GrapeVine : Tracking the Pulse of Businesses using Twitter

Arpan K Ghosh

A Dissertation
Presented to the Faculty
of Princeton University
in Candidacy for the Degree
of Master of Science in Engineering

Recommended for Acceptance
by the Department of
Computer Science
Adviser: Professor Mung Chiang

June 2013

Abstract

Twitter is no longer simply a social network but a 'social-information network' used for sharing information, ideas and opinions on matters of mass importance. This is especially true in the consumer-business sector where companies are leveraging it to reach out and 'talk' to their consumer base at an unprecedented scale and speed. However, the low signal to noise ratio and the lack of structured data in Twitter's data stream makes it much harder to 'listen' to what consumers are saying about the company: their opinions, feedback and sentiment. Almost half of all tweets are personal conversations, users' self-promotion, random observations or spam, which are not of any use for a business-oriented use case. Moreover, Twitter does not formally attempt to organize tweets based on any contextual or categorical information, nor does it collect any detailed information about its users beyond an optional, short textual description. We propose the design of a Twitter-based consumer-business tool which can help companies bridge this communication gap and better monitor and analyze in near real-time, the opinion, sentiment and feedback that exists about them amongst the Twitter consumer base. We implement and evaluate the following basic features of this tool: 1) a classifier which distinguishes between 'relevant' and 'irrelevant' tweets, from a business perspective, and filters the data stream, 2) an algorithm to discover the 'top tweeters' mentioning a company and rank them based on their expertise in the company's domain and 3) an algorithm to detect trending stories pertaining to a company.

Acknowledgements

I would like to express my heartfelt gratitude to my adviser Prof. Mung Chiang for his invaluable mentorship, his confidence in me to pursue ambitious ideas and his patience through all the twists and turns this project has taken.

I would not have ventured down this path had it not been for my parents who have always led by example and set a high standard and instilled in me a love for scientific thinking. My younger brother Arnav deserves a lot of credit as he has taught me to truly value every opportunity that I have been fortunate enough to be blessed with. My wife Renuka, or 'co-advisor' as I like to call her, kindled this dream two years ago and since then has shared my passions and provided keen insight and timely encouragement every step of the way, even though she's working on the other side of the country. To my family, a big Thank You.

This thesis would also not be possible without the love and support of my New Jersey and New York based extended family and friends who gave me a home away from home and much needed R&R when the workload was too overwhelming.

And last, but not the least, a shout-out to the fine folks at WibiData. Working with nascent, open-source software for the first time can be baptism by fire. However, they never let me realize that I was not paying for premium support.

To my family.

Contents

Abstract . . . iii

Acknowledgements . . . iv

List of Figures . . . viii

1 Introduction **1**

1.1 Twitter: From Social to Social Information Network . . . 1

1.2 Using Twitter as a Consumer Business Tool . . . 2

1.3 Limitations of Twitter as a Consumer Business Tool . . . 2

1.4 Why Raw Twitter Data is Unsuitable for Business Applications . . . 4

1.4.1 Lack of Insightful Structured Data . . . 4

1.4.2 Informal Linguistic Style of Unstructured Textual Data . . . 5

1.4.3 Lack of Context in Tweets . . . 5

1.4.4 A Large Part of Twitter is Social, Spam or Foreign . . . 6

1.4.5 Twitter Data is Huge . . . 7

1.5 Research Objectives and Contributions . . . 7

2 Related Work **14**

3 The Data we Collect from Twitter **16**

3.1 Tweets Mentioning a Company . . . 17

3.2 Tweets from Users in Twitter's 'Suggested Categories' . . . 17

3.3 Random Tweets from 1% of the Twitter Firehose . . . 18

4 Handling Twitter Data at Scale **19**

4.1 Storing Tweets at Scale . . . 20

4.2 Collecting Tweets at Scale . . . 21

4.3 Analyzing Tweets at Scale . . . 22

5 Filtering the Twitter Stream by Relevance **24**

5.1 How Noisy is Our Dataset? 25

5.2 Selecting Tweet Features for Relevance Classification 26

5.2.1 Twitter and Tweet Specific Features 26

5.2.2 Linguistic Features of Tweet Text 27

5.3 Choosing A Classification Algorithm 28

5.4 Accuracy of Relevance-Based Tweet Classification 29

6 Extracting Consumer Insight From Twitter **32**

6.1 User Tweeting Patterns while Mentioning Businessess 32

6.2 Determining User Expertise Implicitly through Tweeting Patterns 33

6.3 Determining User Expertise Explicitly from Tweet Text 34

7 Detecting Business-Related Trending Stories and Events **39**

7.1 Using Dictionary Learning for Identifying Emerging Stories 39

7.2 Our Modified Dictionary Learning Algorithm 40

7.3 Empirical Evaluation on Companies' Tweets 42

8 Future Work **46**

9 Summary and Conclusion **48**

Bibliography **50**

List of Figures

1.1 Twitter followings of consumer companies in various verticals 3

1.2 Daily volume of tweets mentioning Amazon, Blackberry and McDonalds. The horizontal axis is the day of the year. The purple bar represents the fraction of total tweets that were re-tweets. Our tweet collecting process was down on days 103-105 . 10

1.3 A huge spike in the daily tweet volume graph for Toyota illustrates how users tweet about a company with increased intensity when there is negative news about it . . . 11

1.4 Graphs displaying the daily and weekly overall tweet volume received by our tweet collector processes during our month-long collection period. Our collectors were down on days 103-106 12

1.5 Graph displaying the overall hourly tweet volume for a subset of our collection period. Notice the spikes occuring around 8pm and Noon on a daily basis 13

4.1 The various components of our framework and how they interact with each other . . 20

4.2 A breakdown of how we can chain together multiple collectors to fetch tweets that require pulling data from multiple Twitter API endpoints 23

5.1 Post-labeling distribution of the relevant, not-relevant and not-English tweets present in the mini-datasets representing the three types of tweets we collect 25

5.2 The Twitter and tweet specific features of each tweet that we obtained from Twitter or are derived by examining the tweet text 28

5.3 A list of the POS tags supported by the Twitter POS tagger. We ignored the 5 *Twitter/online-specific* tags as they are already accounted for in our first set of Twitter and tweet specific features 29

5.4 Relevance classification results based on the Twitter and tweet-specific features. Data_A is the mini-dataset containing the curated users' tweets, data_B is the mini-dataset containing the tweets mentioning companies and data_C is the mini-dataset containing the tweets from 1% of the Twitter firehose 30

5.5 Relevance classification results based on the POS features. Data_A is the mini-dataset containing the curated users' tweets and data_B is the mini-dataset containing the tweets mentioning companies 31

6.1 The long-tail distribution of how many times individual users mention a company: shown here for Samsung and Chase 36

6.2 The top 4 tweeters for Delta Airlines, when they are simply ranked by how many times they mentioned it during our collection period 37

6.3 The top four tweeters for the company Delta Airlines, after we clean up the list of users mentioning it and rank them by their expertise in the company's vertical . . . 37

6.4 Classification accuracy for the 'News' topic while using the SVMs algorithm 38

7.1 The pseudo code describing our dictionary learning based algorithm for trending-story detection 42

7.2 True positive trending stories detected for companies that actually correspond to a completely different event or person because of ambiguity in the company name . . . 43

7.3 True Positive trending stories detected by our modified dictionary learning algorithm 44

7.4 Word clouds of the most frequently occurring words in Toyota related tweets on April 10th (top) and April 11th (bottom). This illustrates the change in tweet content when the trending story about Toyota's airbag recall surfaced on April 11th 45

Chapter 1

Introduction

1.1 Twitter: From Social to Social Information Network

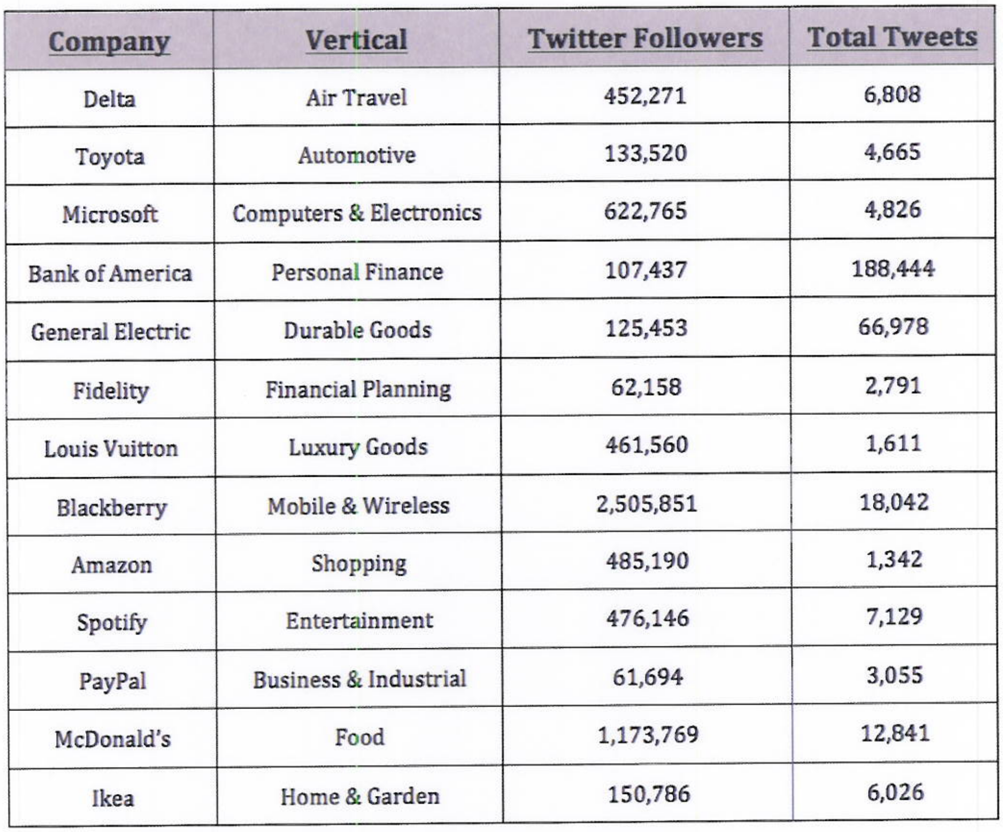
In its early days, Twitter's sole utility lay in being a social network which allowed users to 'follow' the lives of their friends and famous celebrities through frequently published tweets or short, textual messages, the size of a SMS. Tweets came to be known as 'status updates' as they described the publishing user's mood, thoughts, observations or current activity for the most part. However, in the last couple of years, while social interaction and personal promotion remain strong motivators for tweeting, Twitter has grown into what can be called a 'social information network'. Tweets are increasingly being used to share articles on the web, news or otherwise, by simply specifying a URL and a short description. According to statistics released by Twitter in 2010, 25% of the 90 million tweets published daily, contained URLs [20]. Hashtags are short, '#' prefixed strings consisting of a single word or the words of a phrase concatenated together, which enable users to informally label their tweets as pertaining to a certain topic and enable Twitter to group tweets by them to make tweets easily searchable. In this way hashtags have contributed to an increase in the volume of informational tweets, not only ones sharing URLs but also ones containing facts or the publishing user's opinions about issues, objects and happenings of importance to an audience much larger than their social circle. E.g. most tweets concerning the 'Occupy Wall Street' protests in 2011 were labeled with the hashtag #occupy. The shift towards more informational tweet content has also been accompanied by the widespread emergence of what we term 'professional tweeters', or Twitter users who specialize in solely publishing informational content. This, for the most part, includes the usual disseminators of information through traditional media, like journalists, analysts, broadcasters and bloggers who have embraced Twitter as an important and often quicker medium to reach their audience. It also extends beyond individuals to include organizations who have started using Twitter to keep users, interested in them, updated with the latest information concerning the organization. 'Professional tweeters' typically have some of the largest Twitter followings, or the number of users following their accounts, second only to famous celebrities.

1.2 Using Twitter as a Consumer Business Tool

One of the more recent and interesting 'professional' uses of Twitter is by companies in the consumer business sector. While these businesses continue to use traditional broadcast media like television, magazines and newspapers to reach out to people in general, Twitter provides them with a medium to engage a more focused audience: people who are already interested in them and are following their Twitter account. Twitter allows companies to reach out to millions of users in seconds with announcements, deals, promotions, advertisements, contests and pretty much anything they would normally do for maintaining customer loyalty. The dual benefit for the companies and customers has led to the widespread adoption of Twitter by consumer businesses and them also gaining significant followings. The table in Figure 1.1 shows the large number of Twitter users following consumer companies in various verticals.

1.3 Limitations of Twitter as a Consumer Business Tool

Twitter accounts owned by businesses aren't the only ones that publish tweets mentioning companies. Regular users talk about businesses in their tweets on a daily basis. Figure 1.2 shows the daily volume of tweets mentioning three different companies, collected over a month. The tweets were collected using Twitter's *filter* streaming API endpoint, which returns public tweets that match specified string predicates, company names in our case. Looking at the total number of tweets published by the official company accounts in Figure 1.1, we can deduce that the bulk of the tweet volume in Figure 1.2 is generated by regular users. It is clear that Twitter users talk about businesses a decent amount on any regular day. However, they do so in significantly higher volumes when there is negative or disturbing news about a company. Figure 1.3 illustrates how the daily tweet volume for Toyota spikes up on April 11th when there is an announcement of a massive recall due to faulty airbags.



Company	Vertical	Twitter Followers	Total Tweets
Delta	Air Travel	452,271	6,808
Toyota	Automotive	133,520	4,665
Microsoft	Computers & Electronics	622,765	4,826
Bank of America	Personal Finance	107,437	188,444
General Electric	Durable Goods	125,453	66,978
Fidelity	Financial Planning	62,158	2,791
Louis Vuitton	Luxury Goods	461,560	1,611
Blackberry	Mobile & Wireless	2,505,851	18,042
Amazon	Shopping	485,190	1,342
Spotify	Entertainment	476,146	7,129
PayPal	Business & Industrial	61,694	3,055
McDonald's	Food	1,173,769	12,841
Ikea	Home & Garden	150,786	6,026

Figure 1.1: Twitter followings of consumer companies in various verticals

Given these numbers, it stands to reason that simply looking at a company's official account paints a highly incomplete picture about its presence on Twitter. A company should have the ability to 'listen' to what regular Twitter users are saying about it. We define 'listening' to be a set of traditional market research and analysis tasks listed below, but enhanced by the scale and real-time nature of Twitter, as listed below:

- A company should be able to identify tweets containing 'relevant' mentions of itself, its products and brands and also the users who are publishing said tweets. We define a relevant tweet here, as one that actually contains information or opinion about a company as opposed to one that just happens to mention the company's name as part of a personal message or conversation or completely randomly. Filtering the stream to only include relevant tweets improves the quality of any further analysis. In Chapter 5 we explain how almost half of all tweets can be considered irrelevant from a consumer-business standpoint. Filtering out irrelevant tweets also yields significant savings in processing and storage costs.

- A company should be able to identify the top tweeters mentioning it, not just by tweet quantity but also by the quality of their tweets and the quality of their Twitter following. This opens the door to true social marketing as a company can leverage these influential users to effectively build product awareness, brand buzz and new sales. This requires that the influential users also have expertise or strong interest in the vertical that the company operates in. Therefore, a company should be able to gain latent insight about the users mentioning it, specifically their topics of interest on Twitter, so that experts in its domain(s) can be focused on.
- Companies should be the first ones to detect trending stories about themselves, especially negative ones, for a proactive response. It has been shown that major news stories are increasingly breaking faster on Twitter than conventional news sources [15]. More importantly, identifying tweets that are part of a trending story also enables companies to gauge consumer sentiment and reactions as the story unfolds rather than after it.

While Twitter makes it very easy for businesses to reach out and 'talk' to customers following them, 'listening' to regular users is easier said than done. Twitter potentially has all the data required to accomplish these tasks and can be an extremely valuable consumer-business data source, but its communication paradigm and style of content pose unique challenges that we analyze in the next section.

