Breaking Hash-Tag Detection Algorithm for Social Media (Twitter)

by

Piyush Awasthi

A Dissertation Presented in Partial Fulfillment
of the Requirement for the Degree
Master of Science

Approved April 2015 by the
Graduate Supervisory Committee:

Hasan Davulcu, Chair
Hanghang Tong
Arunabha Sen

ARIZONA STATE UNIVERSITY

May 2015

ABSTRACT

In trading, volume is a measure of how much stock has been exchanged in a given period of time. Since every stock is distinctive and has an alternate measure of shares, volume can be contrasted with historical volume inside a stock to spot changes. It is likewise used to affirm value patterns, breakouts, and spot potential reversals. In my thesis, I hypothesize that the concept of trading volume can be extrapolated to social media (Twitter).

The ubiquity of social media, especially Twitter, in financial market has been overly resonant in the past couple of years. With the growth of its (Twitter) usage by news channels, financial experts and pandits, the global economy does seem to hinge on 140 characters. By analyzing the number of tweets hash tagged to a stock, a strong relation can be established between the number of people talking about it, to the trading volume of the stock.

In my work, I overt this relation and find a state of the breakout when the volume goes beyond a characterized support or resistance level.

DEDICATION

I dedicate my dissertation work to my family and many friends. An uncommon feeling of appreciation to my cherishing folks, Anupam Goel, Nishant Mahajan, Pratik Karnawat, Manish Trivedi and Ankit Sharma whose uplifting statements and push for tirelessness ring in my ears.

I dedicate this work and give special thanks to my closest companion Vinuta Chopra and my best friend Abhimanyu Gupta for being there for me all through the whole Masters program. Both of you have been my best team promoters.

ACKNOWLEDGEMENTS

I sincerely thank my advisor Dr. Hasan Davulcu for his continued guidance, support and encouragement during my masters and while writing this thesis. I also would like to thank Dr. Hanghang Tong & Dr. Arunabha Sen for being on my thesis supervisory committee. I would like to thank all my lab-mates Hana, Sravan and all members of Centre for Strategic Communication (CSC) research lab of Hugh Downs School of Human Communication at Arizona State University for making this journey so exciting and making this a great learning experience. I am very grateful for the love and the unconditional support of my family and friends.

TABLE OF CONTENTS

Page

LIST OF TABLES vi

LIST OF FIGURES vii

CHAPTER

1 INTRODUCTION 1

1.1 Motivation 2

1.1.1 Word of Mouth 2

1.1.2 Twitter: A tool for Word of Mouth 2

1.1.3 Word of Mouth, Twitter & Stocks 3

1.1.4 Example 4

1.2 Organization 5

1.2.1 Methodology 5

1.2.2 Document Outline 5

2 BACKGROUND 6

2.1 Twitter 6

2.2 HashTags 6

2.3 Dow Jones 6

2.4 TwitterStreaming API 7

2.4.1 Difference between Streaming and Rest 8

2.4.2 Connecting to Streaming Endpoint 8

2.4.3 Authentication 9

2.4.4 Connections 9

2.4.5 Disconnections 10

2.4.6 Reconnecting 11

2.4.7 Connection Churn 11

CHAPTER Page

2.4.8 Rate Limiting 12
2.4.9 Best Practices 12
2.5 Twitter4J 14
2.5.1 Streaming API 14
3 LITERATURE REVIEW 16
4 BREAKING HASH-TAG DETECTION 18
4.1 Introduction 18
4.2 Twitter Crawling 19
4.3 Breaking Point Detection 19
4.4 URL crawling & Scraping 20
4.5 Bypassing Advertisements 21
4.6 Domain Filtering 23
5 FUTURE WORK 25
5.1 Time Index 25
5.2 Online Application 25
REFERENCES 26
APPENDIX
A DATABASE DESIGN 27

LIST OF TABLES

Table Page

2.1 Types of Twitter Streaming Endpoints 7

2.2 Study Design 9

2.3 Http Error Codes 13

3.1 30 Dow Jones Company symbols 17

LIST OF FIGURES

Figure		Page
2.1	Twitter REST API connection.	8
2.2	Twitter Streaming API Architecture.	9
4.1	Process flow of Algorithm	18
4.2	Screenshot of Advertisement page with Timer	22
4.3	Screenshot of Advertisement page with Skip-Ad button	22

Chapter 1

INTRODUCTION

On August 13, 2013, Carl Icahn made an announcement about Apple's position over Twitter. Within minutes, Apple's stock shot up gaining $17 billion in marketcap. On May 28, 2013, there was a train crash in Maryland. The local respondents started tweeting about that, and within the next 90 minutes , the stock lost $500 million marketcap. On the other hand, a Twitter hoax on April 23, 2013, claiming about president Obama was injured in an explosion in White House caused the Dow Jones industrial average to drop temporarily by 150 points, erasing $136 billion in market value. The ubiquity of social media especially Twitter, in financial market has been overly resonant in past couple of years. With the growth of its (Twitter) usage by news channels, financial experts and pandits, global economy does seem to hinge on 140 characters.

