resources under the given situations and her own persona and context. Situations should be detected by using heterogeneous data streams. We are building a software framework for situation recognition and determining persona and personal context to connect people to resources efficiently, effectively, and promptly. We present our research in situation recognition, EventShop, persona building, and making connections using a few example scenarios.

1 Introduction

Like most other technologies, computing was developed and used for meeting human needs. It has been one of the most effective and pervasive technologies to help in diverse human needs. In the last few years its scope has exploded due to the tremendous growth in sensing, mobile, storage, processing, and communication technologies. In the form of mobile phones, armed with various sensors and actuators, computing can now reach more than 80% humans even in the remotest and most underdeveloped parts of the world [12]. In a very practical sense, we are living through one of the most pervasive social transformations being facilitated by computing related technologies. Abraham Maslow developed the famous hierarchy of needs in 1943 [10]. His theory identifies five levels in basic human needs that start from the most basic needs for survival to highest intellectual and sociological drive resulting from unique accomplishments. After physiological demands the needs at the next level are about various forms of security ranging from bodily safety, to economic security through employment, and emotional security through family. The human aspect of serving and being served by social presence starts at this level, and hence basic social norms are part of this level. When these two basic needs are satisfied, a person can live as a part of the social system. The next stage expands the social needs from mere existence to belongingness, and thus creating a personal social

Bulletin of the IEEE Computer Society Technical Committee on Data Engineering

structure that satisfies emotional needs and creates a sense of collective good. At the top of the needs hierarchy is the need for humans to seek recognition from self, and this recognition comes from the feeling that they are doing really well by contributing to the betterment of the society.

Maslow's hierarchy helps us in understanding currently popular social computing and social media. Starting from early days of civilization, new technology has always served the needs of the society. In a very real sense, most of the issues in nature are the results of needs and resources. For meeting every need, a resource is required. Therefore, it is not surprising that much of human knowledge addresses issues that ultimately lead to either creation of resources or proper distribution of resources so that they can be used to connect to proper needs. A very fundamental issue addressed by different fields of studies is how to connect people to resources effectively, efficiently, and promptly based on their need in specific situations.

In the last few years, there is growing interest in social media, crowd-sourcing, participatory sensing and social networks. Social networks are basically an infrastructure developed for connecting people to other people. Like the last few years, social networks will keep evolving to increasingly meet needs of society. Current social networks focus mainly on the third and fourth level of Maslowian hierarchy. We believe that the computing infrastructure is now ready to meet the needs at the first two levels of the hierarchy also. We call such an infrastructure Social Life Networks (SLNs) [1, 2]. Here we discuss an approach for building SLN.

It is essential that SLN bridge across what is commonly called cyber, physical, and social systems. Increasingly, emerging systems have sensors to provide data about physical world, and humans provide data about the social world. All this data is assimilated and processed in cyber world of computers. This has serious implications for data engineers for designing systems to solve new challenging problems. In the following, we first discuss the changing nature of data and computing and then present our approach to building SLN.

2 Recreating Dynamic Real World

A very important aspect of the real world is that it is dynamic. It is always evolving. Most of the new data is collected to capture the dynamic nature of the world. The dynamic data about the past is important because one can model the world using that data and use those models to understand the world, and more importantly, predict future events. Until a few years ago, information systems limited collection and processing of data to very limited aspects of the world. Moreover, many systems were designed assuming that updates were less frequent than retrieval of data. A good example is the popularity of enterprise data warehouses towards the end of the last century. These systems collected data related to limited aspects of operations of an enterprise and tried to gain insights and understanding to create business intelligence. Around the same time one saw beginnings of stream processing starting from streams of stock market data to the so called complex event processing that dealt with limited number of data streams in an organization. Just in less than two decades, the situation has changed dramatically. Collectively, now we are creating a global data warehouse that involves massive number of heterogeneous data streams that must be analyzed in real time for predicting evolving situations and managing them. Some fundamental differences between the data about two decades ago and now are obvious. These differences are essential to building systems that will span cyber, physical, and social aspects of computing. We believe that some important differences are:

1. Most of the data used to be structured and usually human mediated. Now most of the data comes from disparate sensors as data streams.
2. Data used to be related to well-defined operations and expected events. Now the number of operations and events is vast and one needs to deal with the unexpected all the time. Unexpected is the new expected.
3. Data was either unrelated to geo-location or had only a few geo-spatial streams. Now the world is being covered by sensors producing dense data streams. Soon almost every point will be covered by

multiple sensors. Also, the granularity at which data is being collected is becoming finer and finer every day. Location of data has become a fundamental attribute.

