

Measuring Online Service Availability Using Twitter

Marti Motoyama[†] Brendan Meeder* Kirill Levchenko[†]
Geoffrey M. Voelker[†] Stefan Savage[†]

[†]Dept. of Computer Science and Engineering
University of California, San Diego, USA
{mmotoyam,klevchen,voelker,savage}@cs.ucsd.edu

*Dept. of Computer Science
Carnegie Mellon University, Pittsburgh, USA
bmeeder@cs.cmu.edu

Abstract

Real-time micro-blogging services such as Twitter are widely recognized for their social dynamics — how they both encapsulate a social graph and propagate information across it. However, the *content* of this information is equally interesting since it frequently reflects individual experiences with a broad variety of real-time events. Indeed, events of broad interest are commonly revealed in correlated spikes of semantically-related posting activity. In this paper, we explore one such application of this phenomenon: using Twitter data to infer on-line Internet service availability. We show that simple techniques are sufficient to extract key semantic content from “tweets” (i.e., service *X* is down) and also filter out extraneous noise. We demonstrate the efficacy of this approach at identifying a range of large-scale service outages in 2009 for popular services such as Gmail, Bing and PayPal.

1 Introduction

Real-time micro-blogging, exemplified by Twitter, has quickly achieved widespread popularity among Internet users. Twitter allows users to publish 140-character long messages (known as *tweets*). Individuals that subscribe to a particular user’s updates, called the *followers* of that user, see these updates in real-time. Moreover, these messages are instantly viewable and searchable to anyone on the internet. One recent report [19] estimates that Twitter’s user population climbed to over 75 million in 2009, and the number of legitimate (i.e., non-spam) tweets posted to the site per day has reached over 50 million [52]. However, this distribution is by no means uniform, and a significant fraction of Twitter users send a myriad of updates throughout the day on a vast array of topics. This combination of a broad user base, large message volume and the dynamic range of the information has increasingly made Twitter a “go to” venue for acquiring current information on a topic. Indeed,

a number of reporting services have employed Twitter in their research for mainstream news stories [13]. Conversely, when an event of sufficient magnitude occurs, large numbers of contemporaneous tweets about the event inevitably follow. Thus, aggregating across the messages of Twitter users is a kind of passive crowdsourcing, one in which individual users act as noisy sensors whose updates can be correlated and filtered to extract the presence of key events.

In this paper we focus on one such application of this phenomenon: inferring Internet service availability. Thus, we seek to determine when popular services experience downtime. Understanding the prevalence of such events is increasingly important as consumers and businesses alike move to “cloud-hosted” applications. For example, according to [7], more than 1.75 million businesses utilize Google Apps (and businesses pay \$50 per user annually for access to the Google Apps collaboration suite), while the social networking site Facebook commands over 400 million users. Thus, an outage at one of these sites can translate into both lost revenue and a significant disruption in the lives of users. Moreover, while availability is a key priority for virtually all service providers, it is also clear that 100% availability has not been achievable in practice.

In 2009, a large number of widely used Web services (e.g., Amazon, Gmail, Bing, etc.) suffered from widespread outages. In this paper, we will demonstrate that these outages are in turn revealed across user tweets via a small number of lexical features such as the phrase “is down” and “fail” hash tags (e.g., “#gfail”), to signal an outage.

- T1. gmail is down! ughh... #gmailfail
- T2. It's funny watching the real-time panic on Twitter as everyone realizes Gmail is down.

However, Twitter is by no means the only way to measure such outages. There are a range of *direct* methods for measuring service availability, such as Keynote’s well-known commercial network monitoring service,

that are strictly superior in many ways. In particular, direct and active measurements can be used to monitor a particular service over well-known regular intervals (e.g., every second) and with well-defined failure semantics such as a failed TCP connection to the site. Thus, they offer results that are both timely and precise.

Why then consider the indirect human-mediated approach expressed via Twitter messages? Because inferring service downtime indirectly via human complaints offers a number of distinct advantages as well. First, this approach does not require pre-specifying the service of interest; so long as the service is popular, its failure is likely to appear in the Twitter record. Thus, service dependencies and relationships that may not be known *a priori* are implicitly revealed. For example, a site that uses Google’s AppEngine or Amazon’s AWS may experience downtime when their underlying service infrastructure has an outage. Second, Twitter offers a unique set of vantage points for observing failures — literally millions of locations across the Internet — that would be hard for any explicitly provisioned sensor infrastructure to match. Finally, the human-mediated nature of a Twitter-based failure detector offers a degree of breadth and flexibility in the *definition* of failure that is difficult to encapsulate in a direct sensor. Humans identify a far broader array of experiences as a failure (e.g., site is too slow, messages aren’t being received, messages are corrupted, buttons don’t work, data is out of date, site time’s out, etc.) than are typically or easily encoded as predicate conditions in a measurement device. In a sense the end-user’s perception of downtime is the closest to a true end-to-end measurement and certainly the one that most constitutes customer satisfaction. As Protagoras is known for saying, “Man is the measure of all things.”

Thus, while we would not argue for using Twitter to replace existing measurement services, we do argue that it is a useful complement that offers its own advantages.

The remainder of the paper is organized as follows. Section 2 provides background on the Twitter service and related work focused on mining this data, followed by a description of our technique in Section 3. Sections 4 describe our results and an analysis over a large Twitter data set, and Section 5 summarizes our findings.

2 Background

Social networking Web sites are increasingly important for a range of personal and professional interactions. While many social networks focus on sharing information within a limited circle of well-known connections, the Twitter social network has a different design point focused on broad real-time information dissemination. The units of information produced and consumed in the Twitter network are short textual messages known as *tweets*.