1.4 Why Raw Twitter Data is Unsuitable for Business Applications

Twitter provides a variety of structured information about each published tweet and the user publishing it along with one really important piece of unstructured data: the text of the tweet. More details about the attributes describing tweets and users is provided in Chapter 3. On paper Twitter is a very big, largely public and fresh source of market and consumer data, but the following reasons somewhat hamper its effectiveness and suitability for business oriented applications and make it hard to work with:

1.4.1 Lack of Insightful Structured Data

The structured data about users and tweets, which is easy to extract from a Twitter dataset, is not very insightful, especially for the consumer-business use case at hand. During account creation

Twitter does not ask a user to specify anything more than their name and a description, which is unstructured textual data. The rest of the fields describing a user pertain to the Twitter account, like time of account creation and whether the account is public or protected, rather than the human user. Information about the users 'followed' by a specific user (and equivalently the list of people 'following' said user) should be able to convey something about a user's interests but it is simply presented as a list of IDs, which cannot yield any insight in isolation. The most useful fields of a tweet object are the ones listing the specialized entities found in the tweet, namely hashtags, URLs and mentions of other users. However, the strings that make up the hashtags and URLs are once again unstructured text without any additional labels to categorize them. The user mentions can help one fetch more user objects which as already discussed are devoid of insightful structured data. Additional structured fields in the tweet object include the time when the tweet was published and the coordinates of the publishing location. The latter could be quite useful from a business standpoint but is optional and was absent from almost all of the tweets we collected.

1.4.2 Informal Linguistic Style of Unstructured Textual Data

The text of each tweet, which includes textual entities like hashtags and URLs, is the unstructured data that will primarily be used for fulfilling the tasks of this consumer business use case. In spite of Twitter's recent increase in informational content, it is primarily a social network leading to an informal style of authorship, even by 'professional tweeters' and also in tweets mentioning businesses. Actual entity names are often replaced by their colloquial or slang names, e.g. the Huffington Post is called '*huffpo*'. The 140-character word limit leads to the generation of various unofficial abbreviations to refer to entities, e.g. 'BOFA' for Bank of America. Moreover, since Twitter embodies publishing quickly and frequently, users do not place a great deal of importance on correct spelling. Therefore, the list of string predicates that the Twitter API must track in order to collect tweets talking about a particular company requires constant manual curation and is still not guaranteed to return all such tweets.

1.4.3 Lack of Context in Tweets

Twitter does not attempt to officially categorize tweets at any granularity to provide them with some context based on the topic or otherwise. Hashtags are an informal method employed by users to provide their tweets with context by associating them with a trend, story or an event. However, there is no central authority, which formally assigns hashtags to topics or stories. This leads to the parallel existence of multiple hashtags while a story is developing with one of them eventually gaining enough critical mass to dominate and represent the trend. Since being able to detect stories about companies early on is a desired feature, hashtags cannot be used to conclusively provide context for tweets. Hashtag generation is also subject to the informal linguistic style used across Twitter. They can range from simply being the name of the company referred to in the tweet (which would be desirable) to complete English phrases, making them unsuitable as finite sets of category or company labels for tweets. Previous research has shown that hashtags are extremely short-lived, as 40% of hashtags used on any given day have not been used in the prior 30 days [19]. Because of this ephemeral nature and the entropy involved in their generation, hashtags cannot be used to reliably provide context for tweets in our use case, which involves long-running tweet collection and analysis. The lack of context leads to issues of ambiguity about whether a tweet is referring to a business or something completely unrelated which simply shares the businesses name, the most famous example being 'apple'. Since the Twitter API basically performs a case-insensitive pattern match for the specified string predicates against all tweets, the lack of context creates similar ambiguity issues for companies with short names (e.g. Ford) or those better known by their abbreviations, as these short strings can easily appear in tweets that have nothing to do with the respective companies. E.g. when we started our tweet collection, it looked like 'hp', the computer manufacturer, was one of the most discussed companies on Twitter, however we soon realized that the pattern 'hp' appears in 78 English words and countless shortened URL hashes.

1.4.4 A Large Part of Twitter is Social, Spam or Foreign

We only wish to analyze relevant tweets, as defined in Section 1.3, for this consumer business use case. According to official Twitter statistics released in 2011, 87% of all tweets can be categorized as the following [14]:

- Pointless Babble
- Conversational
- Self-promotion
- Spam

This only leaves 13% of the 350 million daily tweet volume to qualify as relevant content. While this is still a sizeable amount of data to work with, the signal to noise ratio of our data stream is very low and definitely requires some cleanup before any analysis can take place. Additionally, the same statistics reveal that non-English speaking countries make up about half of Twitter's global user base resulting in a significant fraction of non-English tweets. We quantify this by analyzing our collected tweets in Chapter 5. These tweets will end up matching the string predicates we track using the Twitter API, as most company names do not change across languages, but will be useless for any further analysis.

1.4.5 Twitter Data is Huge

Most recent statistics from Twitter put the global daily tweet volume at 400 million [9]. Our consumer business use case certainly does not require collecting and analyzing the entire Twitter firehose, but the daily volume of tweets mentioning businesses are high enough to get the conversation about distributed storage and processing techniques started, which are non-trivial to implement. Figure 1.4 shows the daily and weekly tweet volumes during our month-long collection period. On average we collect more than 100,000 tweets per hour, 2.7 million tweets per day and 17 million tweets per week. The hourly tweet volumes for a subset of the collection period are shown in Figure 1.5. We noticed a consistent pattern in the hourly tweet volumes during a day. The highest tweet volumes are observed between 7 and 9pm and a second smaller spike is observed around Noon. Intuitively this seems to correspond to when users would have free time to tweet during the day.

1.5 Research Objectives and Contributions

This work focuses on developing the building blocks of a Twitter-based consumer business tool which can overcome the challenges presented by Twitter's as a data stream, as described in Section 1.4, and use it to accomplish the tasks of better understanding and listening to a Twitter based consumer base, as defined in Section 1.3. Our research objectives are as follows:

- Collect a sizeable dataset of tweets containing mentions of businesses.
- Utilize state of the art 'Big Data' infrastructure and techniques to store and analyze our tweet dataset at scale.
- Explore techniques to sanitize the collected dataset by distinguishing between relevant and irrelevant tweets.
- Explore techniques to rank the users talking about a company by their expertise in the company's domain.

- Explore techniques to categorize tweets into broad topics based on their content and generate topic profiles for users to substantiate our knowledge of their expertise.
- Explore techniques to detect trending stories about businesses on Twitter.

We began by building a scalable, distributed and fault-tolerant framework for collecting, storing and analyzing tweets. Using the tweet collectors we implemented in this framework, we were able to collect approximately 100 million tweets mentioning a set of 70 companies that we chose to track across 14 verticals. Our framework allowed us to use multiple machines to store replicated segments of our 600 GB dataset to eliminate a single point of failure. This also prevented a single big disk from becoming an I/O bottleneck and allowed us to use multiple machines to parallelize the analysis of our dataset. The next step involved cleaning up the dataset to only consider relevant tweets. We used Amazon's Mechanical Turk to label a sample of 12000 tweets from our dataset as either containing or lacking relevant content. These were used to train various classifier algorithms to distinguish between relevant and irrelevant tweets by using attributes describing the circumstances in which the tweet was published, attributes describing the type of content present in the tweet text and attributes which describe the tweet text from a linguistic and grammatical point of view. Following the data cleanup, we focused on determining, both implicitly and explicitly, the area of expertise of users talking about a company in order to highlight those users who are experts or strongly interested in the same domain as the company. The implicit approach involved selecting users who also mentioned a company's nearest neighbors and ranking them by the average of their tweet volumes across the companies. For the explicit approach we attempted to categorize users' tweets amongst a finite set of broad topics and generate a profile of their topics of interest. We also developed an algorithm for detecting trending Twitter stories pertaining to a company. It is based on Dictionary Learning and checks for the presence of novel tweet content occurring in high volume. Our contributions in this work are listed below. We developed:

- A distributed and fault-tolerant framework for collecting, storing and analyzing Twitter data, which enabled us to collect 100 million tweets with mentions of businesses and perform operations like aggregations, in parallel, on this dataset in under 2 hours using a very basic cluster of 3 desktop machines.
- A filter that can distinguish between relevant and irrelevant tweets, as defined by our specific use case, with 78% accuracy. Apart from cleaning up our dataset, this also provides big

processing cost savings by significantly bringing down the size of our working set of tweets for any further analysis.

- An algorithm to rank the Twitter users talking about a company by their expertise or strong interest in the company's domain.
- A modified Dictionary Learning based algorithm, which successfully detects business-related emerging stories in a stream of tweets mentioning a company.

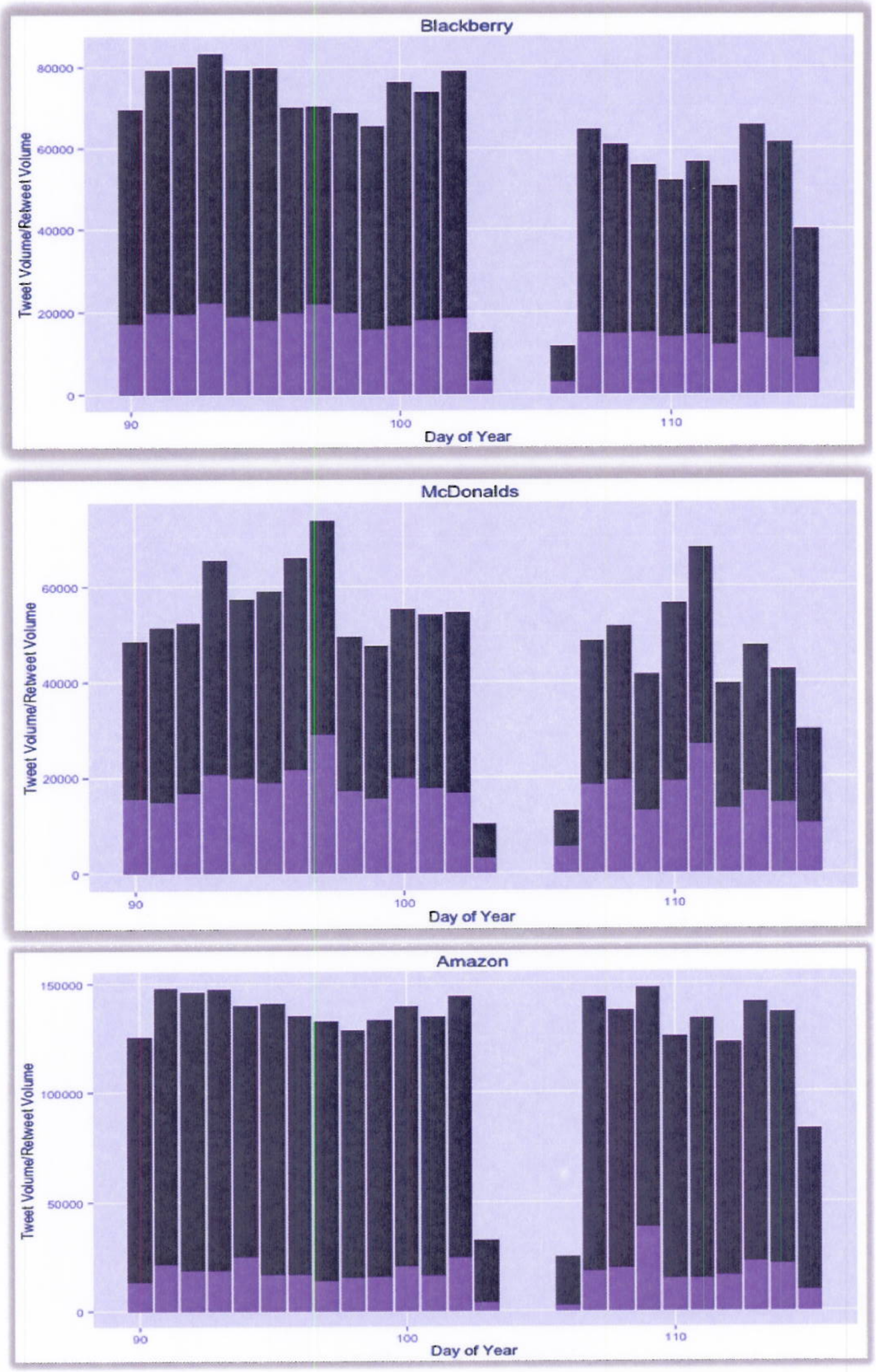
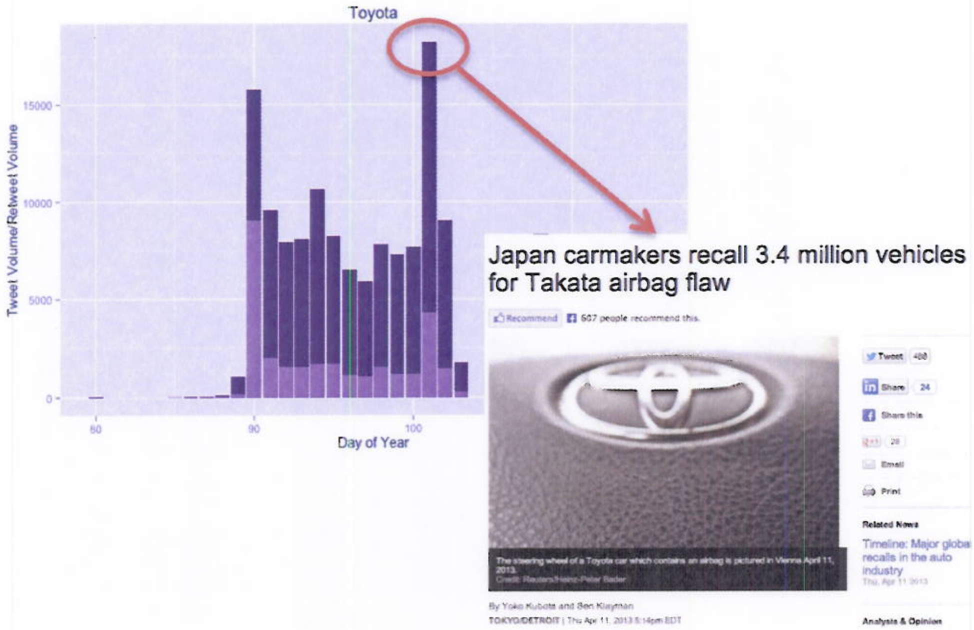


Figure 1.2: Daily volume of tweets mentioning Amazon, Blackberry and McDonalds. The horizontal axis is the day of the year. The purple bar represents the fraction of total tweets that were re-tweets. Our tweet collecting process was down on days 103-105



Japan carmakers recall 3.4 million vehicles for Takata airbag flaw

The steering wheel of a Toyota car which contains an airbag is pictured in Vienna April 11, 2013.
Credit: Reuters/Heinz-Peter Bader

By Yoko Kubota and Ben Klayman
TOKYO/DETROIT | Thu Apr 11, 2013 5:14pm EDT

Figure 1.3: A huge spike in the daily tweet volume graph for Toyota illustrates how users tweet about a company with increased intensity when there is negative news about it