4. Most decisions were required for planning the operations of an enterprise. Increasingly, most decisions are communicated to actuators for action. In many cases similar to human-sensing (in Twitter), human actuation (as in flash mobs) may be used in situations like disaster management. Suggestions and actuation are becoming more common than planning.

An obvious fact is that physical world events and situations take place in physical space. Situations are the result of interactions among several related events. Events are the results of some happenings that are due to significant state changes. State changes can be observed by measuring some physical attributes using a sensor. Usually a sensor measures only a specific attribute, which is one of many different independent or correlated attributes required to detect the state change. Norbert Wiener [11] showed that cybernetics is the theory of control and communication in humans as well as machines. His work may be considered first work in building CPS. Systems theory was strongly influenced by his research. Inspired by his approach, we say that the state at a point in space at a given time can be characterized by N attributes. Thus,

$$S = [a_1, a_2, ..., a_N]^T$$

Where S is the state of a point (x, y, z) and a_i is an attribute at the point as measured by a sensor at time t. For brevity we are dropping (x, y, z) and t from each variable and will be assuming until specifically mentioned that all discussion is related to measurements and sensors at location (x, y, z) at time t.

It must be mentioned here that in CPS, many different types of sensors are used. The values obtained directly from sensors may range from measured facts to expressed opinions. This is a challenge for CPS systems to appropriately deal with these values from disparate sensors to reconstruct the state.

The information obtained from sensors varies in many respects. Methods to convert data to information and the reliability of information could be entirely different for different sensors. Humans used as sensors, as in participatory sensing, provide opinions, not measurements. The goal of many research projects with Twitter data is to develop techniques to convert these opinions to one or more attributes above. Physical sensors provide either direct measurements (as in a thermometer) or indirect measurements that are a result of many-to-one mapping. To convert measurements to attributes at a point in space, complex approaches involving inverse mapping are used. Cameras are the best example of such sensors. Thus, one can say that

$$a_i = f_i(m_i)$$

where a_i is the attribute derived from a measurement m_i using the function f_i for this attribute. The function used to convert the measurement to an attribute could be very simple or it could be extremely complex. Good examples of complex functions required for such processing are computer vision systems and tweet analysis systems. As is well known, in a computer vision system, the intensity at a pixel is the result of a 3-dimensional to 2-dimensional projection as well as the light ray's behavior in the environment. Computer vision systems have been so challenging because of this complex many-to-one mapping that should be inverted to find what the pixel really represents. The data value at a pixel is only suggestive and in itself not much important. When analyzed in concert with other values and context, it becomes valuable and useful.

In the last few years, many computer science researchers have been fascinated by analysis and use of Twitter data [3,4]. We believe that this is because a Tweet is text and computing community finds it easy to deal with. Automatic tweet analysis faces formidable challenges. A tweet is the result of a person expressing her opinion on a topic. Though a tweet may have location and time associated with it, the topic of the tweet may be unrelated to its origin. Another challenge is that different people have different language, style, and motivations. To make this more complex, people need to package their thoughts in less than 140 characters so they need to invent suggestive language. All these factors make tweets a weakly related, uncertain, suggestive source of information. One needs to consider this nature of tweets in using it as a source of measurement coming from human sources.

The above analysis of these two different sources shows that one needs to seriously consider the nature of data sources. In data engineering research, until recently the source and nature of data was usually direct symbolic input in structured form. With emerging systems, this is just not doable for most emerging applications. CPS systems present a formidable challenge in "integrating" uncertain heterogeneous data streams with minimal latency. Given that the problems that could be solved using these systems are so important and rewarding; these challenges should not be ignored.

3 General Architecture of the SLN System

In building SLN, we adopt a perspective that there are sensors, databases, and social networks that are observing, storing, and reporting what is happening in the world. A subset of all these information sources is used to characterize the status of available physical or human resources. Similarly, another special subset is used to characterize needs. Thus, we consider all data sources in three classes: Observers, Resources, and Needs. There is overlap and change of roles among these classes. Since the world is dynamic, we consider these classes only in the context of a specific application.

Situation recognition is a central component of the SLN. Situations are the result of interactions among several related events. Events are the results of some happenings that are due to significant state changes. Based on the situation at a place, the system Identifies needs and available resources to satisfy those needs. Situation recognition is always based on the context of a specific application and so are all other operations. The data sources used by the system are also those publicly available or specifically made available in the context of the application. The system closes the loop by sending actuation or action information to appropriate needs and resources as a result of the matching.