These are limited to 140 characters and are published under the user’s identifier. The sole explicit relationship in the network is that of *following*. A user can choose to follow anyone in the network (with the exception of *protected* users who must approve the relationship). When a user U follows a user V , messages created by V are assembled in a chronologically ordered list of messages called the *timeline* for user U .

Various textual conventions exist that assign specific semantics to different tokens in Tweets. Words or phrases preceded by a # symbol, which are known as *hash tags*, provide the ability to designate that a message discusses or references a particular topic. One popular trend is to communicate failures or malfunctions using the *#fail* hash tag. Our work will use mentions of *#fail* and its variants to detect anomalous conditions.

The ease of composing a short message in any of the myriad of Twitter clients encourages users to share information spontaneously and in real-time. Furthermore, Twitter differs from many other services in that messages are immediately made globally available and searchable. These characteristics together make Twitter an attractive tool for studying a range of social phenomena and especially so for acute and broadly felt events such as the failures of popular Internet services we study here. Moreover, Twitter offers significant advantages over other social networking services from a concrete data collection standpoint as well. While almost all social networks offer an application programming interface (API) to extract and manipulate network data in a structured form, these APIs typically come with a substantial number of restrictions including limitations on the number of requests or duration over which one is permitted to store data. Moreover, rapid page scraping or crawling is typically either prohibited by the terms of service or results in the crawler’s IP address being blocked. Thus, in practice these immense and valuable data sources are largely sealed in virtual walled gardens, available only to the service provider themselves. By contrast, Twitter has embraced an open ecosystem that allows for large-scale data collection and the creation of real-time services.

For many of these same reasons, Twitter has been an attractive data source for a range of academic research which can be roughly broken into two categories. The first concerns studies of Twitter itself, its user base, their relationships and the resulting social network. This work is plentiful, and among the better known examples, Java *et al.* examine the topological, geographical and communal properties of the social network [23], while Krishnamurthy *et al.* present a comprehensive characterization of Twitter users (e.g., followers vs. followings) [27].

A second, but smaller, set of research has looked at applications of the Twitter data itself (typically social applications). For example, Jansen *et al.* analyze more than

150,000 postings to determine branding sentiments [22]. In particular, the authors identify a small number of brands *a priori* and search for tweets containing those words; their search terms are generally unambiguous (e.g. “Banana Republic”). Similarly, Jagen *et al.* investigate building a news processing service based on Twitter [37]. In particular they hand-pick a set of 2,000 reputable Twitter users (e.g., CNN, MSNBC) whose tweets are used to determine that a topic has become “news”.

To the best of our knowledge, the only other paper that considers the use of Twitter for inferring non-social phenomena is Sakaki *et al.*’s study of Earthquake sensing [36]. Their recent paper describes techniques for inferring the epicenter of earthquakes and the trajectories of typhoons based on tweet geo-location. However, most of the complexity in their application is centered around the development of the associated spatial models and not in detecting the presence of the underlying phenomenon. This is because the authors are able to use very uncommon words, “earthquakes” and “typhoons”, to locate pertinent tweets and thus have few problems with filtering noise. By contrast, our application is less complex since we only care about the presence of the event (Internet service outage) but must contend with more noise in the data stream, since all of our terms are commonly found in a wide variety of tweets.

3 Methodology

At a high level, our approach is to treat certain tweets as signals that a particular service is experiencing an outage, whether planned or unplanned. Thus, the Twitter network becomes a large heterogeneous sensor network, where the Twitter users serve as sensor nodes. Our service outage signals consist of two simple lexical features: the phrase “*X* is down” and the hash tag idiom “#*X*fail” where *X* is typically the name of an online service. For example,

- T3. Gmail is down! EVERYBODY PANIC!
- T4. ...oh not again #gmailfail

3.1 Data Collection

To drive our study, we have collected content from the Twitter social network using their publicly available API. However, due to the scale of Twitter, this itself is a significant effort. A complete description of the data collection methodology and infrastructure is beyond the scope of this paper, but we outline the basic approach here.

Twitter implements their API using simple HTTP methods that accept or return data in a structured format, such as XML or strings in JavaScript Object Notation (JSON) form. To avoid excessive use and abuse of this service, the number of requests per client is limited; the

baseline number of requests is 150 per hour per client. However, users can request their screen name or IP address to be *white-listed*, at which point they can make 20,000 requests per hour from that resource.

3.1.1 Crawl Infrastructure

We have created a distributed crawler that can capture user profiles, network topology and message content at scale. Our distributed infrastructure consists of *crawler* machines and a centralized *controller*. The controller prioritizes *workloads*, which are assigned to the crawler machines. These workloads consist of a collection of user identifiers and specific content to retrieve, such as the user’s profile, social links, or messages. We implemented this system in Python. To enable better network concurrency, each crawler launches many worker processes which are each given a small fraction of the crawl list. With this architecture we regularly maintain more than one thousand concurrent connections to the API. Once a crawler receives a response indicating it has no remaining API calls, data retrieved from the API is aggregated and sent to the Amazon Web Services S3 platform for persistent storage. This data is then parsed, filtered, and organized in a distributed filesystem for later consumption by analysis processes.

3.2 Detecting Outage Tweets

To detect outage tweets, we focus on those that *express a predicate* over some set of entities. We define two particular predicates, *IsDown(X)* and *Fail(X)*, corresponding to the associated lexical features we described earlier.

