CLUSTER-DISCOVERY OF TWITTER MESSAGES FOR EVENT DETECTION AND TRENDING

by

Shakira Banu Kaleel

Bachelor of Engineering in Computer Science

Bharathidasan University, India, 1999

A thesis

presented to Ryerson University

in partial fulfillment of the

requirement for the degree of

Master of Science

in the Program of

Computer Science

Toronto, Canada, 2015

AUTHOR'S DECLARATION FOR ELECTRONIC SUBMISSION OF A THESIS

I hereby declare that I am the sole author of this thesis. This is a true copy of the thesis, including any required final revisions, as accepted by my examiners.

I authorize Ryerson University to lend this thesis to other institutions or individuals for the purpose of scholarly research

I further authorize Ryerson University to reproduce this thesis by photocopying or by other means, in total or in part, at the request of other institutions or individuals for the purpose of scholarly research.

I understand that my thesis may be made electronically available to the public.

SHAKIRA BANU KALEEL

Cluster-Discovery of Twitter Messages for Event Detection and Trending

Shakira Banu Kaleel

Master of Science in Computer Science, Ryerson University, 2015

Abstract

Social media data carries abundant hidden occurrences of real-time events in the world which raises the demand for efficient event detection and trending system. The Locality Sensitive Hashing (LSH) technique is capable of processing the large-scale big datasets. In this thesis, a novel framework is proposed for detecting and trending events from tweet clusters presence in Twitter[1] dataset that are discovered using LSH. The experimental results obtained from this research work showed that the LSH technique took only 12.99% of the running time compared to that required for K-means to find all of the tweet clusters. Key challenges include: 1) construction of dictionary using incremental TF-IDF in high-dimensional data in order to create tweet feature vector 2) leveraging LSH to find truly interesting events 3) trending the behavior of event based on time, geo-locations and cluster size and 4) speed-up the cluster-discovery process while retaining the cluster quality.

[1]http://www.twitter.com

Acknowledgement

I would like to express sincere gratitude to my supervisor Dr. Abdolreza Abhari for his excellent guidance throughout the graduate programme studies. He has been very helpful during the course of my study by providing feedback and getting me access to a high computing machine from SHARCNET. His patience, motivation and immense knowledge led me to completion of this thesis, publication in a conference and submission of a journal paper related to my research work. Working under the supervision of Dr. Abdolreza Abhari has been a great pleasure and memorable experience for me.

I would also like to convey my gratitude to the faculty members of the Department of Computer Science at Ryerson University. Attending courses under the guidance of committed professors has helped me greatly to advance my knowledge in computer science.

My sincere appreciation goes to the staff members of Computer Science department and my fellow graduate students for their continuous support and cooperation over the last two years.

A very special thanks to Dr.Sasa Petrovic, Software Engineer, Google Zurich for his support and guidance during the initial period of my research work.

Last but not least, I would like to thank my family for the support they provided me and in particular, I must acknowledge my husband and my son. Without their encouragement and assistance, I would not have finished this thesis. They were always there cheering me up and stood behind me through all my good and bad times.

Dedication

To my family

Contents

List of Tables

List of Figures

List of Abbreviations

TDT	Topic Detection and Tracking
FSD	First Story Detection
NED	New Event Detection
LSH	Locality Sensitive Hashing
IR	Information Retrieval
TF	Term Frequency
IDF	Inverse Document Frequency
TF-IDF	Term Frequency – Inverse Document Frequency
GAAC	Group Average Agglomerative Clustering
NNS	Nearest Neighbour Search
ANNS	Approximate Nearest Neighbour Search
HAC	Hierarchical Agglomerative Clustering
UMass	University of Massachusetts
E2LSH	Exact Euclidean Locality Sensitive Hashing
Pdfs	probability distribution functions
NMI	Normalized Mutual Information
REB	Ryerson Ethic Board
NLP	Natural Language Processing

Chapter 1

Introduction

Online social media provides an abundance of data on public opinions which can be used to extract the occurrences of real-time events in the world. The value of content generated on social media attracts people towards them [1]. Increase in popularity of social media websites such as Twitter, Facebook[2], LinkedIn[3], YouTube[4] and Pinterest[5] accumulates a large amount of user-contributed data on the web. The first systematic work concerning the topic and event detection in the newswire (e.g., web broadcast news) document corpus was conducted during Topic Detection and Tracking (TDT) research which includes the following five tasks [2];

- Story Segmentation: This is the process of dividing news into individual stories.
- First Story Detection (FSD): The stories are monitored to detect the events which have not been seen before, also known as New Event Detection (NED).
- Cluster Detection: This is the process of grouping all stories based on the events they discuss.
- Tracking: News streams are monitored for additional stories which are related to previously existing stories.
- Story Link Detection: The link between stories discussing the same topic of interest is identified.

[2]https://www.facebook.com [3]https://www.linkedin.com [4]http://www.youtube.com
[5]http://www.pinterest.com

Each individual task in TDT has caused a separate research problem in the field of Information Retrieval (IR). This thesis specifically deals with event detection, which is under the category of a TDT cluster detection task.

According to Fiscus and Doddington [3], the definition of topic, story and event are as follows:

Topic: "*A seminal event or activity, along with all directly related events and activities.*"

Story: "*A topically cohesive segment of news that includes two or more declarative independent clauses about a single event*."

Event: "*Something that happens at specific time and place along with all necessary conditions and unavoidable consequences.*"

For example, "Federal Election" is a topic whereas "Canada Federal Election 2015" is an event. The various discussions about the federal election are stories related to that topic. In simple terms, any identifiable occurrence happening at particular time and place is defined as an event. Social media data captures content associated with various types of events including both planned (e.g. presidential elections and concerts) and unplanned (e.g. natural disaster and epidemics) events. Twitter, a micro-blogging website, has experienced tremendous growth in the last few years and users often post tweets related to events in real time. In this thesis, a Twitter dataset is used to conduct the experiments for event detection and trending. This dataset includes an event called "2011 End Times Prediction" [4, 5] which refers to a prediction made by Harold Camping, a Christian evangelist and radio broadcaster, that Judgment Day would take place on May 21st, 2011 and it was claimed by followers of Camping that 3% of the world population would be raptured[4-6]. The event was widely spread over popular news periodicals and social media websites.

Users of social media tend to tweet using highly unstructured language with many typographical errors. A significant amount of tools and infrastructure are required to operationalize social media data due to its rapid growth and to the difficultly of processing its data by using standard relational SQL databases [7, 8]. The typical change in the volume of data in social media related to a specific topic is an indication of occurrences of a real world event.