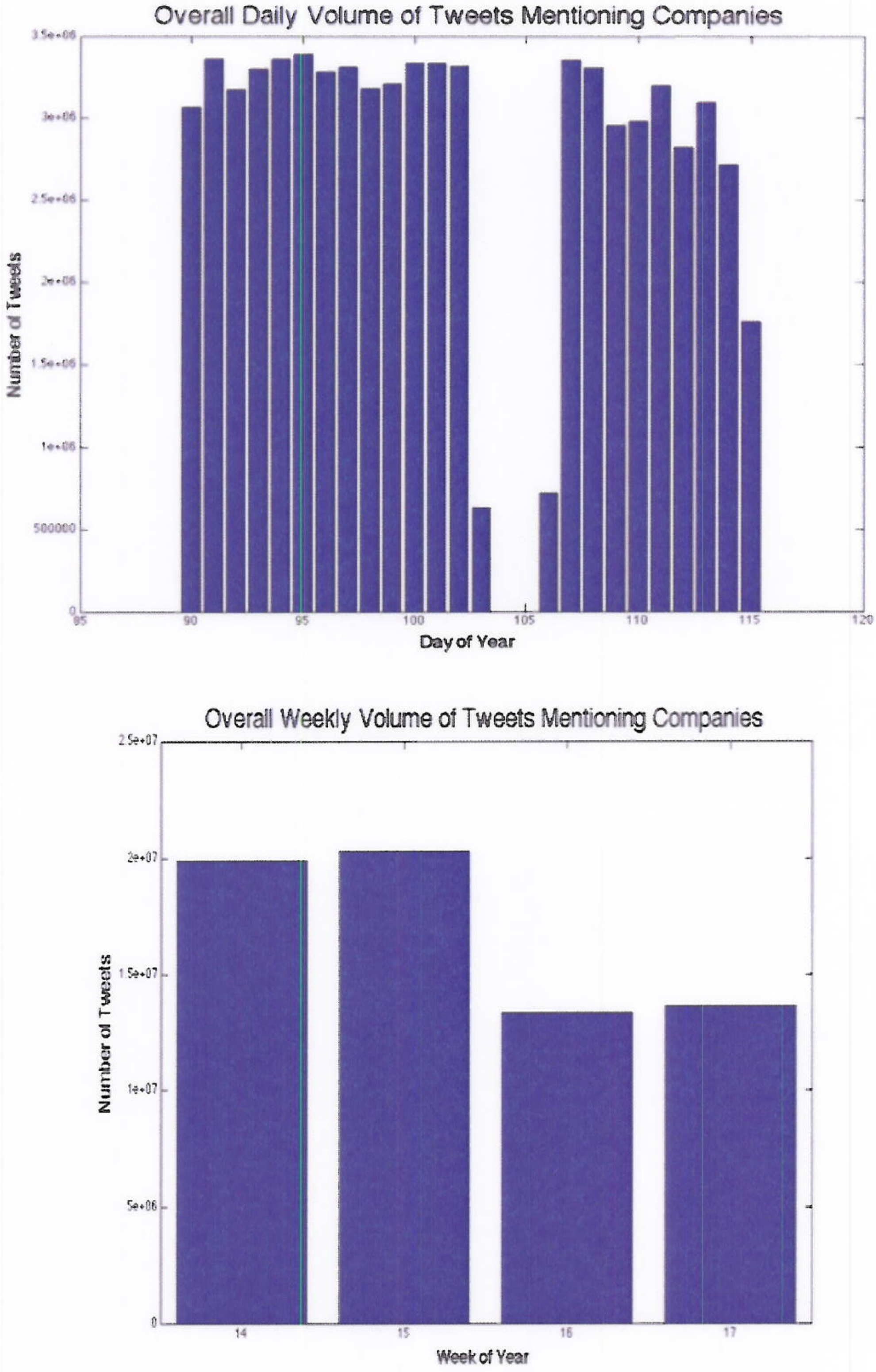


Figure 1.4: Graphs displaying the daily and weekly overall tweet volume received by our tweet collector processes during our month-long collection period. Our collectors were down on days 103-106

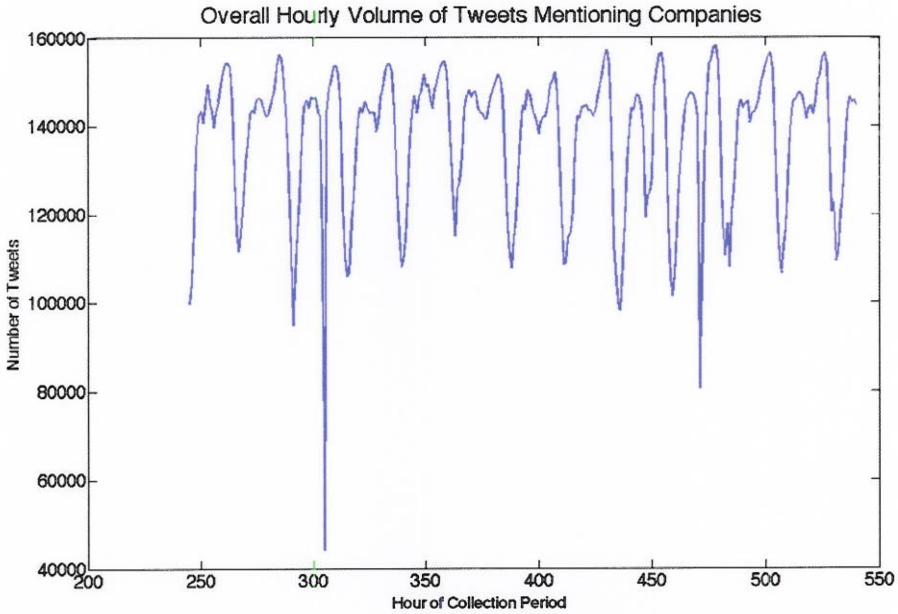


Figure 1.5: Graph displaying the overall hourly tweet volume for a subset of our collection period. Notice the spikes occuring around 8pm and Noon on a daily basis

Chapter 2

Related Work

Most of the related work that we surveyed does not explicitly focus on cleaning up the Twitter stream or dataset prior to analyzing it. Xu et al. classify tweets as 'interest-related' or 'interest-unrelated' and ignore 'interest-unrelated' during any subsequent analysis [42]. However, their method of classification is a little naive. They manually label 1000 tweets with the two categories and then calculate the probability of a tweet being in either category given that it is a re-tweet, it is a reply, it contains a link or it contains a hashtag. These probabilities, conditioned on the presence of the four features, are used to classify the rest of the dataset. In [25] Ramage et al. use LDA to divide a Twitter dataset into 200 topic groups, which are supposed to be used as topic labels for the tweets. They also use 304 additional labels describing the properties of a tweet like whether it contains hashtags, or whether it contains a question etc. Each tweet can be described using a subset of these 504 labels and the authors manually classify each of the labels into one of the following four categories intended to describe a type of tweet: 'Substance', 'Social', 'Status' and 'Style'. The assumptions used for this classification are a little naive. E.g. the label indicating the presence of hashtags is always classified as 'Substance', which refers to tweets with ideas and information and the label indicating the presence of a question mark is always classified as 'Social', which refers to conversational tweets between friends. Moreover, this assignment of labels to categories cannot be easily automated and has to be redone if the model is retrained. In contrast our relevance-based filter does not make any assumptions about what features contribute to a tweet's relevance nor does it require any manual assignment of labels to tweet attributes. Instead, we extract over 20 tweet-specific and language-specific features that describe a tweet and learn how they affect its relevance by training classification algorithms on tweet feature vectors labeled as 'relevant' and 'not-relevant'.

There have been several moderately successful attempts to apply traditional topic modeling algorithms, like LDA, with minor modifications, to figure out what tweets are talking about. Hong et al. [11], Xu et al. [42] and Ramage et al. [25] all try to use LDA or its variations like the Modified Author-Topic Model or Labeled LDA to build a topic model for tweets. However, given the short and sparse nature of tweets, some of the basic assumptions of these algorithms do not hold. Relevant tweets have one author and only mention a single topic. The irrelevant ones do not even mention a single discernable topic. Therefore a generative process that assumes a distribution over multiple authors or topics, as in the case of LDA, does not work very well for tweets. There have been attempts to counter the short and sparse nature of tweets by aggregating all of a user's tweets into a single document [11]. However, except for few 'professional tweeters' the vast majority of Twitter users talk about a diverse range of random topics. There have also been attempts to use an external knowledge base like Wikipedia [24] [13] or search engine results [26] to augment the tweet text, based on keywords present in it. These can work well for relevant tweets but will also end up augmenting the noise in the dataset if the Twitter stream is not pre-filtered. Moreover, these text-augmentation techniques cannot be used in real-time, which is the eventual goal of our consumer-business tool.

A few previous approaches for identifying emerging topics in document streams have involved clustering sets of new documents based on similarity using techniques like LDA [6], Probabilistic Latent Semantic Analysis [10] and Non-Negative Matrix Factorization [12]. While they can find sets of documents with cohesive topics and patterns, they are not guaranteed to produce clusters that contain previously unseen, novel content. There have been attempts [23] to use First Story Detection (FSD) [1], developed for traditional news streams, on a Twitter stream. FSD tries to detect the occurrence of a document in the stream that talks about a previously unseen story or event. The low signal to noise ratio in social network based streams, especially Twitter, make FSD less effective due to the large number of personal and random tweets that could be considered novel content but are of no interest to a significant segment of the population. Our trending-story detection algorithm is a modified and relaxed implementation of Dictionary Learning, which tries to identify large volumes of novel tweets, similar to each other and dissimilar to any previously observed tweets. It is inspired by how Prasad et al. [16] apply Dictionary Learning, a technique primarily used for sparse coding and compression, to the problem of detecting novel documents.

Chapter 3

The Data we Collect from Twitter

Most of the data served up by the Twitter API is represented using the following 4 Twitter Platform objects:

- ***Tweet***: An object representing an individual tweet published by a user. It contains the tweet's text and several attributes like when it was published, how many times it has been re-tweeted etc. The full list of attributes can be obtained from [38].
- ***User***: An object representing a Twitter user. It does not contain many informative attributes about the human user, apart from their name and a string description. Instead the attributes describe the Twitter account. E.g. when it was created, total number of tweets published, whether tweets are private or public etc. The full list of attributes can be obtained from [39].
- ***Entity***: An object representing Twitter-specific entities that occur in tweets like URLs, hashtags and mentions of other users. Some of the useful attributes are the expanded text of the URLs and the positions of these entities within the tweet text. The full list of attributes can be obtained from [30].
- ***Place***: An object representing the geographical location from where a tweet is published, if the publishing client has been granted permission to disclose a user's location. The full list of attributes can be obtained from [36].

The 'Tweet' object is used most commonly as it contains shortened versions of the other three objects embedded within it. The 'fully-hydrated' versions of the other 3 objects can also be fetched from the Twitter API by using their IDs. The API returns all of these objects as JSON encoded dictionaries of their attributes and corresponding values. There are two types of Twitter API endpoints: streaming and regular pull-based. Streaming endpoints need to be provided with a query only once following which they continuously return data satisfying the query. Pull-based endpoints have to be explicitly queried every time some data is required. They return the data in a single response and close the connection. We collect the following types of tweets to generate three different datasets.

3.1 Tweets Mentioning a Company

The streaming *POST statuses/filter* API endpoint [37] returns public tweets that match one or more string filter predicates. It is not meant to be an exhaustive search of all the tweets containing the predicates. The default access level allows up to 400 tracked keywords. We invoke this endpoint with the list of company names we are interested in tracking and a Collector process, described in Chapter 4, continuously receives tweets containing those strings. We track 70 companies in 14 different verticals. We wanted to track companies that are not only significant in and representative of their vertical areas but that also have a significant online presence so that we are able to collect a sizeable dataset. Since there are no publicly available statistics about how often different companies are mentioned on Twitter, we approximated this information by using Google Domestic Trends [35]. It is a tool which tracks Google search traffic across specific sectors of the economy and sorts search keywords, the majority of which are company names, by how often they occur. We derived our 14 verticals from the way this tool divided the US economy into sectors and picked the top 5 companies searched for in each of those sectors. Our final list of keywords to track was slightly bigger than 70 as we added some well known abbreviations used to refer to companies in our list like 'amex' for American Express. Over a period of one month we ended up collecting 100 million tweets mentioning one or more of these 70 companies.

3.2 Tweets from Users in Twitter's 'Suggested Categories'

We discovered that Twitter curates lists of popular users for regular users to follow in about 30 different categories or topics of interest [29]. These categories include topics like 'Music', 'Sports', 'Technology', 'Food and Drink' etc. The topics and the list of recommended users to follow in each topic can be updated every hour. Most of these categorized users are famous and accomplished individuals in the corresponding category. Many of them are 'professional tweeters' who only use their accounts to disseminate information about said category. We felt that the tweets of these 'curated' users could be labeled as being representative of the publishing user's category and be used to train a text-classifier for distinguishing between tweets based on the 30 topics of interest. This in turn would be useful for detecting the topics of regular users' tweets and trying to determine their topics of interest and expertise. Unfortunately there was no API endpoint to directly fetch these tweets. The following pull-based API endpoints had to be called, in this order, to collect the 'curated' users' tweets.

- ***GET users/suggestions***: fetches the list of categories that Twitter has chosen to curate users for [33].
- ***GET users/suggestions/:slug***: fetches the list of user ids in a particular category. 'Slug' is the string identifier for a category [34].
- ***GET statuses/user_timeline***: fetches tweets published by a particular user [32]. The user's id obtained from the previous endpoint has to be specified as a parameter.

Over a period of one month we ended up collecting over a hundred thousand tweets belonging to these 'curated' users across 26 categories.

3.3 Random Tweets from 1% of the Twitter Firehose

We felt that since the first two types of tweets being collected were very specific, we would end up underestimating the amount of noise in Twitter's stream while building the relevance-based filter for tweets. This assumption eventually turned out to be incorrect, as discussed in Chapter 5, but we setup another Collector process to receive tweets from the streaming GET statuses/sample API endpoint that returns a 1% random sample of all the tweets in the Twitter firehose [31]. Over a period of one month we ended up collecting approximately 75 million random, public tweets in this way.

Chapter 4

Handling Twitter Data at Scale

As there is no publicly available Twitter dataset with the specific kinds of tweets we required, collecting these tweets was one of our most important tasks and we felt, as is true with most data-driven projects, the bigger the data the better. We were also interested in exploring state of the art 'Big Data' products and techniques that a commercial implementation of such a consumer business tool would have to employ. With this in mind, we chose not to quickly hack together some scripts for tweet collection and plain text files for their storage but instead decided to build a framework for collecting, storing and analyzing tweets which would serve our needs at scale and also be generic enough to be used against any Twitter API endpoint for future Twitter-based research efforts. We had the following requirements from such a framework:

- Continuous collection of tweets from a variety of Twitter API endpoints without any manual intervention.
- Durable and reliable storage of the collected dataset in the face of machine failure.
- Easily scalable storage for continuously growing datasets.
- Reasonable runtime (few hours) for queries run against the dataset.

The following subsections describe the design and implementation of this framework and how each of the requirements are met. Figure 4.1 illustrates how all the components of the framework fit together. *Kiji* is an open source 'Big Data' library that allows us to talk to all these components and is the glue that holds the framework together [17].

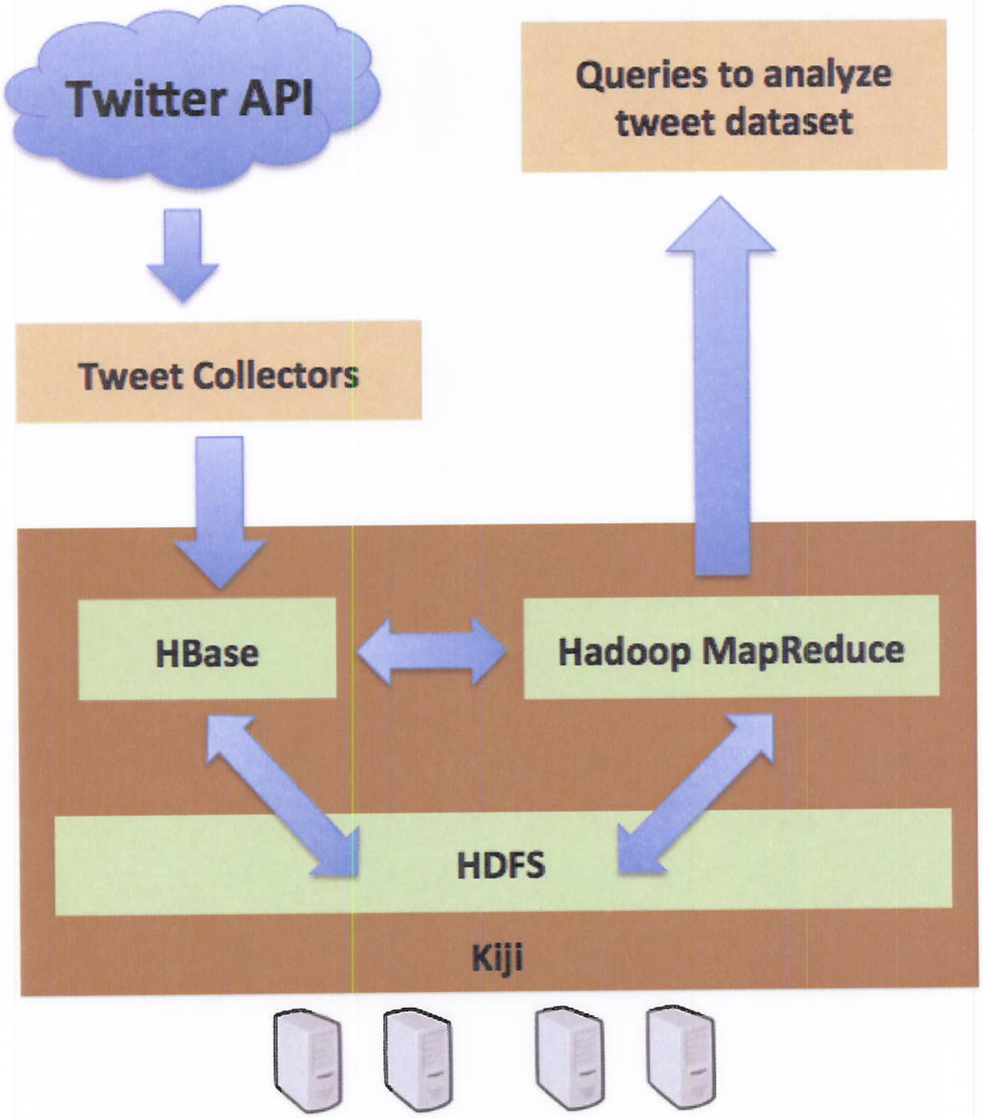


Figure 4.1: The various components of our framework and how they interact with each other

