Bursty Topic Detection using Acceleration and User Influence

by

Rizwan Ali

Bachelor of Software Engineering, Bahria University, 2012

A Thesis Submitted in Partial Fulfillment
of the Requirements for the Degree of

Master of Computer Science

in the Graduate Academic Unit of Faculty of Computer Science

Supervisor: Ali A. Ghorbani, PhD, Faculty of Computer Science

Examining Board: Huajie Zhang, PhD, Faculty of Computer Science, Chair
Bruce Spencer, PhD, Faculty of Computer Science

This thesis is accepted by the
Dean of Graduate Studies

THE UNIVERSITY OF NEW BRUNSWICK

May, 2017

©Rizwan Ali, 2017

Abstract

In this thesis, we present a system which detects bursty topics from real-time social media data. Bursty topics are topics which get a sudden surge in their mentions online in a very short period. They are detected using acceleration of keywords from the real-time data. The acceleration of a bursty topic is measured using the increase in appearance of keywords of a topic over a small period. Along with acceleration, user influence is used to score keywords in our system to improve the detection by increasing the keyword precision of the detected bursty topics. Bursty topics are formed using the top scoring keywords, based on acceleration and influence, and grouping them based on the similarities in their term document vectors. We use soft frequent pattern mining approach for generating topics. The bursty topics are also linked to bursty topics detected on previous time windows by comparing similarities in their keywords. Bursty topics detected using acceleration are evaluated with and without user influence score, with a baseline topic model. We use Latent Dirichlet Allocation topic model as the baseline. It is found that user influence helped the topic detector improve its precision by 11% on average. The results show that user influence can add great value to bursty topic detection methods.

Dedication

I dedicate this thesis work to my parents and my siblings, whose words of encouragement and advice helped me throughout my life.

Acknowledgments

I would like to express my sincere thanks to Dr. Ali Ghorbani, for supervising my thesis and for his insightful guidance and unending encouragement throughout my time at UNB. I would also like to extend my thanks to the members of the faculty for their support. Lastly, I would like to thank my parents, for without their love, support and unflinching belief in me, I would not have succeeded.

Table of Contents

ABSTRACT ..... **II**

DEDICATION ..... **III**

ACKNOWLEDGMENTS ..... **IV**

TABLE OF CONTENTS ..... **V**

LIST OF TABLES ..... **VIII**

LIST OF FIGURES ..... **IX**

LIST OF ABBREVIATIONS ..... **XI**

NOMENCLATURE ..... **XII**

INTRODUCTION **1**

1.1 BACKGROUND AND MOTIVATION 1

1.2 SUMMARY OF CONTRIBUTIONS 2

1.3 THESIS ORGANIZATION 3

LITERATURE REVIEW **5**

2.1 MICROBLOGS 5

2.1.1 Twitter *5*

2.2 TOPIC DETECTION AND TRACKING 6

2.3 PREPROCESSING 8

2.3.1 Crawler *9*

2.3.2 Stemming *10*

2.3.3 Tokenization *10*

2.3.4 Aggregation *11*

2.3.5 Removing stop words *11*

2.3.6 Parts of speech tagger (POS-Tagger) *12*

2.4 TOPIC DETECTION USING BURSTS IN STREAM 12

2.4.1 Document-pivot techniques *13*

2.4.2 Feature-pivot techniques *16*

2.4.3 Clustering Topics *18*

2.4.4 Soft Frequent Pattern Mining *19*

2.4.5 Similarity Computation *23*

2.5 SPECIFIED AND UNSPECIFIED TOPICS 23

2.6 NEW AND RETROSPECTIVE DETECTION 24

2.7 MACHINE LEARNING USED IN TOPIC DETECTION 25

2.8 CONCLUDING REMARKS 26

THE PROPOSED FRAMEWORK 28

3.1 TERM SELECTION AND VECTORIZATION 29

3.1.1 Acceleration of Terms *30*

3.1.2 Term Vectors *33*

3.2 USER INFLUENCE 34

3.2.1 Indegree influence *35*

3.2.2 Retweet influence *35*

3.2.3 Mention influence *36*

3.3 TOPIC CLUSTERS 37

3.4 CONCLUDING REMARKS 40

EXPERIMENTS AND EVALUATION 42

4.1 DATA-SET 42

4.2 SIMPLE BURSTY KEYWORD DETECTOR USING N-GRAM MODEL 43

4.3 EVALUATION METHOD 46

4.4 EVALUATION AND DISCUSSION 47

4.5 BURSTY TOPIC CHARACTERISTICS 55

4.6 VISUALIZATION 57

4.7 CONCLUDING REMARKS 60

CONCLUSION AND FUTURE WORK 62

5.1 CONCLUSION 62

5.2 APPLICATION OF TOPIC DETECTION 64

5.3 FUTURE WORK 65

REFERENCES 67

VITA

List of Tables

TABLE 4.1: DOCUMENT FREQUENCY OF KEYWORDS FOR ENTIRE DAY 45

TABLE 4.2: BURSTY TOPIC EVALUATION METHOD TABLE 46

TABLE 4.3: TOP 4 BURSTY TOPICS OVER ENTIRE DATASET 48

TABLE 4.4: KEYWORD PRECISION AND RECALL OF TOP 4 TOPICS 52

List of Figures

FIGURE 2.1: CLUSTERS C1, C2 AND C3 ARE FORMED 18

FIGURE 2.2: A CLIQUE CLUSTER AND A STAR CLUSTER 19

FIGURE 2.3: EXPANSION OF S WITH THE BEST MATCH FOR $\boldsymbol{DS}$ IS TERM VECTOR $\boldsymbol{Dk}$. [25] . 21

FIGURE 3.1: OVERVIEW OF TOPIC DETECTION FRAMEWORK..... 28

FIGURE 3.2: VOLUME AND ACCELERATION OF WORD "SPICER" FOR 30 MINUTES 31

FIGURE 3.3: VOLUME AND ACCELERATION OF WORD "APPLE" FOR 30 MINUTES 32

FIGURE 3.4: TERM VECTORS FORMED USING THEIR DOCUMENT OCCURRENCE..... 34

FIGURE 3.5: VENN DIAGRAM SHOWS THE OVERLAP OF THREE TYPES OF INFLUENTIAL USERS [29]..... 37

FIGURE 3.6: SIGMOID FUNCTION WITH B = 5 AND C = 2 (BLUE) AND B = 1 AND C = 5 (RED) 39

FIGURE 4.1: DISTRIBUTION OF TWEETS PER DAY FROM 1ST JUNE TO 25TH JUNE 2016 43

FIGURE 4.2: STACKED AREA OF DOCUMENT FREQUENCY OF TOP TOPICS FOR ENTIRE DAY . 45

FIGURE 4.3: BURSTY TOPIC "ORLANDO NIGHT CLUB SHOOTING" AGAINST TIME 49

FIGURE 4.4: BURSTY TOPIC "PRESIDENT OBAMA ENDORSES CLINTON" AGAINST TIME..... 50

FIGURE 4.5: BURSTY TOPIC "DONALD TRUMP RALLY IN FLORIDA" AGAINST TIME..... 50

FIGURE 4.6: BURSTY TOPIC "DISCUSSIONS ON ORLANDO NIGHT CLUB SHOOTING" AGAINST TIME..... 51

FIGURE 4.7: PRECISION AND RECALL OF TOP K TOPICS..... 53

FIGURE 4.8: PRECISION COMPARISON OF INFLUENCE AND ACCELERATION BURSTY TOPICS AND SFPM 54
FIGURE 4.9: RECALL COMPARISON OF INFLUENCE AND ACCELERATION BURSTY TOPICS AND SFPM 54
FIGURE 4.10: RELATIVE STANDARD DEVIATION TOPICS VS TOTAL MENTIONS 55
FIGURE 4.11: RELATIVE STANDARD DEVIATION TOPICS VS MEAN MENTIONS 56
FIGURE 4.12: RELATIVE STANDARD DEVIATION TOPIC VS RELATIVE STANDARD DEVIATION MENTIONS 56
FIGURE 4.13: PACKED BUBBLES CHART OF BURSTY TOPICS 58
FIGURE 4.14: POLAR CHART SHOWING TOPICS OF 24 HOURS 59
FIGURE 4.15: HEAT MAP SHOWING SCORE AND MENTIONS OF TOPICS 60

List of Abbreviations

TDT: Topic Detection and Tracking

SFPM: Soft Frequent Pattern Mining

POS: Part of Speech

LDA: Latent Dirichlet Allocation

FSD: First Story Detection

RAKE: Rapid Automatic Keyword Extraction

RSD: Relative Standard Deviation

NLP: Natural Language Processing

Nomenclature

Symbol	Meaning	Section
T	Set of terms	2.3.3
t_f	Frequency of term t	2.3.3
D	Set of Documents	2.3.3
ΔT	Time window	2.4.2
$V_{\Delta T}(t)$	Velocity of term in time window	2.4.2
$A(t)$	Acceleration of term	2.4.2
S	Set of terms	2.4.4
D_S	Vector representation of all terms in a set of terms	2.4.4
D_t	Binary vector f term	2.4.4
f_i	Frequency of i^{th} term	4.2
$U_{\Delta T}$	Users who posted in time window	3.1.1
u_i	i^{th} user weight	3.1.1
V_t	Term vector	3.1.2

Chapter 1

Introduction

1.1 Background and Motivation

Microblogging social networks are now well established and highly popular social media platforms. A huge chunk of information is being broadcasted and exchanged using them. Millions of users around the world are expressing their opinions, sharing their accounts on real world events and discussing everyday issues on it right now. In 2016 there were around 2.3 billion social media users worldwide. And in two of the top three most used social networks (Facebook and Twitter), users share information using microblogs [1]. This information accounts to millions of microblogs every day and with this sheer volume understanding and analyzing this valuable source information has become a major challenge.

One way to analyze this data is by looking at "bursty" terms or topics. A topic can be defined as a set of closely related terms, then a bursty topic is a topic which has a sudden increase (burst) in the occurrence of the terms related to it in a small timeframe. When large sum of users is using selective terms related to topic in a very short period. For example, the topic "Star Wars" has on average large amount of traffic, but on 27th December 2016 had a burst in the occurrence of its keywords when Carrie Fisher (who played Leia) died. Several systems have been proposed to detect these bursty terms, topics

or events from large amounts of data [4] [5] [6] [9] [10] [14] [15]. These systems mostly rely on probability of keywords and terms in corpus to identify the bursty topics. One system uses sudden increase in acceleration of terms, which has been shown to be a very efficient indicator [12] for bursty topic detection. Even using acceleration the quality of the terms inside the bursty topic is low and it can be hard to relate them with real world topics. We believe popularity of the posts themselves can also play a part to solve this problem. We want to see if acceleration along with influence, calculated using the user's influence, retweet influence and user mentions can be used to improve the bursty topics. In this thesis, we will explore these features and see how they can help us accurately detect bursty topics from the microblogging data.

1.2 Summary of Contributions

The main purpose of this thesis is to develop a framework which detects emerging topics from a stream of microblogging data. We believe that leveraging from user's popularity and connections in the social network like Twitter, we can improve precision at which bursty topics are detected from a real-time stream. The contributions of this thesis can be summarized as follows:

- Propose and implement a system which can identify topics with their terms which have a very high increase in acceleration and velocity of appearance in micro blog streams over a very short period (bursty topic).
- Identify features from user information, which can be used for estimation of influence, which will be used to detect a bursty topic with more precision.

- Perform an experiment and evaluate the bursty topics. Identify the effects of user influence over the topics. Propose a methodology to validate the topics themselves. Our experiments will show:
    1. Characteristics of bursty topics and their correlation with the dataset.
    2. Precision and Recall of our models based on topic volume acceleration and user influence for detecting bursty topics from terms.
    3. Comparison of feature sets based of precision and recall.
- Develop visualizations for bursty topics, which can help understand them and use them more efficiently with real world data.

1.3 Thesis Organization

The rest of the thesis is organized as follows:

- Chapter 2 discusses core topic detection and tracking systems and introduces types of topic detection systems and their characteristics. How they differ from each other and their pros and cons. It also discusses implementations of topic detectors, event detectors and bursty topic detectors proposed by researchers and what we learned from them.
- Chapter 3 first provides a basic overview of the proposed bursty topic detection framework, followed by a detailed description of each module. It explains, how keywords can be extracted from real-time data and converted into vector form, how user influence can play a role in this keywords extraction process and how topics are formed using these terms.

- Chapter 4 explains experiments and validation of the system. The data collection process is explained in this chapter. It also demonstrates how can we evaluate the system and do comparisons on it in detail. It also shows characteristics of bursty topics and ways to visualize the system.
- Finally, Chapter 5 concludes the thesis by discussing its contributions and possible improvements and future work that can be done on it. It also shows uses and real world implementation of the topic detection systems.

Chapter 2

Literature Review

2.1 Microblogs

Microblogging social networks are now well established and very popular social media platforms. A huge chunk of information is being broadcasted and exchanged through them. Millions of users around the world are expressing their opinions, sharing their accounts on real world events and discussing everyday issues on it right now. In 2016 there were around 2.3 billion social media users worldwide. Users share information through microblogs, on two of the top three most used social networks (Facebook and Twitter) [1]. This information accounts to millions of microblogs every day and with this sheer volume understanding and analyzing this valuable source information has become a major challenge.

2.1.1 Twitter

Twitter is a widely used microblogging social networks with more than 100 million active users online [2]. It exclusively uses microblogs for broadcasting information. It allows users to share content which can be text, links, images and videos. It provides a reliable platform for ordinary users to share anything that's happening around them. Major news networks and corporations use Twitter to pick up stories and topics every day and to get a general understanding of what people think about them.

The microblogs shared on it are called tweets and cannot be longer than 140 characters. Hundreds of millions of tweets are uploaded every day on Twitter from its users around the world, therefore Twitter has a very diverse information stream. Users can relate their tweets to a particular topic using hashtags or by tagging other users in them. This provides valuable information about what the tweet is talking about, however not all tweets have hashtags in them and even with hashtags, identifying their true topic is a difficult and complicated task. The percentage of tweets without hashtags varies based on different corpuses. Tweets are public by default and any registered user can see them. Users can also set their tweets to be private, then only users that are following them can see these tweets. In our study, we used only publicly available tweets.

Tweets can be seen as a source of information for news agencies which are tracking real world events and for individuals or organizations who want to know the opinion of the people about them. The information retrieved from Twitter is in real time and ongoing discussions can be tracked. Monitoring and analyzing this continuous flow of user generated data can provide us with valuable insights. Moreover, this data has emerged as a valuable information source for predicting election results, finding new events and analyzing sentiment for entities.

2.2 Topic Detection and Tracking

Topic detection and tracking (TDT) [20] first started event detection from large textual data sets. The program was sponsored by DARPA (Defense Advanced Research Projects

Agency), to detect topics from broadcast news. The aim was to monitor news streams from multiple sources and identify any event happening. TDT had three main tasks:

- Segmentation Task
- Detection Task
- Tracking Task

The segmentation task takes a continuous stream of stories and segments them into cohesive stories. It gives differences between the stories in the corpus which it found. The segmentation is evaluated in two ways, first whether it can evaluate and identify stories from the corpus correctly and then its performance while supporting detection and tracking. Several segmentation methods were proposed including Dragon Approach. In this the corpus is considered as a stream of unlabeled content of a topic. The content stream is then labeled with topics using a topic model. Another segmentation method proposed using local context analysis (LCA) [21]. LCA segmentation was found to be more robust and effective and was unsupervised.