IsDown Predicate. Specifically, we say a tweet expresses the predicate *IsDown(X)* if it simply contains the phrase “*X* is down,” where *X* is a token consisting of alphanumeric characters. Tweet T3 earlier expresses the predicate *IsDown(Gmail)*. In general, *X* is the word immediately preceding the phrase; however, we make exceptions to include two preceding words when *X* is in the set {mail, email, store}. While predominantly used to describe services, the expression “is down” can be used for other purposes. For example, the subject of the abridged tweet “kicking a man while he is down ;)” is Tiger Woods. Thus, our signal detection must be resilient to noise resulting from semantic ambiguity.

Fail Predicate. Users sometimes express that a service is unavailable using hash tags of the form “#*X*fail”, where *X* is a (sometimes abbreviated) service name. A tweet might also convey the same meaning by containing two separate hash tags, “#*X*” and “#fail”. We say that a tweet expresses the *Fail(X)* predicate if it contains either form. These hash tags are more generally used as an expression of the “fail” meme [55], e.g., “#obamafail,” “#gopfail,”

Bing (12/3/09)		Gmail (9/1/09)		Hotmail (3/12/09)		Flickr (10/20/09)		PayPal (8/3/09)	
Phrase	Count	Phrase	Count	Phrase	Count	Phrase	Count	Phrase	Count
maps beta	702	is down	12,836	is down	61	is down	261	and its	318
bing bing	678	is back	2,774	server is	24	photo	227	payfail	190
is down	445	being down	1,630	server too	23	is	98	is down	81
the new	288	down	998	is	14	down battling	83	suffers from	80
with revamped	278	was down	994	is back	14	is back	77	shut down	66
<i>Other</i>	13,934	<i>Other</i>	171,772	<i>Other</i>	1,570	<i>Other</i>	8,235	<i>Other</i>	4,707
Total	16,325	Total	191,004	Total	1,706	Total	8,981	Total	5,442

Table 1: Summary of the top 5, 2-word phrases surrounding each service name.

and “#epicfail.” Since this trend is rather widespread in the meme-aware Twitter community, we do not use this signal in isolation. Rather, we use the $Fail(X)$ predicate to reinforce the signal generated by $IsDown(X)$.

We chose these predicates because they appeared in a large number of tweets concerning service outages. We hand labeled 878 tweets from the Gmail(2), Hotmail, PayPal, and Bing outages. We discovered that the top bigram among this set was “is down” (2.4% of all bigrams), and the top hashtag (not consisting of a company name) was fail (8.2% of all hashtags).

To validate both the subject and topic detection of the predicates, we extracted the two words preceding and following the name of a company during a known outage. Table 1 shows the top five, 2-word phrases surrounding the service name. Note that, if the tweet is comprised of “is gmail down?”, then the preceding phrase is simply “is”, and the following phrase is “down”. The table shows that users who tweeted about the service in question expressed that the service was down, and they did so by tweeting “(service) is down”.

We can incorporate additional predicates into our system (capturing both synonyms such as “crashed” and syntactic variants such as “its down”) to further improve our detection mechanism. However, we find that $IsDown$ is sufficient to capture many service outages, and we restrict our analysis to this single predicate.

We emphasize that expressing an $IsDown$ or $Fail$ predicate is defined *lexically*; we do not attach any formal semantics to each individual tweet. Rather, our premise is that *in the aggregate* an increase in the frequency of expression of a given predicate correlates to a real event. In the next section, we evaluate the validity of this premise. Before doing so, we need to specify how many expressions of the $IsDown$ or $Fail$ predicates are combined to infer that a service outage has taken place.

3.3 Interpreting Predicate Signals

In the normal course of events, we expect that there will be some number of tweets that have the lexical signature

of the $IsDown$ or $Fail$ predicates. In some cases, the user truly intended to express that something was “down”, while in others, the user was expressing something unrelated. Because $Fail$ is more ambiguous than $IsDown$ we focus our analysis on detecting outages using $IsDown$. Later, we describe how to incorporate the $Fail$ predicates to filter out false positives from our detection technique.

3.3.1 Exponential Smoothing

We partition the tweets matching the $IsDown$ predicate into five-minute bins. Subsequently, we use an Exponentially Weighted Moving Average (EWMA), commonly used to predict the next value in a time series, to detect significant deviations away from normal behavior [8, 53]. Let y_n denote the number of tweets expressing the $IsDown$ predicate at time interval n . We then compute the EWMA value, M_n at time n as follows:

$$M_n = \alpha y_n + (1 - \alpha)M_{n-1}$$

where α dictates how much weight the current value is given. We also compute a smoothed deviation σ_n to determine whether an anomaly is occurring at n .

$$D_n = y_n - M_{n-1}$$

$$\sigma_n^2 = \beta D_n^2 + (1 - \beta)\sigma_{n-1}^2$$

Finally, we compute threshold T_n as:

$$T_n = M_{n-1} + \varepsilon \sigma_{n-1}$$

If y_n is greater than equal to T_n for two consecutive intervals, we signal a service outage. We wait for two consecutive threshold violations because flagging the first violation yields a large number of false positives, and waiting for three produces too many false negatives. Once we signal an outage for a particular entity, we do not signal additional outages for that entity for 12 hours. In our experience, most outages are repaired within this period.