4.1 Storing Tweets at Scale

Using the open source Hadoop Distributed File System (HDFS) as the base of our framework's storage layer naturally satisfied the requirements of scaling out our storage capacity for a continuously growing dataset, and storing it in a fault-tolerant manner [5]. HDFS combines the disks of multiple LAN-connected machines to create the impression of a single bigger disk. Machines with HDFS installed can be added to the cluster on the fly to scale up the storage capacity. HDFS divides each file into chunks, replicates the chunks and distributes them across the machines in order to achieve fault tolerance. As a result the entire dataset is still available if a machine goes down. HDFS regenerates replicas for the under-replicated chunks and redistributes them amongst the remaining machines to restore balance. Additionally, we wanted our storage layer to provide more organization and structure than simple flat files as we anticipated the need to lookup individual tweets by their ids and to add collected or derived metadata or labels to the tweets. For this reason we chose to store the collection of tweets in tables in HBase, a HDFS based open source database [4]. HBase tables are designed to handle datasets with billions of rows while providing random, real-time read/write access to the records. Since it uses HDFS as its backing store, the scalability and reliability requirements remain satisfied. The Twitter API returns each tweet as a large JSON dictionary serialized into a string. We felt that it would be nice if someone writing a program to process the dataset did not have to bother with serializing and deserializing the data but could instead directly read and write tweet objects with easily accessible attributes. Upon realizing that HBase tables support 'object' column types we leveraged Avro, and open source serialization library to automatically handle the serialization and deserialization of tweet and user objects [3]. With this enabled, a programmer analyzing the dataset could 'GET' and 'PUT' tweets and users as objects, while the HBase backend stored them in a compact binary format.

4.2 Collecting Tweets at Scale

In order to collect a sizeable Twitter dataset, it is important that the tweet collecting processes can run continuously and autonomously without constant manual intervention. A naively implemented HTTP client making requests to Twitter API endpoints will be disconnected soon enough for any of the following reasons:

- Exceeding the rate limit for an API endpoint: Each endpoint permits a certain number of requests per rate limit window, which is 15 minutes long.
- Twitter's servers going down.
- Twitter's servers being overloaded.
- Not having sufficient permissions to fetch tweets for a specific user.

We have implemented generic Collector classes in Java, using the twitter4j library [40], which can be pointed to most Twitter API endpoints for collecting tweets. The collectors can automatically detect when the rate limit of an API endpoint is exceeded and back off until the next rate limit window. They can also detect errors on the Twitter side of the world, like their servers being overloaded or down, and continue to back off until the servers resume responding. While most Twitter API endpoints require a single collector, fetching certain kinds of tweets may require making calls to multiple endpoints. E.g. fetching a list of users from one API end point and fetching tweets published by these users from another. To handle these complex collection tasks, the collectors have been designed with an input and output queue (anything that implements the Java BlockingQueue interface), which allows them to be chained together. Since the API endpoints involved may have different rate limits, each collector has been designed to run in a separate thread so that they do not end up blocking each other. We use chained collectors to fetch the tweets published by the curated Twitter users, described in Chapter 3, as this requires making calls to three different API endpoints. The first collector fetches the list of categories that Twitter has used to organize its curated users. This data is updated every hour and the API endpoint has a rate limit of 15 requests per window. We fetch this data once every hour and only one request suffices for fetching all the category identifiers. The category identifiers are passed, using a queue, to the second collector, which fetches lists of user ids belonging to the users in each category. The rate limit for this API endpoint is also 15 requests per window and one request fetches all the user ids for a single category, usually around a hundred in number. The user ids are passed, using a queue, to the third collector that fetches tweets published by specific users. The rate limit of this API endpoint is 300 requests per window and a single request can fetch up to 200 tweets belonging to a single user. By using separate threads, none of the collectors are blocked and unable to fetch data before hitting their rate limits. E.g. once the collector fetching user ids, fetches them for 15 categories, it will have to wait for a fresh rate limit window. However, the collector fetching tweets can continue collecting data for the 1500 odd user IDs sitting on its input queue, until it hits its own rate limit after fetching tweets for 300 users. Figure 4.2 illustrates the chained collectors and flow of data in this complex collection task.

4.3 Analyzing Tweets at Scale

While our dataset could potentially fit on a single large hard drive, it would take an unreasonably long time to run any kind of operation over a hundred million tweets on a single machine. Tasks like counting, aggregations and transformations are well suited for parallelization using the Map-Reduce paradigm. The Hadoop Map-Reduce implementation is ideal for our framework as it is designed to work with data stored in HDFS. In order to speed up tasks on 'Big Data', Hadoop launches multiple

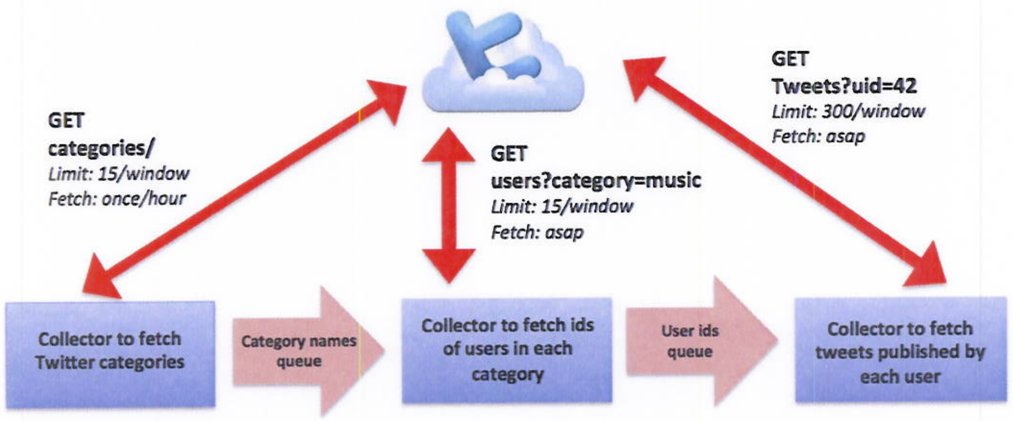


Figure 4.2: A breakdown of how we can chain together multiple collectors to fetch tweets that require pulling data from multiple Twitter API endpoints

versions of the same task, in parallel, on all the machines of the cluster to operate on the individual chunks of the data file stored locally on them. E.g. our Map-Reduce job for sorting users by how many times they tweet about a company takes only 30 minutes, across three desktop machines, while looking at a 50% sample of our dataset or 50 million tweets.

Chapter 5

Filtering the Twitter Stream by Relevance

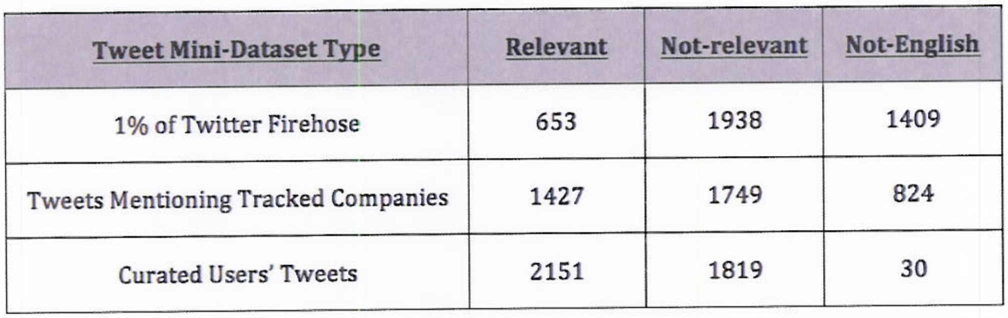
As explained in Chapter 1 we only wish to consider relevant tweets while attempting to gain insight about consumers or detect trending stories concerning a company. Let us further elaborate on our definition of a relevant tweet through a few examples. Relevant tweets are topic-based in that they pertain to a specific topic. E.g. the tweet *"Google Maps head moves over to company's most secretive unit, Google X http://t.co/Q4l1FMlkPO"* is a perfect example of a relevant one. Irrelevant tweets are harder to define as they come in many forms:

- ***Personal conversations***: *"@gregpass Happy V Day!! Made a valentine for you http://t.co/9CRCa6H"*
- ***Status messages or random observations***: *"boring ass Sunday"*
- ***Nonsensical text***: *"RT @adelmz44: ? ???? 24 ????? ?????? ?????????"*
- ***Foreign languages***: *''Je sais pas pourquoi y a #frenchdirectioner en TT alors je vais le tweeter"*

The statistics in Chapter 1 about the proportion of tweets that would be deemed irrelevant by our definition are intimidating. However, those numbers are based on the all the traffic in the Twitter firehose. We expect our dataset to have a significantly higher percentage of relevant tweets as we only collect ones that have a company name in them. The following subsections describe how we estimated the amount of noise in our own dataset and trained a classification algorithm to distinguish between relevant and irrelevant tweets.

5.1 How Noisy is Our Dataset?

Getting an estimate of the amount of noise in our collected dataset required manually labeling a sample of the tweets as relevant or irrelevant. Needless to say this is a laborious task and we required a decently sized (few thousands) labeled sample for an accurate estimate and to use as training data for a relevance-based tweet classifier. Hence, we decided to use Amazon's Mechanical Turk, which is an online marketplace for leveraging thousands of human workers to perform intelligent tasks on datasets, like de-duplication, translation and labeling, for a small dollar reward per data item [2]. We sampled 4000 tweets from each of the three tweet datasets being collected by us, as described in Chapter 3, and uploaded these miniature datasets to Mechanical Turk. The workers were instructed to label the tweets as 'relevant', 'not-relevant' or 'not-English' and were given a detailed description of each category along with examples. We were able to get the 12000 tweets labeled within a day by around 100 unique workers. Figure 5.1 displays the distribution of the three types of tweets in each of the labeled mini-datasets.



Tweet Mini-Dataset Type	Relevant	Not-relevant	Not-English
1% of Twitter Firehose	653	1938	1409
Tweets Mentioning Tracked Companies	1427	1749	824
Curated Users' Tweets	2151	1819	30

Figure 5.1: Post-labeling distribution of the relevant, not-relevant and not-English tweets present in the mini-datasets representing the three types of tweets we collect

If the labeling by the Mechanical Turk workers was accurate, the proportion of relevant tweets in the dataset containing tweets from 1% of the Twitter firehose should be in the vicinity of the official Twitter statistics mentioned in Chapter 1. Only 16% of the tweets in this dataset are labeled as relevant which leads us to believe that the labeling process was quite accurate. 35% of the dataset is made up of non-English tweets, implying that a large number of Twitter users are based in foreign countries, which also agrees with the numbers in Chapter 1. The percentage of relevant tweets is significantly higher, around 35%, in the dataset containing tweets that mention the 70 companies we track. This was expected, as it is less likely that tweets with names of businesses in them would be personal conversations or random status updates. The dataset containing the curated users' tweets is evenly split between relevant and irrelevant tweets. This is most likely because a large number of curated users are 'professional tweeters' who only publish relevant, informational tweets. The high proportion of irrelevant tweets, even in the company dataset, justifies the need for building a tweet relevance filter. This would help clean up the data to improve the results of any further analysis and also reduce the runtime of those jobs due to a smaller working set.

5.2 Selecting Tweet Features for Relevance Classification

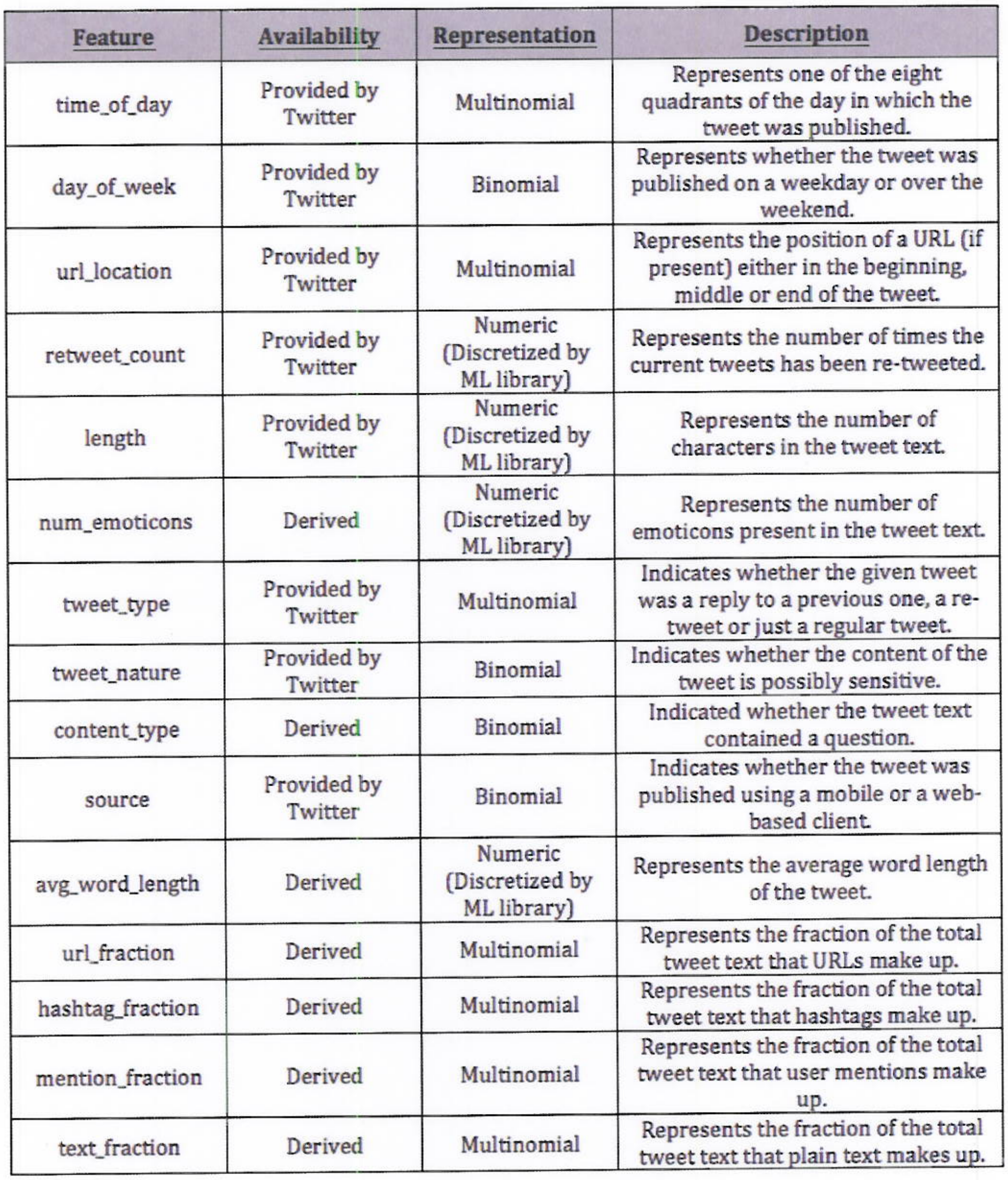
The success of any classification algorithm largely depends on selecting appropriate features or attributes to describe the data and representing them appropriately. The features we use to describe the tweets for the purpose of relevance classification can be grouped into the following two broad categories.