An SLN system, as shown in Figure 1, then essentially implements the following loop:

1. Observe multiple real-time data sources
2. Determine interesting atomic events
3. Combine events using complex event definitions
4. Identify situations from these complex events based on personal, social and global context
5. Determine needs using the situations and a set of needs definitions
6. Monitor known resources based on their capabilities, availability, reachability and cost of access
7. Find maximally matching resource for each need detected
8. Communicate decisions and perform actions commensurate with needs by feeding the information back to the social network

Here, not only people, but other objects like mobile applications (e.g. body activity monitors if allowed by the owner), databases, and the Internet of Things (e.g., traffic sensors) also observe, store and report information about the state of entities in the world. In this setting, we conceive of a world where (a) a significant body of information today comes from sensors, (b) the number of sensors is huge and the number of events generated by them is even larger, (c) a large fragment of data, both human and device generated, have associated locational information, (d) most situation and needs assessment decisions are for controlling and managing real time and evolving situations, and (e) keeping pace with the real-time nature of our problem space, planning and decision processes need to be viewed like a real-time control system that interoperates with the publish- subscribe and update-propagation model of standard social networks.

A user update in a social network is analyzed to create a microevent (or a personal event), which is then fed to the situation recognizer. The situation recognizer evaluates this microevent with respect to other events from different sources and creates an action (e.g., a message, a recommendation, an alert) that goes back to the sender or a potential resource that can service the needs of the original message sender.

In the next section we describe two prototypes of situation recognition modules that take two different approaches to detect situations. The first, called EventShop [7, 8], is designed to detect situations that are global and spatio-temporal from sensor streams that observe some portion of a real-world phenomenon. The second, called Personal EventShop, is designed to detect situations about a person's private world. It can serve as a personal agent that recognizes and handles a situation "locally" when possible, or creates a micro-event.

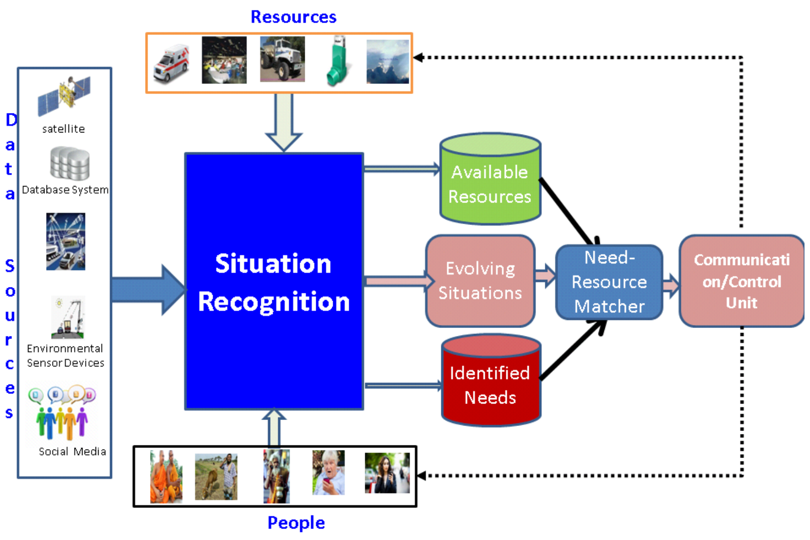


Figure 1: A Social Life Network connects people to resources based on their need in specific situations. Here we show a high level architecture of a SLN system that identifies needs and available resources in a specific situation and then connects needs to resources.