Traditional method of stock analysis involved analysts brainstorming past factual papers of the companies and remotely trying to predict the behavior of the stock upon their best knowledge. The mechanism is well oiled and functioning for the low frequency trades as the behavior is stable or in other words shows very less skewness in majority. However, this accuracy seems to falter for high trading stocks where trades are based on keywords within milliseconds. The decision-making becomes a complex problem with insufficient data as input and less accurate results as output. With the advancement in the fields of Data Mining and Machine Learning, and with the companies sharing their data over the digital net, algorithms have been designed and developed to use these data sources to analyze and execute trades. Companies like Goldman Sachs use these high functioning algorithms with an exogenous feedback

from analysts to provide analysis and prediction. The use of this hybrid approach is becoming more widespread and researches are already underway in automatizing this process with highest accuracy.

One of the key components in automatizing the process of prediction is detecting that one point where the change in behavior is most likely to happen. In my research, I design an algorithm to detect this point, which inflates or deflates the behavior of stock over the tweets feed from Twitter.

1.1 Motivation

1.1.1 Word of Mouth

Informal showcasing is a standout amongst the most alluring exercises to brands, why? Since exploration on trust demonstrates that customers (people like you and me) believe the conclusions of individuals we know more than any other individual. It comprehends course, consider whenever you're going to purchase an auto, who's assessment would you say you are going to trust, those of your companions or the sentiment of the business fellow speaking to the item?

1.1.2 Twitter: A tool for Word of Mouth

Twitter, which I'm seeing casual details of around 288 million monthly active users, has kept on showing its viral abilities, with Motrin mother's image punking of a commercial to news being spread about common debacles speedier than customary news, this toolset permits substance to spread quicker and more distant than we've ever seen. Observing how Al Gore's TV coordinated tweets live on their TV telecast and how CNN and CSPAN said this micro blogging administration amid the decision months is a gesture to its energy. In a few ways, in length structure blog entries like

this appear to be so much slower and trudging contrasted with how rapidly data can go back and forth in Twitter.

1.1.3 Word of Mouth, Twitter & Stocks

Stock costs change consistently as a consequence of business powers. By this we imply that stock costs change in view of supply and demand. On the off chance that more individuals need to purchase a stock (demand) than sell it (supply), then the value climbs. Alternately, if a greater number of individuals needed to sell a stock than purchase it, there would be more noteworthy supply than demand, and the value would fall. Seeing supply and demand is simple. What is hard to fathom is the thing that makes individuals like a specific stock and aversion another stock. This boils down to making sense of what news is positive for an organization and what news is negative.

In customary frameworks this positive or the negative news goes through organization sites or authority stock news channels, which is then broke down by the investigators to anticipate the stock. The main thrust again being whether a speculator would be occupied with purchasing the stock in the wake of listening to the news or not. This framework extremely well taps the information of examiners with great forecast score yet imagine a scenario in which there were intends to take advantage of the opinions of expert and additionally speculators. This positively can enhance the precision of expectation. In addition, with customary framework, a speculator is forgotten with data from distinctive experts and a choice to run with which expert is one-sided with their information & involvement in the individual field.

With 288 million month to month dynamic clients, Twitter has turned into a solid apparatus of communicating one's feeling. Stock exchange pandit/specialists use it to post their examination and speculators tail them to choose whether to purchase

stocks or drop stocks. In addition, an open spot to post singular feelings pulls in the speculators/investors to express their perspectives. Hitherto, my work can be confused as only one more semantic examination of twitter content of the specialists and speculators. Truth being, my proposal just focusses on the measure of discussion over a certain stock. A question that takes after straightforwardly to the last sentence is, the way to find that certain stock?- An issue of right group of onlookers. Twitter addressed this by reverse engineering. They developed hash tags in 2007 - a tool to find right source of information than audience or in another words it's way to organize content where it finds its way to people who are interested in listening/reading about it.

In this way, by close observing the discussion, hash tagged to distinctive stocks, one can undoubtedly quantify the essential insights i.e. which stock is being talked most, what is the change in measure of discussion and so on. In this work, I estimate that change in measure of tweets over a certain stock over a certain period of time could conceivably anticipate whether a change is normal in stocks conduct (this shouldn't be mistaken for forecast of measure of progress/digress).

1.1.4 Example

A stock showcasing no/less action over twitter for recent past hours begin transforming its conduct. Tweets begin pouring in and there is a high swelling in measure of individuals discussing it. A simple supposition taking after this movement would be that there is something incorrect/right with this separate stock. As settled that informal discussions influences the cost of stock, we model and develop this idea to tell whether a stock will break (have a tendency to change its conduct) or not.

1.2 Organization

1.2.1 Methodology

The work proposed and presented in this thesis is done following an Incremental model of Software Development Life Cycle (SDLC). The algorithm proposed is done in an evolutionary fashion. So rather than considering the entire work as a single project, it is divided into multiple projects which in turn follow the usual phases: requirement gathering, design, implementation and evaluation, as identified in a typical SDLC. Subsequent projects take the results of previous projects as inputs and the software continues to evolve until the final product.