5.2.1 Twitter and Tweet Specific Features

These features describe a tweet in ways that are specific to the Twitter network or describe the presence of Twitter specific entities in it. Figure 5.2 lists the tweet features in this category and the way they are represented in a feature vector when presented to a classification algorithm. These features were selected based on our intuition of their ability to indicate or affect the relevance of a tweet. The *time_of_day* and *day_of_week* features were chosen to investigate whether Twitter users' tendency to publish informational vs. personal tweets was affected by how busy they were. In this case the morning and afternoon hours and weekdays implicitly represent people being busy. While most tweets with URLs are typically trying to share some information, the *url_location* feature tries to distinguish between informational tweets, which typically have a short description of what is being shared with the URL at the end and the large volume of URL-only tweets we observed in our dataset. We consider the latter kind of tweet to be irrelevant as there is no context and no way to determine what the tweet is about without following the link, which is currently not in the scope of our research. The *retweet_count* of a tweet indicates how popular it is. Informational tweets typically get re-tweeted a lot as they are shared amongst users, but so do funny observations and jokes which would should be classified as irrelevant. A lot of spam tweets or random exclamations by users are much shorter compared to tweets actually talking about a topic and the *length* feature tries to capture this. The presence of emoticons in a tweet tends to indicate its personal nature, which potentially makes the *num_emoticons* feature a good identifier of irrelevant tweets. In order to count the number of emoticons in a tweet, we obtained a public dataset of the emoticons used in 1.6 billion tweets over a three-year period [21] and processed it using a MapReduce job to rank the emoticons by how often they were used. We took the 500 most popular emoticons and searched for them in our collected tweets to calculate the value of the *num_emoticons* feature. The *tweet_type* feature conveys whether the current tweet is a re-tweet, indicating that the content was interesting enough to be passed on, or a reply to another tweet, indicating that it is part of a conversation and consequently making it less likely to be relevant. The *tweet_nature* feature is an attempt to identify tweets that may contain sensitive content and would be more opinionated than informative. We were not sure how the presence of a question in a tweet would affect its relevance but wanted to capture this information using the *content_type* feature in case it provided any insight. The *source* feature was chosen to represent whether the tweet was published using a mobile or web client as we thought that people may be publishing shorter and more personal or conversational tweets while on the go. In order to determine whether the source of a tweet is a mobile or web client, we sampled 10 million tweets from our dataset and sorted the names of the clients by how often tweets published by them occurred using a Map-Reduce job. Following this we manually labeled the 100 most popular clients as web or mobile and compared our collected tweets' clients against this list to determine a value for the *source* feature. Clients not found in this list were labeled as web clients. We felt that informational tweets might have a higher average word length due to the presence of more proper nouns and adjectives, which is captured in the *avg_word_length* feature. The *url_fraction*, *mention_fraction*, *hashtag_fraction* and *text_fraction* features represent the fraction of the tweet text that corresponds to the respective entities. These features normalize tweets of different lengths while capturing the effect of URLs, user mentions and hashtags on the relevance of a tweet.

5.2.2 Linguistic Features of Tweet Text

We were curious to see if the linguistic and grammatical properties of a tweet's text could shed any light on its relevance. Parts of Speech (POS) tagging is a technique used for identifying the components of language and grammar present in textual data and the tagged text can easily be converted into a feature vector. However, the language used in tweets is highly informal and unconventional making this task unsuitable for any generic POS tagger. We were able to find a Twitter-specific POS tagger, which was developed as part of a recent NLP research project and reported tagging results nearing 90% accuracy [7]. We incorporated the tagging code from that project into ours and generated feature vectors containing numerical features representing the proportion of 20 different parts of speech, like nouns, adjectives, verbs etc. present in a tweet. The POS tags supported by the Twitter POS tagger are listed in Figure 5.3.

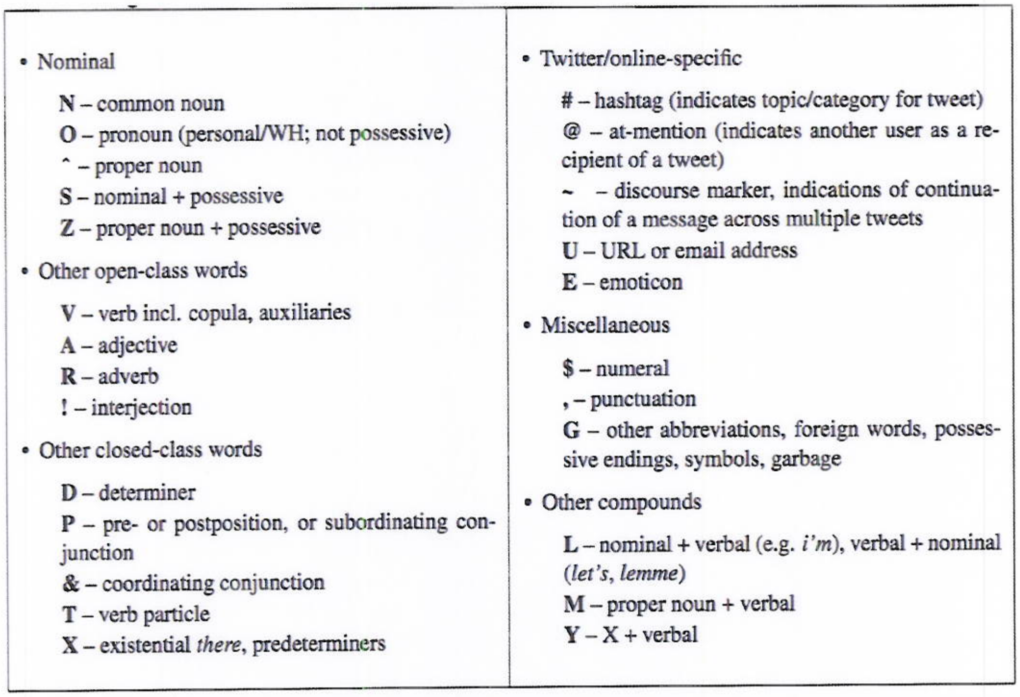


Feature	Availability	Representation	Description
time_of_day	Provided by Twitter	Multinomial	Represents one of the eight quadrants of the day in which the tweet was published.
day_of_week	Provided by Twitter	Binomial	Represents whether the tweet was published on a weekday or over the weekend.
url_location	Provided by Twitter	Multinomial	Represents the position of a URL (if present) either in the beginning, middle or end of the tweet.
retweet_count	Provided by Twitter	Numeric (Discretized by ML library)	Represents the number of times the current tweets has been re-tweeted.
length	Provided by Twitter	Numeric (Discretized by ML library)	Represents the number of characters in the tweet text.
num_emoticons	Derived	Numeric (Discretized by ML library)	Represents the number of emoticons present in the tweet text.
tweet_type	Provided by Twitter	Multinomial	Indicates whether the given tweet was a reply to a previous one, a re-tweet or just a regular tweet.
tweet_nature	Provided by Twitter	Binomial	Indicates whether the content of the tweet is possibly sensitive.
content_type	Derived	Binomial	Indicated whether the tweet text contained a question.
source	Provided by Twitter	Binomial	Indicates whether the tweet was published using a mobile or a web-based client.
avg_word_length	Derived	Numeric (Discretized by ML library)	Represents the average word length of the tweet.
url_fraction	Derived	Multinomial	Represents the fraction of the total tweet text that URLs make up.
hashtag_fraction	Derived	Multinomial	Represents the fraction of the total tweet text that hashtags make up.
mention_fraction	Derived	Multinomial	Represents the fraction of the total tweet text that user mentions make up.
text_fraction	Derived	Multinomial	Represents the fraction of the total tweet text that plain text makes up.

Figure 5.2: The Twitter and tweet specific features of each tweet that we obtained from Twitter or are derived by examining the tweet text

5.3 Choosing A Classification Algorithm

Since the Twitter and tweet-specific features are mostly nominal in their representation, we chose 3 classification algorithms for this feature set that are designed for handling nominal features: Bayesian Networks, RIPPER, a rule-based learner and Random Forest. The POS feature set on the other hand has purely numeric features leading to the selection of the Logistic Regression, Multilayer Perceptron and Support Vector Machines (SVM) algorithms. We trained these algorithms on the



- Nominal

  N – common noun
  O – pronoun (personal/WH; not possessive)
  ^ – proper noun
  S – nominal + possessive
  Z – proper noun + possessive

- Other open-class words

  V – verb incl. copula, auxiliaries
  A – adjective
  R – adverb
  ! – interjection

- Other closed-class words

  D – determiner
  P – pre- or postposition, or subordinating conjunction
  & – coordinating conjunction
  T – verb particle
  X – existential *there*, predeterminers

- Twitter/online-specific

  # – hashtag (indicates topic/category for tweet)
  @ – at-mention (indicates another user as a recipient of a tweet)
  ~ – discourse marker, indications of continuation of a message across multiple tweets
  U – URL or email address
  E – emoticon

- Miscellaneous

  $ – numeral
  , – punctuation
  G – other abbreviations, foreign words, possessive endings, symbols, garbage

- Other compounds

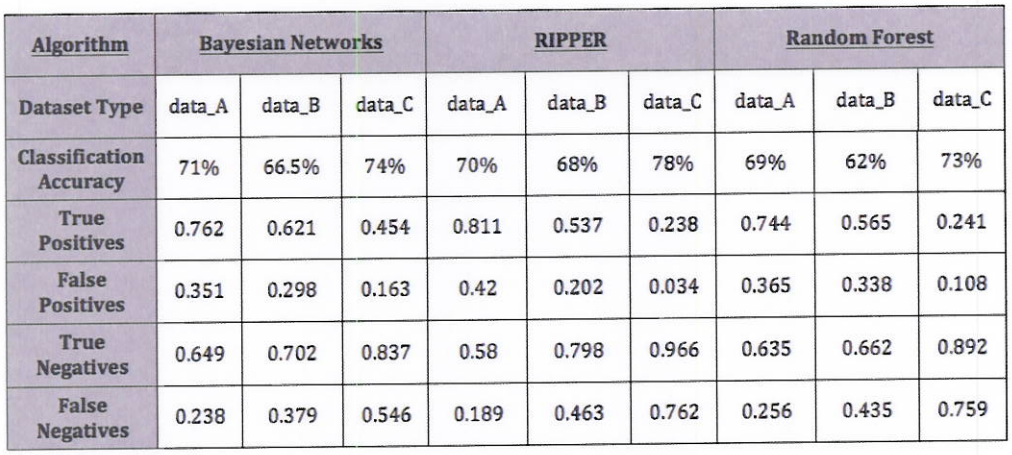
  L – nominal + verbal (e.g. *i'm*), verbal + nominal (*let's*, *lemme*)
  M – proper noun + verbal
  Y – **X** + verbal

Figure 5.3: A list of the POS tags supported by the Twitter POS tagger. We ignored the 5 *Twitter/online-specific* tags as they are already accounted for in our first set of Twitter and tweet specific features

three miniature datasets, each 4000 tweets in size and labeled using Mechanical Turk, individually to evaluate if it was easier to classify certain kinds of tweets based on relevance. Feature vectors were not generated for the tweets labeled as 'not-English'. The classification accuracy was evaluated using 5-fold cross validation. We used the Weka machine-learning library to train and evaluate all of these classifiers [41].

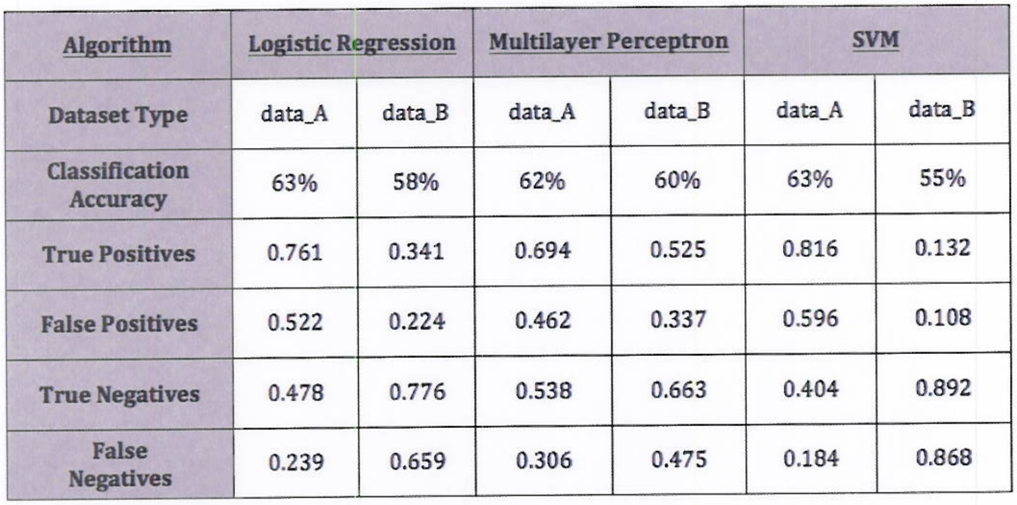
5.4 Accuracy of Relevance-Based Tweet Classification

The classification accuracy results for the chosen algorithms and mini-datasets, based on the Twitter-specific features and POS features, are listed in Figure 5.4 and Figure 5.5 respectively. We were unable to train a classifier using the POS features on the mini-dataset with tweets from 1% of the firehose as it contained a significant amount of non-ASCII characters, which the Twitter POS tagger was unable to process. The overall results indicate that POS features are not very good indicators of tweet relevance as the highest classification accuracy achieved across all algorithms and datasets is only 63%. The Twitter and tweet-specific features are able to do a much better job with the highest classification accuracy being 78% using the RIPPER algorithm. While using this feature set, the mini-dataset with tweets from 1% of the firehose is the easiest to classify, across all the algorithms. This is probably because this dataset contains the most noise and very 'obviously irrelevant' tweets. The high rate of true negatives across all algorithms, 84, 89 and 96%, while using this dataset is a clear indicator of the significant presence of these 'obviously irrelevant' tweets. The mini-dataset containing the curated users' tweets is also classified at a decent accuracy of 70% on average. This can be explained by this dataset having a large proportion of relevant tweets and a lot of those being 'obviously relevant' tweets as 'professional tweeters' publish them. The high rate of true positives, 76, 81 and 74%, while classifying this dataset is a clear indication of these 'obviously relevant' tweets being present. The dataset containing company-related tweets is the hardest to classify, as it does not contain extremely relevant or extremely irrelevant tweets. Consequently both the true positive and true negative rates for this dataset are not extremely high, ranging between 50 and 80%. However, we are still able to achieve classification accuracy close to 70%, using the RIPPER algorithm, with 80% of the irrelevant tweets being classified correctly. While using the Twitter and tweet-specific feature set, the feature that surprisingly provides the highest information gain, across all algorithms and datasets, is *avg_word_length*. While this is not very intuitive, non-relevant tweets tend to be shorter in length and use more common nouns, slang, abbreviations and emoticons and also contain fewer URLs, all of which contribute to a low average word length.



Algorithm	Bayesian Networks			RIPPER			Random Forest		
Dataset Type	data_A	data_B	data_C	data_A	data_B	data_C	data_A	data_B	data_C
Classification Accuracy	71%	66.5%	74%	70%	68%	78%	69%	62%	73%
True Positives	0.762	0.621	0.454	0.811	0.537	0.238	0.744	0.565	0.241
False Positives	0.351	0.298	0.163	0.42	0.202	0.034	0.365	0.338	0.108
True Negatives	0.649	0.702	0.837	0.58	0.798	0.966	0.635	0.662	0.892
False Negatives	0.238	0.379	0.546	0.189	0.463	0.762	0.256	0.435	0.759

Figure 5.4: Relevance classification results based on the Twitter and tweet-specific features. Data_A is the mini-dataset containing the curated users' tweets, data_B is the mini-dataset containing the tweets mentioning companies and data_C is the mini-dataset containing the tweets from 1% of the Twitter firehose



Algorithm	Logistic Regression		Multilayer Perceptron		SVM	
Dataset Type	data_A	data_B	data_A	data_B	data_A	data_B
Classification Accuracy	63%	58%	62%	60%	63%	55%
True Positives	0.761	0.341	0.694	0.525	0.816	0.132
False Positives	0.522	0.224	0.462	0.337	0.596	0.108
True Negatives	0.478	0.776	0.538	0.663	0.404	0.892
False Negatives	0.239	0.659	0.306	0.475	0.184	0.868

Figure 5.5: Relevance classification results based on the POS features. Data_A is the mini-dataset containing the curated users' tweets and data_B is the mini-dataset containing the tweets mentioning companies

Chapter 6

Extracting Consumer Insight From Twitter

It is extremely desirable for a company to be able to identify influential people talking about it, who are also experts in the domain that it specializes in. On a social network like Twitter, these individuals can be leveraged to effectively build product awareness, brand buzz and new sales. More importantly these are the people to pay attention to and appease when negative stories about the company are trending. This section describes our efforts to identify these 'top tweeters' for a company. We currently only focus on finding users with high expertise in the company's domain. Evaluating users' influence through metrics like 'true reach' requires collecting tweets for a lot of specific users and also their follower and followee graphs, which is a significant collection effort that we plan to undertake going forward. In the following subsections we look at users' tweeting patterns while talking about businesses and our efforts to identify individuals with expertise in the company's domain in the following two ways:

- By only looking at users' tweeting patterns
- By examining the content of individual tweets published by users.