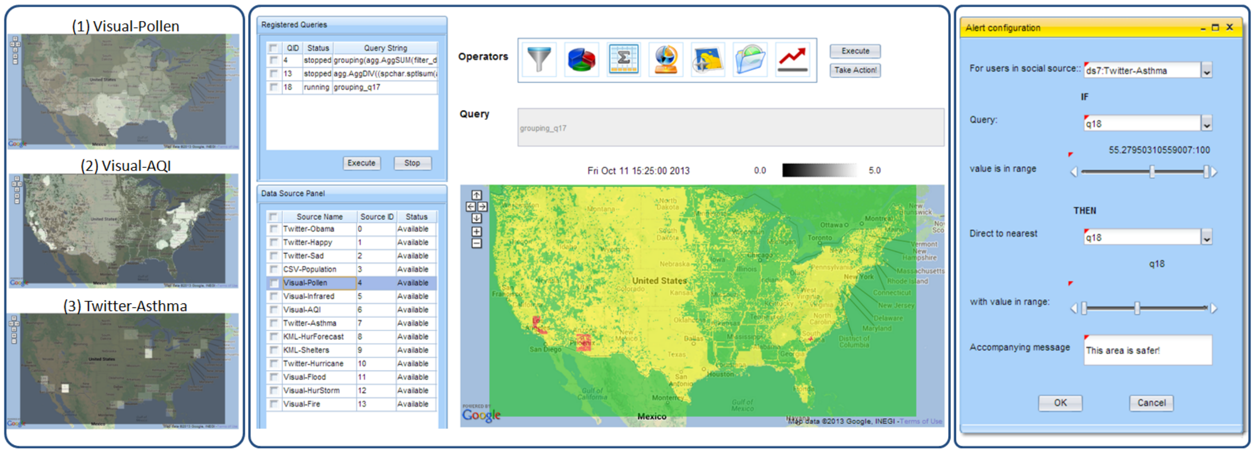
4 Recognizing Situations

4.1 EventShop: The Prototype of a Global Situation Recognizer

Inspired by programming models for distributed systems, we have developed an open source situation modeling and recognition platform called EventShop (ES). The ES framework hides the complex implementation in the background allowing users to simply create massive distributed systems for situation recognition as well as allowing users to focus solely on their application logic and the semantics of their systems. The ES framework has a) a programming model that allows for complex event streaming applications to be created without distributed systems expertise, and b) a prototype implementation of the ES framework that proves its capability to handle wide variety of heterogeneous data streams observing real world events on the internet scale in real-time.

Unlike other CEP systems, ES uses a spatial grid stream as its data model because it is naturally suitable for representing various geo-spatial data; each cell of the grid stores value of certain measure taken from the corresponding geo-location. We adopt the grid structure, and call it E-mage (an event data based analog of image) [6]. We believe that this generic data model can be used to integrate heterogeneous data coming from spatio-temporally distributed web streams. For handling data streams, a special attention needs to be paid to the semantics of aggregated data. Since the data streams are unbounded, combining such data to find simple aggregated value such as summation and average is unclear. This problem is normally resolved by introducing windows which transform an unlimited sequence of data streams into windows of data. In this work, we use tumbling [5] window that splits data streams into non-overlapped contiguous windows.

There are four main components in the ES framework which are data ingestor, stream processing, internal storage, and situation output. In the data ingestor component, original raw spatio-temporal data, either real-time or near real-time streams from the Web are translated into unified STT (Space-Time-Theme) format along with their numeric values using an appropriate data wrapper. A data wrapper for a sensor converts a measurement in a semantic attribute as shown in equation 2 above. Based on users' defined spatio-temporal resolutions mapper, the system aggregates each STT stream to form an E-mage stream. This E-mage stream is then pulled by the stream processing and/or transferred to internal storage. Based on the situation recognition model determined by the domain expert using something akin to equation 1 above,



Registered Queries

	QID	Status	Query String
	4	stopped	grouping(agg.AggSUM(filter_d
	13	stopped	agg.AggDIV((spchar.sptlsum(
	18	running	grouping_q17

Execute Stop

Data Source Panel

	Source Name	Source ID	Status
	Twitter-Obama	0	Available
	Twitter-Happy	1	Available
	Twitter-Sad	2	Available
	CSV-Population	3	Available
	Visual-Pollen	4	Available
	Visual-Infrared	5	Available
	Visual-AQI	6	Available
	Twitter-Asthma	7	Available
	KML-HurForecast	8	Available
	KML-Shelters	9	Available
	Twitter-Hurricane	10	Available
	Visual-Flood	11	Available
	Visual-HurStorm	12	Available
	Visual-Fire	13	Available

Alert configuration

For users in social source:: ds7:Twitter-Asthma

IF

Query: q18

55.27950310559007:100

value is in range

THEN

Direct to nearest q18

q18

with value in range:

Accompanying message This area is safer!

OK Cancel

Figure 2: Asthma Relief Application (from left to right): Data Sources E-mages, ES Web UI, and Situation Action Rule.

appropriate operators are applied on the E-mage streams to detect situation. In most of the cases, the final step is a segmentation operation that uses domain knowledge to assign appropriate class to each pixel on the E-mage. This classification results in a segmentation of an E-mage into areas characterized by the situation there. Once we know the situation, appropriate actions can be taken depending on application action control rules in the situation output component. From the application developers' point of view, a workflow of moving from heterogeneous raw data streams to actionable situations consists of three simple steps: 1) register or select appropriate data stream sources, 2) register complex event model by combining a rich set of built-in operators and 3) define event condition action rules to send personalized alerts to relevant people.