1.2.2 Document Outline

With the methodology explained, the rest of the document is organized in the following way. Chapter 2 gives an overview of the Background information about Dow Jones and Twitter API along with its protocol to obtain data. Also it explains the libraries used with sample code to setup the twitter crawling process.

Chapter 3 provides the literature review of the earlier work.

Chapter 4 explains Breaking Topic Detection algorithm. It has been divided to explain different sections that deals with different requirements.

Chapter 5 gives a formal conclusion of this thesis along with a brief outline on the future work that can be done to improve Breaking Point Detection algorithm in capturing breaking stocks which can provide greater accuracy of market value prediction.

Chapter 2

BACKGROUND

2.1 Twitter

Twitter is an online interpersonal interaction benefit that empowers individuals to send and read short 140-character messages called "tweets". Registered users can read and post tweets, yet unregistered users can just read them. Clients access Twitter through the site interface, SMS, or cell phone application. Twitter Inc. is situated in San Francisco and has more than 25 workplaces around the globe.

Currently there are 288 million monthly active users, with 500 million tweets sent per day supporting 33 languages on Twitter.

2.2 HashTags

A hashtag is a sort of name or metadata tag utilized on social network and microblogging services, which makes it simpler for people to discover messages with a particular subject or substance. People make and utilize hashtags by setting the hash character (or number sign) # before a word or unspaced expression, either in the primary content of a message or toward the end. Scanning for that hashtag will then present every message that has been labeled with it.

2.3 Dow Jones

"The Dow Jones Industrial Average, also called the Industrial Average, the Dow Jones, the Dow Jones Industrial, the Dow 30, or simply the Dow, is a stock market index, and one of several indices created by Wall Street Journal editor and Dow Jones

& Company co-founder Charles Dow. The industrial average was first calculated on May 26, 1896. Currently owned by S&P Dow Jones Indices, which is majority owned by McGraw-Hill Financial, it is the most notable of the Dow Averages, of which the first (non-industrial) was first published on February 16, 1885." [1] The averages are named after Dow and one of his business partners, statistician Edward Jones.

It is a record that shows how 30 substantial freely possessed organizations situated in the United States have exchanged amid a standard exchanging session in stocks. It is the second most seasoned U.S. business sector record after the Dow Jones Transportation Average, which was likewise made by Dow.

2.4 TwitterStreaming API

The Streaming APIs give programmers an ability to retrieve Twitter's stream with least amount of low latency. It is a different service than REST without involving any of REST's overhead. A successful connection results in messages being pushed which are different Tweets or any other event that occurs on Twitter. The Different type of streaming endpoints offered by Twitter are :

Table 2.1: Types of Twitter Streaming Endpoints

Public Streams	It allows to capture public streams on Twitter. It is most suitable for cases which involves specific users or topics.
User Streams	It provides the data of a single respective user. This stream is most suitable when following a specific user.
Site Streams	This streaming endpoint is more of requirement when a server on behalf of many users wants to connect to Twitter user.

[1] http://answerparty.com/question/answer/what-did-the-dow-jones-industrial-average-close-at-up-or-down

2.4.1 *Difference between Streaming and Rest*

Connecting with the streaming API requires keeping a determined HTTP association open. As a rule this includes considering your application uniquely in contrast to on the off chance that you were connecting with the REST Programming interface. For an illustration, which takes user request to receive some information. This request only needs one or more request rather than a persistent connection:

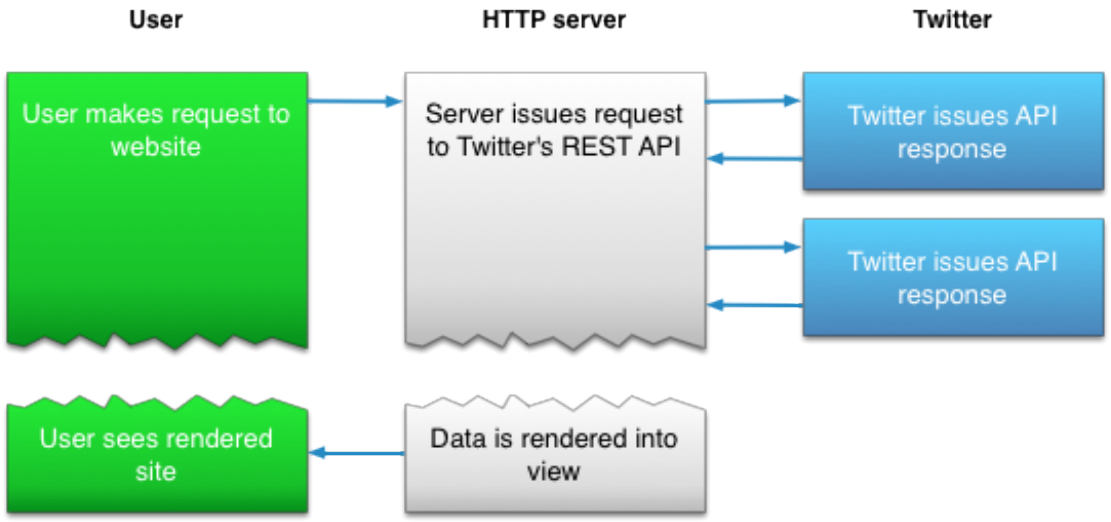

Figure 2.1: Twitter REST API connection.