6.1 User Tweeting Patterns while Mentioning Businessess

We wrote a Map-Reduce job to identify the users tweeting about each of the 70 companies we were tracking and ordered them by the number of times they had mentioned it during our collection period. We observed that for each of the companies, the number of times it is mentioned by individual users follows a long-tail distribution. This is illustrated in Figure 6.1. The top 10% of users clearly stand out by virtue of their high tweet volume and this is the group we want to focus on for determining the 'top tweeters'. The remaining 90% of the users in the tail all blend together and their daily rate of mentioning the company is not high enough to indicate expertise or strong interest in it or its domain.

6.2 Determining User Expertise Implicitly through Tweeting Patterns

Simply ranking users by how often they mention a company does not work very well. We examined the top 10% of users for several companies and discovered that the highest-ranked users are almost always bots, accounts owned by the company itself, accounts set up to re-tweet an official company account or users talking about some entirely different topic or company which shares its name with the concerned company. Half of these are spurious Twitter accounts and the other half are of no interest to the concerned business. This problem is evident from Figure 6.2, which displays the top 4 users mentioning Delta Airlines, simply ranked by tweet volume. None of these users make any sense as 'top tweeters' for an airline. Clearly there is a need for cleaning up this list of users based on their expertise so that it is more relevant to the company.

As we wish to rank users by expertise, we reasoned that a user with an interest in a particular company's domain would also mention other similar companies in that domain with high probability. Since we were collecting tweets for 70 companies across 14 verticals, we already have companies grouped together, in sets of 5, by their domains. We perform item-item collaborative filtering on the lists of users mentioning the companies in a particular vertical, except in reverse, as we already know the items (companies) that are similar and want the users common to them. The resulting list is made up of users who have tweeted about all five companies in the vertical that the company in question belongs to and the average of their tweet volumes across the five companies is used to rank them. The resulting list contains userswho are actually experts or are strongly interested in the company's domain, as illustrated in Figure 6.3 by the new top 4 tweeters for Delta Airlines. This technique removes company owned Twitter account(s) from the list as they typically only exhibit high tweet volume for the company in question. It also removes users who got included due to ambiguity in the name of the concerned entity. We can see how ambiguity causes problems in the case of Delta Airlines. As Figure 6.2 illustrates, there are three water fixture companies amongst the users who publish the most tweets mentioning Delta, which is unusual for an airline company. However, it turns out that Delta is also a brand of bathroom and kitchen fixtures. Our algorithm is able to get rid of these users as the ambiguity does not persist throughout the company's vertical and the expertise-based top tweeters for Delta in Figure 6.3 are all related to aviation. Not only are these users more relevant to the concerned company but they also seem to be more influential, if we consider the number of followers to be an initial estimate of popularity on Twitter.

6.3 Determining User Expertise Explicitly from Tweet Text

The 'follow' and 're-tweet' features in Twitter have resulted in an information-flow paradigm of few producers and many consumers. We processed the dataset containing tweets from Twitter's curated users, as described in Chapter 3, and generated some statistics to support the existence of this paradigm. The curated users, who Twitter recommends as users for regular users to follow, have 1.05 million followers on average and each of their tweets gets an average of 371 re-tweets. As a lot of keywords, URLs, hashtags and user mentions from their tweets are repeated and retransmitted through tweets published by the remaining regular users, we felt that it might be possible to classify tweets into broad topics by using the curated users' tweets as training data. By classifying a regular user's tweets into these broad topics, we could build a topic distribution for them and gain more accurate insight about their area of expertise. Each tweet in the dataset was converted into a sparse bag-of-words feature vector with TF-IDF weighting and labeled with the category that the publishing user belonged to. We created three additional datasets by artificially accentuating the presence of URLs, hashtags and user mentions in the tweets, by repeating them five times, to evaluate whether the presence of these Twitter-specific features helped increase classification accuracy. We trained four well-known text-classification algorithms, namely C4.5 (decision tree based), Nave Bayes, k-Nearest Neighbor search and SVMs, on these datasets but the overall classification accuracies were very unimpressive, never rising above 50%. The 'News' topic was the only class to achieve decent individual classification accuracy. It performed the best with the SVMs algorithm and the results across the four datasets are listed in Figure 6.4. As illustrated in Chapter 5, almost half of the mini-dataset containing the curated users' tweets was not relevant. This means that half the tweets labeled as belonging to a particular category were not really representative of that category at all, which explains the classifiers' poor performance. In most categories like 'Music' or 'Sports' the tweets are not focused solely on that category but take on the personality of the publishing users who are popular musicians and athletes. The 'News' topic is an exception as it mostly consists of Twitter accounts belonging to popular journalists, broadcasters, newspapers and news channels whose tweets are solely used to share headlines or other newsworthy events and articles. The tweet format in this category is also very consistent: a short description followed by a URL. Links from major news sources are re-tweeted a lot and circulated through the network even thought the tweet text describing the link might change. This explains why 'News' was the only topic to get classified with decent accuracy and why the accuracy was highest for the dataset that accentuated the occurrence of URLs.

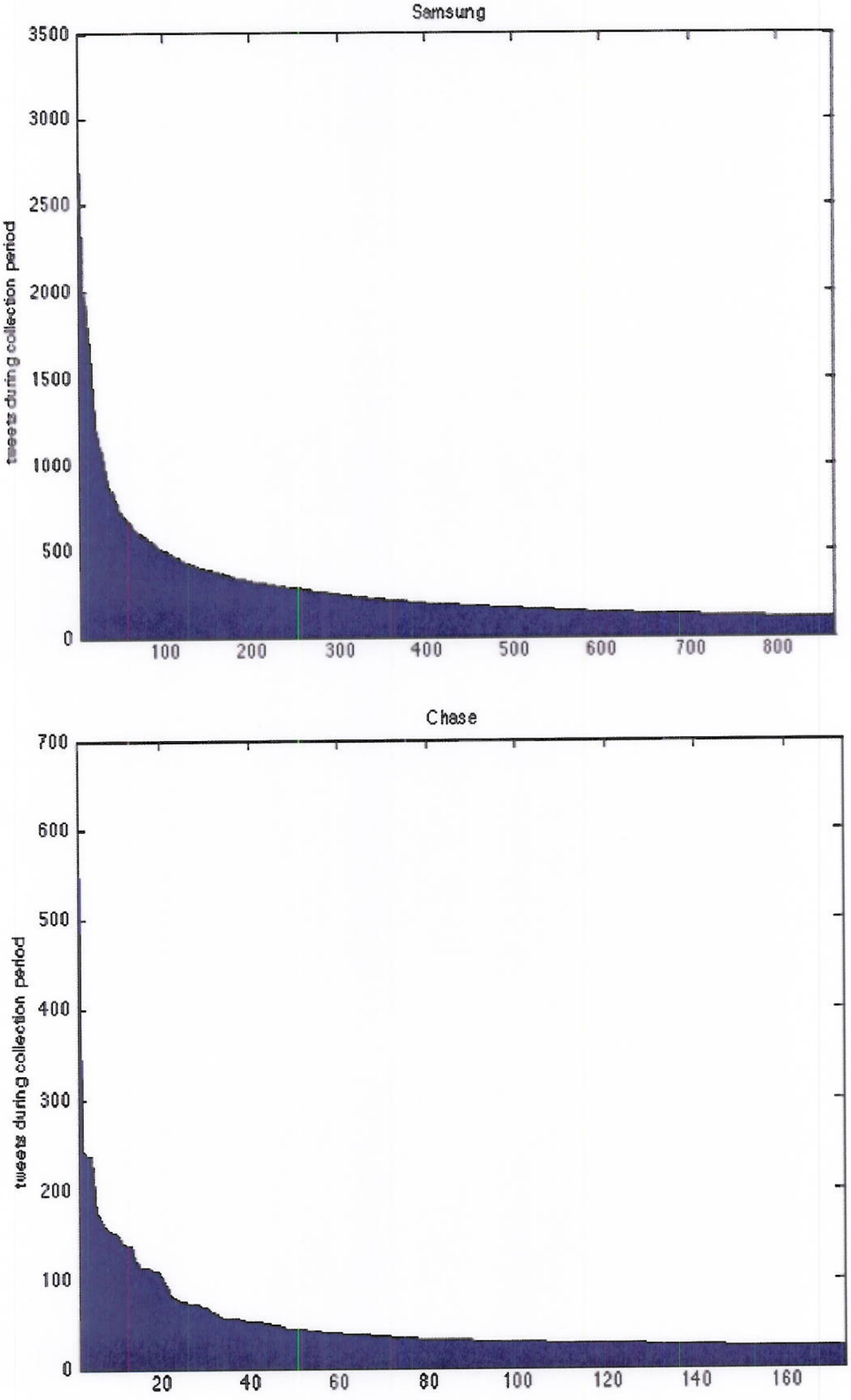
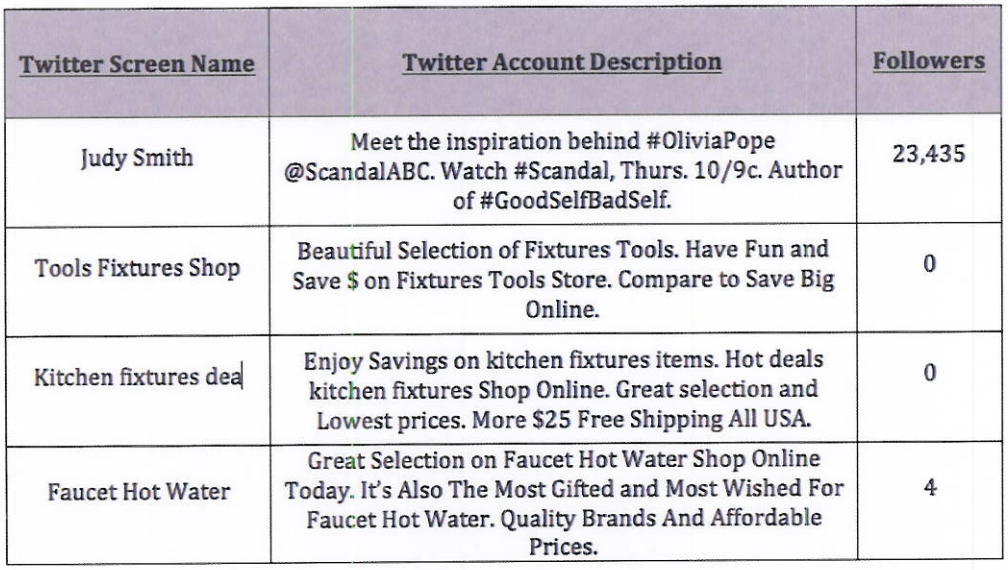
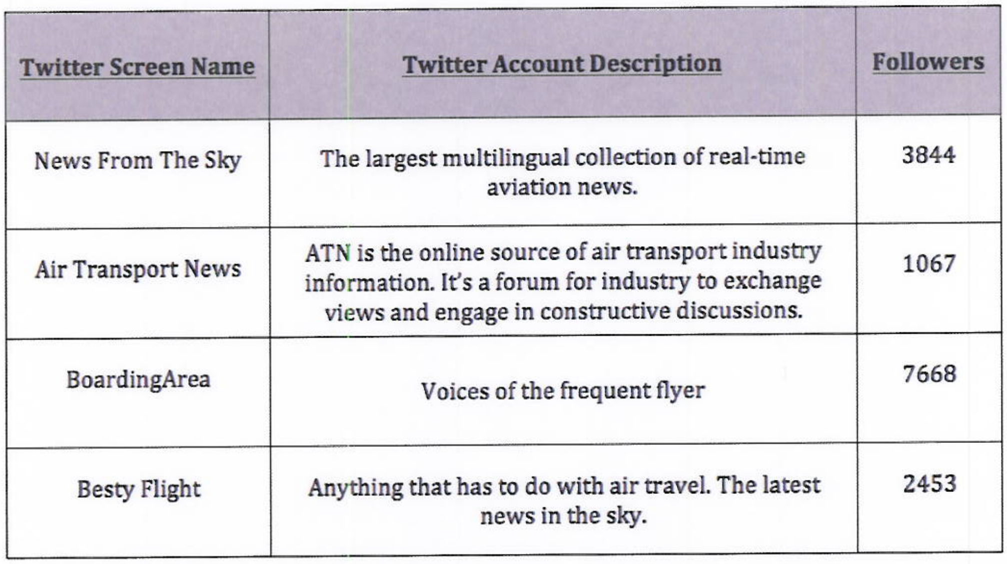


Figure 6.1: The long-tail distribution of how many times individual users mention a company: shown here for Samsung and Chase



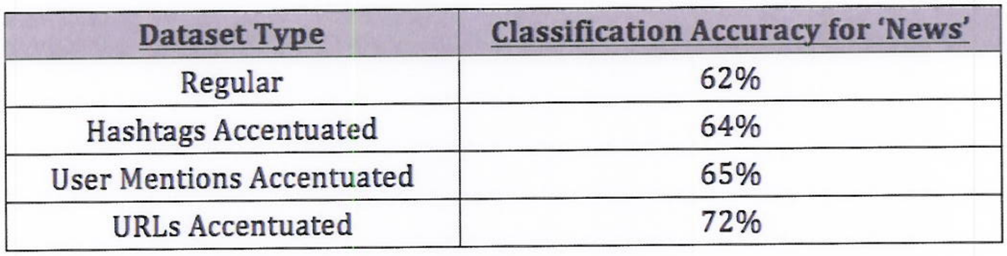
Twitter Screen Name	Twitter Account Description	Followers
Judy Smith	Meet the inspiration behind #OliviaPope @ScandalABC. Watch #Scandal, Thurs. 10/9c. Author of #GoodSelfBadSelf.	23,435
Tools Fixtures Shop	Beautiful Selection of Fixtures Tools. Have Fun and Save $ on Fixtures Tools Store. Compare to Save Big Online.	0
Kitchen fixtures dea	Enjoy Savings on kitchen fixtures items. Hot deals kitchen fixtures Shop Online. Great selection and Lowest prices. More $25 Free Shipping All USA.	0
Faucet Hot Water	Great Selection on Faucet Hot Water Shop Online Today. It's Also The Most Gifted and Most Wished For Faucet Hot Water. Quality Brands And Affordable Prices.	4

Figure 6.2: The top 4 tweeters for Delta Airlines, when they are simply ranked by how many times they mentioned it during our collection period



Twitter Screen Name	Twitter Account Description	Followers
News From The Sky	The largest multilingual collection of real-time aviation news.	3844
Air Transport News	ATN is the online source of air transport industry information. It's a forum for industry to exchange views and engage in constructive discussions.	1067
BoardingArea	Voices of the frequent flyer	7668
Besty Flight	Anything that has to do with air travel. The latest news in the sky.	2453

Figure 6.3: The top four tweeters for the company Delta Airlines, after we clean up the list of users mentioning it and rank them by their expertise in the company's vertical



Dataset Type	Classification Accuracy for 'News'
Regular	62%
Hashtags Accentuated	64%
User Mentions Accentuated	65%
URLs Accentuated	72%

Figure 6.4: Classification accuracy for the 'News' topic while using the SVMs algorithm