3.3.2 Choosing Parameters

There are three parameters to configure in the above equations: α , β , and ε . We obtain a validation set

of labeled events by searching for outages affecting a small set of varied services: Flickr, Hotmail, LiveJournal, Ning, PayPal, and TMobile. To construct the validation set, we located days where the *IsDown* predicate was expressed more than 2 times, then we searched for news articles pertaining to the outages. Additionally, Flickr [15], LiveJournal [28], and Ning [32] maintain blogs that mention downtimes; we use these to check for false positives and to find false negatives. In total, we discovered eight outage events for Flickr, four for Hotmail, 10 for LiveJournal, 14 for Ning, three for PayPal, and six for TMobile. Note that we do not differentiate between planned and unplanned downtime.

To find the best settings for the three parameters, we attempted various combinations and picked the set that yielded the highest average F-Score across the six validation sets. Note that we express α and β in terms of their half lives; a half life of x for either parameter means that an observation from x minutes ago receives half the weight of the current one. We varied the half lives for α and β in five-minute increments, iterating over the range [5,30] for α and [5,90] for β . Additionally, we varied ϵ from one to three in 0.5 increments. We found that α with a half life of 15 minutes, β with a half life of 60 minutes and ϵ equal to two produced the best results, with an average F-Score of 0.510.

4 Evaluation

In this section, we evaluate how well our technique identifies service outages. Using the data collected in Section 3 we ran our analysis on tweets authored during 2009 calendar year. To assess how well our system detects real outages, we identified *a priori* eight events when a known service disruption took place, and examined the strength of the corresponding signal generated using our technique. Subsequently, we applied our analysis on the entire corpus of tweets. We manually inspected the top 50 events (as measured by the volume of tweets matching the *IsDown* predicate) identified by our system, as well as 50 random events detected by our methodology.

4.1 Known Outages

We begin by evaluating the performance of our system on known outages. For this part of the evaluation, we selected several of the top services listed by Alexa. Using Google News, we attempted to find articles pertaining to outages for the services in question. We chose the events shown in Table 2, as they span a wide range of affected users, service types, and interests. The source column refers to a news article discussing the event. The time

Service	Date	Reported Time	Source
Gmail(1)	Feb. 24, 2009	1:30-4:00	[35]
Hotmail	Mar. 12, 2009	8:44-10:15	[29]
Google	May 14, 2009	7:48-9:48	[11]
PayPal	Aug. 3, 2009	11:00-14:00	[40]
Gmail(2)	Sep. 1, 2009	12:30-14:30	[26]
Flickr	Oct. 20, 2009	8:51-10:50	[14]
Bing	Dec. 3, 2009	18:30-19:00	[9]
RackSpace	Dec. 18, 2009	13:37-14:12	[33]

Table 2: Known service outages used in the evaluation, listed in chronological order. Times are in the US Pacific time zone.

column lists the hours (US Pacific time) that the article claims the outage took place.

Recall that we signal an outage when our observations exceed a threshold defined by both a smoothed average and variance. To decrease the number of false positives, we require that a threshold violation span two consecutive bin intervals (5 minutes) before we raise an alarm. To further minimize false positives, we can check whether the *Fail* predicate is expressed within 60 minutes of the triggered *IsDown* alarm.

Figures 1 and 2 show our results on the known events. For each top graph, three lines are shown: the line labeled *isdown* displays the volume of tweets matching the *IsDown* predicate, while the lines labeled *avg* and *threshold* show the computed EWMA and threshold values, respectively. Below each graph we show 3 impulses. The one labeled *reported* indicates when the downtime transpired according to the news articles. The one labeled *isdown* indicates when our system triggers an outage alarm using only the EWMA violation. The final impulse labeled *isdown+fail* shows when we signal an error using the added *Fail* predicate.

We can see that the technique triggers an outage alarm for each known event. The outages are detected anywhere from 10 minutes (Bing, PayPal) to 50 minutes (Gmail(1), Hotmail) after the reported downtime. There is an opportunity improve upon our delay times by utilizing additional predicate expressions; we leave this objective to future work. Adding in the *Fail* predicate to filter false positives adds little to no delay, as demonstrated by the overlaid signals on Gmail(1), RackSpace, and Bing. In the worst case, we delayed triggering our alarm for an additional 25 minutes (Hotmail).

4.2 Unknown Outages

We now utilize our threshold-based signal detection on the entire corpus of tweets by extracting all tweets matching any of the predicates. First, we ran our analysis without checking for the *Fail* predicate, discovering 5,358 “outages” over 1,556 entities. However, we found a