Streaming API architecture is different that REST. In Streaming, the response is not processed based on user's request. Rather, the whole process of connecting with Streaming is a different process done in separate with HTTP requests (See Figure 2.2 for picture representation).

2.4.2 *Connecting to Streaming Endpoint*

A connection to streaming API requires a dedicated and persistent HTTP request. The response from the request is required to be processed incrementally.

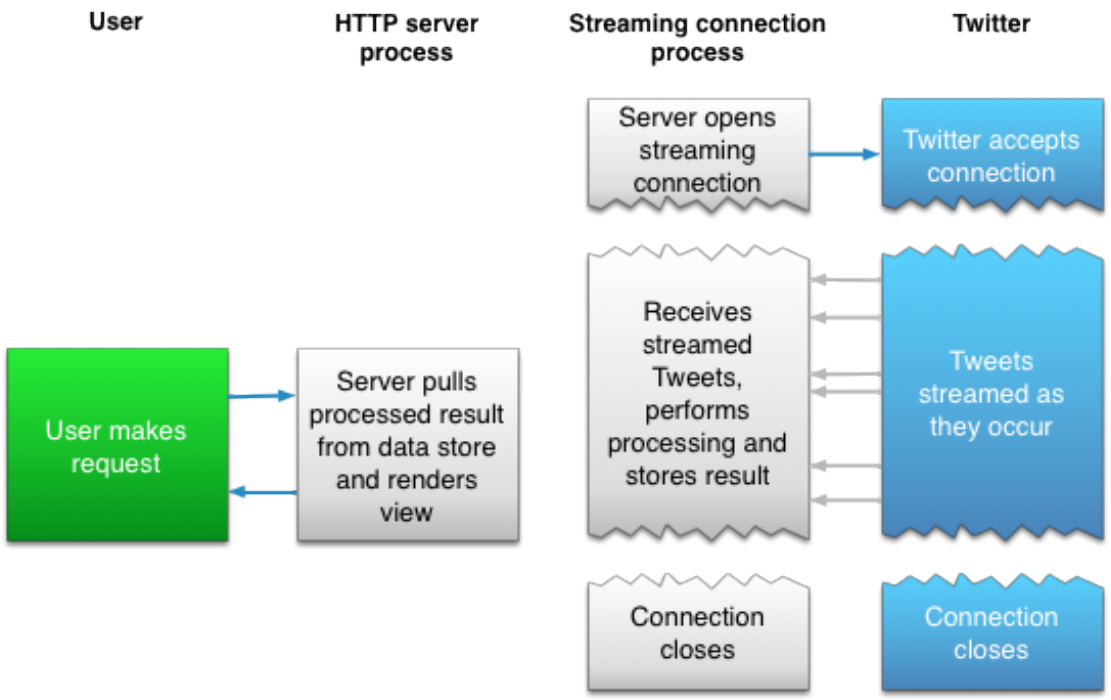

Figure 2.2: Twitter Streaming API Architecture.

2.4.3 Authentication

Streaming API supports following authentication methods -

Table 2.2: Study Design

Auth Type	Supported API	Description
OAuth	Public Streams, User Streams, Site Streams	Request must be authorized according to the OAuth specification

2.4.4 Connections

To interface with the Streaming API, structure a HTTP ask for and expend the subsequent stream the length of is viable. The servers will hold the connection open

inconclusively, notwithstanding server-side mistake, unreasonable customer side slack, system hiccups, routine server support or copy logins.

The technique to structure a HTTP request and parse the reaction will be distinctive for each language or system, so counsel the documentation for the HTTP library you are utilizing.

Few HTTP client libraries give back the response body after the connection has been shut by the server. These clients won't work for getting to the Streaming API. You must utilize a HTTP client that will return reaction information incrementally. Most powerful HTTP client libraries will give this usefulness.

2.4.5 Disconnections

The Streaming connection is limited by Twitter. Twitter has the right to disconnect any connection in following cases -

- Multiple connection requests using one OAuth connection credentials. Twitter terminates the oldest connection in this case.
- If the user stops parsing the streaming data. Twitter disconnects this type of connection under the assumption that the user isn't using the stream.
- If the user is processing the data very slowly. Resulting in the overflow of data in the backup queue.
- When the server is restarted.
- If Twitter changes its network configuration which is very rare.

Setting up a timer of 90 second on TCP level socket or application level. If there is no stream for 90 seconds, it is recommended to disconnect and reconnect. The

Streaming API usually sends a newline every 30 seconds to keep alive the connection.You ought to hold up no less than three cycles to counteract spurious reconnects in the occasion of system blockage, nearby CPU starvation, neighborhood GC delays, and so on.