Chapter 7

Detecting Business-Related Trending Stories and Events

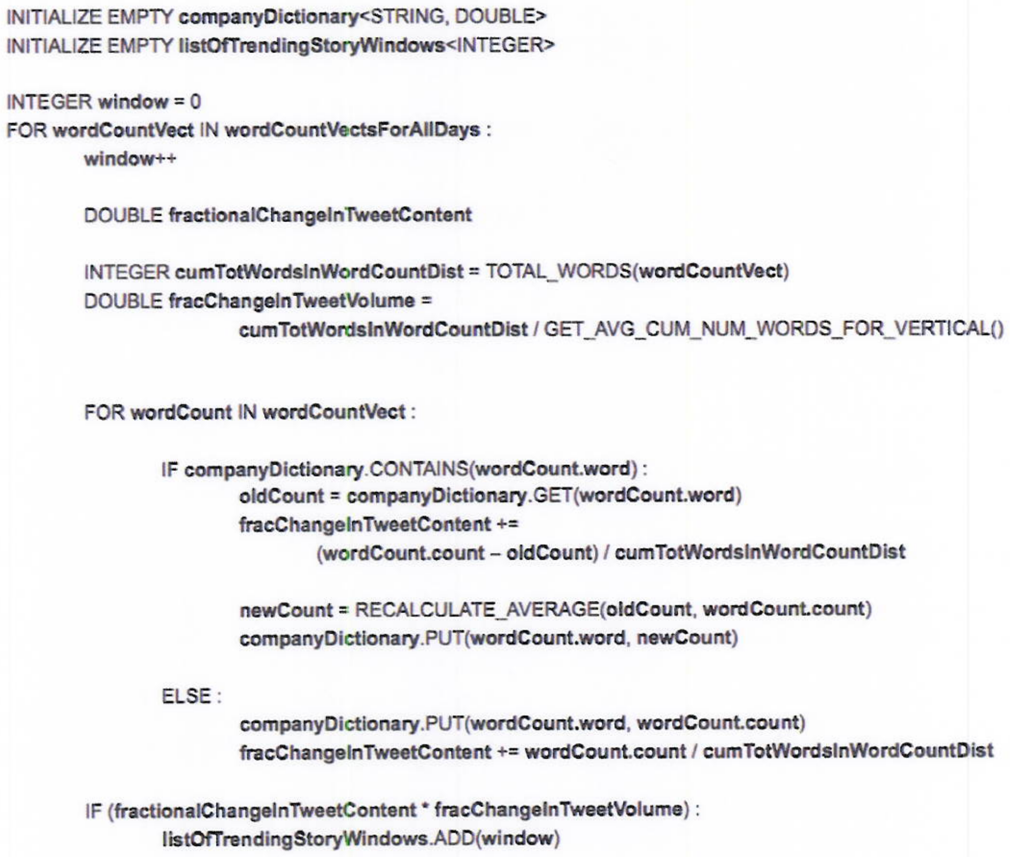
A company can greatly benefit if it is able to detect trending stories about itself on Twitter rather than hear about them when the stories have been read by most people. Information spreads extremely fast in a broadcast oriented social network like Twitter and being able to detect trends, especially negative information or opinions, can enable a company to react promptly. More importantly, this can help a company gauge the sentiment and reactions of consumers at scale as the story breaks, without having to explicitly survey people or wait a significant amount of time to see an implicit change in statistics like sales, orders or revenue. For the content of a tweet to be considered a trending story, it must have support, i.e. appear in high volume from multiple sources and it must be novel, i.e. dissimilar to previously seen content. We have implemented an algorithm for detecting emerging stories about a company, based on Dictionary Learning, which takes both these factors into account. The following subsections discuss how dictionary learning works, our modified implementation of dictionary learning to detect trending stories and the actual business-related stories that it was able to uncover during our one-month collection period.

7.1 Using Dictionary Learning for Identifying Emerging Stories

Dictionary learning is the process of building a concise dictionary of basic elements or atoms to represent a sparse, high-dimensional dataset, e.g. most text-based datasets, such that the documents can be approximately represented by a linear combination of a few atoms. It is primarily used for sparse coding and compression. It was recently applied to the problem of detecting emerging topics in text document collections and a small, focused Twitter dataset [16]. While building a dictionary from a stream of documents, if a new document is encountered which cannot be represented with low error as a sparse linear combination of the existing atoms, it is a good indication of the novelty of that document. Novel documents thus identified can be used to learn a new dictionary of atoms representing the new content, which is in turn used to cluster novel documents together. The size of a cluster indicates how much support that novel document has. Big clusters of novel documents are likely to be emerging stories. The original dictionary is eventually updated with atoms from the new dictionary, depending on what the resolution of story detection is: minutes, hours or days.

7.2 Our Modified Dictionary Learning Algorithm

The crux of using dictionary learning for detecting emerging topics lies in being able to determine whether a document can or cannot be represented as a linear combination of the atoms in the current dictionary. Unfortunately the optimization problem that represents this satisfiability constraint involves non-negative matrix factorization and is in general non-convex. A relaxed version of this optimization, which is convex, is discussed in [16] and can be solved using techniques like Robust PCA or SVD. However, the repeated need to solve linear/quadratic programs and access all the data per iteration does not allow these algorithms to scale to large datasets. In order to get around this, our algorithm calculates a numeric value to represent how novel the content of a new document is when compared to the existing dictionary, instead of trying to represent it using the dictionary's atoms. In order to quantify this difference in content between the new tweet and tweets seen so far, the dictionary needs to provide, for each word seen so far, an estimate of the expected likelihood of observing the word, instead of simply storing the atoms or individual words. It also does not make sense to compare a single tweet against this dictionary to determine a change in content. Instead we aggregate tweets over the desired window of story detection, which can range from minutes to days, and build a vector of the 1000 most frequently occurring words (along with their counts) in the tweets published during the window. For the first window, the dictionary is empty and the words from the vector are simply loaded into the dictionary along with their counts, which help us calculate the likelihood of observing the word during a window. After each subsequent window, the new word count vector is compared against the dictionary. For words that are already present in the dictionary, their counts are averaged with the existing count to get an updated estimate of the likelihood of their occurrence in a window. Newly encountered words are added to the dictionary along with their counts. For every window the change in tweet content is calculated by adding up the fractional changes observed for each word in the word count vector being compared to the dictionary. For newly observed words the numerator of this fractional change is simply their count and the denominator is the cumulative number of words that make up the word count vector distribution. Simply put, what fraction of words observed during the window did this newly observed word make up. For words that were already present in the dictionary, the denominator of the fractional change is the same as when dealing with new words, but the numerator is the difference between the current window's count and the prior expected count of this word during a window, as stored in the dictionary. The fractional change in tweet content thus calculated is further multiplied by a fractional change in overall tweet volume, where the numerator is the cumulative number of words that make up the word count vector distribution during this window and the denominator is a running average of the cumulative number of words making up the word count vector distributions for all companies in that vertical. While our approach already accounts for support by incorporating the word counts in the fractional content-change calculation, multiplying by a fractional change in overall tweet volume helps prevent certain false positives. These false positives arise when a certain company's overall tweet volume during a window is low, but a few common words occur a large number of times (heavily skewed long-tail word count distribution). This can lead to a decently high value of the window's fractional change in content and give the impression of a trending story. Windows that have a high value for the product of the fractional change in tweet content and the fractional change in overall tweet volume are classified as emerging or trending stories. This algorithm is run separately for each company's collected tweets and can use a story-detection window of any resolution, although a very small window will not have enough signal to detect anything meaningful and a very long window may end up detecting multiple stories or wait too long before informing the user of a trend. In order to represent the dictionary, we use a Cache data structure that is part of Google's guava-libraries for Java [8]. Cache is basically identical to a Java HashMap, but it evicts key-value mappings, using a LRU policy, if its size grows beyond a pre-specified limit or if certain mappings have not been accessed for a specified amount of time. Cache makes use of a Ticker class to determine the amount of elapsed time for evicting mappings. Ticker can be overridden to simulate a custom time source, which proved to be extremely essential as we were running this algorithm offline, after all the data had been collected. Figure 7.1 illustrates the key state updates and data structures involved in this algorithm.

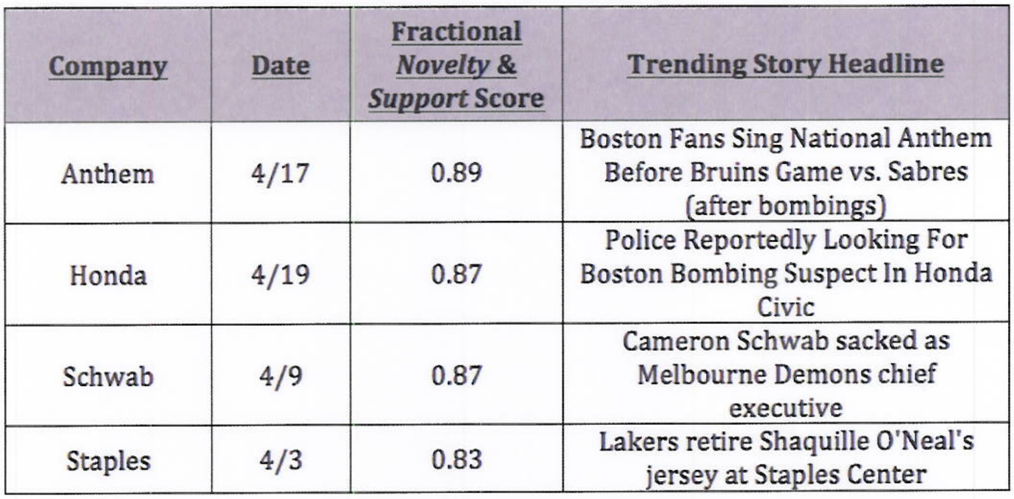


```
INITIALIZE EMPTY companyDictionary<STRING, DOUBLE>
INITIALIZE EMPTY listOfTrendingStoryWindows<INTEGER>

INTEGER window = 0
FOR wordCountVect IN wordCountVectsForAllDays :
        window++

        DOUBLE fractionalChangeInTweetContent

        INTEGER cumTotWordsInWordCountDist = TOTAL_WORDS(wordCountVect)
        DOUBLE fracChangeInTweetVolume =
                        cumTotWordsInWordCountDist / GET_AVG_CUM_NUM_WORDS_FOR_VERTICAL()


        FOR wordCount IN wordCountVect :

                IF companyDictionary.CONTAINS(wordCount.word) :
                        oldCount = companyDictionary.GET(wordCount.word)
                        fracChangeInTweetContent +=
                                (wordCount.count – oldCount) / cumTotWordsInWordCountDist

                        newCount = RECALCULATE_AVERAGE(oldCount, wordCount.count)
                        companyDictionary.PUT(wordCount.word, newCount)

                ELSE :
                        companyDictionary.PUT(wordCount.word, wordCount.count)
                        fracChangeInTweetContent += wordCount.count / cumTotWordsInWordCountDist

        IF (fractionalChangeInTweetContent * fracChangeInTweetVolume) :
                listOfTrendingStoryWindows.ADD(window)
```

Figure 7.1: The pseudo code describing our dictionary learning based algorithm for trending-story detection

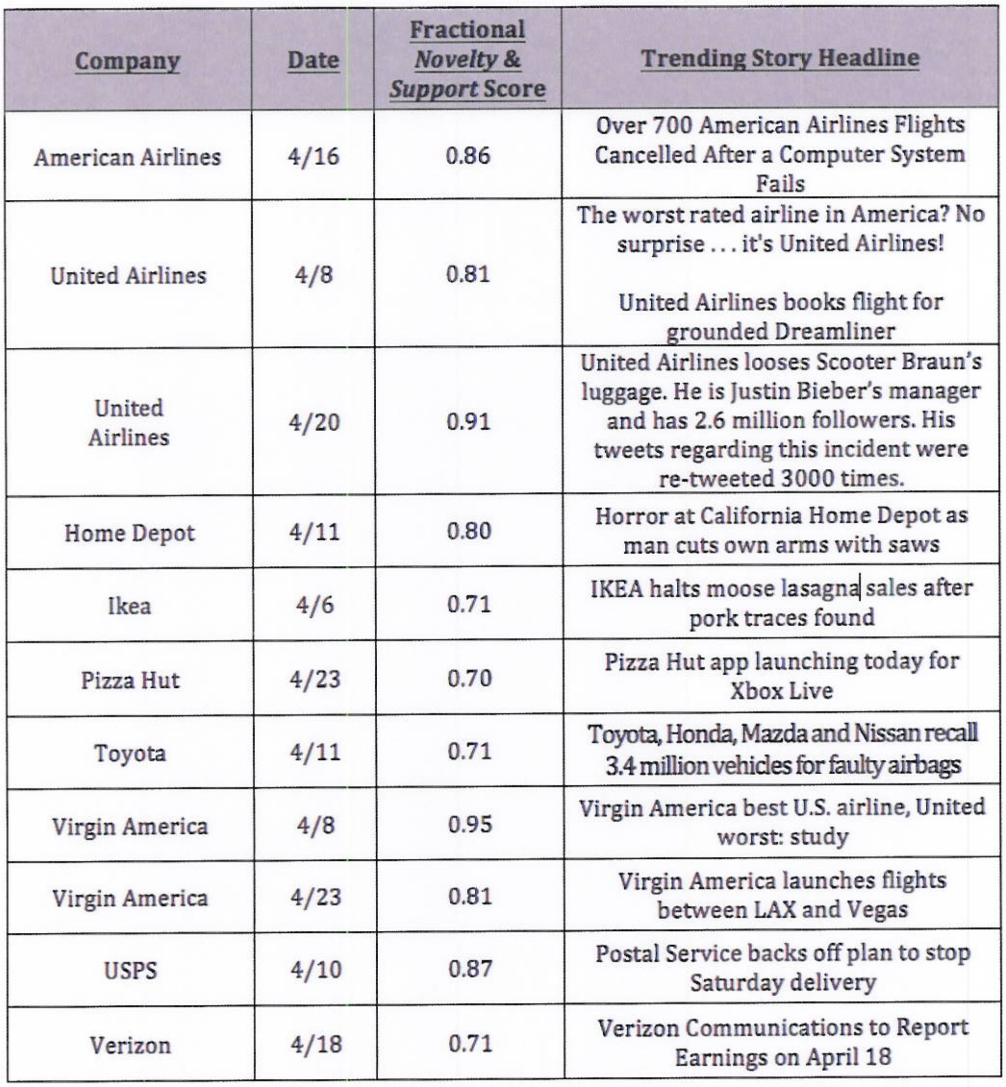
7.3 Empirical Evaluation on Companies' Tweets

Since we do not have an exhaustive list of the trending stories concerning our 70 companies during the tweet collection period, we ran our algorithm on the 100 million-tweet dataset of company-related tweets and manually verified whether the identified windows actually correspond to a trending story for a company. For this evaluation we set the window size to be 24 hours, which meant stories would be detected on a daily basis. We set the maximum size of the dictionary to be 3000 words as we wanted the first three windows to just load the dictionary to its normal state, i.e. the expected word occurrences in the absence of a trending story. We have assumed that no trending stories occur during the first three days of data collection. We configured the dictionary to evict words if their counts have not been modified for 3 days as content on Twitter is very short-lived and words specific to an old story should not hamper the detection of new emerging stories. We wrote a 2 phase Map-Reduce to detect the emerging stories. The first Map-Reduce job calculates word count vectors of the 1000 most frequently occurring words in every company's tweets for every day of our collection period. The second Map-Reduce job groups these word count vectors by company, sorts them by the day of the year and runs the trending-story detection algorithm. The events that we detected for various companies are displayed in Figure 7.3. A few detected stories do not actually correspond to the concerned company due to ambiguity in the name, but we include these in the true positives, as it is a success for the story detection algorithm. These stories are displayed in Figure 7.2. Company tweet disambiguation is one of our top priorities as we enhance this work going forward. We played around with the threshold value of the fractional change in tweet content and volume during a window, to include the most number of real trending stories and keep the false positives low. We settled at a threshold value of 0.7 and got the 15 true positive trending stories listed in Figures 7.2 and 7.3 and 5 false positives. Figure 7.4 illustrates how the content of a tweet changes when a trending story surfaces, specifically the tweets of Toyota a day before and on the day of the announcement of the massive airbag recall.