Events in Twitter can be detected by clustering similar tweets. The text in social media is often noisy, and high volume of such data renders the challenge of finding a suitable technique for event detection to the researchers. Today, some solutions such as Topic Detection and Tracking [2, 9, 10] over newswire system and event detection in social media [11-13] have been employed in data mining related research. However, there remains a great potential in research for extracting events from tweet clusters and trending them.

1.1 Motivation

People record their opinions or statuses about events as they occur or sometimes earlier than the coverage of the event is published in mainstream media. For example, tweets related to earthquake in Japan were recorded on Twitter before the earthquake was officially announced by Japan Metrological Agency [12]. Analysis of Twitter data for presence of events related to natural disaster, epidemic, presidential election etc. could be extremely useful to promote public awareness. However, analyzing tweets to efficiently identify the events is a challenging task due to the following reasons:

- Tweets are short in length with maximum of 140 characters which limits contextual meaning related to an event
- Tweets are written in highly unstructured language and they are noisy in the sense that words may be misspelled and the text may be grammatically incorrect.
- The technique used to analyze tweets needs to meet potential issues of scalability as the volume of tweets can rapidly change at the time of events

According to the cluster detection task of TDT, we believe that clustering of tweets will help to identify events in Twitter and that trending of identified events will produce substantially more insights into events.

1.2 Research Statement

Rapid exchange of information in social media captures events almost in real-time. The early detection of events and their trends on social media websites can allow business or administrative decision-makers to get additional time for them to make informed decisions. The TDT research focuses on finding the IR method for organizing the news articles by the events that they discuss. TDT is explored only in well-written documents arriving from news streams such as newswire, radio broadcasts and TV news shows. As social media data is recorded by millions of users all over the world and the content is highly unstructured, the TDT approach is an inefficient choice for achieving the task of event detection in online social media websites.

Clustering similar text messages in social media data can help to identify the truly interesting events. However, it is important to take into a consideration on the volume of data produced by social media while finding the technique to solve the problem of event detection. The traditional hypothesis-driven clustering techniques face the challenges of recognizing events in social media due to the continuous flow of information arrival, scalability, noise and availability of limited time and resources to process the data. It leaves room in the research to locate a more suitable technique for mining events on social media websites.

Event trends provide the information about the behavioural changes of events over the time. For instance, trends can illustrate the events popularity over time in social media (e.g. popularity of newly launched Apple iPhone in North America). Event detection in social media is fairly a new area of research [11-13]. The solutions previously attempted only focused on the task of event detection. Therefore, it requires finding a modern approach which combines both event detection and trending tasks. Twitter produces massive collection of unstructured data which contains public opinions about events. This research work leverages the availability of Twitter dataset for finding events and their trends.

1.3 The Proposed Approach

The representation of text documents as vectors in a common vector space is known as vector space model [14] and is used in information retrieval for document clustering. The objective of this research is to propose a novel framework for detecting and trending events from tweet clusters. The most meaningful representative terms or lexical items from a tweet are generally considered as features of that tweet. A tweet feature vector is a vector which mathematically encodes a set of features. In this thesis, tweets are represented as vectors in a d-dimensional vector space model, where d is the number of different terms (i.e. features) that appear in the tweets collection. Twitter data is high dimensional in size due to the nature of tweets which results in a large size for the dictionary. Hence, it is a challenging task to build a dictionary for a Twitter dataset.

The Twitter dataset used in this research work is divided into multiple chunks. Each chunk goes through the process of discovering the tweet clusters using the LSH technique and the clusters' details are stored in a MySQL database. During cluster discovery, each tweet in a chunk is preprocessed and features are extracted through a sequence of processes which includes tokenization, language detection and filtration. The Incremental Term Frequency - Inverse Document Frequency (TF-IDF) term weighting scheme is a product of Term Frequency(TF) and Inverse Document Frequency(IDF) and is applied to build the dictionary of each chunk by using which the tweet feature vector is constructed. The high dimensional tweet feature vector is converted to the low dimensional K-bit signature vector. The cluster of each tweet is identified using the Locality Sensitive Hashing (LSH) method described in Algorithm-1(1-23). As a result of this process, a set of tweet clusters are discovered.

Each cluster's centroid is identified by using the average document frequency method and is used to label the cluster. The cluster attributes such as cluster label and details of each tweet {user-id, geo-coordinates, timestamp} that appear in that cluster are stored in the MySQL database. A novel part of this thesis work is to find interesting event by matching its keywords on cluster labels and trend it based on time, geo-locations and cluster size described in Algorithm-1(24-36). Also, the proposed framework has a significant advantage in that a number

of clusters need not be supplied in advance as it leverages the LSH technique to discover the tweet clusters. Thus, it is an unsupervised model.

A Twitter dataset containing an event called the "2011 End Times Prediction" [4, 5] is used for experiments. In this thesis, a novel framework is proposed to identify and trend the presence of non-trivial events in Twitter. Similar tweets are grouped using the LSH technique. The proposed approach of cluster-discovery algorithm using LSH and K-means are compared with the Group Average Agglomerative Clustering (GAAC).

1.4 Objectives

The primary objective of this study is to find truly interesting events from Twitter and trend them. The LSH technique is incorporated to find the tweet clusters from which events are identified.

The plan is to meet the objectives described below;

1. Constructing the tweet feature vector in high dimensional vector space model. The dictionary of vocabulary is constructed by using the incremental TF-IDF technique.
2. Finding suitable values for LSH parameters to obtain better quality tweet clusters. Generally, cluster quality measures how well the similar tweets are grouped and its metrics are discussed in Chapter 4 Section 4.2.1. LSH parameters such as number of buckets, signature length and threshold for cosine similarity are required to be determined before the clustering process starts.
3. Leveraging Locality Sensitive Hashing technique to form the tweet clusters
4. Storing the tweet cluster's attributes such as cluster label and details of each tweet {user-id, geo-coordinates, timestamp} that appear in the cluster into the MySQL database.
5. Analyzing the tweet clusters using their attributes for identifying and trending the events.
6. Comparing the quality of clusters discovered using LSH and K-means with GAAC of hierarchical clustering technique.
7. Trending the identified events to proliferate their level of understandings.

1.5 Research Contributions

This study focuses on proposing a framework for detecting and trending the events from the tweet clusters using the LSH technique. The major contributions of this study are outlined as follows:

- It proposed a novel framework to find an interesting event by matching its keywords on cluster labels and trending it based on time, geo-locations and cluster size.
- This research work examined and confirmed that the LSH technique requires less running time for the cluster discovery process compared to a traditional K-means clustering algorithm for the given Twitter dataset which shows the suitability of this method to analyse big data of Twitter in real-time.