2.4.6 Reconnecting

Reconnection must be established immediately if the connection drops. If reconnection fails, reduce the rate of reconnection tries based on the type of error experienced:

- Linearly back off for TCP/IP level network errors. Such errors are mostly temporary and tend to clear quickly. Increase the delay in reconnection by 250ms each attempt, up to 16 seconds.
- Exponentially back off for HTTP errors. Start by waiting for 5 seconds and double each attempt, up to 320 seconds.
- Exponentially back off for HTTP 420 errors. Start by waiting for a minute and double each attempt. Every HTTP 420 errors received increases the waiting time until limiting the rate will no longer be in effect for your account.

2.4.7 Connection Churn

Server resources are wasted by repeatedly opening and closing a connection (churn). Connections must be as stable and long-lived as possible.

Mobile (cellular network) connections from mobile devices must be avoided. WiFi is generally OK.

In cases where the user may quit the application quickly, delay opening a streaming connection.

Flaky connections must be detected if the client works in an environment where the connection quality changes over time. After detection, fall back to REST polling until the connection quality improves.

2.4.8 Rate Limiting

Customers who don't execute backoff and endeavor to reconnect as frequently as could be expected under the circumstances will have their associations rate restricted for a little number of minutes. Rate restricted customers will get HTTP 420 reactions for all association demands.

Customers who break an association and after that reconnect much of the time (to change inquiry parameters, for instance) run the danger of being rate constrained.

Twitter does not make open the quantity of association request that will bring about a rate restricting to happen, yet there is some resistance for testing and improvement. A couple of dozen association endeavors now and again won't trigger a cutoff. In any case, it is fundamental to stop further association attempts for a couple of minutes if an HTTP 420 reaction is gotten.improves.

2.4.9 Best Practices

Following some of the best practices that can be practiced during establishing connection and keeping it alive without any lag, delays and disconnection:

- **Test backoff strategies** - Invalid authorization credentials and examination of the reconnect attempts is a good way to test a backoff implementation. An appropriate implementation will not get any 420 responses.

- **Issue alerts for multiple reconnects** - If the upper threshold of a client is reached of its time between reconnections, notifications must be sent so you can

triage the issues affecting your connection.

- **Handle DNS changes** - Testing must be done to see if the client process honors the DNS Time To live (TTL). A resolved address will be cached by a couple of stacks cache during the process. The DNS changes within the proscribed TTLwill not be picked up. Service disruptions are caused on the client because of such aggressive caching while Twitter shifts load between IP addresses.

- **User Agent** - Issues on Twitter will be detected by ensuring the User-Agent HTTP header includes the clients version. X-User-Agent header mus be sent if your environment precludes setting the User-Agent field.

- **HTTP Error Codes** - Twitter usually provides a little description with every code. For further reference please refer Table 2.3

Table 2.3: Http Error Codes

Status	Text	Description
200	Success	Self evident
401	Unauthorized	HTTP authentication failed due to either: • Invalid auth credentials, or an invalid OAuth request. • Out-of-sync timestamp in OAuth request. • Too many incorrect passwords entered or other login rate limiting.
403	Forbidden	The connecting account is not permitted to access.

404	Unknown	There is nothing at the URL.
406	Not Acceptable	When atleast one parameter is invalid in the request.
413	Too Long	This error occurs when the parameter list is too long.
416	Range Unacceptable	For example, an endpoint returns this status if: • A count parameter is specified but the user does not have access to use the count parameter. • A count parameter is specified which is outside of the maximum/minimum allowable values.
420	Rate Limited	If a client tries to make too many connection request over a short period of time or tries to use same credentials for multiple log ins.
503	Service Unavailable	The server is too loaded with the requests.

2.5 Twitter4J

An unofficial Java library for the Twitter API is Twitter4J. It can be used to easily integrate the Java application. It is an unofficial library.

2.5.1 Streaming API

TwitterStream class has many methods prepared for streaming API. To do so, we need a class implementing StatusListener. Twitter4J will create a thread and consume the stream. Provided below a reference code from Twitter4J to implement the status listener.

```
public static void main(String[] args) throws
    ↪ TwitterException, IOException{
    StatusListener listener = new StatusListener(){
        public void onStatus(Status status) {
            System.out.println(status.getUser().getName() + "
                ↪ : " + status.getText());
        }
        public void onDeletionNotice(StatusDeletionNotice
            ↪ statusDeletionNotice) {}
        public void onTrackLimitationNotice(int
            ↪ numberOfLimitedStatuses) {}
        public void onException(Exception ex) {
            ex.printStackTrace();
        }
    };
    TwitterStream twitterStream = new TwitterStreamFactory().
        ↪ getInstance();
    twitterStream.addListener(listener);
    // sample() method internally creates a thread which
        ↪ manipulates TwitterStream and calls these adequate
        ↪ listener methods continuously.
    twitterStream.sample();
}
```

Chapter 3

LITERATURE REVIEW

The vastness of Twitter in social media and its involvement in almost every field be it news, sports or personal have gained lot of attention especially in stock market. Lot of research on how the social media deviates the market has been done with results showing positivism. Early research were based on as simple as bag of words textual classification on the data provided by the companies in media and public forums. Results showed no greater improvement than the chances of guessing but paved path for other researchers to improve the classifiers. Existing approaches mainly differ in three aspects, 1. Text mining approach, 2. Feature extraction & processing and 3. machine learning algorithm.