The detection task processes the stories and identifies events in them. The task is divided into two parts, retrospective and new event detection. Retrospective events use supervised learning algorithms, the user provides most of the information about the event. New events are unsupervised because no prior information about the new events is available in advance. Clustering methods are mainly used in detection of new events.

The tracking task associates an incoming new story to an existing event which was already detected by the system.

2.3 Preprocessing

Most of the text gathered from social media is in an unstructured format. This means data is raw and has not been processed. In microblogs like Twitter we may get some structure from tweets that use hashtags or user references, but the data still cannot be used by text analysis algorithms. Documents with some structure in them are called semi-structured. Recent estimates show 75 to 80 percent of the data on the Internet is in an unstructured or semi-structured format [3]. Therefore, performing preprocessing on the data, to prepare it for further analysis, is required and a very important task.

The goal of preprocessing is to convert the text into structured format. The most popular structured format is the bag-of-words model. In this model each item in a vector indicates a word that is occurring in the text. The down side of this model is, if we have a large corpus which is using a diverse vocabulary, each vector will have high dimensions and will become sparse. Each element of the vector represents the number of occurrences of the word in the document.

In the vector space model the order of the text is lost. For a text analysis task like event and topic detection, it is not a big problem as the task can be still performed without any information of the sequence of the words, since a greater focus is given to the frequency of the words rather than to the actual syntax of the sentence. This assumption is called the bag-of-words assumption.

The steps required to perform the preprocessing may vary for the specific task that is being performed. For text analysis and burst detection, the following steps are generally used:

- Web crawler

- Stemming
- Tokenization
- Aggregation
- Removing stop words

2.3.1 Crawler

Microblogging social networks like Facebook and Twitter, provide API access to their public data. Different scopes of the data can be gathered from the API depending upon the goal we need to achieve from the data. For classification tasks we often require the entire data set to be available. However, for tasks like burst detection a smaller sample of the data can be sufficient. Further we can focus our search on entities or topics, if we want to identify bursty terms about them. For example, we want to detect bursts of terms in documents only related to cybersecurity. In this case the bursty terms might be related to data leaks, hacks or names of viruses affecting devices used by people.

Twitter streaming API can be used to get real-time tweets and tweets related to entities. This data can be used to identify bursts in the entire Twitter stream. Another API which Twitter offers to developers is the Twitter search API. This API takes general search terms, like hashtags or user references, and returns either the entire data set or a smaller subset depending upon the size of the result we want. We can routinely use the search API to get data of a smaller time window in order to get near real-time data of specific search terms.

2.3.2 Stemming

Stemming is used to reduce words to their basic form. For this a rule based process is used to reduce a given word, like a rule could be defined as “removing ING and ED from the end of words”. Porter stemming algorithm is one of the widely used algorithms which uses a rule-based technique [6]. The algorithm was further refined and is now available as an open source software called Snowball.

Similar to stemming, lemmatization is a process which detects the lemma (reduced form) of the words given to it based on their meaning and use in sentences. With higher computation cost it gives much better results than rule based stemming algorithms. Stanford NLP tools provides both stemming and lemmatization.

2.3.3 Tokenization

The text obtained from the crawler has to be converted into many tokens containing terms. White spaces and punctuations are used to divide the documents into tokens. We can calculate the frequency of these tokenized terms in a time window for our analysis. We call this term frequency tf of each term t in document d_i in set of documents D. Such that term-vector of document d_i can be given as

$$\vec{d}_i = \{tf(t_n, d_i)\} \quad 1 \leq n \leq N$$

where N is the number of terms in our Vocabulary, and $tf(t_n, d_i)$ is the term frequency of term t_n in document d_i.

2.3.4 Aggregation

Microblogs posted on social media typically contain very short posts. On Twitter each tweet can only have 140 characters. N-gram and co-occurrence models on such documents cannot be applied if we are comparing entire documents to perform text analysis tasks. This problem is generally solved by concatenating different documents into a large super-document. An easy way to develop a super-document is by using temporal features. Documents which are posted in the same time window are concatenated into a single document. We can also use social media specific features to aggregate documents, for example documents from the same group or documents with same hashtags on Twitter. Another way to create the super document is by using document similarity. We will discuss them further in "Document Pivot Techniques" (section 2.3.1).

2.3.5 Removing stop words

In text analysis, for tasks like burst detection we can ignore words such as 'the', as they provide no information or usage in the tasks. Removing these words can help improve the performance of the task, as these words are very frequent present in any block of text. We call these non-content-bearing words as stop words. The list of stop words was originally taken from [19], but in our system we used an open source version of it, with more words added in it related to micro blogs. Below is a list of 25 most common stop words.

a an and are as at be by for from

has he in is it its of on that the

to was were will with

2.3.6 Parts of speech tagger (POS-Tagger)

Parts of speech taggers are software designed to mark a word in given text to its particular part of speech. The word is evaluated with its meaning and surrounding text. Many systems have presented reaching up to an accuracy level up to 97% [16]. A POS tagger specifically designed for informal text from microblogs like Twitter was presented by Owoputi et al, 2013 [13], which gave a 93% accuracy. The tagging software and word cluster data sets are also available online[1].

2.4 Topic detection using bursts in stream

Topic detection was first addressed in the research program Topic Detection and Tracking (TDT) [17]. The program focused on conventional media and broadcast news stories as an information source. However, detection from online social media and microblogs like Twitter has brought unique challenges. In conventional methods, every document used must have a rich content and the corpus should be of a static size. The corpus size is also very small compared to the real-time data from social media. And the documents used in these traditional topic detectors, in many cases, are structured (preprocessed) like RSS. In contrast microblogs have a very short and unstructured content. Many documents have very little or no information in them, so just using the text of documents alone can mislead to false topics. On top of this there is a large amount of noise (spam content). Due to these

[1] http://www.cs.cmu.edu/~ark/TweetNLP/

difficulties, traditional topic detection methods cannot be efficiently applied to detect topics and events based on bursts.

Topic detection techniques can be classified in two main types based on their outputs, *document-pivot* and *feature-pivot* [18]. Document-pivot techniques form clusters of documents in the corpus, using a similarity measure between the documents. The final output will be several clusters of documents in which each cluster represents a topic and all documents in that cluster have the same topic. In feature-pivot techniques individual features of documents are first extracted. Common features used are keywords extracted from the document or correlation between words and keywords. These features are then clustered to form topics. Both techniques have advantages on different types of corpuses. Several frameworks have been proposed of these systems and we can analyze the pros and cons of them to find out the best fit for detecting bursts, as well as study the implementations of the systems.

2.4.1 Document-pivot techniques

Document-pivot techniques cluster documents based on textual similarity between them. The similarity is measured between two documents or between a document and a cluster. The cluster then can be labeled by preexisting keywords or the keywords can be extracted from the documents. The Similarity between two documents can easily be measured after converting them into a simple bag-of-word representation. In this vector representation, similarity can be measured by using a commonly used measure, cosine similarity.

TwitterStand [4] is a news processing system based on Twitter's stream to capture tweets that correspond to the latest breaking news. They employ a naive Bayes classifier to

separate news from irrelevant information and an online clustering algorithm based on a similarity metric using a weighted term vector according to $tf - idf$ and cosine similarity to form clusters of news. Term frequency and inverse document frequency ($tf - idf$) tells us how important a word is to a document in a corpus. In addition, hashtags are also used to reduce clustering errors.

The technique suffers from cluster fragmentation, which means there could be two clusters with the same topics competing with each other. To accommodate this issue, a separate task is proposed to identify such clusters and to merge them into one.

First story detection (FSD) technique [5] for Twitter captures documents related to emerging stories. In this method new stories are created by documents which have lower similarity with all the existing documents. To cater to the issue of high performance cost of similarity detection between documents, they make some ground rules for the algorithm of their system.

- Each document will be compared with a limited number of documents in a cluster.
- There will be a limited number of clusters, dying and old clusters will be discarded.

The performance cost is a major issue in document-pivot technique. These rules can significantly decrease the performance cost but will affect the accuracy.

Becker et al [6] proposed a technique which continuously clusters similar tweets and then classifies the clusters content into real-world events or non-events. These non-events involve Twitter-centric topics which are trending activities that do not reflect any real-world occurrences.

The features of tweets are also used in order to reveal characteristics that may help detect clusters that are associated with events. For this the temporal features are used that rely on

term frequency that appear in the set of messages associated with a cluster over time. The social features include the percentage of messages containing user interaction (i.e., retweets, replies, and mentions) out of all the messages in a cluster. The topical features are based on the hypothesis that event clusters tend to revolve around a central topic, whereas nonevent clusters often center on various common terms (e.g., “sleep” or “work”) that do not reflect a single theme. The Twitter-centric features are based on the frequency of multiword hashtags with special capitalization.

Because the clusters constantly evolve over time, the features are periodically updated for old clusters and computed for newly formed ones. Finally, a support vector machine (SVM) classifier is trained on a labeled set of cluster features and used to decide whether the cluster contains real-world event information.

In microblogging data stream from social media, document pivot techniques suffer from several disadvantages. The clusters formed based on document similarity, have cluster fragmentation, this is solved using separate processes which use thresholds to find such clusters and merge them into one topic. Several methods use separate process to include new documents into existing clusters or topics using arbitrary thresholds. This is also for some cases based on the assumption that documents will always belong to a topic which is not true in our case. Additionally, smaller topics (at the initial stage of their burst) may not get identified at all in these techniques. To accommodate these issues, we propose to use Feature pivot techniques.

2.4.2 Feature-pivot techniques

Feature-pivot techniques focus on using natural language models, co-occurrence of terms and term frequency to extract set of keywords which can be labeled as topics. They create clusters of these keywords using the co-occurrence patterns on term pairs. Thus the final output is a set of keywords as a single topic. This method can perform extremely well on streaming data where document size is very small [22]. Though they may capture misleading terms and co-occurrence relations. We can fine tune these techniques using other non-textual feature sets. We enlist the general features which are being used in the techniques we studied:

- Word frequency and acceleration
- Word correlation
- Parts of speech tagging
- N-gram models
- Hashtags
- Timestamps
- Followers of users (specific to Twitter)

A model for hot topic discovery using concepts of velocity and acceleration was proposed by Zhu et al [9]. In their technique, from each post they gather at most 20 keywords and form a vector. A topic detected from a series of posts, which can evolve its subtopics with the passage of time. A topic is evaluated for its potential popularity based on their velocity. Hot topics are detected and are given statues about how high their acceleration is, compared

to the threshold for the acceleration. The threshold for the acceleration is computed using the acceleration of hot topics detected in the past.

An alternative framework which uses a personalized threshold calculation to find if a document is related to an event has been studied and is called threshold-resilient classification [10]. This threshold is the main classification feature in their classifier. The framework uses $tf - idf$ weights of documents of events to calculate the threshold. A document is classified using its similarity with event terms and temporal data, if the similarity beats the threshold of an event the document is assigned to it. For new events they propose to develop an "association link network" between documents and find community centers [11].

Another solution used bursty topic detection to increase efficiency by getting topics without examining the entire set of documents in the corpus [12]. A sketch is developed using the acceleration in frequency of terms in documents and their co-occurrence. This sketch is used to predict the top topics and latent topics. The equation below shows how they calculate velocity $V_{\Delta T}(t)$ and acceleration $A(t)$ of word *x* at time *t.*

$$V_{\Delta T}(t) = \sum X_i \frac{\exp\left(\frac{t_i - t}{\Delta T}\right)}{\Delta T}$$

$$A(t) = \frac{V_{\Delta T_1}(t) - V_{\Delta T_2}(t)}{\Delta T_1 - \Delta T_2}$$

Where X is the frequency of word x in the i-th document, t_i is the time stamp of i-th tweet containing the word x. The exponential part of the velocity $V_{\Delta T}(t)$ will give less weight to old tweets and more weight to new tweets. ΔT is the window size. The acceleration is the difference of two velocities with different window sizes, similar to divergence calculation

where two moving averages, one of smaller time period and the other of a larger time period, are used to determine the future trend. We can also use this to determine acceleration of pair of words.

Due to its high efficiency and early detection, we believe burst detection method used in this solution is most promising. We will use their approach of detecting velocity and acceleration to further analyze all other features which can be used to determine topic burst detection with a higher accuracy.

2.4.3 Clustering Topics

Clustering is an unsupervised learning task which classifies objects into groups. In topic detection, clustering can be used to group documents about the same topic. Clustering algorithms develop groups based on their similarities between the documents. Looking at Figure 2.1 we can see three clusters. The documents closer to each other are grouped in one cluster.

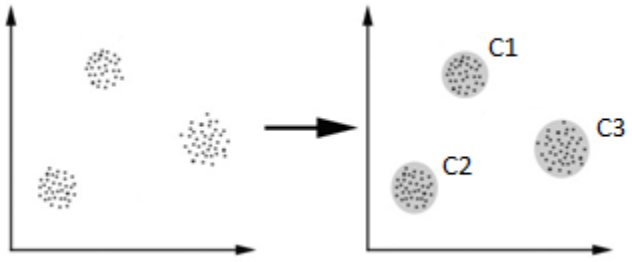

Figure 2.1: Clusters C1, C2 and C3 are formed

Clusters can be formed by different liking methods between documents. A cluster where all documents in the cluster are linked to each other is called a Clique. In this type of cluster, for a new document to become a member it has to be similar with every single document

in the cluster. Another more efficient approach is if we take a central document, called centroid, into the cluster and then every new document will have to be similar only to the centroid to become a member of the cluster. This becomes a Star cluster.

The centroid does not have to be an actual document, and can be formed using the average values from all the documents in the cluster. The centroid can also be used as the label (topic) of the cluster of documents, having either most common words of the cluster or words with highest $tf - idf$ score [23]. A document-pivot technique has been presented which use centroid based k-means clustering algorithm to detect topics [26]. Another document-pivot approach has also been presented which uses db-scan for clustering [24].

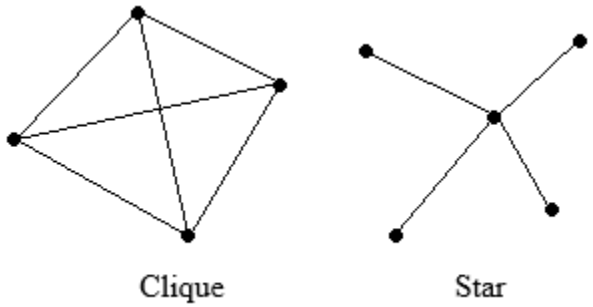

Figure 2.2: A Clique cluster and a Star cluster

2.4.4 Soft Frequent Pattern Mining

Soft frequent pattern mining (SFPM) is an approach used to determine topics from co-occurrence patterns of large number of terms [25]. In this approach K top terms are selected from the corpus C that will be grouped into topics. The top terms are selected using the likelihood of appearance of each term t estimated as

$$p(t|C) = \frac{N_t + \delta}{(\sum_{i=1}^{n} N_i) + \delta n}$$

where N_t is the frequency of the term t has appeared in the corpus, n is total number of terms in the corpus and δ is a constant. To select the top terms in the current corpus C the ratio of likelihood of appearance of each term is compared with likelihood from a randomly generated corpus C_{ref}.

$$r(t) = \frac{p(t|C)}{p(t|C_{ref})}$$

The terms with a significantly higher frequency than usual (C_{ref}) will have higher ratios and will be ultimately selected as top terms. The terms with the highest ratio will be the ones with significantly higher than usual frequency of appearance and it is expected that they are related to the most actively discussed topics in the corpus.