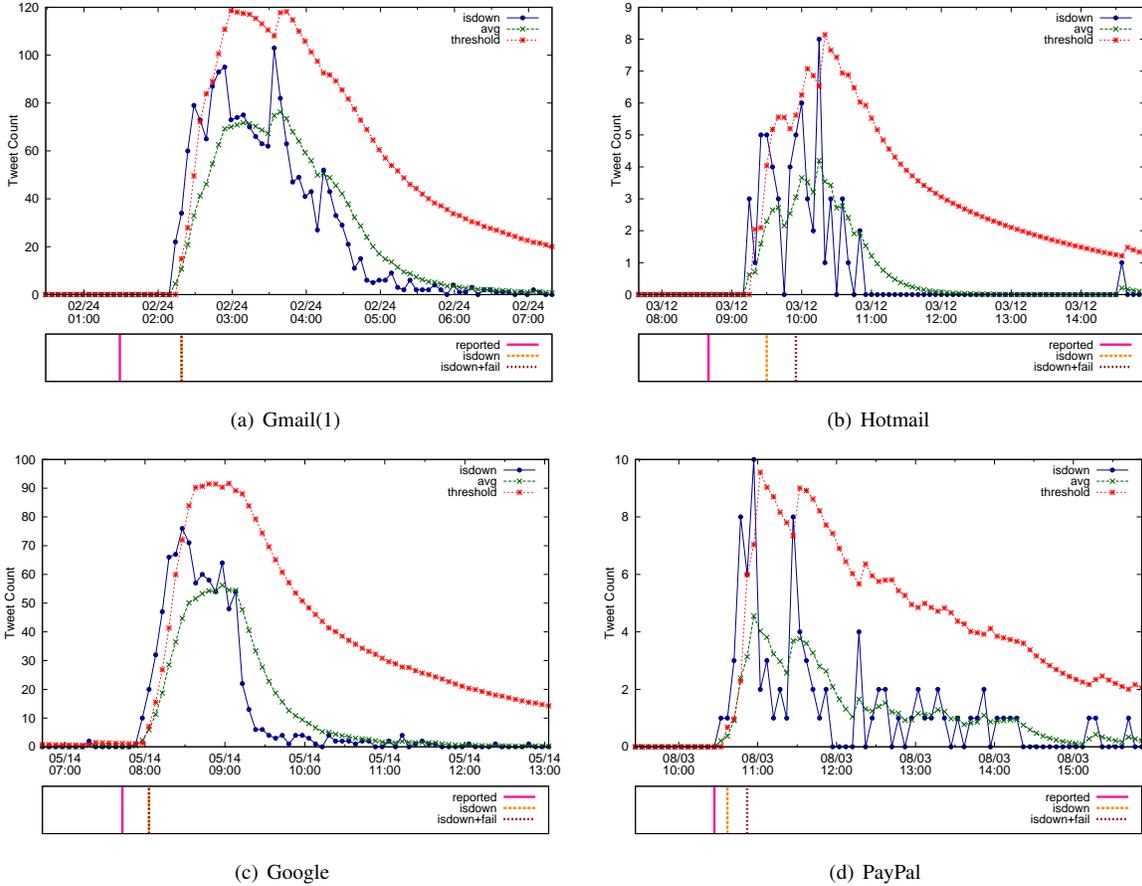


Figure 1: Graphs demonstrating how the EWMA and threshold computations change over the course of an outage. Below each main graph, we show when a news source reported a downtime, and when our system triggers an alarm. Note that the several of the triggers are overlaid on each other.

fair number of invalid entities in this set, such as “attendance”, “visibility” and “tourism”. To filter out these false positives, we employed the *Fail* predicate, thus ensuring that at least 1 *Fail* occurs within 60 minutes of the *IsDown* trigger. By including this predicate, our methodology produced 894 “outages” spread across 245 entities. We inspected the 245 entities; 59 did not appear to be related to technology or Internet services. For example, a myriad of sports teams appeared in the outage set (such as USC, Michigan, Duke, Liverpool), as did subjects related to fiscal matters (such as the DOW, economy).

We then analyzed the top 50 service outages, as determined by the volume of *IsDown* tweets that occurred within 12 hours of the trigger time. Table 3 details 25 of the events captured by our methodology. Of the top 50 events, we were able to validate 48 of them with online news references. YouTube downtimes comprised 11 of our top 50 events (or 22%), while Gmail outages contributed nine (18%). One of the more unexpected downtimes in the top 50 was the Wikipedia outage on July 31st, 2009 [38]; we did not anticipate such a vociferous reaction to the event.

Surprisingly, we are able to detect Twitter service outages. Our experiences crawling and utilizing Twitter, as well as the commentary provided on the Twitter blog, indicate that Twitter often succumbs to sporadic API or website failures (affectionately known as ‘fail-whales’). These errors tend to be transient; a user who encounters such an error might be able to successfully post a message on his next attempt. As users can vocalize problems with Twitter during periods of degraded performance, we are able to, and frequently do, detect such events.

Next, we randomly selected 50 outages that affected valid entities. We successfully located news articles or forum postings that confirmed 35 of these downtimes. We show these outages in Figure 3, and color code them according to whether they were confirmed. In the random set of outages, we find many references to social media sites (LiveJournal [24], YouTube [51]), online games (Azphel [1], World of Warcraft [17]), and hosting services (1and1 [5] and BlueHost [21]).

Of the remaining outages that we could not confirm, one in particular seems rather large: YouTube on June 19th, 2009. Over 165 tweets expressing the *IsDown*