1.6 Scope and Assumptions

The main focus of this thesis is to find events from tweet clusters using the LSH technique and provide information about detected events to increase the level of understanding. Henceforth, the collected information of events can either be used for social awareness or for corporate decision making depending on the type of events. For example, if an event falls under the category of natural disaster, such as US Hurricane Sandy, the information can be used for social awareness. On the other hand, if an event falls under the category of market brands, such as the launch of a new smartphone, the information can be used for growth of the products related to the brands. In this thesis, the Twitter dataset containing an event named "2011 End Times Prediction" is analyzed. The scope of the research is restricted to analyze only this specific event in the Twitter dataset.

Tweets written in the English language are considered for the clustering process. Language detection of tweets is performed based on a criterion of having a minimum of 30% of the words in a tweet in recognizable English.

The entire Twitter data corpus is divided into 'N' number of smaller chunks. Each contains maximum of n number of tweets (the value for n is set to 100 in this thesis i.e. a

maximum of 100 tweets/chunk) in order to achieve the better computational time. A series of experiments are conducted to find tweet clusters from the first tweet chunk by varying the values for LSH parameters. As a result, LSH parameters are set based on the value which produces the better quality of clusters and are retained for all other chunks (i.e. N-1 chunks), assuming that it will yield a similar result for all other chunks. We did not find the suitable values for LSH parameters in entire data corpus as the LSH technique is applied on incremental fashion and it makes more sense for us to find the values based on smaller chunk.

1.7 Organization of Chapters

The thesis is organized as follows:

- Chapter 2 presents background information and a series of literature related to this research work. The first section of this chapter provides background information instrumental in understanding the rest of the thesis. It starts with the introduction of Twitter, an online social network. The following section provides the background information of the LSH technique as it is the main technique used in the proposed framework for finding the tweet clusters, followed by the prefix tree data structure. The final section of this chapter reviews a series of significant published research works in the area of event detection and trending.

- Chapter 3 presents the methodology used in the proposed framework which includes the tweet feature vector construction and the cluster discovery process using LSH technique followed by the algorithm for event detection and trending.

- Chapter 4 presents the implementation and results of this research work. The first section explains the implementation of the proposed framework. The following section describes the performance metrics used in this research work for cluster evaluation and the results of the experiment conducted for event detection and trending in Twitter. It explores the different values of LSH parameters and their resultant effects on the quality of clusters. It provides a comparison of the cluster-discovery process using LSH with other clustering

techniques. After analyzing clusters related to a specific event, the event trends based on time, geo-locations and tag-clouds are presented at the end of this chapter.

- Chapter 5 concludes the thesis work by summarizing the contributions and outlining the proposals for potential future research.

Chapter 2

Background & Related Works

In this chapter, some preliminary information is discussed regarding:

- Twitter, a social networking platform,
- The approximate nearest neighbour search approach,
- The date structure - prefix tree and
- Two well know clustering techniques.

The background information presented here is necessary for understanding the information presented in the remainder of the thesis. The later part of this chapter reviews a number of related works in the area of event detection and trending.

2.1 Background

2.1.1 Online Social Network – Twitter

Social network is defined by boyd and Ellison [15] as "*web-based services that allow individuals to (1) construct a public or semi-public profile within a bounded system, (2) articulate a list of other users with whom they share a connection, and (3) view and traverse their list of connections and those made by others within the system*". Although there are many definitions of a *social network* that emphasize different aspects of social components such as social graphs, content and users communities, the common definition is a web-based platform

where users can generate and share content, as well as express their views or opinions. In recent years, social networks have become increasingly more popular partly due to availability and affordability of internet access and devices such as personal computers, laptops, smartphones and tablets. There are hundreds of social networking sites available today, focusing on different niches and aspects of social life. For example, Facebook, Twitter, YouTube, LinkedIn and Pinterest are some of the leading and best-known social networks. Social networks allow users to post content and express their view on existing conversations or social issues in real-time from any location.

In this thesis, a dataset from Twitter is analyzed to find the presence of events and later, their trends are presented. Twitter, a micro-blogging website, has experienced tremendous growth in the last few years and has over 200 million active users who tweet over 400 million short-messages every day as of March 2013[16]. Twitter users post short form of message, most commonly known as, "tweet" with a maximum length of 140 characters in Twitter. Tweet can contain text, emoticons (a combination of keyboard characters intended to represent a facial expression. e.g. :-) represents a smiling face), external links and hashtags. When a new account is created in Twitter, the account type is set to public by default, meaning that the user's tweets are visible on the web to anyone. However, users are able to protect the privacy of their Twitter account and make their tweets available only to a designated list of users by changing the profile type from public to private. The dataset used in this thesis contains the tweets extracted from publically available Twitter accounts.

A Twitter data model consists of two main entities: i) users and ii) tweets. A user entity has a set of attributes such as name, screen-name, unique id assigned by the Twitter internal system, location, short description, tweet counts, friends and other profile settings (picture, background image, colour etc.). The relationship between users is an interesting part of Twitter. Unlike other social network, such as Facebook or LinkedIn, users in Twitter can follow others, becoming "followers" or are followed by others, known as "followees" without mutual consent. By becoming a follower of a Twitter account, a user receives the tweets from the account that the user has chosen to follow.

The tweet entity contains an actual message to be exchanged among users. In addition, the tweet includes the creation time, web source, user id of the user who posted the tweet, reply tweets and geo-location information. Tweets can automatically be annotated with the current location of the user if the user has enabled the geo-location option in Twitter. Republishing another user's tweet in Twitter is referred as a ReTweet and is commonly used to further spread the original message to a greater audience through Twitter. It is represented by starting the tweet with the keyword "RT". Thus, the ReTweet count is considered as an important influence factor of the tweet that expresses the user's intention of republishing the message. Using certain specific symbols in a tweet carries special meaning in Twitter. The use of the @ symbol followed by a Twitter user at the beginning of the tweet is known as reply. If the @ symbol is used within the body of the tweet, it is referred to as a mention. A reply is used to indicate that the tweet is a response to another user, such as an answer to an inquiry or a comment in response to a posted tweet. Reply tweets appear on the timeline of the sender, the recipient and users who are followers of both the sender and the recipient. Mentions can be used to endorse the existing members in an ongoing community tweet conversation. For example, one can use the mention to give their support to the product or person or event. Another useful symbol in Twitter is a hashtag which is denoted by '#'. A word prefixed with the # symbol is called a hashtag and represents a trendy topic. These hashtags provide ability to the user to search the tweets related to the hashtags and allows them to take part in conversation regarding the topic. Hashtags are more common in Twitter.