A set of terms S is maintained which will ultimately represent topics. To add a new term t in S, a greedy approach is used. For this, two vectors D_S and D_t are formed with length equal to the size of documents in the corpus. D_S stores in each element the number of terms in S that occur in i-th document. D_t is a binary vector that stores 1 in element when the term t occur in the i-th document. The vector D_t will have a high cosine similarity with D_S, if the term t frequently co-occurs with terms in set S. The vector D_t will also be added into D_S. Figure 2.3 shows a single expansion of set S

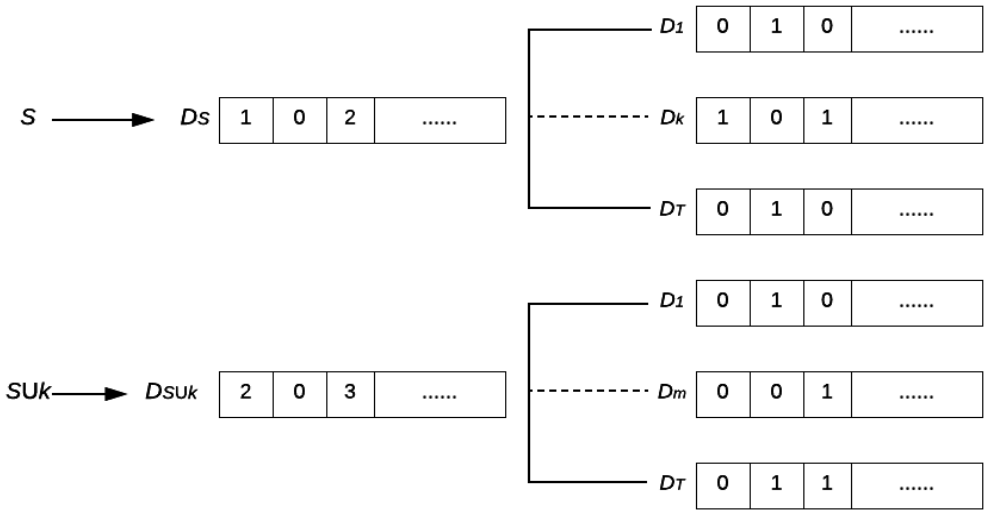

Figure 2.3: Expansion of S with the best match for D_S is term vector D_k. [25]

To terminate the expansion of S, a sigmoid function is used. The threshold θ is given as

$$\theta(S) = 1 - \frac{1}{1 + e^{\frac{|S|-b}{c}}}$$

The parameters b and c can be adjusted to change the size of the clusters. The threshold function will make it harder to add terms as the cardinality of S increases.

Algorithm 2.1 SFPM Algorithm [25]

1: **function** SFPM (C, K)
2: T = SelectTerms (C, K)
3: **for** each term *t* in *T* **do**
4: Compute Occurrence Vector D_t for *t*
5: **end for**
6: Topics = []
7: **for** each term *t* in *T* **do**
8: $S = t$;

```
    D_S = D_t ;
    ContinueExpanding = true;
    repeat
        t* = GetBestMatchingTerm (D_S, S, T)
        sim = CosineSimilarity (D_S, D_t*)
        if sim > θ(S) then
            S = S ∪ t*;
            D_S = D_S + D_t*;
            n = size (C);
            for i = 1 to n do
                if D_S[i] < |S| /2 then
                D_S[i] = 0;
                end if
            end for
        else
            ContinueExpanding = false;
        end if
    until ContinueExpanding
        Topics = Topics ∪ S
end for
```

In experiments this approach was compared to frequent pattern mining (FPM) and Latent Dirichlet Allocation (LDA). LDA is a generative probabilistic model, in which each document is modeled as a mixture of multiple topics [8]. SFPM showed more promising results for microblog data for both topic detection and representation [25]. It achieved a higher topic recall (percentage of true topics) compared to LDA. They also measured keyword precision (correctly detected topic keywords over total keywords of true topic) and SFPM performed consistently well over multiple datasets.

2.4.5 Similarity Computation

In previous section, we saw cosine similarity, a similarity measure, used in SFPM algorithm. Similarity measures are used to define a degree of similarity between two objects based on their characteristics. For example, consider two text documents represented as a probability distribution over their terms, then the similarity of the two documents is the degree of similarity between their probability distributions.

The cosine similarity between two documents or terms represented as vectors, is the measure of the cosine of the angle between them. Given two vectors A and B, the cosine similarity is given as below.

$$similarity = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} {A_i}^2} \sqrt{\sum_{i=1}^{n} {B_i}^2}}$$

The degree of similarity ranges from 0, having no similarity, to 1, being identical. The magnitude of the vectors does not matter, two vectors with same orientation will have a cosine similarity of 1.

2.5 Specified and Unspecified Topics

Topics detection techniques can classify topics on prior information available about the topics. The prior information can be keywords, temporal information or descriptions. The Topic detection process can use these features for classifying documents. This information is provided in advance by the user. Depending upon the requirements of prior information topic detection can be classified into two types, Specified and Unspecified techniques.

Specified topic detection techniques relies on specific information and features that are known about the topic. For unspecified topic detection techniques, no prior information is available, these techniques rely on word frequencies, word correlations and the temporal signals to classify topics. These techniques typically require monitoring for bursts or trends in frequencies and grouping the features together with identical trends into topics, and ultimately classifying the topics into different categories. In our work we will focus on unspecified topic detection.

2.6 New and Retrospective Detection

Many topic detection techniques involve identification of new events from a stream of documents. The detection is made after the documents are processed and grouped together. The groups formed provide the topic for it as various methods. These methods are called New Event Detection Methods. Several methods discussed before are new event detection methods [4] [5] [6] [14].

In retrospective event detection historical documents are searched for events already known. The documents are then classified as related to the known event or not. Ideally a document is related to only one event. Many of the existing search services, like Google and Twitter, provide the functionality of searching retrospective documents in their databases. Retrospective event detection technique was proposed using hierarchical document clustering by Yang et al [15]. Each cluster represented an event. The data, provided with temporal information, was clustered using semantic and temporal properties and similarity between documents.

2.7 Machine Learning used in Topic Detection

Machine learning techniques are widely used in the topic detection process. They involve learning a mapping function which will be used to classify data. They are divided into two main types, supervised and unsupervised. In supervised learning, we provide a data set to the algorithm which is already classified. The algorithm is trained from this dataset. Kasiviswanathan et al. [16] proposed a dictionary learning based framework for detecting emerging topics. The framework is divided into two sub-tasks, first to identify documents which are related to emerging topics and then identifying clusters in these documents. Both of these tasks use dictionary learning for classification.

Unsupervised machine learning algorithms use the input data to identify patterns in it and then classify the data. In topic detection, unsupervised learning algorithms are more focused towards new event detection from input streams using clustering. The clustering algorithm has to classify new topics from a stream of input documents in a limited amount of time. Several features are used for clustering. In techniques proposed by Becker et al. [6] clusters were formed by identifying similarity between documents using features such as a hashtag occurrence. In addition to frequency of words, entropy, weighted topical words and proper names are used as features [17]. Wartena et al. [18] also proposed a technique which performed clustering using keyword extraction without any prior knowledge of the document. Keywords were extracted using frequency of words and their co-occurrence probability.

2.8 Concluding Remarks

Topic detection in social media microblogs has become a difficult and expensive operation because of the sheer quantity of data. Detecting the topic of every document will be an impossible task. A more informative way to analyze this data is to only extract interesting bursty topics from it. One way to get these topics is to identify the bursts in the topics and terms in the data stream.

The data stream contains documents in an unstructured format and performing preprocessing is essential to further process it using text mining and topic detection techniques. The topics can be detected using two types of detection techniques. Document-pivot techniques generally compare documents from the data stream with each other to find topics whereas, feature-pivot techniques use word frequencies and term co-occurrence to determine topics. In topic detection for microblogs acceleration based feature-pivot techniques are more efficient in detecting interesting bursty topics than document-pivot techniques, as they can be far more sensitive to bursts. The acceleration with other features, user popularity and influence can be used to predict bursts more accurately.

In document-pivot techniques we can cluster documents into topics using k-means and db-scan clustering algorithms. For feature-pivot techniques we can use soft frequent pattern mining (SFPM) to find relations in word co-occurrences to identify interesting topics. Beyond pairwise co-occurrences, SFPM examines simultaneous co-occurrence terms in documents, which helps it improve the keyword quality of the topics and has a higher keyword precision than similar technique LDA.

Topic detectors can also be used to detect already specified topics. Topics of interest with their keywords are already listed for the topic detector algorithms. We can also detect retrospective events from historical documents that are searched for events already known. We conclude the literature review with the hypothesis that feature-pivot techniques can be further tuned to accurately detect, predict and report bursty topics.

Chapter 3

The Proposed Framework

In Chapter 2, we studied several methodologies which use bursts for topic detection. We saw various drawbacks in them which we want to address and combine methods that not only focus on bursts of keywords in the data stream, but also on the probability of users triggering these bursts. We propose a framework which uses both acceleration and user influence to detect bursty topics. Major components of this framework involve a selection of interesting terms using a combination of two different matrices, term acceleration and an added influence score based on the influence of term's originator. The selected terms will be converted into vector format and clustered into topics. Figure 3.1 shows an overview of the process.

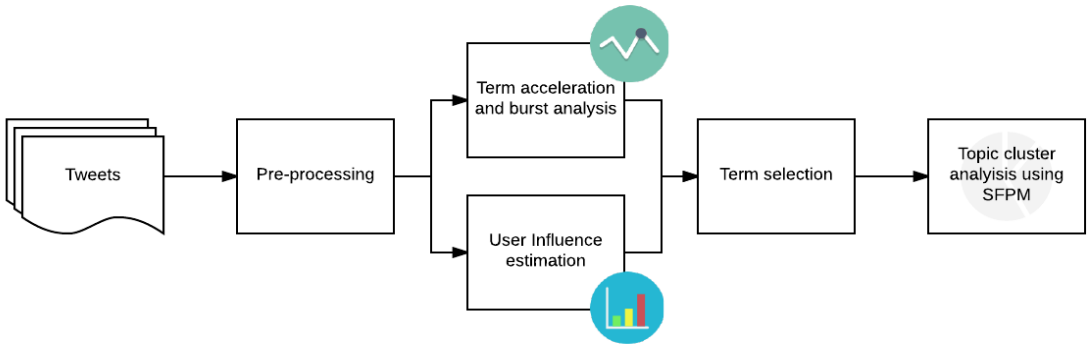

Figure 3.1: Overview of topic detection framework

The tweets are collected and preprocessed. These components will be further discussed in Section 4.1 in detail and have been seen in Section 2.3. The other major components of our framework are as follows:

- Term Selection and Vectorization
- User Influence
- Topic Clusters

In this chapter, we will first discuss these components in detail. We also discuss different evaluation methods for our framework with can be used to analyze different feature sets.

3.1 Term Selection and Vectorization

Corpuses extracted from social networks, like Twitter, can contain a large amount of unique words. It could be a good practice to drop the non-content bearing and less important words earlier before words are converted into vector form. This sharply reduces the complexity and size of the input vectors for text mining algorithms. The simplest way to remove these words is by using a stop-words list. After this the remaining words are converted into tokens, with each word having a single token. As discussed in Section 2.3.5 we can use preexisting stop-word list for this purpose.

The down side of this technique is that; phrases can be lost. For example, the noun phrase "A penny for your thoughts" would change to "penny thoughts". The original phrase means, asking for someone's advice, while the latter phrase could potentially mean that the advice given is not worth much.

Another way to extract terms is by using Rapid Automatic Keyword Extraction (RAKE). RAKE can extract terms of multiple words [27]. The text is split into tokens using delimiters and some stop words. This way each token can carry multiple words. The keywords are scored using the degree of the tokens and frequency of the words. The tokens

are then ranked using the score. In our framework, we want to give greater focus to the scoring of the term, not just by frequency and degree, but by its acceleration over time periods. Therefore, we use combination of stop-words and acceleration score to filter words.

3.1.1 Acceleration of Terms

In TopicSketch [12], terms are weighted using velocity and acceleration as discussed in Section 2.4.2. Terms which are related to general topics and do not have a burst, will have a constant velocity over many time windows. In contrast, emerging terms related to bursty topics which have bursts in their mentions will have a sudden surge in velocity and acceleration. In the equation below we see TopicSketch [12] uses exponential part in calculating velocity. This makes velocity and acceleration very sensitive to bursts, but can also be misled by noise. The noise contributes in decreasing the value of the measures. Generally, there are two types of noise producers, one is a bot and the other is a topic hijacker. Bots generally post ad URLs and unrelated text in popular discussions. Topic hijackers post large volume of low quality content making it harder to accurately detect the topic.

To filter out noise we track a score of posts u_i by each user i from users $U_{\Delta T}$ in time window ΔT. We will discuss the possibilities in the next section (Section 3.2). We calculate velocity $V_{\Delta T}(x)$ of term x from Documents in time window $D_{\Delta T}$.

$$V_{\Delta T}(x) = \sum_{d \in D_{\Delta T}} d(x)\, u_i \,. \frac{\exp(\frac{t_i - t}{\Delta T})}{\Delta T}, \quad u_i \in U_{\Delta T}$$

$$A(x) = \frac{V_{\Delta T_1}(x) - V_{\Delta T_2}(x)}{\Delta T_1 - \Delta T_2}$$

Where $d(x)$ is the frequency of term x in document d. ΔT_1 and $\Delta T_2$are two different but overlapping time windows. This is similar to divergence of 5 days and 10 days moving averages used in Moving Average Convergence Divergence (MACD).