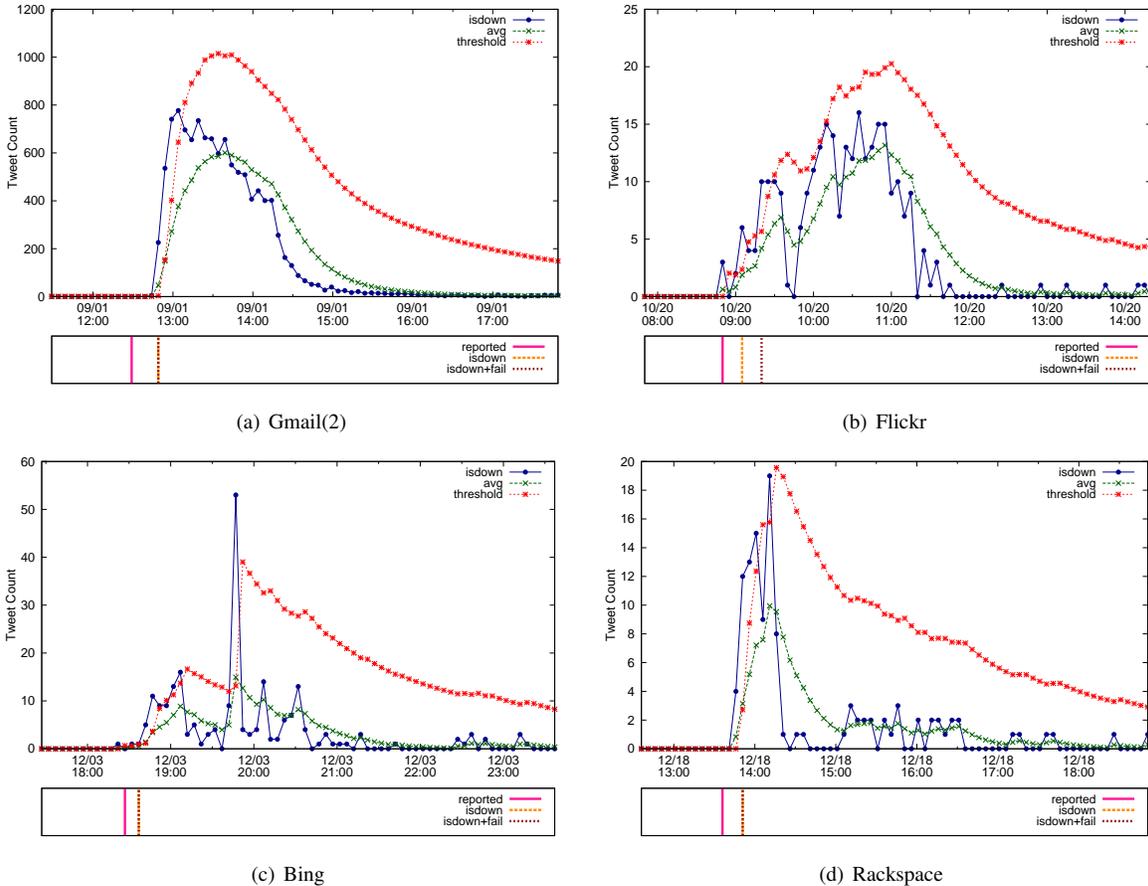


Figure 2: Additional graphs for known outages.

predicate were posted within the 12 hour window after the trigger time. Also, we could not verify either of the outages detected for the sizable service LinkedIn on Oct. 7th or Dec. 22nd, 2009, although the magnitudes of the events were rather low.

5 Discussion and Conclusion

Real-time microblogging represents a new “kind” of communication on the Internet, one that encapsulates information that was not previously shared broadly. We believe that the “user as implicit sensor” model has tremendous potential for capturing a broad array of current events and sentiments *as they happen*. This paper has explored one such application focused on Internet service outages, and we have demonstrated that even simple techniques can identify important events.

Looking forward we identify three key areas of further work. The first is in better extracting the semantics of tweets. For our application, there was sufficiently large message volume and sufficiently narrow range of common expressions that a single phrase (“is down”) was commonly found during true failures. Trying to

identify events with less broad impact or that engender a broader array of expression will require more language pre-processing. Such processing can be particularly challenging because tweets are short, and make extensive use of a quickly evolving cultural short-hand that influences lexicon and grammar both. One possible technique we can employ is latent semantic analysis. Using LSA on a bag-of-words model of Tweets, we can possibly extract predicates that are word-position independent and capture additional ways of expressing “is down” in an unsupervised manner.

Second, we would like to apply our techniques to a real-time stream of messages, to provide users with up-to-date status information about their frequented sites.

The last issue where we see a need is in maintaining state, both about the lexical conventions of individual users as well as capturing which users are typically “primary reporters” vs. users who simply provide “hearsay” tweets. The first represents a real sample of a user’s experience while the latter is simple a repetition of someone else’s information (indeed, we have seen tweets reporting that they have heard a service “is down” even after it is back up).