Any given tweet can include retweets, replies, mentions or hashtags. The use of these aspects of a Tweet are illustrated in the sample tweets included below, which are taken from the experimental Twitter dataset used in this thesis.

Sample Tweets:

Tweet-1:

> RT @andersoncooper: The ridiculist RT @heighmichael: @andersoncooper So how will you be celebrating the end of the world?

Tweet-2:

@carolhanna i'm doing Economics. I don't finish till thursday #endoftheworld

Tweet-3:

Saturday Rockpile: It's Not the End of the World http://FANpeeps.com/-fVeb (Via @ROCKIESpeeps) #mlb #rockies

The RT at the beginning of Tweet-1 shows that the Twitter user is republishing the tweet of another Twitter user, in this case @andersoncooper. Tweet-2 is a reply tweet to the user @carolhanna as demonstrated by its presence at the beginning of the tweet. On the other hand, the use of @ROCKIESpeeps in Tweet-3 is a mention as it is contained within the body of the tweet. The hashtags #endoftheworld in Tweet-2 and #mlb and #rockies in Tweet-3 denote different topics. A URL of any length posted in the tweet is shortened due to the limitation of tweet length. The web address http://FANpeeps.com/-fVeb used in Tweet-3 is a short and unique user generated URL which can be used while tweeting. The human readable status object of the tweet can be retrieved from Twitter using the Twitter API. The message is in the format of JSON (JavaScript Object Notation) which is a text-based open standard data-interchange format. An extended documentation of each attribute in JSON can be found in [17].

2.1.2 Approximate Nearest Neighbour Search Approach

In this section, the problem of Nearest Neighbour Search (NNS) is explained as well as its relaxed version, known as Approximate Nearest Neighbour Search (ANNS). An algorithm under the category of ANNS is applied in this research work to deal with the problem of event detection as a clustering problem where each cluster containing a set of documents represent the event.

Definition of NNS-problem:

For given a set of N-points $P = \{P_1, P_2, P_3, \ldots, P_N\}$ in a matrix S and a query point Q, find the point in P closest to the given query point $Q \in S$ [18].

In case of *d*-dimensional document vector space model R^d, the points are the documents and a query point is a searching point. Figure 2.1.a represents the NNS scenario. The nearest point to the query point can be found by simply computing the distance between all points in *P* to the query point *Q* and finding the one $P_i \in P$ which is closest to *Q*. The NNS solution for finding the nearest point to the given query point is computationally expensive in a high dimensional vector space model.

The NNS approach suffers heavily when the dimension of space increase and this challenge is known as the "*Curse of Dimensionality"* [19]. However, researchers have developed a solution by using a relaxed version of the NNS problem, known as Approximate Nearest Neighbour Search (ANNS) which is defined as follows:

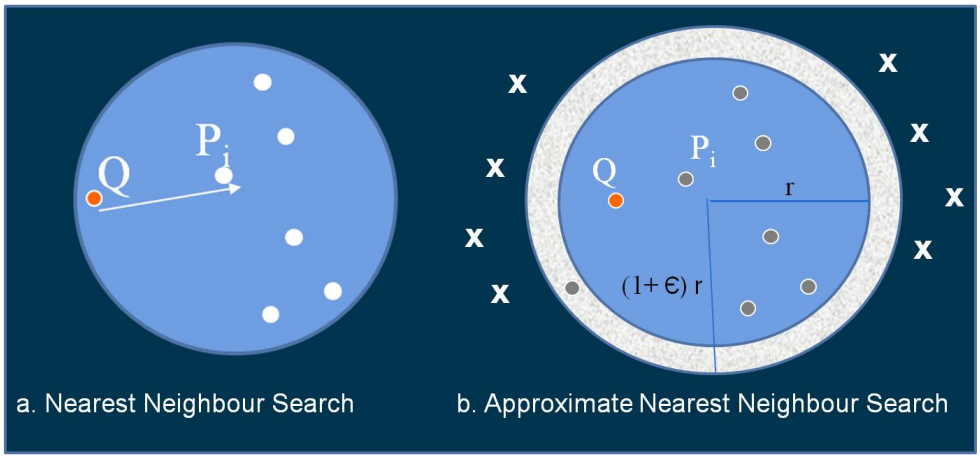

Figure 2.1: Nearest and approximate neighbour search representation

Definition of ANNS Problem:

For given a set of N-points $P = \{P_1, P_2, P_3, \ldots, P_N\}$ in a matrix S, query point Q and the value of r (1+ Є), find an approximate nearest neighbour point in P closest to the given query point Q such that $\forall P' \in P$, $d(P_i, Q) \leq (1 + Є)d(P', Q)$ where d(P',Q) is the distance between P' and the query point Q [18].

Figure 2.1.b represents the ANNS scenario. To clarify further, a point P_i is an approximate nearest neighbour of query point *Q* if its distance is within a constant factor *(1+ Є)*

of the true nearest distance *r*. The points outside the larger sphere with the radius of *r (1+ Є)* in Figure 2.1.b are not searched for finding the approximate nearest neighbour of query point *Q*.

One of the approaches to address the ANNS problem was introduced by Indyk and Motwani [18] and is known as Locality Sensitive Hashing (LSH). In this research work, the LSH technique is employed to find tweet clusters from which events are detected and trended. As this thesis work is heavily focused on LSH for cluster-discovery process, the following sub-section is dedicated to discussing LSH in greater detail.

2.1.2.1 Locality Sensitive Hashing

The hashing function for finding similar documents was first introduced by Broder et al. [20]. Indyk and Motwani used the LSH technique, one of the ANNS approaches, to detect unseen stories in a stream of news stories [18]. The LSH technique can be applied to information retrieval, pattern recognition, dynamic closest pairs and fast clustering [18]. The key idea of LSH is to apply the hash functions in such a way that the probability of collision is much higher for similar objects than for the objects that are dissimilar [18] i.e. the objects close to each other will most likely fall into the same bucket. Intuitively, a hash function is locality sensitive if two points that are close under the similarity distance measure *D* are more likely to collide. Formally the LSH is defined as follows [18]:

LSH Definition:

Let us consider any pair of points (p,q) and a ball of radius *r* for distance measure *D* as $B(q,r) = \{p: D(q,r) \geq r\}$ and $\mathcal{H}$ be a family of hash functions which maps the given space S to U for any randomly selected hash function h i.e. $\mathcal{H} = \{h: S \rightarrow U\}$. The family of hash functions $\mathcal{H}$ is called locality sensitive for the parameters (r_1, r_2, p_1, p_2) if for any pair of points $p,q \in S$ satisfies the following properties:

- if $p \in B(q, r_1)$ then $Pr_H[h(q) = h(p)] \geq p_1$,
- if $p \notin B(q, r_2)$ then $Pr_H[h(q) = h(p)] \leq p_2$.

where p1 > p2 and r1 < r2. The $Pr_H[h(q) = h(p)]$ is the probability that the value of the hash function on two points *p* and *q* returning the same value i.e. h(p) and h(q) yields the same result.