The operators of EventShop are used to detect interesting E-mages based on value-patterns on the spatial grid stream. We analyzed situation recognition models across multiple domains and defined the initial set of operators, which are generic enough to capture most of the common requirements. These operators are: selection, segmentation, aggregation, spatial characterization, spatial pattern matching, temporal characterization, and temporal pattern matching. We developed the EventShop system so that a situation modeler can define an algebraic plan using the operators above to specify a situation. The output of such a situation detection plan is an E-mage that can subsequently be transformed to a convenient form and sent directly to a client, to a social network system, or to an end application used by the client.

4.2 Use Case: Asthma Relief Application

For the past couple of years, several applications have been designed and developed using the EventShop framework to recognize real world situations. for example, hurricane detection and migration, demand hot-spots of business products identification, flu outbreak, wildfires detection, flood migration, and allergy risk and recommendation [7, 8]. Here, we demonstrate asthma relief application, the most motivating one. The goal is to suggest safer areas to people who are in asthma risk zones. From asthma study, the severeness of an environment to an asthma patient is related to the pollen count and air quality in that area. From crowd sourcing aspect, if many people from a specific area discuss about asthma, it usually suggests that the situation in that area is not very friendly to asthma patients. We combine data from these three data sources, and then segment the aggregated data over entire US into three danger zones based on the values in aggregated E-mages as shown in Figure 2. Then, a situation action rule is created to broadcast a safe area to an individual. Although, these messages is partially personalized based on end-userâĂŹs location, we could provide more specific and useful recommendations if more detailed information about end-user's context was available.

4.3 Personal EventShop

To close the loop in a SLN, the system should send personalized action instructions/requests to users. This requires knowing persona and personal context. We realized that the persona can be computed by using several data streams related to a person. For this task many processing operations are similar in concepts to EventShop. We are building Personal EventShop (PES) for meeting this need. PES is designed with the assumption that the states, events and situations recognized by it come from observing personal information streams. It uses an individual's life events from sensors and mobile devices that capture fitness data, personal events, eating habits, sleep patterns, and every day activities.These information comes from more structured information sources like personal calendars, as well as less structured information such as activities on her social networks. All these data sources represent data streams related to the person's life. The persona is built by collecting and analyzing these streams in a long term data warehouse for the person. For converting different data streams to related event and situation streams, mathematical and computational operators similar to those used in EventShop are required. The event data from Personal EventShop will be stored in a personal data warehouse where all data and important results of intermediate computations will be stored. Key aggregates and results required to build persona may also be saved for rapid computations of requested information and insights when needed.

The major difference between the EventShop and PES is that PES deals with an individual related stream so is basically one dimensional stream, while ES deals with physical space so is either 2-dimensional or 3-dimensional grid. All activities of a person in her persona should be related to her life events.

5 Research Challenges

We believe that there are several challenges in building a SLN platform that will address many emerging applications. We consider the following as particularly interesting and important research challenges.

1. Massive Heterogeneous Geo-spatial Stream Processing: Traditional data processing techniques considered data streams as a sequence of data items. Increasingly geo-spatial heterogeneous data streams at different granularity must be combined to detect emerging situations. This requires a different perspective on processing and managing data.
2. Situation Recognition: Complex Event Processing was satisfactory when we had a few data streams and the result of CEP was of interest to an organization. Evolving situations affect large number of people and resources and must be determined using complex spatio-temporal semantics of streams. This is a new research area.
3. Persona and Personal Context: Most search engines create very narrow persona for users to serve them advertisements or for recommending them products. Emerging applications have large number of personal data streams that could be used to build more sophisticated persona that could be used in diverse applications considering more specific user context.
4. Chronicle Analytics and Visualization: Enterprise data warehouses started making analytics and visualization of data popular. Big data has created significant interest, some say hype, around analytics and visualization. Now one should consider chronicles at different level and in different situations and develop techniques for analysis, prediction, and visualization with as little latency as possible.
5. Dynamic Need-Resource Optimization: Now that we are developing techniques for situation dependent need identification as well as resource availability, we need to extend techniques for optimal satisfaction of needs in given situations with as little cost as possible. This becomes a challenging task.