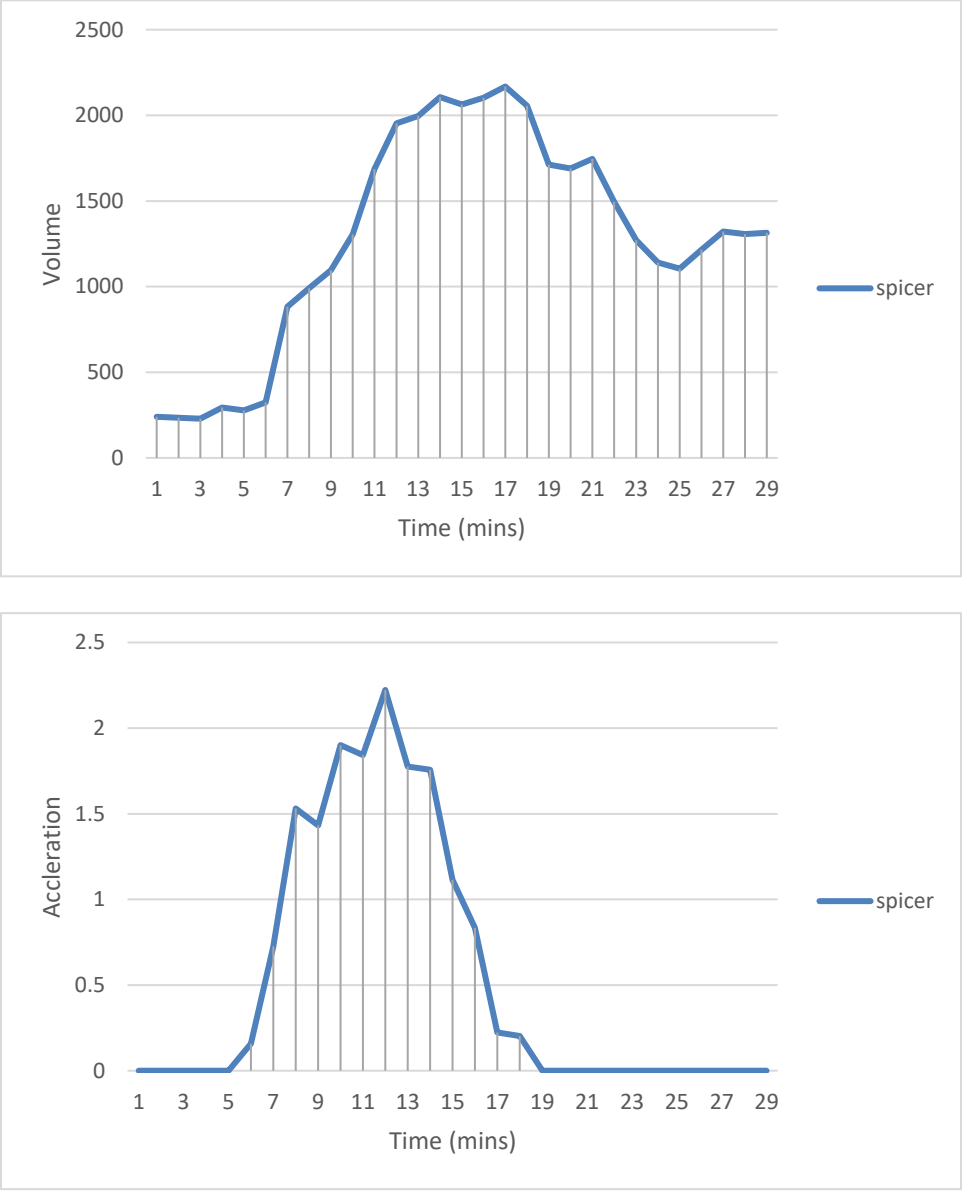

Figure 3.2: Volume and acceleration of word "spicer" for 30 minutes

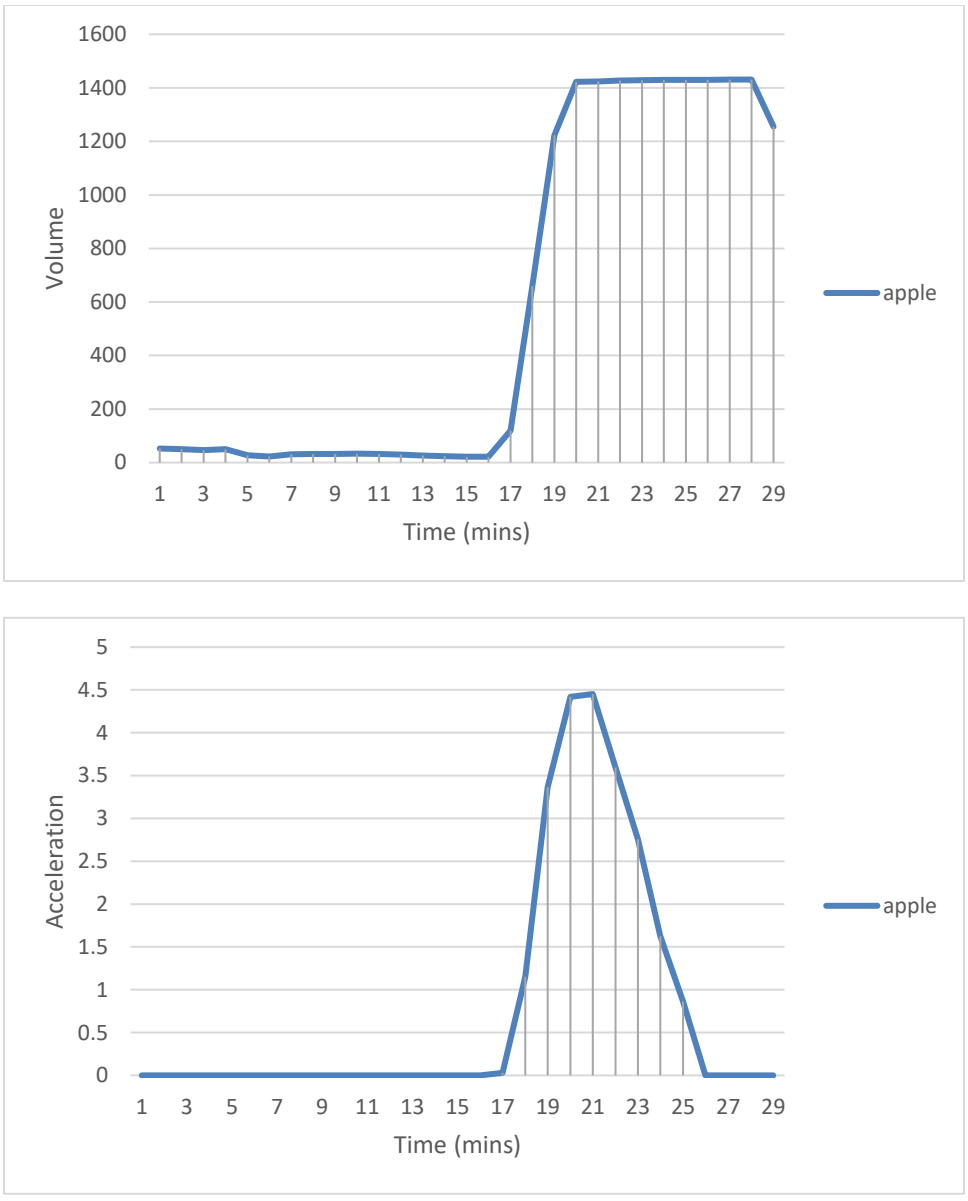

Figure 3.3: Volume and acceleration of word "apple" for 30 minutes

In Figure 3.2 and Figure 3.3 we can see a comparison between the actual volume and acceleration calculated from the velocity. The comparison is done using two real keywords. One keyword is 'spicer' which is related to white house press secretary Sean Spicer. The second key word is related to technology company Apple. During the data collection time, tweets relating to Sean Spicer's news conference started to come, as well as information

about Apple's first quarter results. A smaller random sample from all of the data was used for the calculation only. Looking at the acceleration chart we see two different types of bursts. Keyword 'spicer' had a burst for a relatively longer period than 'apple'. But 'apple' had a more sudden burst and therefore got a higher value of acceleration. The acceleration gives a maximum reading as soon as the burst starts, providing an early detection and rapidly gets closer to zero as soon as the burst of volume ends.

It is clear that acceleration can provide an early indication for identifying bursts in keywords. Using this, we can easily identify the starting point of the burst as well. Term selection based on acceleration can help us filter out keywords which have more potential to be a part of a bursty topic.

3.1.2 Term Vectors

After selecting an n number of terms using the acceleration of the term occurrence, they are converted into vector form so that their similarity can be computed. As we will compute the similarity of terms in a single timeframe, vectors can be formed using only documents D from that timeframe. For term t, term-vector V_t contains N number of elements, where N is the number of documents in D. The i^{th} element of V_t denotes if i^{th} document contains the term t in it. Irrespective to the frequency of the term, the value of each element will be a binary indicator of the term occurrence. Using these term vectors, we can compute cosine similarity between terms required for our framework. Figure 3.4 shows some examples of term vectors.

	D_0	D_1	D_2	.	.	.	D_N
V_0	1	1	0				0
V_t	0	1	0				1
.							
V_n	1	0	0				1

Figure 3.4: term vectors formed using their document occurrence

3.2 User Influence

The Six degree of separation[2], The Chance of central person[3] and The Law of few are some of the theories which highlight the importance of Influence as a means of spreading information [28]. These theories are based on the principle that everyone in the world is connected through a small number of individuals or things. Detecting these vital or essential components is important, for the early detection of influencers.

User influence in social media can be defined as the probability of compelling other users to participate in a conversation. If we only focus on Twitter, influence can be seen as the likelihood of a user's post sparking conversations, discussions, mentions, likes and retweets by other users. Users that can regularly start conversations can be attributed to a higher influence.

[2] https://www.theguardian.com/technology/2008/aug/03/internet.email

[3] Watts, D. J. (2004). *Six degrees: The science of a connected age*. WW Norton & Company.

Meeyoung Cha et al. analyzed several measures from Twitter which can be attributed to influence [29] [30]. Focusing on individual's potential to have influence over other users, they highlighted three main features on Twitter which can be associated to it; indegree, retweet and mention influence.

3.2.1 Indegree influence

Indegree influence is the number of followers of a user. This tells us the number of users which will instantly receive a post from influential user. This can be considered closely related to a bursty topic because a topic in a post with larger initial audience of has more chances to get discussed among user. On Twitter celebrities and news organizations are mostly the users with highest followers. In the top ten users list, CNN (@cnnbrk) and New York Times (@nytimes) can be seen.

To add weight for indegree influence in our burst calculations, we take the log of the number of followers at the time of the post of the users as weight for this influence. This influence is applied as a user score for the calculation of term velocity for each post. Minimum score is given if the user has no followers.

3.2.2 Retweet influence

In this influence, we measure the effect from the tweets by other users which contained the original influencer's message. Users can retweet tweets of other users. These retweets start with 'RT' or 'via'. Retweets can also be considered as other users giving their indegree influence to influential user.

For bursty topic detection we will analyze the indegree influence of each user which has contributing a post to the topic. The indegree influence of the users who retweet the original tweet will also be included during the bursty topic detection. As we are getting real-time data from Twitter, calculating weight based on retweets is not possible, however our algorithm will add weight to the original tweet's topic.

3.2.3 Mention influence

Mention influence is the number of appearance of influential user's name in tweets of other users. The username is linked using the ate the rate (@) symbol before the username. This can show how much an influential user is discussed by other users. In our bursty topic detection algorithm, additional weight is given to usernames when keywords are being analyzed.

Features like the number of tweets by a user and the number of people users follow were not found to be useful at all, as the often led to spam. To analyze which users have the most influence and find a co-relation between them, Meeyoung Cha et al. aggregated a list of top users from all three influence categories [29]. They saw only a marginal overlap between these three categories, showing that these three categories capture different types of users. In Figure 3.5 we can see the normalized percentage of these users.

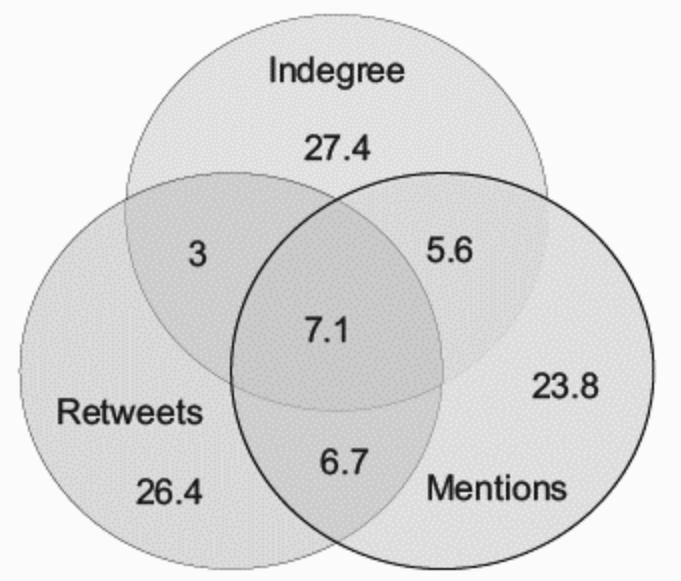

Figure 3.5: Venn diagram shows the overlap of three types of influential users [29]

Analyzing our dataset, we find similar trends and therefore we conclude that all three influence features must be used on topic detection as mostly users will belong to only one of these categories.

3.3 Topic Clusters

Soft Frequent Pattern Mining (SFPM) will be used to cluster terms into topics. As discussed in Section 2.4.4, in SFPM the relevant terms are first selected from the tweets in a time window. This is done by selecting terms with highest likelihood of appearance in the corpus of the current time window, in reference to the corpus of the previous time window. But selecting relevant terms using reference corpus gives preference existing terms in the reference corpus over new terms in the current corpus [31]. In order to cater to this issue,

Gaglio et al. proposed using likelihood ratio with $tf - idf$ which enables selecting emerging terms in the current corpus.

We propose to select the emerging terms which only have bursts in their frequency and not to focus on terms in the reference corpus at all. As discussed in Section 3.2 acceleration can be used to determine burst in frequency of a term. Using acceleration, we will get only the emerging terms from the corpus. The term selection process is summarized in Algorithm 3.1.

Algorithm 3.1 Term Selection

1: **function** SelectTerms ($\Delta T_1, \Delta T_2, T$)
2: $D_{\Delta T1}$ ← DocumentsInWindow (ΔT_1)
3: **for** each term x in $D_{\Delta T1}$ **do**
4: $V_{\Delta T1}(x)$ ← VelocityOfTerm (x, $D_{\Delta T1}$);
5: $V_{\Delta T2}(x)$ ← VelocityFromPreviousWindow (ΔT_2);
6: $A(x)$ ← AcclerationOfTerm($V_{\Delta T1}(x)$, $V_{\Delta T2}(x)$, $\Delta T_1, \Delta T_2$);
7: **if** ($A(x) > T$)
8: f_x ← [$A(x)$, x];
9: **end if**
10: **end for**
11: Sort (f, Ascending by $A(x)$)
12: **return** f

To cluster terms into topics we use SFPM algorithm summarized in Algorithm 2.1. SFPM uses a similarity threshold for cosine similarity between two terms to classify them into a single topic. This threshold is dynamically set using a sigmoid function [25].

$$\theta(S) = 1 - \frac{1}{1 + e^{\frac{|S|-b}{c}}}$$

The constants b and c are used to limit the size of the clusters and how close similarity should be between terms. The function allows lower requirement of similarity when the cluster size is small, but as the cluster size increases the similarity requirement also increases until it reaches its limit. Petkos et al. recommended value of b and c to be 5 and 2 respectively and used these values in their experiments [25]. We want to use a higher similarity requirement from the initial formation of a topic, therefore we used parameter values of 1 and 5. Figure 3.6 shows comparison of the two sigmoid functions. Using this sigmoid function, the average length of topics was 15.

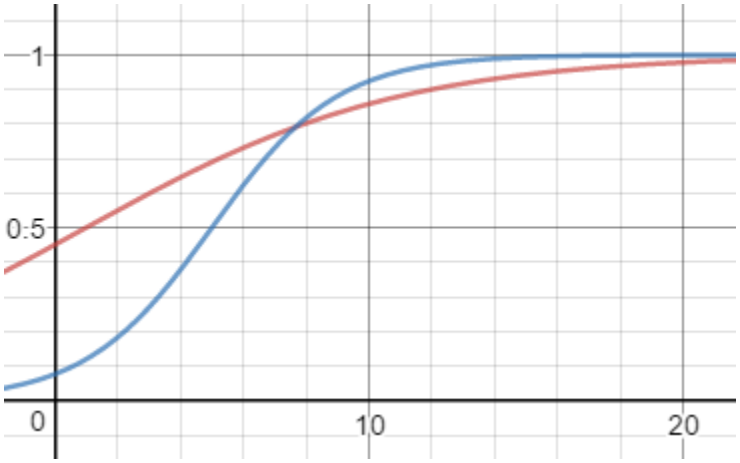

Figure 3.6: Sigmoid function with b = 5 and c = 2 (blue) and b = 1 and c = 5 (red)

The topics in this system are detected in time windows. The size of these windows can have huge impact on topics. Small time windows can have performance issues and large time windows may be slow to detect topics themselves. One solution proposed was using a dynamic detection window using a sigmoid function [41]. The function gives large time windows for topics which are discussed among users over a long period of time (elections, sports) and short windows for topics which have a very sudden burst (tragedies, events) in them. This technique is very useful when the topics are already known, but detecting new

topics can be difficult. As we focus only on new topics that have sudden bursts, we decide to calculate the time window based only on the number of posts in the previous time window. When the posts start to increase in the previous time window, the size of the window decreases and vice versa. This way we completely focus on new topics.