In LSH, a set of points in S will be preprocessed and stored into *L* number of buckets. For each point p ∈ S is hashed using *k* independent, uniform, randomly selected hash functions $g_i(p)=(h_1(p),h_2(p),\ldots,h_k(p))$ and stored in bucket i ∈ L. For any query point *q*, after getting the hash value of *q*, all buckets are searched to retrieve the points similar to *q* and perform the similarity distance measure among all retrieve collection of points and select the one close to *q*. Figure 2.2 shows the example for concatenating three hash functions $g(p)=(h_1(p),h_2(p),h_3(p))$ in two-dimensional vector space divided by three randomly drawn hyperplanes. The buckets are defined by the subspaces created by the random planes.

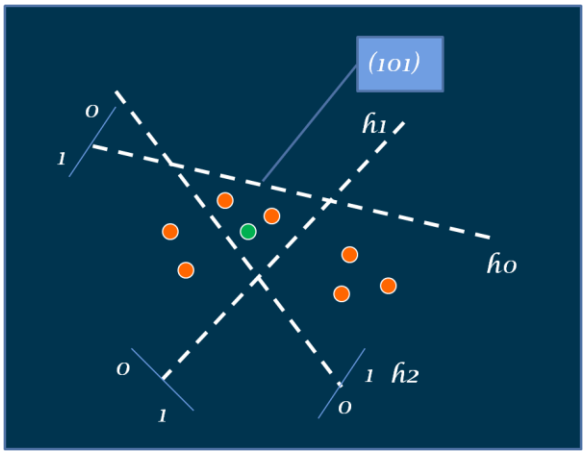

Figure 2.2: LSH with random projection [21]

The basic understanding of the LSH algorithm is visually represented in Figure 2.3 [21]. Let us consider *N* is a set of data points in *S* vector space model. The data points are preprocessed using the randomly chosen independent hash function $hr_1,r_2,..,r_k$ and stored in a hash table. The hash value of the data points acts as the index of the hash table. For any query point *Q*, the hash function $hr_1,r_2,..,r_k$ is applied to determine the index of the hash table from which the set of closely related data points is fetched. From the search result, the most similar *n* neighbours can be fetched and the results are called as the nearest neighbour of the query data point *Q*.

Charikar proposed LSH functions using random hyperplanes to map high dimensional vectors to low dimensional space while preserving similarity between vectors in original space [22]. In Charikar's LSH scheme, the similarity measure between two document vectors is calculated by computing the cosine angle distance between them [22]. The value of cosine similarity becomes one (i.e. cos(0) = 1) when two vectors are parallel and zero (i.e. cos(90) = 0) when orthogonal. Therefore, for a given pair of vectors $\vec{u}$ and $\vec{v}$, the cosine similarity of them is defined as the dot product of feature vectors normalized by their length [23].

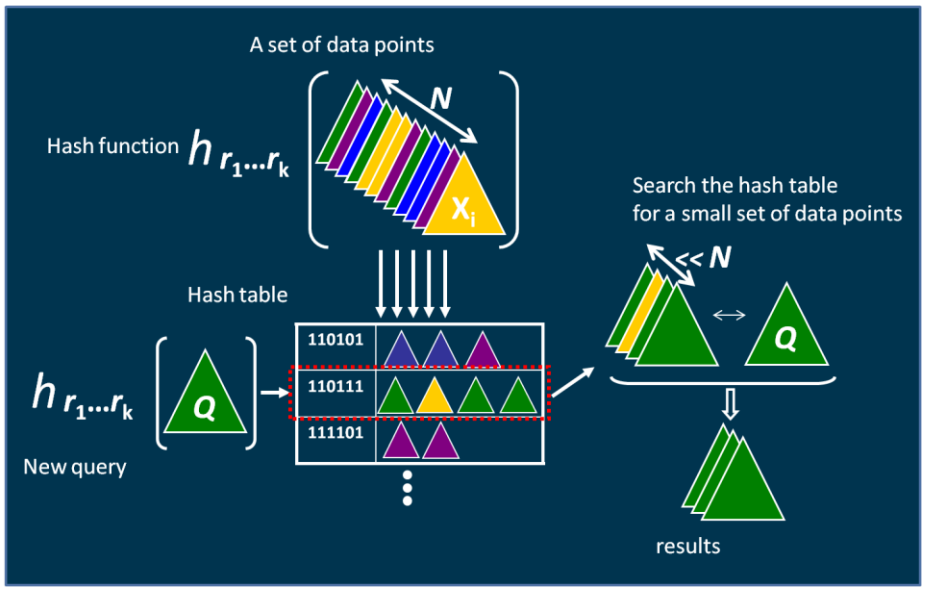

Figure 2.3: LSH algorithm [21]

Generally, for a given *n* points to cluster with *k* features, the traditional similarity measure takes $O(n^2k)$ times. Ravichandran et al. [24] made use of the LSH algorithm defined in [22] for searching similar nouns from a large web corpus (70 million pages) and reduced the time complexity to O(nk). In short, the LSH algorithm has been used to find all pairs of vectors whose cosine similarity is greater than a given threshold for *n* vectors in *k* dimensional space. This thesis work uses LSH technique to perform the online clustering of similar tweets for the purpose of event detection and trending.

In recent research work, the prefix tree based LSH algorithm has been used for detecting a newly formed community of users in real-time social media [25]. Kamath et al. [25] applied Charikar's approach to compute the K-bit signature for a tweet feature vector and the similarity between two tweets is computed using the cosine angle distance. Similarly, this thesis work

attempts to use Charikar's approach to compute the K-bit signature for a tweet feature vector which is further used as input for the prefix tree based LSH approach proposed by Kamath et al. [25] to discover the tweet clusters from which the event is detected and it is trended. The details of the proposed algorithm are explained in Chapter-3. Background information on the prefix tree data structure which is used in LSH approach to replace the hash table is discussed in the next section.