Bag of Words classification ignores the underlying semantics in the text for e.g., if increase is mentioned in conjunction with cost or with earnings. Hence, algorithms with complex feature extraction and modeling were developed to provide greater accuracy (Hagenau et al. (2013)). Hagenau et. al work proposed greater accuracy with complex feature selection process and 2-gram classification to unhide underlying semantics.Another vertical of research, to capture the sentiments from the real time data instead of static showed promising correlation between the direction of market sentiment and direction of stock market (Nagar and Hahsler (2012)). Nagar et al. display an automated content mining based way to deal with aggregated news stories from diverse live sources and make a News Corpus. The Corpus is sifted down to pertinent sentences and investigated utilizing Natural Language Processing (NLP) procedures. A similar approach of capturing the stock behavior of previous month and predicting the behavior of the next month using logistic regression model on news

articles was presented by Gong and Sun (2009).

My thesis, will be in continuation to the previous work and concentrates on analyzing Twitter trend and designing an algorithm to shorten the prediction time for the stocks on real time data. This work currently focuses on 30 Dow Jones company stocks (Please refer Table 3.1 for their symbols).

Table 3.1: 30 Dow Jones Company symbols

NYSE:MMM	NYSE:AXP	NYSE:T	NYSE:BA
NYSE:CAT	NYSE:CVX	NASDAQ:CSCO	NYSE:KO
NYSE:DD	NYSE:XOM	NYSE:GE	NYSE:GS
NYSE:HD	NASDAQ:INTC	NYSE:IBM	NYSE:JNJ
NYSE:JPM	NYSE:MCD	NYSE:MRK	NASDAQ:MSFT
NYSE:NKE	NYSE:PFE	NYSE:PG	NYSE:TRV
NYSE:UNH	NYSE:UTX	NYSE:VZ	NYSE:V
NYSE:WMT	NYSE:DIS		

Chapter 4

BREAKING HASH-TAG DETECTION

4.1 Introduction

Breaking Topic Detection algorithm is sectioned into 4 parts - 1. The web crawler extraction from the Twitter stream based on HashTags of the 30 DOW JONES company. Table 1 shows all the HashTags under consideration. 2. Breaking point detection, a running calculation to detect the ball point where the analysis & prediction will be made. 3. Web crawling & scraping of the URLs provided on the tweets. 4. Text classification of the text & prediction. Figure 1 below gives the flow of the process.

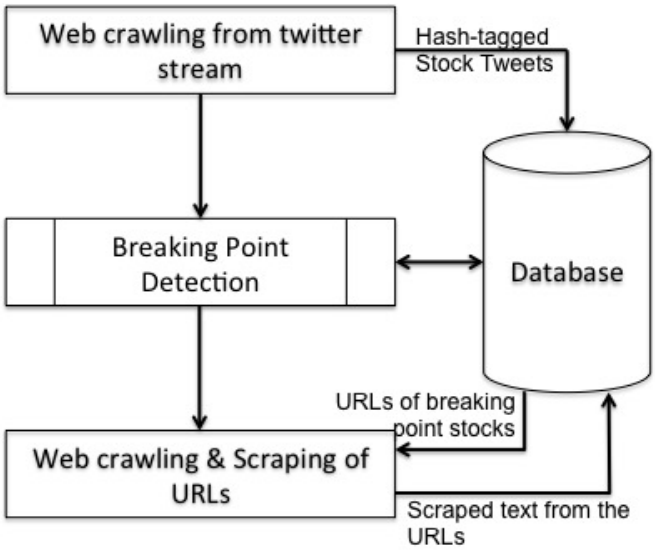

Figure 4.1: Process flow of Algorithm

The classification step has been skipped out of the flow chart as it excludes my responsibility of the work. Further I explain each of the process in more detail.

4.2 Twitter Crawling

We use TwitterStream API to catch all the real time tweets based on the 30 Hash-tag symbols provided in table 3.1. All the relevant tweets are pushed to database with their timestamp and tinyURL if mentioned.

Twitter regularizes the tweet size to 140 characters, constraining pandits/experts of stock market to put down their whole analyses or story. More than often post from these experts includes the URL to their actual story or analyses. These URL becomes very important because once detected a break point, these URL are crawled to get all the text on which a text classifier is run to provide the prediction.