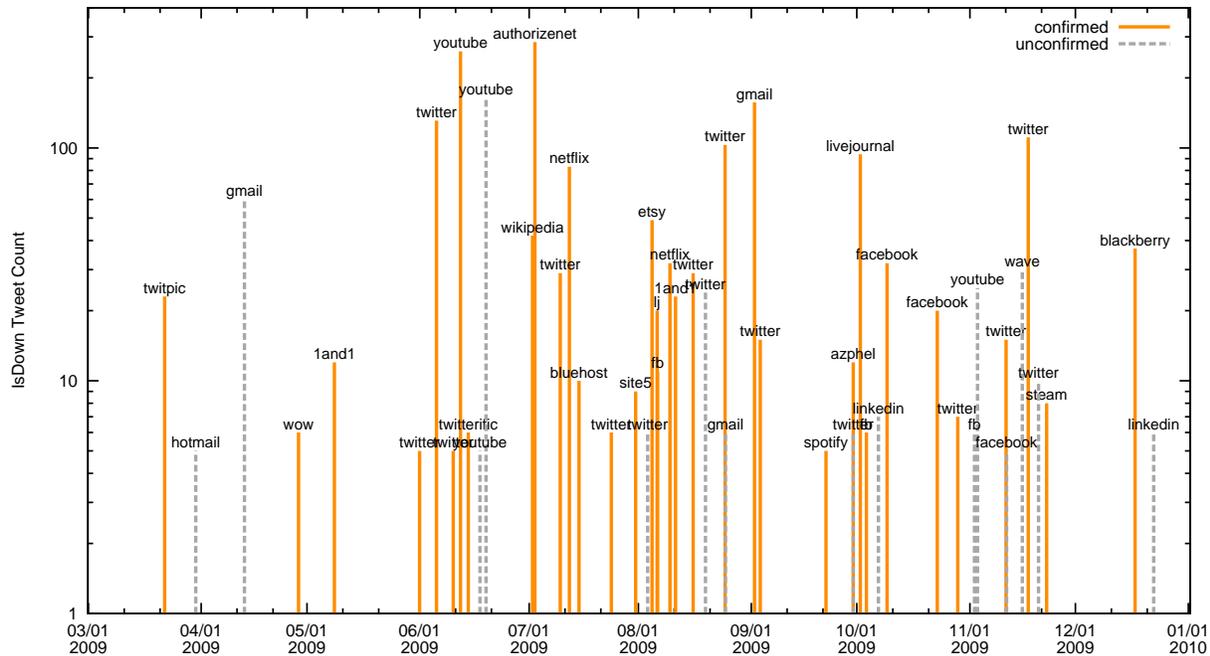


Figure 3: Fifty random outages as measured by the volume of tweets containing *IsDown*.

Entity Name	Detected Time (GMT)	Source
Gmail	2008-08-11 21:34:22	[30]
Gmail	2009-02-24 10:19:22	[20]
Facebook	2009-04-07 15:04:35	[2]
Gmail	2009-04-16 19:44:22	[12]
Gmail	2009-05-08 10:09:22	[4]
Google	2009-05-14 15:03:49	[48]
Gmail	2009-05-14 15:04:22	[49]
Twitter	2009-06-16 14:34:48	[43]
YouTube	2009-07-09 01:16:58	[10]
YouTube	2009-07-12 11:26:58	[34]
Twitter	2009-08-06 13:19:48	[44]
Twitter	2009-08-07 11:24:48	[46]
YouTube	2009-08-11 01:41:58	[16]
Twitter	2009-08-15 17:19:48	[47]
Facebook	2009-08-19 12:04:35	[6]
Gmail	2009-09-01 19:49:22	[50]
Apple Store	2009-09-09 15:03:48	[42]
Facebook	2009-09-19 03:44:35	[31]
Gmail	2009-09-24 14:19:22	[39]
YouTube	2009-10-03 16:16:58	[41]
Twitter	2009-10-08 14:54:48	[45]
YouTube	2009-11-01 17:41:58	[3]
TMobile	2009-11-03 23:29:26	[18]
Facebook	2009-11-23 22:39:35	[25]
bbm	2009-12-22 20:51:18	[54]

Table 3: Top 25 service outages, sorted chronologically.

We believe the most significant research opportunities revolve around identifying new applications that exploit

the real-time nature of Twitter. What kinds of questions about the world at large can be answered via mining this modality? It is here that we expect to focus most of our future effort.

References

- [1] AIONSOURCE. Azphel. http://www.aionsource.com/topic/66323-azphel-again/page__p__1399400&#entry1399400?s=a55a586be192be9d091e9468b794adcb.
- [2] ANSWERS, M. Was Facebook down on April 7, 2009? <http://www.mahalo.com/answers/technology-and-internet/was-facebook-down-on-april-7-2009/>.
- [3] ARRINGTON, M. Youtube down for maintenance. right. <http://www.washingtonpost.com/wp-dyn/content/article/2009/11/01/AR2009110101390.html>.
- [4] BARAK, S. Gmail down worldwide. <http://www.theinquirer.net/inquirer/news/1137250/gmail-worldwide>.
- [5] BESTHOSTRATINGS. 1and1. <http://uptime.besthostratings.com/viewmonth.php?host=1and1&month=20098>.
- [6] BLOG, X. Facebook: Service unavailable- DNS failure. <http://blog.xanda.org/2009/08/20/facebook-service-unavailable-dns-failure/>.
- [7] BOULTON, C. Latest Google Gmail outage caused by insufficient router capacity. <http://www.eweek.com/c/a/Messaging-and-Collaboration/Latest-Google-Gmail-Outage-Caused-By-Insufficient-Router-Capacity-180014>.
- [8] BRUTLAG, J. Aberrant behavior detection in time series for network monitoring. In *Lisa 2000* (2000).
- [9] CHAN, S. P. Microsoft bing outage this morning. http://seattletimes.nwsources.com/html/microsoftpri0/2010424771_microsoftbingoutagethismorning.html.