2.1.3 Data Structure – Prefix Tree

A prefix tree (or trie) is a tree-like data structure to store strings for fast pattern matching. Trie is commonly pronounced as "tree" or "trie" from the root word "retrieval" [26]. A trie data structure associates the key with a value and was created by Edward Fredin [27]. Tries can be used to perform the query operations for pattern matching and prefix matching. Therefore, it is also known as *prefix tree* which can be used in information retrieval application where it requires storing key and value pair information and the same can be retrieved quickly. For example, for prefix matching, the given string 'A' in prefix tree 'P', we can retrieve all the strings in 'P' prefixed with 'A'. Let's consider a set of strings S = {'prefix', 'preamble', 'preach', 'preface'}. The total number of characters in S is 27 (6+8+6+7). While using the prefix tree, it would be pre + {'fix', 'amble', 'each', 'face'} and the total number of characters is 16 saving a total of 11 characters. Figure 2.4 shows the prefix tree for S. The main advantages of prefix tree include short access time and ease of adding or removing items with variant lengths [26]. Hence, we leverage the prefix tree data structure in this thesis to find the nearest neighbour of the given tweet. In the next section, two traditional well-known clustering algorithms which are used in this research work for comparison purpose are discussed in greater detail.

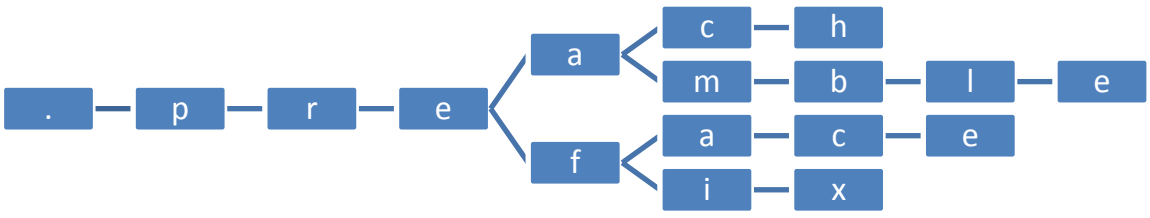

Figure 2.4: Prefix tree for the set of strings 'S'

2.1.4 Traditional Clustering Techniques

The two traditional clustering techniques that are well-documented in literature, K-means and Group Average Agglomerative Clustering, are discussed in this section. The former is classified under flat clustering and the latter falls under the hierarchical clustering classification [28]. Since the clustering task is the primary focus of this research work, the tweet clusters discovered using LSH and K-means are compared against GAAC.

2.1.4.1 K-means

K-means [28] is one of the most widely used clustering algorithms in data mining. It assumes that the number of clusters *k* is known in advance. The algorithm is outlined below for *n* number of points in euclidean space where euclidean distance is used to measure the similarity distance between any two points.

Step 1: Randomly select the initial *k* points, also known as "seeds" to represent the clusters and assume that each point acts as a centroid of the cluster to which it belongs to. Figure 2.5 a shows the selection of seeds for *k*=2.

Step 2: Perform the following steps for all other remaining *(n-k)* points in Euclidean space:

Step 2.1: Assign each point to the cluster with the nearest centroid.

Step 2-2: Recalculate the cluster centroid or mean $\hat{\mu}$ of the documents in the cluster $C_i$using [28]:

$$\hat{\mu}(C_i) = \frac{1}{|C_i|} \sum_{\vec{x} \in C_i} \vec{x}$$

where $|C_i|$ represents the number of data points present in i^{th} cluster

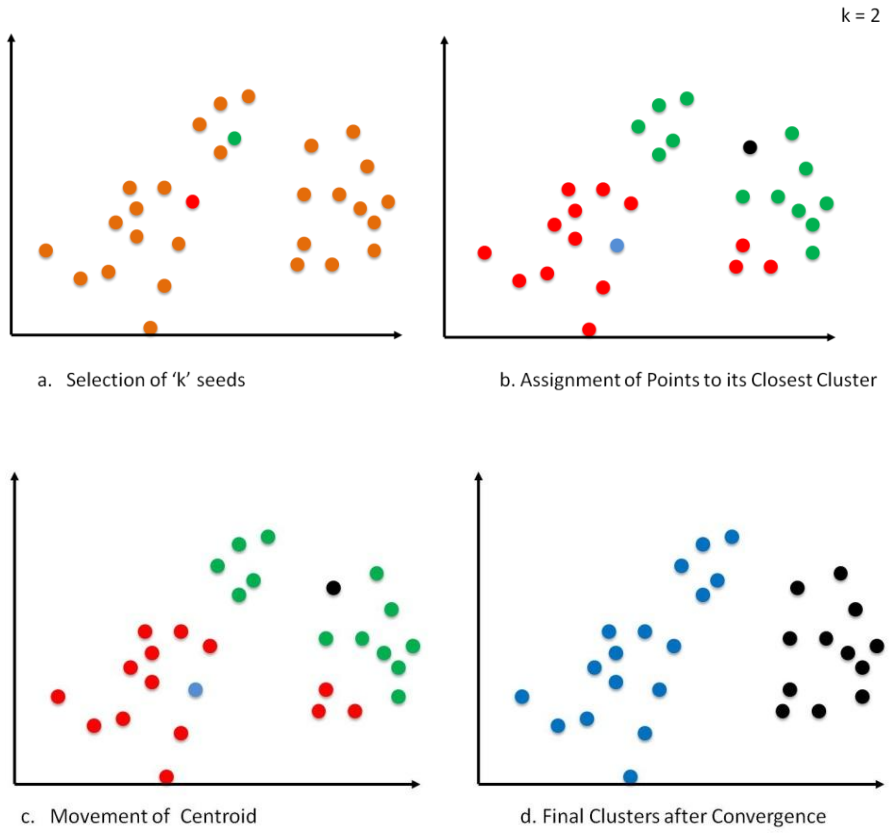

Figure 2.5: Steps involved in K-means algorithm

Figure 2.5b shows the assignment of the data points to its nearest centroid and identifying the two new centroids.

Step 3: For each point in the clusters, compute its Euclidian distance from the centroid of each of the clusters. Switch the point to its closest cluster and update that cluster's centroid. It is shown in Figure 2.5c for k=2.

Step 4: Repeat Step 3 until convergence is achieved i.e. until there are no new assignments to the cluster or a given number of iterations is exhausted as shown in Figure 2.5d for k=2.