Algorithm 1: TweetExtraction

Given: OAuth Parameters for Twitter Stream Connection

1. Establish Connection with Twitter Stream
2. for each *Status s* do
   (a) for each *30 Stocks st* do
      i. if *status s* contains *stock st*, insert in tweets table

4.3 Breaking Point Detection

Breaking point detection is a running calculation that is performed at first second of every new hour for all 30 stocks. This calculation is dependent on a 20-hour window i.e. the data for the last 20 hours. At every new hour we count the number of tweets of all 30 stocks for the past hour. The tweet count is then aggregated with last 19 hours of data, which makes it a total of 20-Hour window to calculate the average and standard deviation. Now, whether a stock has broken from its behavior or not depends on the following equation -

$$current\_hour\_count >= mean\_of\_20hours + 2 * stdDev\_of\_20hours$$

If the above value is true then the state is declared as breaking point. To explain the above procedure consider this example.

Lets consider NYSE:MSFT (Microsoft) stock for this particular example. To follow the example more easily, lets define a timeline of March 22, 2014 09:00:00. At 09:00:01 AM the calculation will be performed. The number of tweets mentioning NYSE:MSFT for the hour 08:00:00 to 09:00:00 will be counted and stored following up with moving average & standard deviation calculation for the 20-hour window i.e. March 21, 2014 13:00:00 to March 22, 2014 09:00:00. Based on the calculated moving average and standard deviation, breaking point calculation is performed.

Algorithm 2: Breaking Point Detection

Given: Tweets table

1. At new second of every hour do
   (a) for each *Stock st* do
      i. Calculate number of tweets for last hour T_{lh}
      ii. Calculate *moving average* M_{avg} and *standard deviation sd* for last 20 hours
      iii. if $T_{\text{lh}} \geq M_{\text{avg}} + 2sd$, then *breaking point bp* is *1* else *0*
      iv. Update hourly table with breaking point value

4.4 URL crawling & Scraping

Once a breaking point is detected for a stock, the program extracts all the URLs from the database for the last one hour i.e. from 08:00:00 to 09:00:00 in our earlier presented example. Twitter has a character restriction of 140 characters, which restricts the users text as well as the URL. To solve the URL restriction TinyURLs were innovated which shortens a long URL to a short URL which when

clicked automatically redirects to the actual URL. For e.g. a TinyURL for `http://tex.stackexchange.com/questions/54328/long-code-that-needs-to-wrap` is `http://tinyurl.com/lrf28ar`. These tiny URLs are first converted to extended URLs followed with crawling and all text scraping from these URLs.

Algorithm 3: URL Extraction

Given: Hourly Table

1. For each *breaking point bp* do
   (a) $domaincountlistD_{\text{count}} \leftarrow DomainCalculation()$
   (b) For each URL in D_{count} do
      i. Crawl URL
      ii. if content present then create text file

4.5 Bypassing Advertisements

In our study we found that not all TinyURLs lead to actual URL page, instead they lead to ad-pages. Reason being, most of TinyURL are third party generated and to gain revenue these short URLs lead to ad pages first and then eventually leading to the actual pages after few seconds. Few sites also have Skip-Ad button, which only gets activated after few seconds of advertisement. Figure 4.2 & 4.3 gives example for both of the cases.

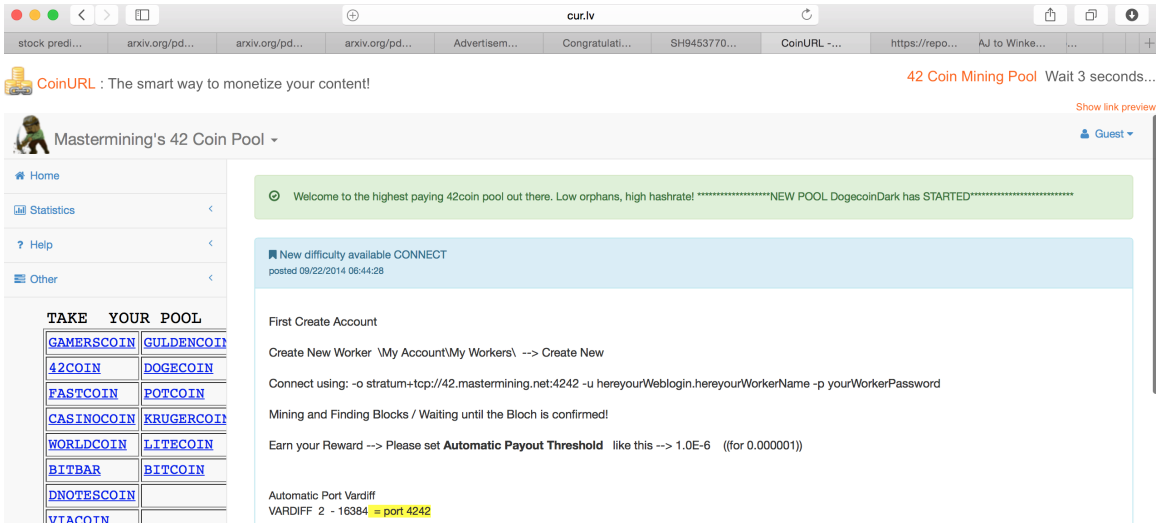

Figure 4.2: Screenshot of Advertisement page with Timer

Figure 4.3: Screenshot of Advertisement page with Skip-Ad button

Bypassing these advertisement programmatically becomes a tedious task as the source code of these pages is dynamically created using javascript. Normal web crawlers or bypassing method fail to overcome this problem as there is no source code to work with. To overcome this problem we use Selenium Library. Because Selenium allows creating an instance of web browser with specific URL, the dynamically generated source code can then be downloaded and manipulated to bypass the advertisement.