3.4 Concluding Remarks

This chapter presents the proposed framework for topic detection by sensing sudden bursts in their activity on Twitter. The overall framework is divided into three main parts. First we crawl our dataset from Twitter and apply pre-processing over it. After that we select the bursty keywords from the terms. The terms that have a sudden surge of occurrence in the dataset. After this the keywords are clustered into topics based on the similarity between them computed using term vectors. We will see details about the data crawling through Twitter and its pre-processing in Section 4.1.

The term frequency over a short time window is used to calculate the velocity and acceleration. Comparing the acceleration and the term frequency we saw that the acceleration will only increase when there is a sudden surge in the term frequency. Even if the term frequency is high but not increasing, the acceleration will be zero. Hence the acceleration represents the rapid increase of term frequency. The terms with acceleration are selected and term vectors are formed. Each element in the term vector is a binary indicator which shows whether or not the document contains the term. The length of the term vector is the number of documents in the time window.

We also use user influence to better estimate bursty terms and improve term selection quality. The influence is integrated in the term selection process using the followers of the users that post documents about the topic. The topic also benefits from the users who retweet the posts as well. We also give importance to the mention count of the users themselves, giving extra weight to the usernames. The weight is given to the selected terms to get a score for each term. The topics are ranked based on these scores.

The selected terms with their vectors are clustered into topics. Each topic is a set of keywords. The cluster is formed using a soft frequent pattern mining (SFPM) approach discussed in Section 2.4.4. SFPM uses cosine similarity to group terms together in a window of time. A threshold is used to set a minimum requirement for the cosine similarity. The threshold is set using a sigmoid function, which make requirement for topic low at start and high as the topic size increases.

We implemented this framework using Java and Mongodb. In the next section, we perform experiments on this framework and analyze input data and results. We also evaluate the results and see different characteristics of bursty topics.

Chapter 4

Experiments and Evaluation

4.1 Data-set

We have collected the data-set for our experiments using Twitter4J[4], a third-party tool that connects to Twitter API and provides integration with java applications. We collected real-time Twitter data of only English tweets mainly from the US and Canada. The data was extracted from 1st June to 25th June 2016. During this time the US presidential primaries were being held. This data is a sample from the complete data set. We can see frequency of documents for each day in Figure 4.1.

The raw data collected is then preprocessed using the following steps, we have already discussed preprocessing techniques in detail in section 2.1.

- Remove tweets not in English.
- Remove all non-content-bearing stop words.
- Remove Twitter specific keywords, like “RT” and “via”.
- Remove URLs and usernames. (identifying usernames with @ character).
- Remove emoticons.

[4] http://twitter4j.org/en/index.html

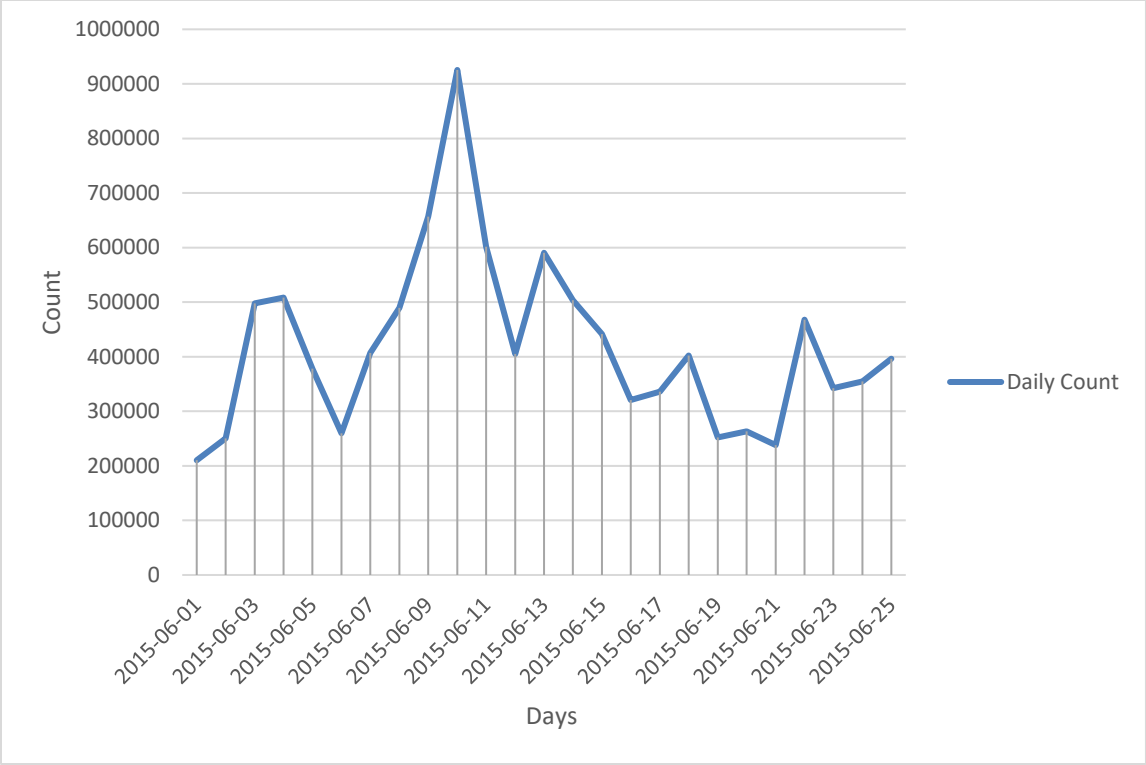

Figure 4.1: Distribution of tweets per day from 1st June to 25th June 2016

4.2 Simple Bursty Keyword Detector using N-gram model

We first implemented a simple framework for detecting bursty topic keywords as a test project. The framework used frequency distribution of unigrams and bigrams to calculate bursty words. First we detect a sudden burst in word frequency, this measure provides us with the surge of popularity of words and the second measure provides us with keyword correlation information and their surge of popularity in the correlated pairs.

To focus only on the content which is of higher importance we used a set of rules to filter out the words which we will consider for the topic detection process.

- Words which have correlation sets will only be considered in the topic detection process.
- A threshold is applied to the number words that will be considered. The threshold will make sure only the words with a higher frequency and correlation are tracked.
- Outliers are removed using the frequency of the words to make sure the spam words are removed.

The keywords are selected using bigram and unigram frequency. The bigram frequency is adjusted using the equation below. The adjusted frequency F_{ij} is decreased if the individual word frequency is much higher than the bigram frequency.

$$F_{ij} = f_{ij} - w\left[\frac{f_i - f_{ij}}{f_{ij}}\right] - \ w\left[\frac{f_j - f_{ij}}{f_{ij}}\right],$$

Where f_i and f_j are individual word frequency and are always equal to or larger than the bigram frequency f_{ij}. The weight, w, is used to increase or decrease the effect of the difference and can vary depending on the dataset.

We extracted bursty keywords and their frequency to see their relations with search terms. Figure 4.2 shows sudden bursts of keywords using stacked graph for data of one day only. We can see certain keywords have their bursts on a small period. For example, keyword "zodiac killer" was bursty only in the first two hours of the day. On the other hand, keyword "illegal immigrants" had small bursts throughout the day. We also list the total document frequency of each keyword in the Table 4.1. We see that both the topics we discussed

earlier in the example have close document frequency but can be separated (or ranked) by the size of their bursts more easily.

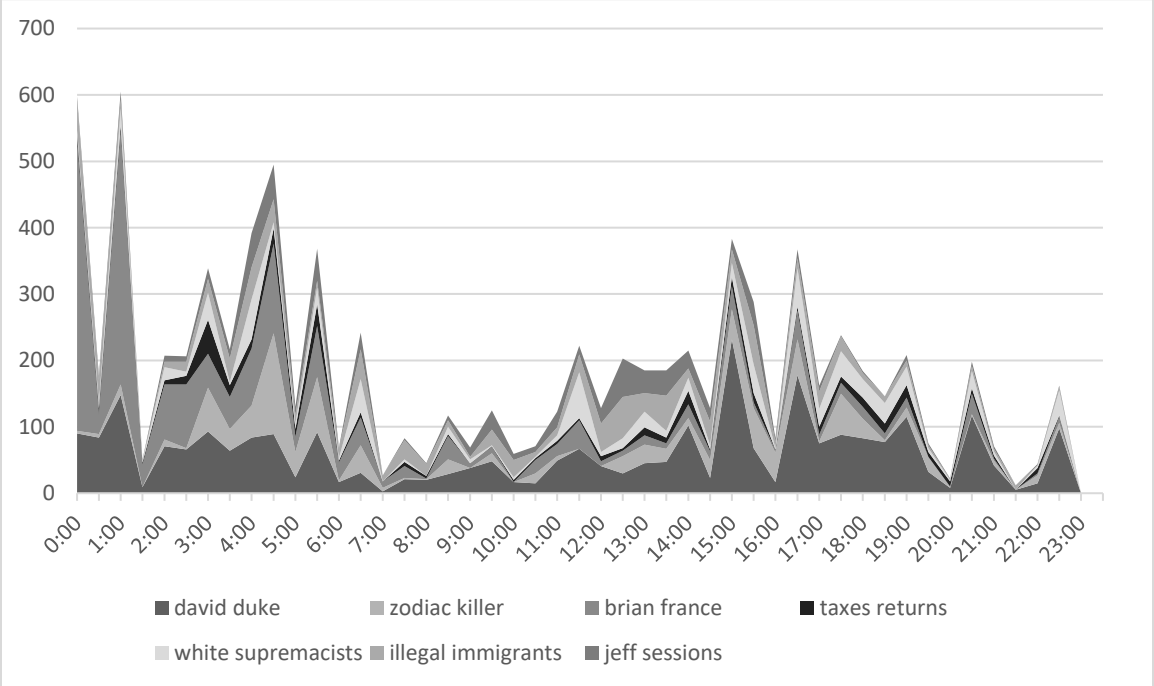

Figure 4.2: Stacked area of document frequency of top topics for entire day

Keywords	Document Frequency
david duke	2807
zodiac killer	1042
brian france	1989
taxes returns	440
white supremacists	813
illegal immigrants	948
jeff sessions	683

Table 4.1: Document frequency of keywords for entire day

4.3 Evaluation Method

We implemented our proposed system on Java 1.8 using 64-bit addressing and used mongodb 3.2 for storage. The process was executed on an Intel Core i7 3.4 GHz CPU with 16 GB of RAM.

The main application of our system is to find top K bursty topics over time. The bursty topics are ranked using a score based on the burstiness of the topic. We can evaluate the bursty topics by analyzing their keywords against baseline topics and keywords obtained using a topic model. We took inspiration from other systems [25] [32] [33] [34] for method of evaluation. For this we used LDA (Latent Dirichlet Allocation) as our baseline. LDA uses a probabilistic model to calculate topics from a corpus. We used LDA over the entire data set to detect 30 topics, with each topic having 15 keywords. We had an average of 15 keywords in the top 100 detected bursty topics as well. The precision and recall of each bursty topic was calculated using Table 4.2.

	Keywords in bursty topic	Keywords not in bursty topic
Keywords in topic	a	b
Keywords not in topic	c	d

Table 4.2: Bursty topic evaluation method table

$$P = \frac{a}{(a+c)}$$

$$R = \frac{a}{(a+b)}$$

The keyword precision of bursty topics is the percentage of keywords in both the bursty topic and the LDA topic over the total keywords in LDA topic. The recall is the percentage

of keywords in both the bursty topic and the LDA topic over the total keywords in a bursty topic.

4.4 Evaluation and Discussion

In this section, we evaluate our framework by first analyzing the bursty topics obtained over the dataset using only acceleration to estimate bursts. We first enlist the top four topics obtained from the entire dataset in Table 4.3. Each topic is listed with information about its starting state and its peak state. The start state is when our system categorized the topic as bursty and peak state is when the topic obtained its maximum score over the course of its time. We list the keywords as well as the hour, score and mentions at the start and peak state of the bursty topic.

Date (UTC)	Topic	Keywords		Time (UTC)	Score / Mention
2016-06-12	Orlando night club shooting	Start	Peak	Start	6.06
		terror, orlando, nightclub, radical, islam	terror, orlando, horrific, nightclub, toughness, radical, islam	8:00	8340
				Peak	17.14
				11:00	19187
2016-06-09	President Obama	Start	Peak	Start	0.98
		obama, qualified, endorses, clinton		19:00	2024
				Peak	4.92

	endorses Clinton		obama, endorsed, clinton, courage, compassion	20:00	9108
2016-06-11	Donald trump rally in Florida	Start	Peak	Start	1.6
		immigrants, tampa, outstanding, trump, florida	immigrants, tampa, trump, volunteers, florida	18:00	4330
				Peak	5.93
				21:00	9125
2016-06-13	Discussions on Orlando night club shooting	Start	Peak	Start	1.18
		families, trump, toughness, terror, islam, puppet	radical, islam, toughness, trump, hard, obama, terror	3:00	1570
				Peak	6.97
				9:00	5474

Table 4.3: Top 4 bursty topics over entire dataset

As expected, Orlando night club shooting incident is the highest bursty topic in the dataset. We started tracking the topic at 8:00 (UTC) on 12th June in under two hours the incident itself started. The Orlando police tweeted the incident at 7:58 (UTC) with the following tweet "Shooting at Pulse Nightclub on S Orange. Multiple injuries. Stay away from area."[5]. The topic reached its peak at 11:00 (UTC) with thousands of tweets from users around the world condemning the incident.

[5] https://twitter.com/OrlandoPolice/status/741902485070045184

The topic picked up again next day with more information coming up on Twitter about the people involved in that incident. Other bursty topics are about president Obama endorsing Hilary Clinton and a discussion on a Trump rally in Florida and his statements. Both topics were reported in news media live and their discussions followed through on Twitter.

We show the mention count of keywords in the topic and the score (calculated using the acceleration of mention count of keywords in the topic) of each topic against hours in Figure 4.3 to Figure 4.6.