2.1.4.2 Group Average Agglomerative Clustering

Hierarchical clustering produces a set of nested clusters organized in the form of a hierarchical tree, which is more informative than flat clustering. Hierarchical clustering algorithms are either bottom-up (agglomerative) or top-down (divisive). Bottom-up hierarchical clustering is also known as Hierarchical Agglomerative Clustering (HAC) where each document is considered as a singleton cluster and each pair of clusters successively merge until all are merged into a single cluster containing all documents. An HAC produces a set of nested clusters organized as a hierarchical tree which can be visualized as a dendrogram where the merge between two document clusters is represented by a horizontal line. The nested clusters and the dendrogram representation of a sample HAC are shown in Figure 2.6a and Figure 2.6b respectively. The x-coordinate of the dendrogram represents the document id and the y-coordinate of horizontal line represents the similarity value of two clusters that were merged in Figure 2.6b. For example, the similarity of the cluster consisting of document 2 and 6 is 0.1. It is not required to specify the number of clusters in hierarchical clustering and it is deterministic in IR. There are four different types of HAC algorithms which differ in the similarity measures they apply; single-link, complete-link, group-average and centroid similarity. The merge criteria of these four HAC variants are listed below [28];

a. *Single-link clustering:* distance between closest neighbours
b. *Complete-link clustering:* distance between farthest neighbours
c. *Group-average clustering:* average distance between all pairs of neighbours
d. *Centroid Clustering:* distance between centroids is the most common measure

Group Average Agglomerative Clustering (GAAC) is used in this thesis to evaluate the clusters discovered using LSH and K-means. GAAC merge criteria considers all similarities between documents while computing the average distance, thus avoiding the pitfalls of single-link and complete-link criteria where the merge is based on the similarity of a single pair of documents. The group average distance between the clusters *Ci* and *Cj* is defined below:

$$Dist_{avg}(C_i, C_j) = \frac{1}{|C_i| \times |C_j|} \sum_{a \in C_i, b \in C_j} d(a,b)$$

where $Dist_{avg}$ is the average distance between any point in C_i and any point in C_j. $d(a,b)$ is a distance between any point a from C_i and any point b from C_j.

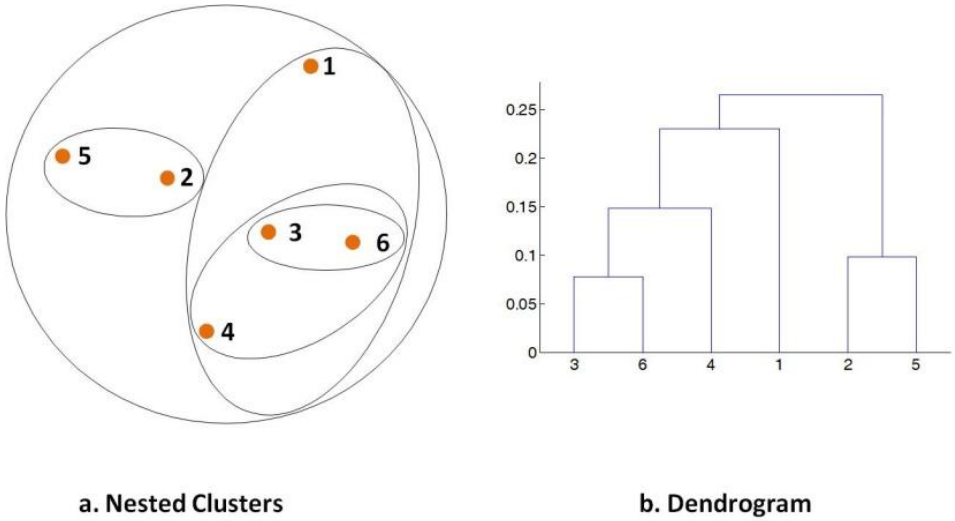

Figure 2.6: Sample of HAC clusters

Most common hierarchical clustering algorithms have a complexity of quadratic in the number of documents compared to the flat clustering algorithms. For a given *n* number of documents, the HAC algorithm takes O(n^2) time. The drawbacks of the hierarchical clustering algorithms are: i) the distance matrix is used for deciding which cluster to merge ii) the time complexity is atleast a quadratic in the number of documents. and iii) they are not usable for large datasets.

2.2 Related Work

In this section, background works related to event detection in the newswire and the different approaches to achieve event detection and trending in social networks are reviewed. One of the ongoing latest topics of research is finding a suitable approach for analyzing the large volume of data. As the proposed framework in this thesis is heavily based on LSH, the last subsection is dedicated to discussion of previous work done related to use of the LSH technique in high dimensional data. We then explain how our work is distinguished from all previous works on the topic.

2.2.1 Event Detection in Newswire

Topic Detection and Tracking (TDT) is a body of research that began in 1997 which includes the five different tasks: i) Story Segmentation ii) First Story Detection iii) Cluster Detection iv) Tracking and v) Story Link Detection. These tasks were previously defined in Chapter 1. These tasks have been combined together to detect the events occurring around the globe. Each component task in the TDT has presented a separate research problem in the field of information retrieval. Allan et al. [9] investigated the detection, first story detection and story link detection TDT tasks used by a system at University of Massachusetts, colloquially referred as the UMass System. In their approach, the stories are represented on vector space mode. A term weight is calculated using a TF-IDF scheme which measures how important a term is to the document in the collection. The TF-IDF is a product of term frequency and inverse document frequency. Term frequency measures the number of times a term appear in a document. Inverse document frequency is computed as the logarithm of the number of the documents in the document collection divided by the number of documents where the specific term appears. All terms are considered equally important while computing the TF value. However, certain terms may appear more times but have little importance (e.g. term like 'a', 'may' or 'that'). Thus, the IDF is used to weigh down the frequent terms and scale up the rare ones. Allan et al. [9] used the k-nearest neighbour (k-NN) approach where the incoming stories are directly compared with all other previously seen stories from which 'k' similar neighbours are found. A story is declared as new when its similarity to its neighbours is below a specified threshold. The similarity is measured using cosine distance which is an inner product of two vectors. The TDT initiatives provided a general outline and fundamentals regarding event detection in earlier days. However, it is not a suitable solution for social media data as TDT is designed for topic detection over newswire. The problem of NNS is previously explained in Chapter 1. The text in social media is often noisy and the high volume of data presents challenges to researchers attempting to find a technique suitable for event detection in social media.

Yang et al. [10] added two tasks in IR for event detection: i) retrospective detection and ii) online detection. The former task detects events from accumulated data and the later finds events from news feeds in real-time. An automatic detection of events from a temporally-ordered

stream of news stories, a CNN dataset in this case, was investigated using two clustering methods: an agglomerative algorithm based on GAAC and a single-pass algorithm (incremental clustering) [10]. Yang et al. [10] adapted hierarchical bottom-up clustering technique called Group Average Agglomerative Clustering (GAAC) to group the documents in a collection. GAAC is a bottom-up hierarchical clustering, which starts clustering with each document as singleton cluster and agglomerates pairs of clusters based on their average similarity until all clusters are merged into a single cluster containing all of the documents [28].GAAC is used for batch processing and hence is used for retrospective detection. Incremental clustering is suitable for both retrospective detection and online detection. Though the GAAC algorithm appears to be a natural solution for clustering, it has time and space complexity quadratic to the number of input documents as described in [29]. Brants et al. [30] adapted an incremental TF-IDF model to perform the FSD task in TDT datasets. In their approach, document frequency changes over time while calculating TF-IDF for a term. In this thesis, incremental TF-IDF is used for term weight as explained in Chapter 3 Section 3.1.1.2.