In my particular work, after creating the instance of browser through selenium with the TinyURL, I wait for 15 seconds at least before capturing the source code of the page, as the Skip-Ad buttons only gets activated then. After the source code is grabbed, I drill down to find a button with name as Skip-Ad or id as Skip-Ad. Once found, I use Selenium library to programmatically click on it to redirect to actual page from where I scrape the content to store in a text file that is then used as input to classifier.

4.6 Domain Filtering

During analyses of the URLs we discovered that not all urls were by authoritative sources. Few links just lead to sources with random content that does not relate to stocks at all. To overcome this solution we consider the top 80% URLs only.

To calculate the top 80% URLs we look into domain names of the URLs. What we really are doing here is following the idea of more URLs with a specific domain, more the authority. So even before the URLs are crawled and scraped we convert the TinyURLs to actual URLs and extract the domain names from there. Once all the domain names for all the URLs are extracted, we perform the basic calculation to keep the top 80% mentioned domains only and discard the others.

The above mentioned TinyURL to expanded URL is done in two ways in my research. Some TinyURLs are easy to expand using a small piece of code which decrypts it. Others fail to convert by the simple decryption rather it requires loading it in the browser which we follow using Selenium Library. Following 80% spike calculation we perform the crawling & scraping.

One thing to note here is because we are already converting the TinyURLs to actual URLs during this process either by decrypting or Selenium library, we can use normal crawling libraries to scrape the text as the bypassing of the advertisement is

already done during 80% spike calculation. Hence, increasing the efficiency of our program.

Algorithm 4: Relevant Domain Calculation

Given: Hourly Table & Tweets Table

1. Create domain count list
2. For each *breaking point bp* do
   (a) Extract all URLs of breaking stock for past one hour
   (b) For each *URL u*
      i. If Tiny URL then convert to actual URL
      ii. Extract Domain of URL u
      iii. If domain present in domain count list then update the count by 1 Else insert the domain with count set to 1
   (c) Calculate 80% of the domain count list. Remove last 20% of the domains from the list.

Chapter 5

FUTURE WORK

5.1 Time Index

Currently, thesis work involves predicting breaking stock based on counts of last hour and last 20 hours of data. Further challenge lies in testing the program for different combination of time indexes. For example, Predicting stock on the basis of last 15 minutes vs last 20 hours of data etc. It will be very early to say what prediction accuracy we obtain but if positive Breaking Topic Detection can be a powerful tool in predicting very high frequency trades.

5.2 Online Application

The whole application is an offline application with different parts of program run manually based on inputs at each step. The next step would be making an online application with least of manual requirements supported with good visualizations.

REFERENCES

Bollen, J., H. Mao and X. Zeng, "Twitter mood predicts the stock market", Journal of Computational Science **2**, 1, 1–8 (2011).

Gong, J. and S. Sun, "A new approach of stock price prediction based on logistic regression model", in "New Trends in Information and Service Science, 2009. NISS'09. International Conference on", pp. 1366–1371 (IEEE, 2009).

Hagenau, M., M. Liebmann and D. Neumann, "Automated news reading: Stock price prediction based on financial news using context-capturing features", Decision Support Systems **55**, 3, 685–697 (2013).

Nagar, A. and M. Hahsler, "Using text and data mining techniques to extract stock market sentiment from live news streams", International Proceedings of Computer Science & Information Technology **47** (2012).

Si, J., A. Mukherjee, B. Liu, Q. Li, H. Li and X. Deng, "Exploiting topic based twitter sentiment for stock prediction.", in "ACL (2)", pp. 24–29 (2013).

Zhang, X., H. Fuehres and P. A. Gloor, "Predicting stock market indicators through twitter i hope it is not as bad as i fear", Procedia-Social and Behavioral Sciences **26**, 55–62 (2011).

APPENDIX A

DATABASE DESIGN

There are two tables: tweets & hourly, that are used for entire process of the algorithm. The structure of both the tables with their table creation query is mentioned below.

```
CREATE TABLE IF NOT EXISTS 'tweets' (
'userName' varchar(50) NOT NULL,
'content' varchar(141) DEFAULT NULL,
'dateTime' datetime DEFAULT NULL,
'stockSymbol' varchar(12) NOT NULL,
'urls' varchar(1500) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```

```
CREATE TABLE IF NOT EXISTS hourly (
'stockSymbol' varchar(20) NOT NULL,
'Date' datetime NOT NULL,
'Hour' int(11) DEFAULT NULL,
'Count' int(11) DEFAULT NULL,
'movingAvg' decimal(10,4) DEFAULT NULL,
'breakOut' bit(1) DEFAULT NULL,
'stdDev' decimal(10,4) DEFAULT NULL
) ENGINE=InnoDB DEFAULT CHARSET=latin1;
```