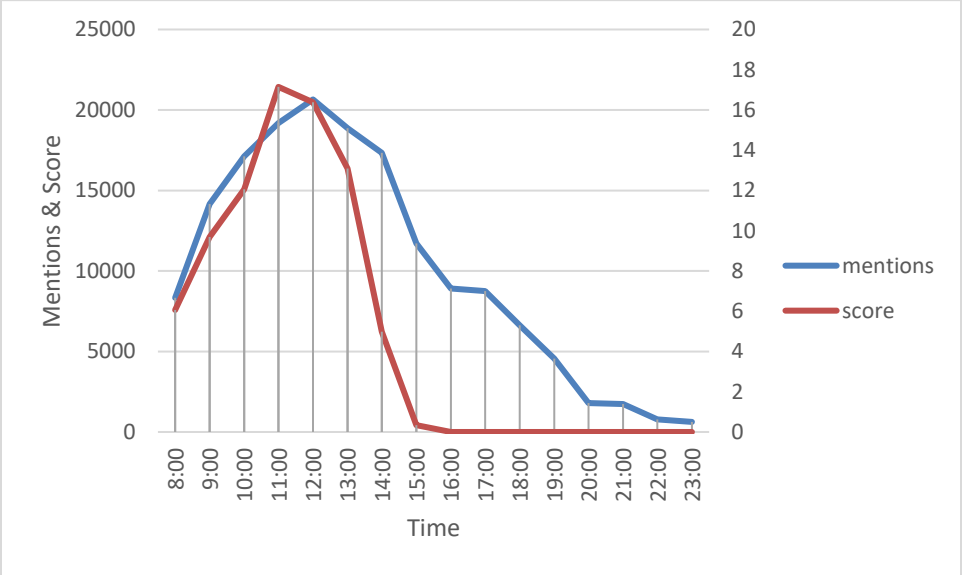

Figure 4.3: Bursty Topic "Orlando night club shooting" against time

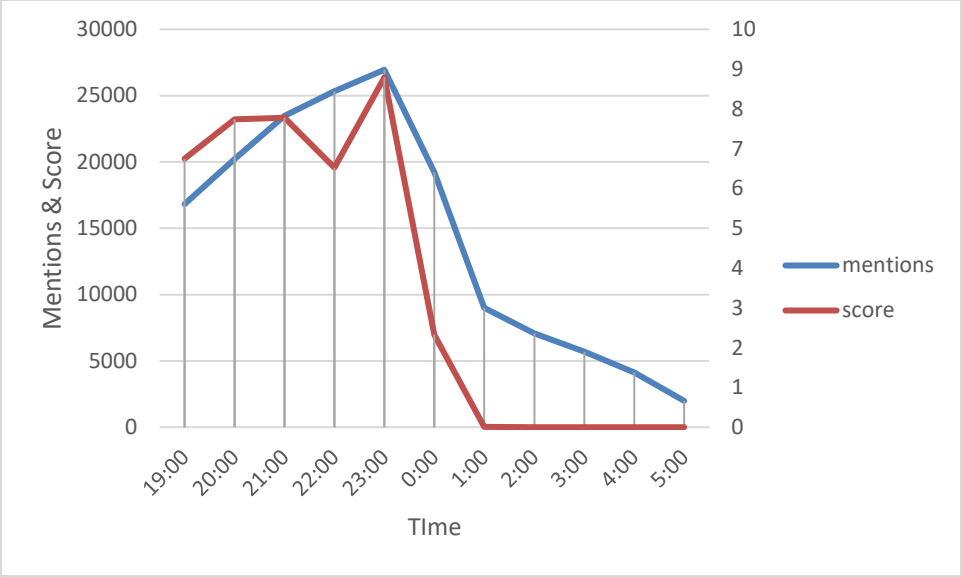

Figure 4.4: Bursty Topic "President Obama endorses Clinton" against time

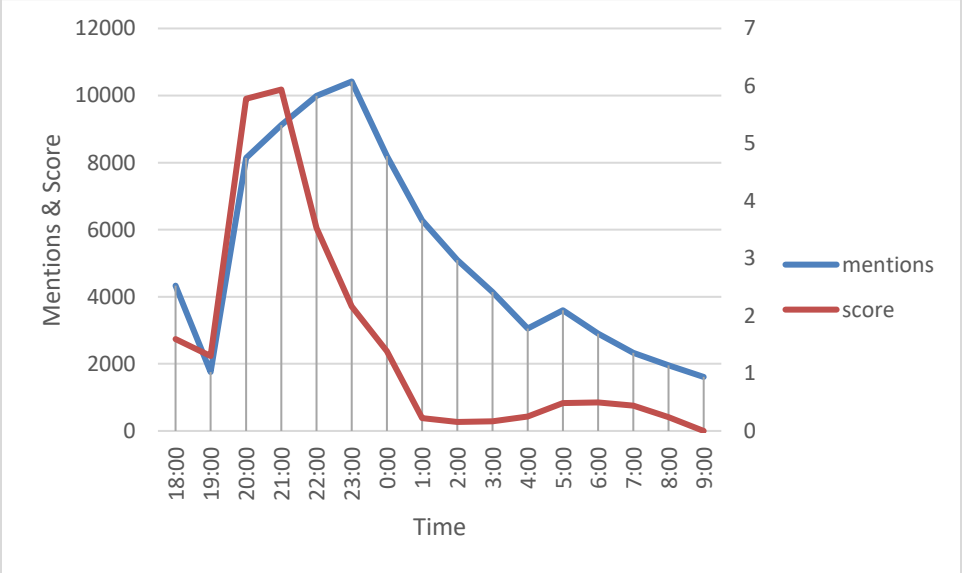

Figure 4.5: Bursty Topic "Donald trump rally in Florida" against time

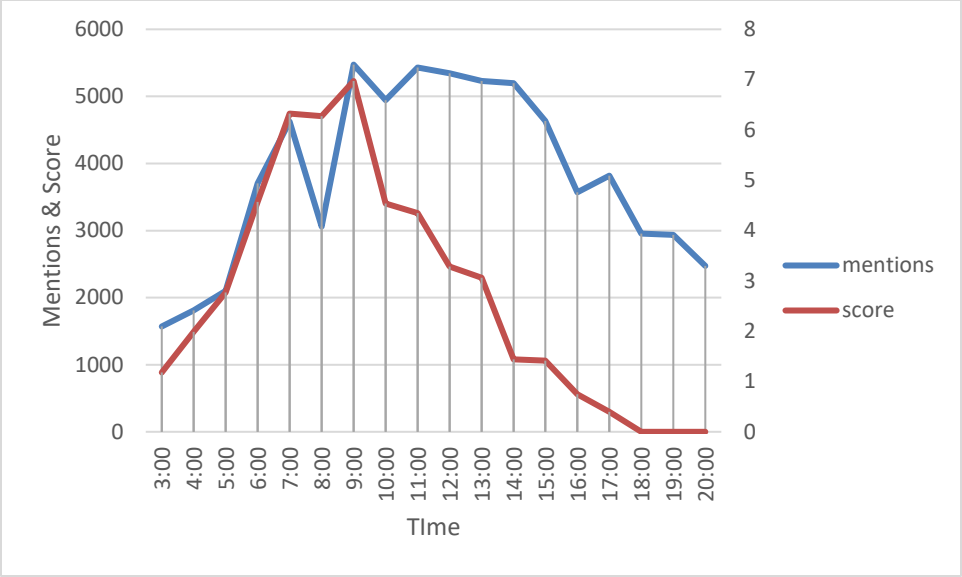

Figure 4.6: Bursty Topic "Discussions on Orlando night club shooting" against time

We can see how score is closely related to the acceleration of the mentions of the topic. The topic is detected as the acceleration starts to increase and then quickly reaches to 0 as the acceleration decreases. We adjusted our score to be a weighted average over time, so that the fall because of no acceleration becomes smooth rather than a sudden fall to 0.

As discussed earlier we have used LDA topic model to extract 30 topics (not based on temporal information) from our entire corpus. Each topic has 15 keywords in them. We can now calculate the precision and recall of the keywords of the bursty topics (discussed is section 4.2) using LDA topics as baseline, we show their calculations in Table 4.4.

We see that the top 4 bursty topics could detect more than half of the keywords from LDA topics. In Topic 4 has only half of the bursty topic keywords are in the LDA topic keywords and has the lowest precision. This is because bursty topics only detect keywords from a smaller time window as compared to LDA topics and we still consider above 40% precision high. The precision could be higher among the most bursty topics if we put more keywords

in the LDA topics but as we have 15 keywords on average in bursty topics, we limited LDA topics to only have 15 keywords as well. Also, if the LDA topic is larger than the bursty topic it will result in a low recall. Like in Topic 2 the total keywords detected by the bursty topic are only 11, this leads to a fewer matches with LDA topics as compared to the other topics and hence lowest recall.

	Topic 1	Topic 2	Topic 3	Topic 4
Matched bursty topic keywords with LDA topic	13	6	11	8
Keywords in bursty topic	23	11	21	19
Keywords in LDA topic	15	15	15	15
Precision	0.565	0.545	0.523	0.421
Recall	0.866	0.4	0.733	0.533

Table 4.4: Keyword Precision and Recall of top 4 topics

Continuing this we can calculate the precision and recall of top K bursty topics. Figure 4.7 shows this in a chart. The highest precision is 0.480 and the highest recall is 0.616 for the top 10 bursty topics. As our system is detecting each bursty topic from documents belonging to a very small time window and LDA has the entire dataset for topic calculation, the precision and recall will be low in most cases in top K bursty topics. We see that both precision and recall decrease as we start considering lower ranked bursty topics. The precision sharply decreases between top 10 to 30 bursty topics. It is then steady around 0.4 between top 30 to 80 bursty topics and then slightly decreases again. The recall also decreases sharply at the start and then is steady around 0.4 between top 30 to 80 bursty topics and then sharply decreases again.

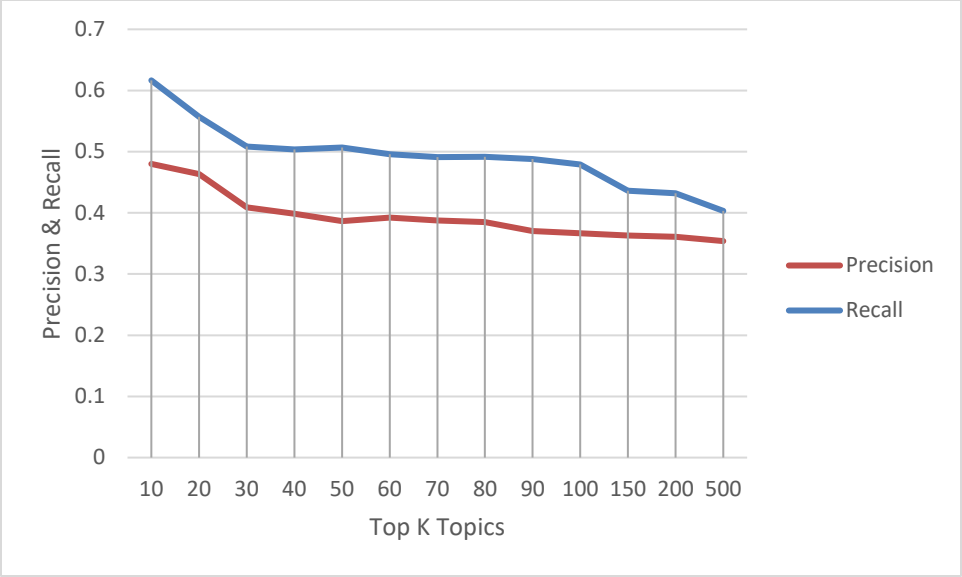

Figure 4.7: Precision and Recall of top K topics

We can now compare the above precision and recall with the bursty topics obtained using user influence weight added in the framework calculations. We believe that adding this weight will improve both precision and recall as more weight will be behind more popular tweets and users. As we will be able to get more relevant keywords related to the topics even in a small time frame.

Figure 4.8 and Figure 4.9 compare the precision and recall respectively, of influence and acceleration based bursty topics, as well as the original soft frequent pattern mining (SFPM) approach discussed in Section 2.4.4. We see an 11.3% increase in precision and a 4% increase in recall on average with influence based bursty topics over acceleration based and both perform better than SFPM approach. The precision in the top 10 to 60 bursty topics is very high with influence based bursty topics as compared to acceleration based and SFPM. After the top 60 bursty topics precision stays similar, with influence based bursty topics having a slightly high precision. The recall of influence based bursty topics

is also higher from the top 10 to 60 topics and only gives a slight difference after that. In all cases influence and acceleration based bursty topics outperform bursty topics based on only acceleration and bursty topics based on only the SFPM approach.

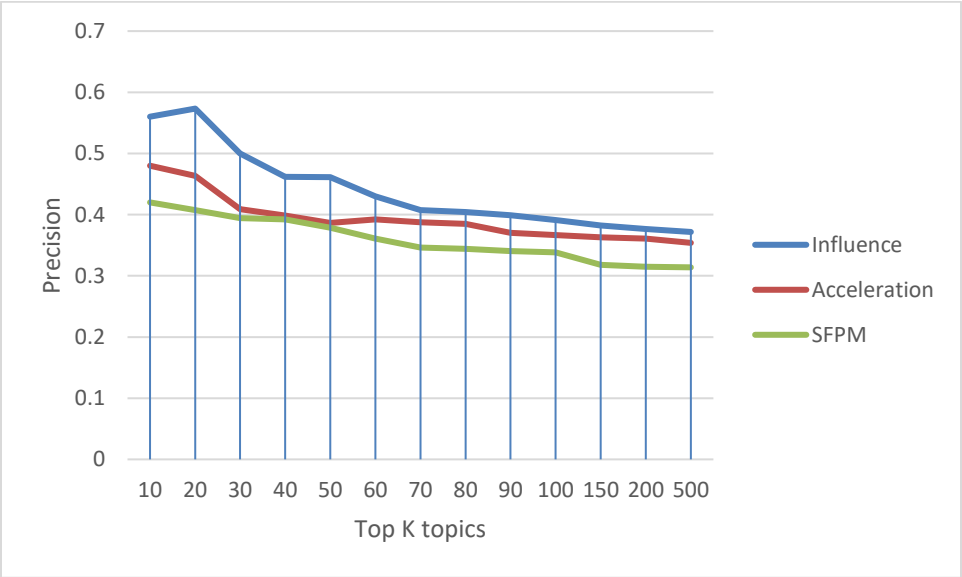

Figure 4.8: Precision comparison of influence and acceleration bursty topics and SFPM

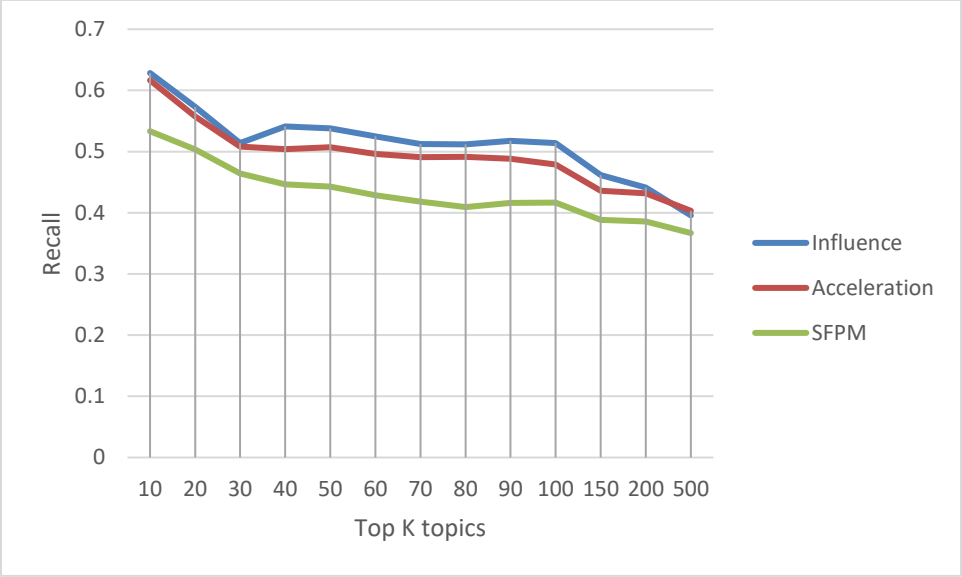

Figure 4.9: Recall comparison of influence and acceleration bursty topics and SFPM

4.5 Bursty Topic Characteristics

To further understand bursty topics we use statistical analysis to see relationships between bursty topics and the total mentions of their keywords in the entire dataset. We analyze the volatility of the bursty topics using relative standard deviation against total mentions, mean mentions and relative standard deviation of the mentions. Relative standard deviation is calculated from standard deviation normalized by dividing it by the mean. The topic is more volatile if it has a large relative S.D. In Figure 4.10 we can see chart between relative standard deviation of topics and total mentions. We see that topics having less relative standard deviation and a low volatility can still have a high total mention count. This is possible when the topic is consistently discussed throughout the time window but does not experience a significant acceleration. Looking at the trendline, the R^2 value is 0.0891. A higher value of R^2 shows that data is fitting the model. But in this case, low R^2 value means there is a very weak relation between volatility and mentions of the bursty topics.