Company	**Date**	**Fractional *Novelty* & *Support* Score**	**Trending Story Headline**
Anthem	4/17	0.89	Boston Fans Sing National Anthem Before Bruins Game vs. Sabres (after bombings)
Honda	4/19	0.87	Police Reportedly Looking For Boston Bombing Suspect In Honda Civic
Schwab	4/9	0.87	Cameron Schwab sacked as Melbourne Demons chief executive
Staples	4/3	0.83	Lakers retire Shaquille O'Neal's jersey at Staples Center

Figure 7.2: True positive trending stories detected for companies that actually correspond to a completely different event or person because of ambiguity in the company name



Company	Date	Fractional *Novelty* & *Support* Score	Trending Story Headline
American Airlines	4/16	0.86	Over 700 American Airlines Flights Cancelled After a Computer System Fails
United Airlines	4/8	0.81	The worst rated airline in America? No surprise . . . it's United Airlines! United Airlines books flight for grounded Dreamliner
United Airlines	4/20	0.91	United Airlines looses Scooter Braun's luggage. He is Justin Bieber's manager and has 2.6 million followers. His tweets regarding this incident were re-tweeted 3000 times.
Home Depot	4/11	0.80	Horror at California Home Depot as man cuts own arms with saws
Ikea	4/6	0.71	IKEA halts moose lasagna sales after pork traces found
Pizza Hut	4/23	0.70	Pizza Hut app launching today for Xbox Live
Toyota	4/11	0.71	Toyota, Honda, Mazda and Nissan recall 3.4 million vehicles for faulty airbags
Virgin America	4/8	0.95	Virgin America best U.S. airline, United worst: study
Virgin America	4/23	0.81	Virgin America launches flights between LAX and Vegas
USPS	4/10	0.87	Postal Service backs off plan to stop Saturday delivery
Verizon	4/18	0.71	Verizon Communications to Report Earnings on April 18

Figure 7.3: True Positive trending stories detected by our modified dictionary learning algorithm

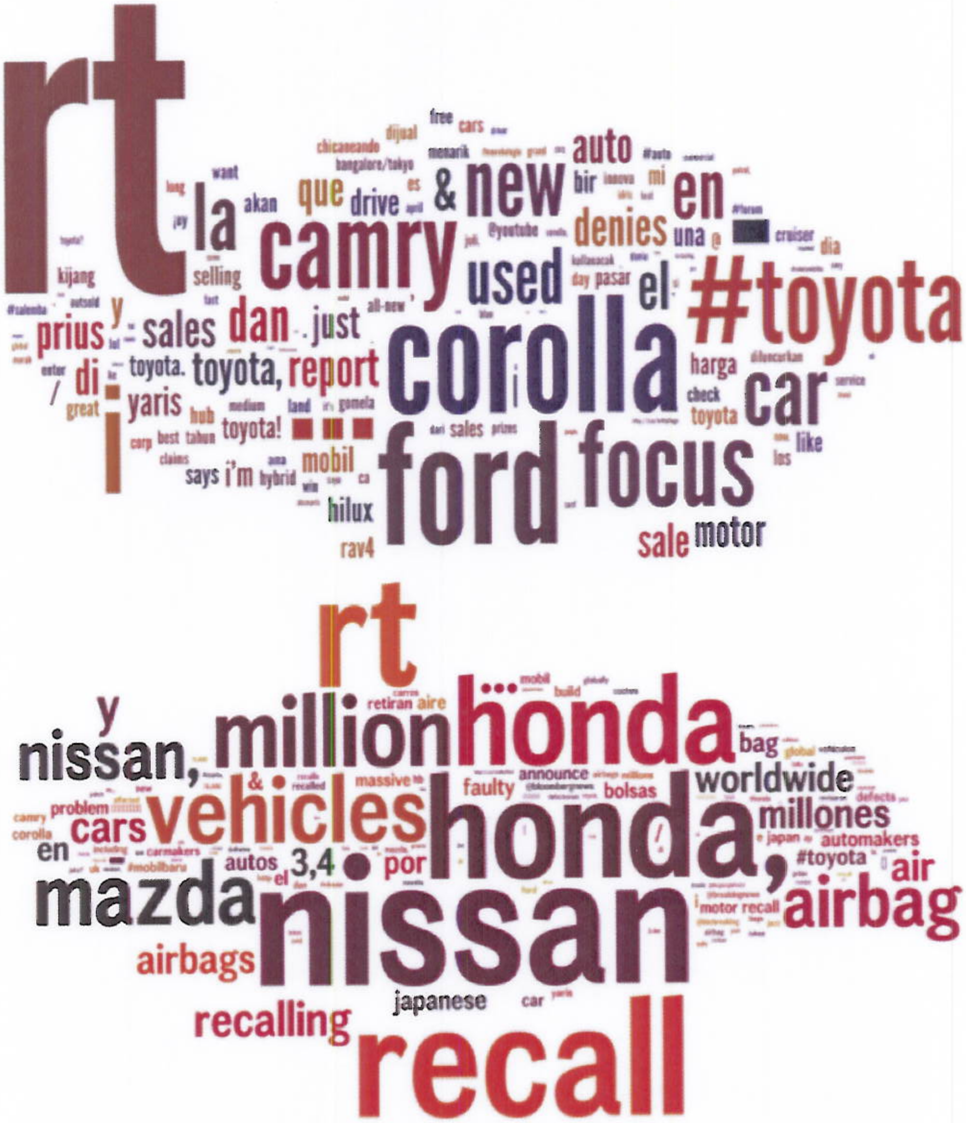


Figure 7.4: Word clouds of the most frequently occurring words in Toyota related tweets on April 10th (top) and April 11th (bottom). This illustrates the change in tweet content when the trending story about Toyota's airbag recall surfaced on April 11th

Chapter 8

Future Work

The ambiguity introduced because some names of companies are also commonly used English words or names of people or because two companies share a name, has compromised the quality of our results while ranking users based on expertise and while detecting trending stories. In order to overcome this problem we would like to build upon the efforts of Zhang et al. [28] who use external sources like a company's webpage and Wikipedia article to build prior context about the text that should accompany tweets that pertain to the company.

We currently neglect influence while ranking the 'top tweeters' for a company. We need to start collecting tweets for individual users mentioning a company and their follower and followee graphs in order to add influence to the mix. After collecting and constructing a sufficiently large section of the Twitter graph, a natural step is to apply an algorithm like HITS [18] to try and distinguish 'authorities' or users who are experts in a particular domain from 'hubs', users with more generic interests who link to several users with expertise in various areas. While the tweets of the set of users 'curated' by Twitter under various categories did not help classify tweets into topics, these users could prove useful while running an algorithm like Topic-Sensitive PageRank on the Twitter network for assigning a-priori importance estimates to users for those categories. Topic-Sensitive PageRank could help identify users who are both influential in the network and experts in a company's domain.

As mentioned in Chapter 7, one of the primary benefits of detecting trending stories about companies is to evaluate the sentiment and reactions of consumers as the story unfolds. In addition to making improvements to our story-detection algorithm, we want to extract the tweets in a window that best represent the trend and run Twitter-specific sentiment analysis techniques like [27] on them in order to provide accurate user feedback to the company.

One of our more futuristic goals is to take the building blocks of this consumer business tool that have shown good results, like trending-story detection, and move the computation from a 'collect and batch-process' paradigm to an online paradigm on a real-time distributed stream processing engine like Storm [22].

Chapter 9

Summary and Conclusion

Twitter is no longer simply a social network but a 'social-information network' used for sharing information, ideas and opinions on matters of mass importance. Our work quantifies the large volume of tweets discussing businesses, published by regular users on a daily basis and validates the need for a Twitter-based consumer business tool, which can provide insight about the Twitter consumer base and also detect stories trending on Twitter pertaining to companies. We have developed a scalable, distributed and fault-tolerant framework in order to collect, store and analyze tweets at a scale that an actual implementation of such a business tool would require. Our framework is generic enough to be adapted to any future Twitter-based research and we were able to use it to collect 100 million tweets, discussing a set of 70 companies over a month. Despite collecting such a specific dataset, Twitter is a social network at its core and almost half of the data is made up of tweets that are personal conversations, users' self promotion, random observations or spam. Such tweets are 'irrelevant' for our consumer-business use case, as we only want to consider 'relevant' tweets pertaining to a specific topic. We have built a relevance-based tweet classifier that is able to distinguish between 'relevant' and 'non-relevant' tweets with almost 80% accuracy. Not only does this improve the quality of any further analysis because of a cleaner tweet stream, but also yields a potential 40% saving in processing and storage costs by removing 'irrelevant' tweets from the working set. Being able to rank Twitter users, mentioning a company, by their influence in the network and expertise in the company's domain is extremely useful as these individuals can be leveraged to effectively build product awareness, brand buzz and new sales. Simply ranking these users by how often they mention a company typically yields spurious accounts like bots and automatic re-tweeters or Twitter accounts that are owned by the company itself. We have designed an algorithm that applies item-item collaborative filtering, in reverse, to the lists of users mentioning companies similar to the one in question. The users that tweet about a majority of the similar companies with high volume implicitly display expertise in the company's domain. Ordering these users by an average of their daily tweet volumes across the similar companies yields a significantly more meaningful list of 'top tweeters' for a company. Recently news has started to break faster on Twitter, than conventional media, and businesses need to be aware of what consumers are saying about their products and react promptly, especially to any emerging negative information or opinions. We have developed an algorithm for automatically detecting emerging topics, hot topics or buzz in a stream of tweets mentioning a specific company. It uses a variation of dictionary learning to identify sudden changes in tweet content and volume. We were able to accurately identify 15 real-life trending stories, with 5 false positives, in our dataset, which captures a month of company-related tweets. The experimental and empirical evaluation of the performance of the relevance-based tweet classifier, the algorithm to identify a company's true 'top tweeters' and the emerging story detection algorithm, illustrate the potential effectiveness and utility of a Twitter-based tool to track the pulse of consumer businesses.

Bibliography

[1] James Allan, editor. *Topic detection and tracking: event-based information organization.* Kluwer Academic Publishers, Norwell, MA, USA, 2002.

[2] Amazon. Introduction to Amazon Mechanical Turk. `http://docs.aws.amazon.com/AWSMechTurk/2008-08-02/AWSMechanicalTurkRequester/IntroductionArticle.html`.

[3] Apache. Apache Avro? 1.7.4 Documentation. `http://avro.apache.org/docs/current/#intro`.

[4] Apache. Chapter 9. Architecture. `http://hbase.apache.org/book/architecture.html#arch.overview`.

[5] Apache. HDFS Architecture Guide. `http://hadoop.apache.org/docs/r1.0.4/hdfs_design.html`.

[6] David M. Blei, Andrew Y. Ng, and Michael I. Jordan. Latent dirichlet allocation. *J. Mach. Learn. Res.*, 3:993–1022, March 2003.

[7] Kevin Gimpel, Nathan Schneider, Brendan O'Connor, Dipanjan Das, Daniel Mills, Jacob Eisenstein, Michael Heilman, Dani Yogatama, Jeffrey Flanigan, and Noah A. Smith. Part-of-speech tagging for twitter: annotation, features, and experiments. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies: short papers - Volume 2*, HLT '11, pages 42–47, Stroudsburg, PA, USA, 2011. Association for Computational Linguistics.

[8] Google. Guava-Libraries. `https://code.google.com/p/guava-libraries/`.

[9] Hayley Tsukayama. Twitter turns 7: Users send over 400 million tweets per day. `http://articles.washingtonpost.com/2013-03-21/business/37889387_1_tweets-jack-dorsey-twitter`, March 2013.

[10] Thomas Hofmann. Probabilistic latent semantic indexing. In *Proceedings of the 22nd annual international ACM SIGIR conference on Research and development in information retrieval*, SIGIR '99, pages 50–57, New York, NY, USA, 1999. ACM.

[11] Liangjie Hong and Brian D. Davison. Empirical study of topic modeling in twitter. In *Proceedings of the First Workshop on Social Media Analytics*, SOMA 2010, pages 80–88, New York, NY, USA, 2010. ACM.

[12] Patrik O. Hoyer. Non-negative matrix factorization with sparseness constraints. *J. Mach. Learn. Res.*, 5:1457–1469, December 2004.

[13] Xia Hu, Nan Sun, Chao Zhang, and Tat-Seng Chua. Exploiting internal and external semantics for the clustering of short texts using world knowledge. In *Proceedings of the 18th ACM Conference on Information and Knowledge Management*, CIKM '09, pages 919–928, New York, NY, USA, 2009. ACM.

[14] Jeff Bullas. 11 New Twitter Facts, Figures and Growth Statistics plus [INFOGRAPHIC]. `http://www.jeffbullas.com/2011/09/21/11-new-twitter-facts-figures-and-growth-statistics-plus-infographic/`, September 2011.

[15] Karl Hodge. 10 news stories that broke on Twitter first. `http://www.techradar.com/us/news/world-of-tech/internet/10-news-stories-that-broke-on-twitter-first-719532`, September 2010.

[16] Shiva Prasad Kasiviswanathan, Prem Melville, Arindam Banerjee, and Vikas Sindhwani. Emerging topic detection using dictionary learning. In *Proceedings of the 20th ACM international conference on Information and knowledge management*, CIKM '11, pages 745–754, New York, NY, USA, 2011. ACM.

[17] Kiji. The Kiji Project. `http://www.kiji.org/`.

[18] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. *J. ACM*, 46(5):604–632, September 1999.

[19] Kywe, Su Mon and Hoang, Tuan-Anh and Lim, Ee-Peng and Zhu, Feida. On recommending hashtags in twitter networks. In *Proceedings of the 4th international conference on Social Informatics*, SocInfo'12, pages 337–350, Berlin, Heidelberg, 2012. Springer-Verlag.

[20] Leena Rao. Twitter Seeing 90 Million Tweets Per Day, 25 Percent Contain Links. `http://techcrunch.com/2010/09/14/twitter-seeing-90-million-tweets-per-day/`, September 2010.

[21] mrflip. TWITTER CENSUS: SMILEYS. `http://www.infochimps.com/datasets/twitter-census-smileys`.

[22] Nathan Marz. Storm. `http://storm-project.net/`.

[23] Saša Petrović, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, HLT '10, pages 181–189, Stroudsburg, PA, USA, 2010. Association for Computational Linguistics.

[24] Xuan-Hieu Phan, Le-Minh Nguyen, and Susumu Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web*, WWW '08, pages 91–100, New York, NY, USA, 2008. ACM.

[25] Daniel Ramage, Susan T. Dumais, and Daniel J. Liebling. Characterizing microblogs with topic models. In *ICWSM*, 2010.

[26] Mehran Sahami and Timothy D. Heilman. A web-based kernel function for measuring the similarity of short text snippets. In *Proceedings of the 15th international conference on World Wide Web*, WWW '06, pages 377–386, New York, NY, USA, 2006. ACM.

[27] Hassan Saif, Yulan He, and Harith Alani. Semantic sentiment analysis of twitter. In *Proceedings of the 11th international conference on The Semantic Web - Volume Part I*, ISWC'12, pages 508–524, Berlin, Heidelberg, 2012. Springer-Verlag.

[28] Dequan Zheng Yao Meng Yingju Xia Shu Zhang, Jianwei Wu and Hao Yu. Supervised and Semi-supervised Methods based Organization Name Disambiguity. In *Proceedings of the 25th Paci?c Asia Conference on Language, Information and Computation*, pages 615–621, Singapore, December 2011.

[29] Twitter. Browse Categories. `https://twitter.com/who_to_follow/interests`.

[30] Twitter. Entities. https://dev.twitter.com/docs/platform-objects/entities.

[31] Twitter. GET statuses/sample. https://dev.twitter.com/docs/api/1.1/get/statuses/sample.

[32] Twitter. GET statuses/user_timeline. https://dev.twitter.com/docs/api/1.1/get/statuses/user_timeline.

[33] Twitter. GET users/suggestions. https://dev.twitter.com/docs/api/1.1/get/users/suggestions.

[34] Twitter. GET users/suggestions/:slug. https://dev.twitter.com/docs/api/1.1/get/users/suggestions/%3Aslug.

[35] Twitter. Google Domestic Trends. https://www.google.com/finance/domestic_trends?ei=H42QUeikIIat0AGfBw.

[36] Twitter. Places. https://dev.twitter.com/docs/platform-objects/places.

[37] Twitter. POST statuses/filter. https://dev.twitter.com/docs/api/1.1/post/statuses/filter.

[38] Twitter. Tweets. https://dev.twitter.com/docs/platform-objects/tweets.

[39] Twitter. Users. https://dev.twitter.com/docs/platform-objects/users.

[40] Twitter4j. Twitter4j. http://twitter4j.org/en/index.html.

[41] weka. Use WEKA in your Java code. http://weka.wikispaces.com/Use+WEKA+in+your+Java+code.

[42] Zhiheng Xu, Long Ru, Liang Xiang, and Qing Yang. Discovering user interest on twitter with a modified author-topic model. In *Proceedings of the 2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology - Volume 01*, WI-IAT '11, pages 422–429, Washington, DC, USA, 2011. IEEE Computer Society.