6 Conclusion

We are living in a very exciting time. In the last few years data storage, processing, and communication technologies have transformed the nature and volume of data that will be routinely used in emerging applications. The new data is massive number of geo-spatial heterogeneous data streams. This forces us to rethink not only about the data, but also about the processing, communication, storage, analysis, and visualization. This also opens up unprecedented opportunities for addressing serious societal problems that we could not consider earlier. Very challenging, exciting, and rewarding time is ahead for people interested in data.

References

[1] A. Gupta and R. Jain. Social life networks: A multimedia problem? In *Proc. of ACM Conf. on Multimedia (SIGMM'13)*, 2013.

[2] R. Jain and D. Sonnen. Social life networks. *IT Professional*, 13(5):8–11, 2011.

[3] H. Purohita, A. Hamptonb, V. L. Shalinb, A. P. Shetha, J. Flachb, S. Bhatta. What kind of conversation is Twitter? Mining psycholinguistic cues for emergency coordination. *Computers in Human Behavior*, 29(6):2438Ð-2447, 2013.

[4] C. H. Lee. Unsupervised and supervised learning to evaluate event relatedness based on content mining from social-media streams. *Expert Systems with Applications*, 39(18):13338Ð-13356, 2012.

[5] J. Li, D. Maier, K. Tufte, V. Papadimos, and P. A. Tucker. No pane, no gain: efficient evaluation of sliding-window aggregates over data streams. *ACM SIGMOD Record*, 34(1):39–44, 2005.

[6] V. K. Singh, M. Gao, and R. Jain. Social pixels: genesis and evaluation. In *Proc. of the ACM Int. Conf. on Multimedia (MM'10)*, pages 481–490, 2010.

[7] M. Gao, V. K. Singh, Singh, and R. Jain. EventShop: From heterogeneous web streams to personalized situation detection an control. In *Proc. of the 3rd Annual ACM Web Science Conf. (WebSci'12)*, pages 105–108, 2012.

[8] S. Pongpaichet, V.K. Singh, M. Gao, and R. Jain. EventShop: Recognizing situations in web data streams. In *Proc. of the 1st WWW workshop on Web Oservatories (WOW'13)*, 2013.

[9] L. Jalali, and R. Jain. Building health persona from personal data streams. In *Proc. of the 1st ACM Workshop on Personal Data Meets Distributed Multimedia (PDM'13)* , 2013.

[10] A.H. Maslow, A theory of human motivation. *Psychological Review*, 50(4), 370Ð96, 1943.

[11] N. Wiener. Cybernetics or the control and communication in the animal and the machine. *Second Edition, MIT Press*, 1948.

[12] R. Scoble, S. Israel. Age of context: mobile, sensors, data and future privacy. *First Edition, CreateSpace Independent Publishing Platform*, September 5, 2013.

Data Engineering

It's FREE to join!

TCDE

tab.computer.org/tcde/

The Technical Committee on Data Engineering (TCDE) of the IEEE Computer Society is concerned with the role of data in the design, development, management and utilization of information systems.

- Data Management Systems and Modern Hardware/Software Platforms
- Data Models, Data Integration, Semantics and Data Quality
- Spatial, Temporal, Graph, Scientific, Statistical and Multimedia Databases
- Data Mining, Data Warehousing, and OLAP
- Big Data, Streams and Clouds
- Information Management, Distribution, Mobility, and the WWW
- Data Security, Privacy and Trust
- Performance, Experiments, and Analysis of Data Systems

The TCDE sponsors the International Conference on Data Engineering (ICDE). It publishes a quarterly newsletter, the Data Engineering Bulletin. If you are a member of the IEEE Computer Society, you may join the TCDE and receive copies of the Data Engineering Bulletin without cost. There are approximately 1000 members of the TCDE.

Join TCDE via Online or Fax

ONLINE: Follow the instructions on this page:

www.computer.org/portal/web/tandc/joinatc

FAX: Complete your details and fax this form to **+61-7-3365 3248**

Name	
IEEE Member #	
Mailing Address	
Country	
Email	
Phone	

TCDE Mailing List

TCDE will occasionally email announcements, and other opportunities available for members. This mailing list will be used only for this purpose.

Membership Questions?

Xiaofang Zhou
School of Information Technology and Electrical Engineering
The University of Queensland
Brisbane, QLD 4072, Australia
zxf@uq.edu.au

TCDE Chair

Kyu-Young Whang
KAIST
371-1 Koo-Sung Dong, Yoo-Sung Ku
Daejeon 305-701, Korea
kywhang@cs.kaist.ac.kr