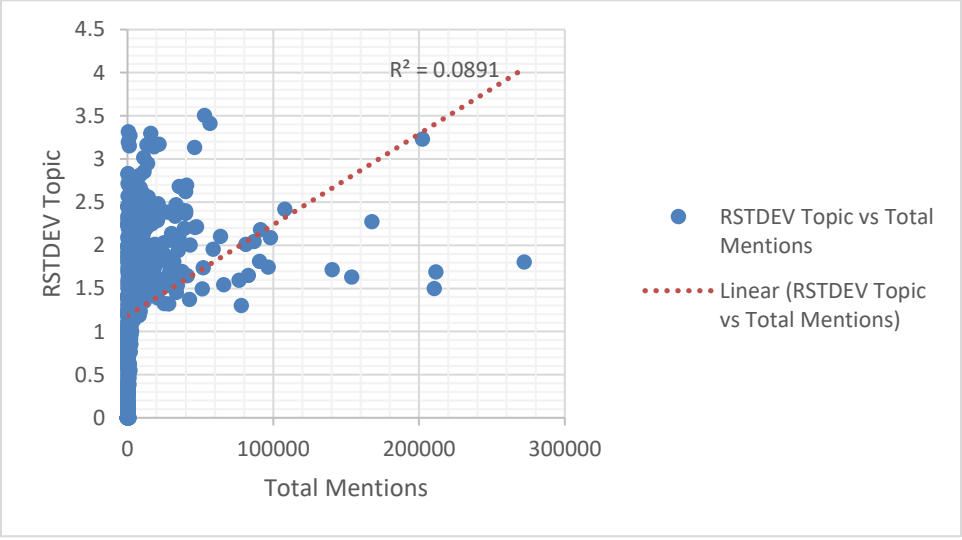

Figure 4.10: Relative Standard Deviation Topics vs Total Mentions

In Figure 4.11 we see chart between relative standard deviation of topics and mean mentions. The R^2 value is 0.1441, again the low R^2 value is because there is a very weak relation between volatility and mentions of the bursty topics. Topics with high mean can have low volatility in them.

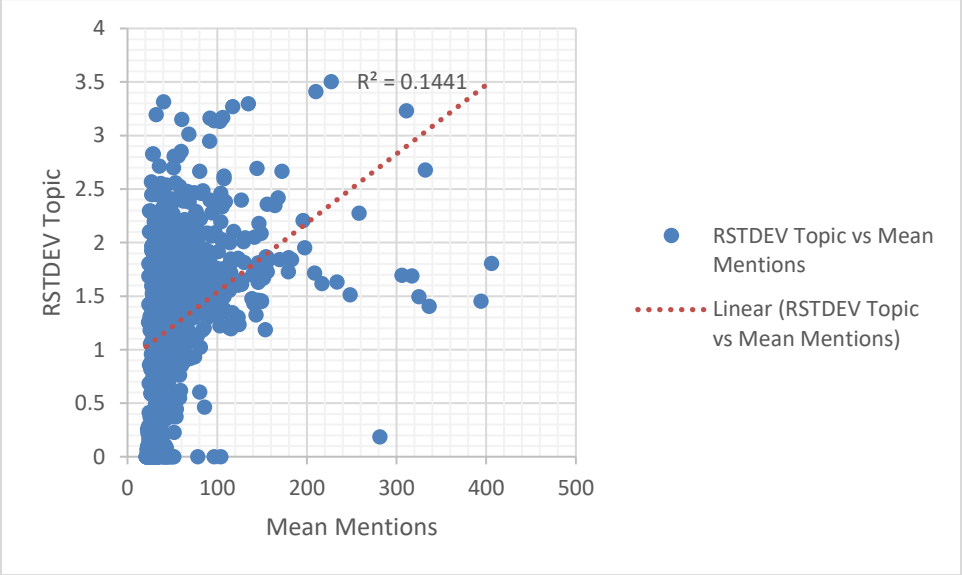

Figure 4.11: Relative Standard Deviation Topics vs Mean Mentions

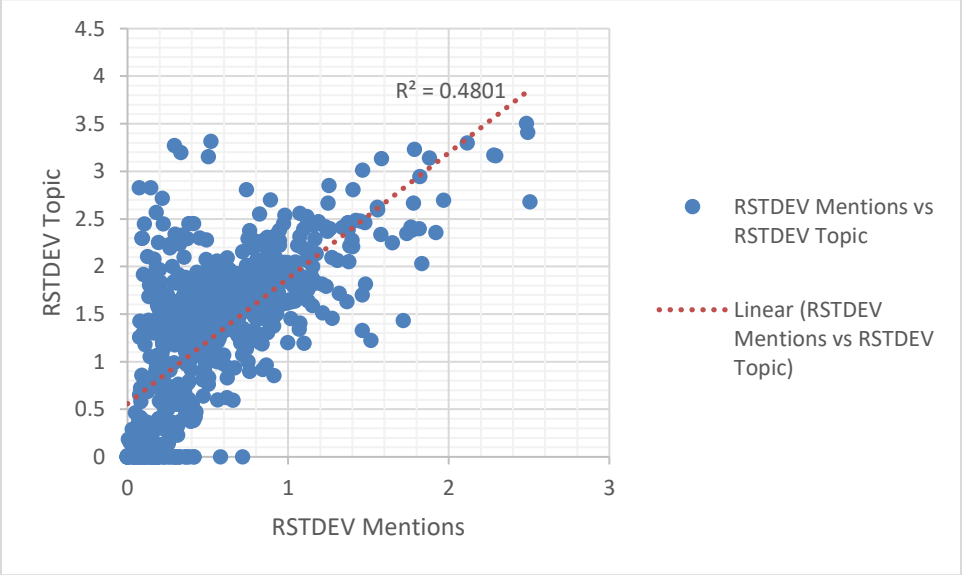

Figure 4.12: Relative Standard Deviation Topic vs Relative Standard Deviation Mentions

In Figure 4.12 we see chart between relative standard deviation of topics and relative standard deviation of mentions. Both measures in the chart show volatility and can be considered related. This is because the relative standard deviation of mentions shows the variation in the counts. The R^2 value 0.4801 shows the close relationship between these two measures. The positive sign of R^2 shows that volatility of both topic and keyword mentions increases at the same time.

4.6 Visualization

We further see interesting visualizations of bursty topics which help us understand them and to give a user interface concept that can help users use this system. In Figure 4.13 we see a stacked bubble chart of bursty topics retrieved from this system. Each bubble represents a topic at some time window in our dataset. The color of the bubble tells the topic of the bubble with nearby bubbles indicating the same topic. The size tells the mentions and the shade of the color tells the score of that topic in a time window. As we can see a very large chunk (al shade of green bubbles near the center) of topics detected were related to the US presidential primaries. Still we have a diverse set of topics, with some bubbles even larger than US presidential primaries. These large topic bubbles include the Orlando shooting on 12th June of 2016. We used Tableau[6] to generate this graph. Some outliers were also removed to simplify the chart.

[6] https://www.tableau.com

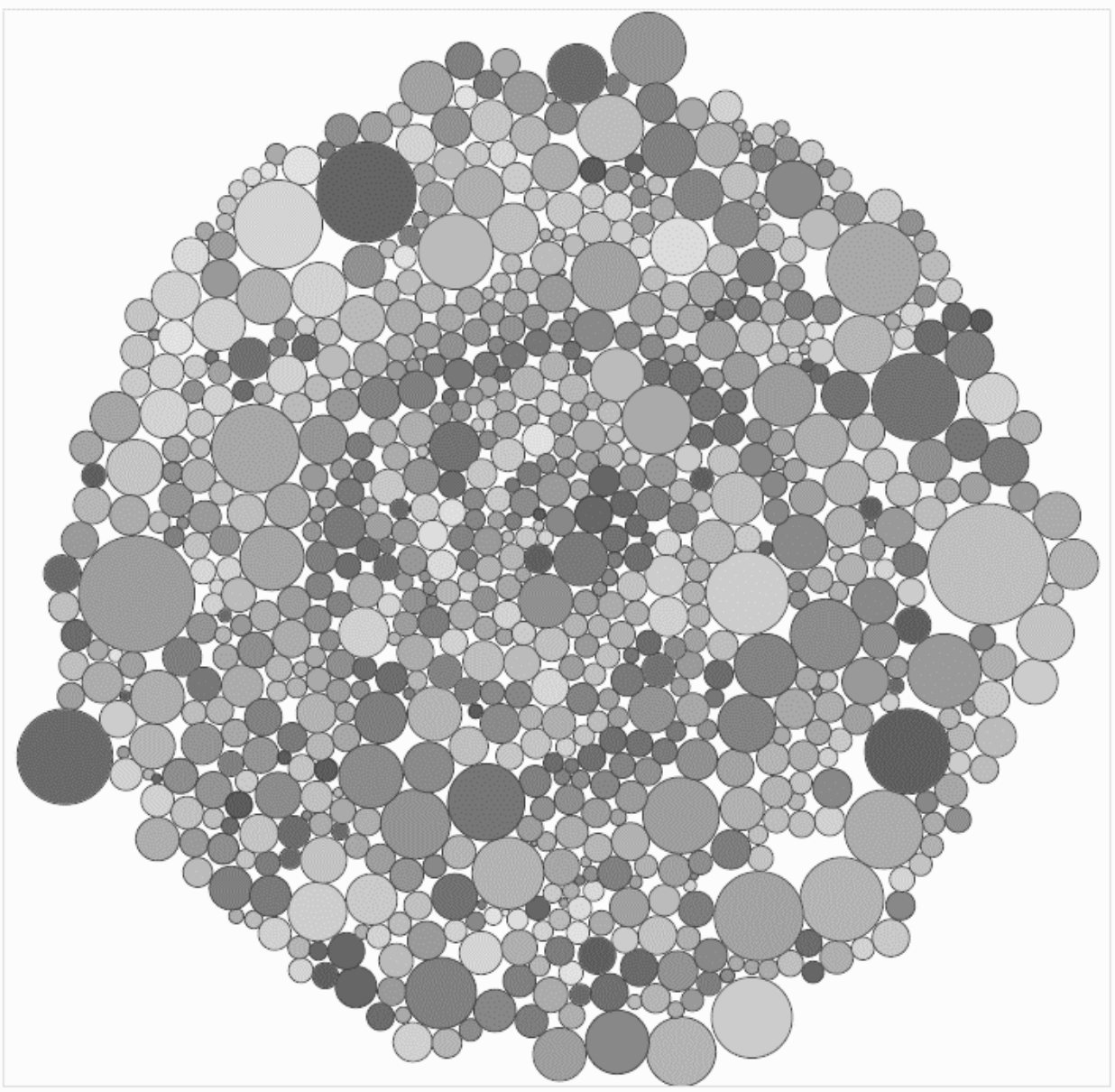

Figure 4.13: Packed bubbles chart of bursty topics

We also developed two web-based representations of bursty topics using highcharts[7]. Highcharts is tool used to develop charts on web applications using java script. We first

[7] http://www.highcharts.com/

used polar chart with 24 intervals in it which represent 24 hours of a day. Each point in the chart represents a topic at that hour and the closer it gets to the center the higher the score of the topic is. We can see a polar chart in Figure 4.14.

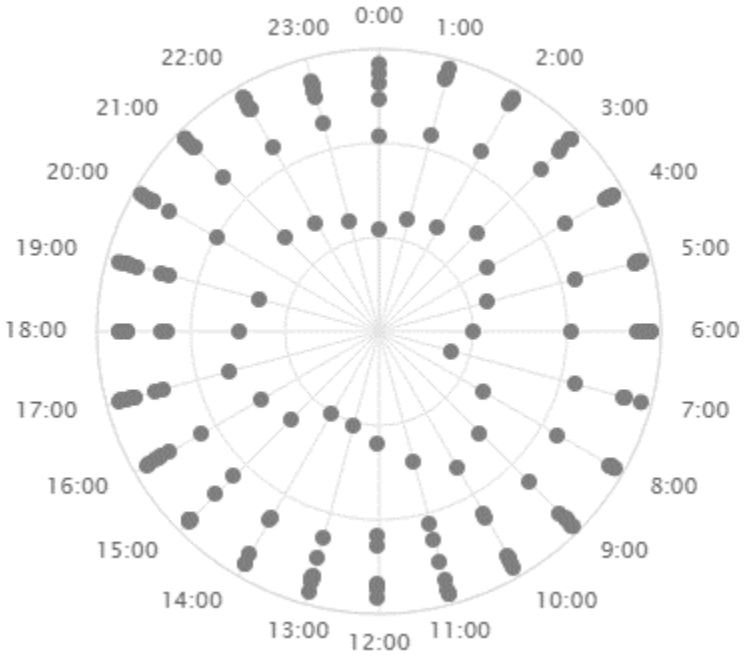

Figure 4.14: Polar chart showing topics of 24 hours

In Figure 4.15 we can see a heat map also representing 24 hours of a day. Each cell in the heat map is a topic and the heat color of the cell represents the score of the topic. The topics are also arranged for each hour in descending order of the mentions. We found the heat map to be very useful to quickly skim through days and hours of detected topics.

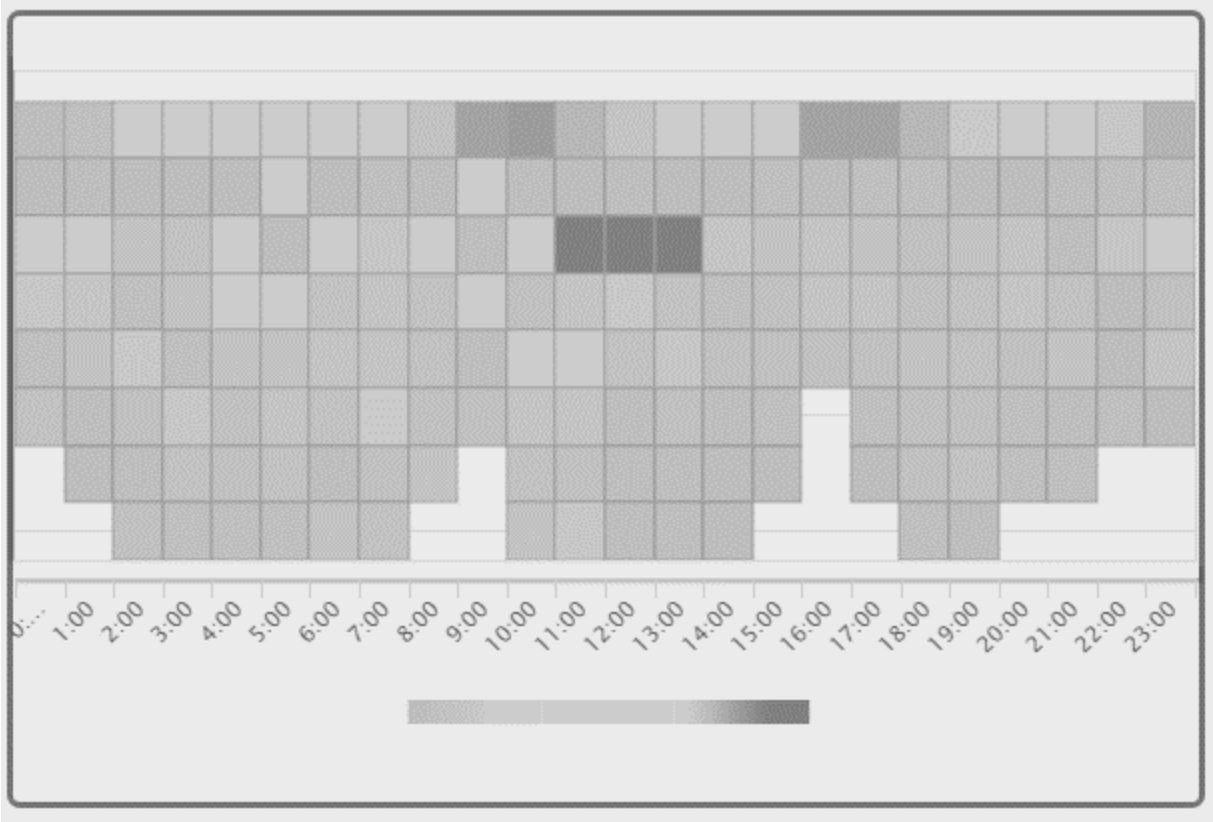

Figure 4.15: Heat map showing score and mentions of topics

4.7 Concluding Remarks

In this chapter, we presented comprehensive experiments and results of our framework on a dataset extracted from Twitter between June 1st to 25th 2016. The evaluation method we used analyzed top K topics, calculating their precision and recall. The precision and recall varied as we increase the K, from top 10 topics to top 500 topics. The precision and recall was calculated using another topic model called Latent Dirichlet Allocation (LDA). Using LDA we got topics from the entire corpus, without using any of the temporal information which our framework is based upon, and we assume these topics as our baseline.