- [10] CHAT, S., AND BLOG. Oops! YouTube down for maintenance? for how long? <http://www.somchat.com/2009/07/oops-youtube-down-for-maintenance-for.html>.
- [11] CLABURN, T. Google service outage stirs tweet storm. <http://www.informationweek.com/news/internet/google/showArticle.jhtml?articleID=217500195>.
- [12] CNET. Gmail outage afflicts some users. http://news.cnet.com/8301-17939_109-10221616-2.html.
- [13] DUNFORD, K. Journalists increasingly utilize social media sites. <http://media.www.gwhatchet.com/media/storage/paper332/news/2010/02/04/News/Gspm-Study.Journalists.Increasingly.Utilize.Social.Media.Sites-3864656.shtml>.
- [14] ERIC. Photo sharing service flickr down amid server issues. <http://erictric.com/2009/10/20/photo-sharing-service-flickr-down-amid-server-issues/>.
- [15] FLICKR. Flickr blog. <http://blog.flickr.net>.
- [16] FORUM, D. Youtube uploads currently unavailable. <http://drupal.org/node/545100>.
- [17] FORUMS, W. Wow. <http://forums.worldofwarcraft.com/thread.html?topicId=16474244539&sid=13>.
- [18] FRIED, I. T-Mobile experiencing widespread outage. http://news.cnet.com/8301-13860_3-10390097-56.html.
- [19] GAUDIN, S. Twitter now has 75m users. http://www.computerworld.com/s/article/9148878/Twitter_now_has_75M_users_most_asleep_at_the_mouse.
- [20] GMAILBLOG. Gmail. <http://gmailblog.blogspot.com/2009/02/update-on-todays-gmail-outage.html>.
- [21] GOLEMLABS. Bluehost. <http://www.golemlabs.com/community/showthread.php?t=25108>.
- [22] JANSEN, B. J., ZHANG, M., SOBEL, K., AND CHOWDURY, A. Twitter power: tweets as electronic word of mouth. In *Journal of the American Society for Information Science and Technology* (2009).
- [23] JAVA, A., SONG, X., FININ, T., AND TSENG, B. Why we Twitter: understanding microblogging usage and communities. In *Proc. of the 9th WebKDD and 1st SNA-KDD 2007 Workshop* (2007).
- [24] JOURNAL, L. http://community.livejournal.com/lj_maintenance/125681.html.
- [25] KAR, M. Facebook server problem annoy users. http://www.thaindian.com/newsportal/tech-news/facebook-server-problem-annoy-users_100279206.html.
- [26] KOBIE, N. Google opens up about reason for yesterday's gmail outage. <http://www.itpro.co.uk/614602/updated-gmail-knocked-offline-again>.
- [27] KRISHNAMURTHY, B., GILL, P., AND ARLITT, M. A few chirps about Twitter. In *Proc. of 1st WOSN* (2008).
- [28] LIVEJOURNAL. Livejournal blog. http://community.livejournal.com/lj_maintenance.
- [29] MONTALBANO, E. Microsoft restores service after hotmail outage. <http://www.itworld.com/saas/64218/microsoft-restores-service-after-hotmail-outage>.
- [30] NEEDLEMAN, R. Gmail is down, Twitter sizzling with the news. http://news.cnet.com/8301-17939_109-10014389-2.html.
- [31] NEWSNIDEA. Facebook outage and Facebook down September 18. <http://newsnidea.com/14912/facebook-outage-and-facebook-down-september-18-2009/>.
- [32] NING. Ning blog. <http://status.ning.com>.
- [33] PCWORD. Rackspace. http://www.pcworld.com/businesscenter/article/185171/rackspace_outage_has_limited_impact.html.
- [34] PROJECT, T. D. YouTube down? <http://thedaneshproject.com/posts/youtube-down/>.
- [35] RAPHAEL, J. Gmail outage marks sixth downtime in eight months. <http://abcnews.go.com/Technology/PCWorld/story?id=6951480>.
- [36] SAKAKI, T., OKAZAKI, M., AND MATSUO, Y. Earthquake shakes Twitter users: real-time event detection by social sensors. In *WWW 2010* (2010).
- [37] SANKARANARAYANAN, J., SAMET, H., TEITLER, B., LIEBERMAN, M., AND SPERLING, J. Twitterstand: News in tweets. In *ACM GIS 2009* (2009).
- [38] SCOTT, J. Wikipedia site recovers from crash. <http://www.itpro.co.uk/613423/wikipedia-site-recovers-from-crash>.
- [39] SHANKLAND, S. Gmail outage 'hits small subset of users'. http://news.cnet.com/8301-30685_3-10360729-264.html.
- [40] SHANKLAND, S. Paypal suffers from e-commerce outage. <http://www.zdnetasia.com/news/internet/0,39044908,62056619,00.htm>.
- [41] STERN, A. YouTube down for maintenance. <http://www.cenetworks.com/youtube-down-for-maintenance>.
- [42] TIPB. Apple store down! 2009 special music event nigh! <http://www.tipb.com/2009/09/09/apple-store-2009-special-music-event-nigh/>.
- [43] TWITTER BLOG, O. Down time rescheduled. <http://blog.twitter.com/2009/06/down-time-rescheduled.html>.
- [44] TWITTER BLOG, O. Ongoing denial-of-service attack. <http://status.twitter.com/post/157191978/ongoing-denial-of-service-attack>.
- [45] TWITTER BLOG, O. Recovering from unplanned downtime. <http://status.twitter.com/post/207018761/recovering-from-unplanned-downtime>.
- [46] TWITTER BLOG, O. Restoring API and SMS. <http://status.twitter.com/post/157979213/restoring-api-and-sms>.
- [47] TWITTER BLOG, O. Working on unexpected downtime. <http://status.twitter.com/post/163603406/working-on-unexpected-downtime>.
- [48] USATODAY. Usa can't hold on against brazil in confederations final. <http://news.cnet.com/widespread-google-outages-rattle-users/>.
- [49] VAUGHAN-NICHOLS, S. Google down! http://blogs.computerworld.com/google_down.
- [50] WAUTERS, R. Gmail hitting some turbulence. <http://techcrunch.com/2009/09/01/gmail-hitting-some-turbulence/>.
- [51] WEBMASTERWORLD. Youtube. <http://www.webmasterworld.com/google/3932315.htm>.
- [52] WEIL, K. Measuring tweets. <http://blog.twitter.com/2010/02/measuring-tweets.html>.
- [53] YE, N., VILBERT, S., AND CHEN, Q. Computer intrusion detection through ewma for autocorrelated and uncorrelated data. In *IEEE Transactions On Reliability* (2003).
- [54] ZIEGLER, C. BlackBerry services down in North America yet again? <http://www.engadget.com/2009/12/22/blackberry-services-down-in-north-america-yet-again/>.
- [55] ZIMMER, B. How fail went from verb to interjection. *New York Times* (August 7, 2009).