The precision and recall of both acceleration and influence base bursty topic detector is calculated. The precision in influence based bursty topic detector improved 11% from acceleration based bursty topic detector. The recall only slightly improved. We conclude that adding user influence improved both precision and recall as more weight is behind more popular tweets and users. This is because we can get more relevant keywords related to the topics even in much smaller time windows. We also analyzed come characteristics of bursty topics score with its relation to overall mentions.

We also experimented with different visualization techniques for our system. We used a stacked bubble chart to show the overall bursts detected and how they are divided into different topics. We used a polar chart to show topics on a 24-hour clock and finally we also used a heat map that can help skim through large number of topics very quickly.

Chapter 5

Conclusion and Future Work

5.1 Conclusion

Bursty topic detection is used for highlighting interesting bursty trends on social media. Several researchers have proposed methods based on different feature selection methods or a topic clustering algorithm to detect bursty topics. We can divide them into either Document-pivot or Feature-pivot techniques. Due to high performance cost, document-pivot techniques are not beneficial for computing topics in real-time data where millions of documents can be generated in hours. Feature -pivot techniques, focusing on keywords extracted from the corpus, are a cheaper alternative. After extracting interesting features, clustering algorithms are used to form topics from the keywords.

We presented a framework that uses acceleration as a primary indicator for feature selection. The acceleration of terms in the corpus is calculated over several smaller time windows. It guides the system to find bursty keywords that can be used as features in our framework. Vector of keywords are formed. Each vector has length equal to the number of documents in the time window. The term vector represents documents where the term was used. The term vector can only have binary values.

We also used user influence as a weight to calculate a score, using acceleration as well in it. The user influence guides the system to better quality keywords. To form topics from

the keywords and their vectors we use soft frequent pattern mining approach (SFPM). It uses cosine similarity to steadily group similar keywords into topics.

We implemented the system using Java and Mongodb to perform experiments on them. The data was extracted from Twitter from 1st June to 25th June 2016. The evaluation method we used calculated precision and recall using Latent Dirichlet Allocation (LDA). We used LDA with 30 topics from the entire corpus each topic having 15 keywords in them. The LDA topics were not generated for smaller time windows, they generated for the entire collection at once without using any temporal information. We used these LDA topics as our baseline.

The precision of the influence based bursty topic detector was 11% higher on average from acceleration based bursty topic detector. The recall improved by only 4%. We conclude that adding user influence improved both precision and recall as more weight is behind tweets from more popular users. This is because we give more importance to tweets that can possibly get more relevant keywords related to the topics in an even smaller time windows. We also analyzed some characteristics of bursty topic score with its relation to overall mentions and found that bursty topics are not closely related to the total mentions of the topics, instead they relate to its volatility which we analyzed using relative standard deviation.

We also experimented with different visualization techniques including stacked bubble chart, polar chart and a heat map. We found that these visualizations can help users further understand bursty topics in the entire dataset as well as on an hourly time bases.

Our system had a maximum precision of 57% on top 20 bursty topics and on average of top 500 bursty topics our precision was 45%. User influence helped us improve the precision and slightly improve the recall.

5.2 Application of Topic Detection

In recent years, we have seen a surge in topic detection applications on social media especially ones using Twitter. With the enormous amount of user generated content produced every day, topic detection has become a major challenge. Many applications have been developed to cater the requirements of corporations who want to know people's opinion of them on social media.

Abrahams et al. [35] developed a framework which takes social media content to give information about product defects. The framework uses text mining and topic detection techniques to discover product defects. The framework uses distinctive terms related to the product, its features as indictor of defects which are being discussed. They also proposed a similar framework that discovers vehicle defects [36]. The application was targeted on vehicle quality management professionals. The framework helped prioritize defects in vehicles which were being discussed online by users.

Another interesting application was proposed by Sakaki et al. [37]. The application proposed using Twitter feed to detect earthquake shakes real-time. To detect tweets relating to the target event, the classifier used features such as keywords related to topic detection, number of words in the tweet and its context. A probabilistic model was used to determine the location of the earthquake.

Nichols et al. proposed a system that used Twitter stream to summarize sporting events [38]. The system used status updates posted by users on social media to fill details of a sporting event and provide a summary and opinion of users about the event. Twitter users largely give opinion about sports events on Twitter. The application tries to leverage from this rich content to provide a detailed summary.

Khare et al. proposed an event detection method which can help journalists get breaking news and understand the opinion of people about specific news or an event [39]. News organizations and journalists heavily rely on Twitter to get news in real time. But directly accessing data from Twitter requires large amount of manual labor. Therefore they propose a solution that takes stream of documents from social media as input and provides likely events by clustering documents as well as providing context to the events.

Ciceri et al. [40] proposed a system that performs topic detection on food related content. It tracks tweets related to food that are being talked about on Twitter, as well as a general discussion and opinion of people about the food. Promotions of food products are also detected.

5.3 Future Work

In future following research work and experiments should be performed to extend the work of this system with:

- Experiment on data set extracted after US elections. In our data set a large number of English tweets were related to Trump and Clinton. Using a data set that has a more diverse range of topics can validate our system more thoroughly.

- Use a longer data-set for an entire year, our system was validated on almost one month of data. Validating it for longer period will help to further improve the system and help find more interesting characteristics of bursty topics.
- Analyze other types of data. For example, network data for cybersecurity and malware detection, in online advertising based real-time user interests, in elections to find, peruse and change public perception of political leaders and in research communities to identify emerging new research work.
- Develop more interactive tools and visualizations for better usability, analysis and understanding of the system, which can help adaptation of this system to the public. Also, develop mobile versions of the visualizations.

Chapter 6

References

[1] Chaffey, D. (2016). Global social media research summary 2016. *Smart Insights: Social Media Marketing*.

[2] De Cristofaro, E., Soriente, C., Tsudik, G., & Williams, A. (2012, May). Hummingbird: Privacy at the time of twitter. In *Security and Privacy (SP), 2012 IEEE Symposium on* (pp. 285-299). IEEE.

[3] Miner, G., Elder IV, J., & Hill, T. (2012). *Practical text mining and statistical analysis for non-structured text data applications*. Academic Press.

[4] Sankaranarayanan, J., Samet, H., Teitler, B. E., Lieberman, M. D., & Sperling, J. (2009, November). Twitterstand: news in tweets. In *Proceedings of the 17th acm sigspatial international conference on advances in geographic information systems* (pp. 42-51). ACM.

[5] Petrović, S., Osborne, M., & Lavrenko, V. (2010, June). 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* (pp. 181-189). Association for Computational Linguistics.

[6] Becker, H., Naaman, M., & Gravano, L. (2011). Beyond Trending Topics: Real-World Event Identification on Twitter. *ICWSM*, *11*(2011), 438-441.

[7] Porter, M. F. (1980). An algorithm for suffix stripping. *Program*, *14*(3), 130-137.

[8] Fox, C. (1989, September). A stop list for general text. In *ACM SIGIR Forum* (Vol. 24, No. 1-2, pp. 19-21). ACM.

[9] Zhu, T., & Yu, J. (2014). A prerecognition model for hot topic discovery based on microblogging data. *The Scientific World Journal*, *2014*.

[10] Liu, Y., & Luo, X. (2013, December). A framework for online hot event discovery on the web. In *Computational Science and Engineering (CSE), 2013 IEEE 16th International Conference on* (pp. 989-996). IEEE.

[11] Raghavan, U. N., Albert, R., & Kumara, S. (2007). Near linear time algorithm to detect community structures in large-scale networks. *Physical review E*, *76*(3), 036106.

[12] Xie, W., Zhu, F., Jiang, J., Lim, E. P., & Wang, K. (2016). Topicsketch: Real-time bursty topic detection from twitter. *IEEE Transactions on Knowledge and Data Engineering*, *28*(8), 2216-2229.

[13] Owoputi, O., O'Connor, B., Dyer, C., Gimpel, K., Schneider, N., & Smith, N. A. (2013). Improved part-of-speech tagging for online conversational text with word clusters. Association for Computational Linguistics.

[14] Phuvipadawat, S., & Murata, T. (2010, August). Breaking news detection and tracking in Twitter. In *Web Intelligence and Intelligent Agent Technology (WI-IAT), 2010 IEEE/WIC/ACM International Conference on* (Vol. 3, pp. 120-123). IEEE.

[15] Yang, Y., Pierce, T., & Carbonell, J. (1998, August). A study of retrospective and on-line event detection. In *Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 28-36). ACM.

[16] Toutanova, K., Klein, D., Manning, C. D., & Singer, Y. (2003, May). Feature-rich part-of-speech tagging with a cyclic dependency network. In *Proceedings of the 2003*

Conference of the North American Chapter of the Association for Computational Linguistics on Human Language Technology-Volume 1 (pp. 173-180). Association for Computational Linguistics.

[17] Allan, J., Carbonell, J. G., Doddington, G., Yamron, J., & Yang, Y. (1998). Topic detection and tracking pilot study final report.

[18] Aiello, L. M., Petkos, G., Martin, C., Corney, D., Papadopoulos, S., Skraba, R., ... & Jaimes, A. (2013). Sensing trending topics in Twitter. *IEEE Transactions on Multimedia*, *15*(6), 1268-1282.

[19] Fox, C. (1989, September). A stop list for general text. In *ACM SIGIR Forum* (Vol. 24, No. 1-2, pp. 19-21). ACM.

[20] Allan, J. (2002). Introduction to topic detection and tracking. In *Topic detection and tracking* (pp. 1-16). Springer US.

[21] Xu, J., & Croft, W. B. (1996, August). Query expansion using local and global document analysis. In *Proceedings of the 19th annual international ACM SIGIR conference on Research and development in information retrieval* (pp. 4-11). ACM.

[22] Atefeh, F., & Khreich, W. (2015). A survey of techniques for event detection in twitter. *Computational Intelligence*, *31*(1), 132-164.

[23] Salton, G., & Buckley, C. (1988). Term-weighting approaches in automatic text retrieval. *Information processing & management*, *24*(5), 513-523.

[24] Madani, P. (2015, June). A Framework for Discovering Bursty Events and Their Relationships from Online News Articles. In *Canadian Conference on Artificial Intelligence* (pp. 352-355). Springer International Publishing.

[25] Petkos, G., Papadopoulos, S., Aiello, L., Skraba, R., & Kompatsiaris, Y. (2014, June). A soft frequent pattern mining approach for textual topic detection. In *Proceedings of the 4th International Conference on Web Intelligence, Mining and Semantics (WIMS14)* (p. 25). ACM.

[26] Long, R., Wang, H., Chen, Y., Jin, O., & Yu, Y. (2011, September). Towards effective event detection, tracking and summarization on microblog data. In *International Conference on Web-Age Information Management* (pp. 652-663). Springer Berlin Heidelberg.

[27] Rose, S., Engel, D., Cramer, N., & Cowley, W. (2010). Automatic keyword extraction from individual documents. *Text Mining*, 1-20.

[28] Gladwell, M. (2006). *The tipping point: How little things can make a big difference*. Little, Brown.

[29] Cha, M., Haddadi, H., Benevenuto, F., & Gummadi, P. K. (2010). Measuring user influence in twitter: The million follower fallacy. *Icwsm*, *10*(10-17), 30.

[30] Bakshy, E., Hofman, J. M., Mason, W. A., & Watts, D. J. (2011, February). Everyone's an influencer: quantifying influence on twitter. In *Proceedings of the fourth ACM international conference on Web search and data mining* (pp. 65-74). ACM.

[31] Gaglio, S., Re, G. L., & Morana, M. (2015, June). Real-time detection of twitter social events from the user's perspective. In *Communications (ICC), 2015 IEEE International Conference on* (pp. 1207-1212). IEEE.

[32] Papadopoulos, S., Corney, D., & Aiello, L. M. (2014, April). SNOW 2014 Data Challenge: Assessing the Performance of News Topic Detection Methods in Social Media. In *SNOW-DC@ WWW* (pp. 1-8).

[33] Makkonen, J., Ahonen-Myka, H., & Salmenkivi, M. (2003, April). Topic detection and tracking with spatio-temporal evidence. In *European Conference on Information Retrieval* (pp. 251-265). Springer Berlin Heidelberg.

[34] He, Q., Chang, K., & Lim, E. P. (2007, October). Using burstiness to improve clustering of topics in news streams. In *Data Mining, 2007. ICDM 2007. Seventh IEEE International Conference on* (pp. 493-498). IEEE.

[35] Abrahams, A. S., Fan, W., Wang, G. A., Zhang, Z. J., & Jiao, J. (2015). An integrated text analytic framework for product defect discovery. *Production and Operations Management*, *24*(6), 975-990.

[36] Abrahams, A. S., Jiao, J., Wang, G. A., & Fan, W. (2012). Vehicle defect discovery from social media. *Decision Support Systems*, *54*(1), 87-97.

[37] Sakaki, T., Okazaki, M., & Matsuo, Y. (2010, April). Earthquake shakes Twitter users: real-time event detection by social sensors. In *Proceedings of the 19th international conference on World wide web* (pp. 851-860). ACM.

[38] Nichols, J., Mahmud, J., & Drews, C. (2012, February). Summarizing sporting events using twitter. In *Proceedings of the 2012 ACM international conference on Intelligent User Interfaces* (pp. 189-198). ACM.

[39] Khare, P., & Heravi, B. R. (2014, August). Towards Social Event Detection and Contextualisation for Journalists. In *In the proceedings of the AHA! Workshop on Information Discovery in Text, at the 25th International Conference on Computational Linguistics, August 2014*.

[40] Ciceri, E., Catallo, I., Martinenghi, D., & Fraternali, P. (2015). When food matters: identifying food-related events on Twitter. *Dipartimento di Elettronica, Informazione e Bioingegneria, Politecnico di Milano*.

[41] Gaglio, S., Re, G. L., & Morana, M. (2016). A framework for real-time Twitter data analysis. *Computer Communications*, *73*, 236-242.

Vita

Candidate's full name: Rizwan Ali

University attended:

Master of Computer Science

University of New Brunswick

2015-2017

Bachelor of Software Engineering

Bahria University

2008-2012