2.2.2 Event Detection in Social Media

Online social media has attracted researchers due to its openness and the availability of data (e.g. Twitter access through TwitterAPI [17] and Facebook access through FacebookAPI [31]). Recent research works have shown that social media data captures vital information about events in real-time. For example, Bollen et al.[32]'s experimental results demonstrated that stock market values can be predicted using public moods extracted from Twitter feeds. Choi and Varian showed how search queries submitted on the Google search engine can be used to predict the near-term values of economic indicators. For example, early prediction on automobile sales for a current month was predicted from *Google Trend*[6] [33]. Additionally, Preis, Moat and Stanley used Google Trends data to predict the early warning signs of stock market moves [34].

Twitter, a micro-blogging online social media is considered as a platform in this research for event detection and trending. Twitter generates large amounts of data, which is extremely valuable for mining events and trending them. The ease of publishing messages in Twitter makes

[6]http://www.google.ca/trends/

it a popular data source for real-time event detection. Challenges in detecting an event in Twitter include:

1. A system is needed to process the enormous volume of tweets that are submitted at a rapid rate
2. A powerful keyword extraction tool is required to represent the events, as tweets are very brief due to their 140 character length limitation and
3. An efficient technique is needed to handle unstructured, noisy and grammatically incorrect text messages.

A large body of work pertaining to event detection has been carried out in the last few years. Parikh et al. [35] developed an automatically scalable system for event detection from tweets, called the ET system. In their approach, a hierarchical clustering technique is used to identify the events. The ET system was able to detect event accurately without any human expertise or knowledge from other sources. In the ET system, the following steps are used to detect the event [35];

1. The time span of the tweets from a Twitter dataset is divided into multiple time blocks of fixed-length.
2. Keywords that show a sudden increase in their frequencies across consecutive time blocks are considered as event representative keywords. The ET system uses bigrams (i.e. two word phrases such as Plane Crash) as candidate keywords which reduces the computations caused by bursty unigrams (i.e. increase in number of unigrams) and avoids the bottleneck of using n-grams.
3. Hierarchical clustering technique is applied to cluster the keywords which represent the events. The algorithm starts to merge the keywords with highest similarity score. The advantage of using hierarchical clustering is that it does not require specifying the number of clusters beforehand.

Sriram et al. [36] proposed an approach to classify the tweets into generic categories such as news, events, opinions, deals and private messages based on the author information and tweet features. Tweet features used in the classification process follow the categories' details. For example, the token "deal" and special characters such as $ and % in the tweet features would classify a tweet as belonging to the deals category [36].

Sankaranarayanan et al. [37] built a system to discover news-related tweets in Twitter. In their approach, tweets from 2000 handpicked Twitter users, known as seeders, are taken as input to their system in addition to the regular tweets coming from other Twitter users. The incoming tweets are classified as either news or junk based on a prebuilt training Twitter data set. Thus, their approach requires a proper training set and a collection of seeders in order to construct a more generalized system. Also, it would be very expensive to construct a training set for a volatile data stream.

Ozdikis et al. [38] presented a method for event detection in Twitter based on agglomerative clustering of hashtags. Ahmed et al. [39] built a unified framework which combines both clustering and topic model approaches to perform clustering, categorization and analysis of news articles. The topic model is a simple approach to discover the topics that occur in a collection of documents. Clustering is used to aggregate related news into the same story. The topic model is used to tie the new stories into events from the past (e.g. news articles related to Union of European Football Association(UEFA) soccer game were clustered as the same story and tied to the 'sports' topics). An agglomerative clustering has the time complexity of quadratic in the number of documents. Also, the time complexity is not reported in either of the approaches as it is an essential factor for a high volume of Twitter data stream. Therefore, both approaches of Ozdikis et al. [38] and Ahmed et al. [39] do not explicitly work on large volume of social mediadata.

Query expansion is the process of expanding the search terms to improve the retrieval performance in IR. Packer et al. [11] proposed an approach which associates tweets to the given event using query expansion with the following three steps:

1. The search terms related to events are identified and are expanded by adding entities related to them using semantic web knowledge base
2. Search for tweets containing the search terms identified in Step-1

Events are identified using the frequency of the tweets over times which are grouped by each entity.

Packer et al. [11] investigated a small subset of tweets within the context of one of Europe's largest musical events, known as "Rock am Ring". They used the musical bands listed on the "Rock am Ring" event and reported the tweets related to those bands. The search terms of the event are expanded using YAGO2 [40]. YAGO2 is a semantic knowledge base containing millions of entities (like persons, organizations, cities etc.,) which are derived from Wikipedia[7], WordNet[8] and GeoNames[9] and can be used to reduce ambiguity as it contains sets of alternative names for entities [41]. In their approach, the tweets are collected using Topsy[10], a web service that searches content published on Twitter. The correlation between the time close to the middle of the actual event and the publication time of tweets was investigated using Pearson's correlation coefficient. It is reported that their system is able to detect the event after the chronological middle of the event. Packer et al. [11] conducted the experiment with a small subset of data and their methodology was not tested on a large collection of tweets. Therefore, their methodology cannot be adopted for real-time continuous social streams.

2.2.3 Monitoring Trends in Social Networks

Monitoring events on social network gives insights into trends and how trending changes over time. The temporal pattern represents how the data changes over the passage of time. Some of the research that has already been performed has shown the importance of finding the temporal pattern in the social media data stream for society's information and awareness [42, 43]. For example, Yardi et al. [42] examined the temporal pattern of tweets about local events such as a shooting that took place in Kansas and a building collapse in Georgia.

[7]http://www.wikipedia.com [8]http://wordnet.princeton.edu/ [9]http://www.geonames.org
[10]http://www.topsy.com

Kwak et al. [44] analyzed the tweets of top trending topics over a nearly four month period and reported the behaviour of trends in terms of users' participation, the active period of the trend and the number of tweets relevant to the topic.

A system for trend monitoring based on the characteristics of pubic posts on Facebook was proposed in [45]. The authors generated a list of terms from public posts after filtering out the stopwords (i.e. the words which carry the least information. For example, words such as 'a', 'the' or 'may') and URLs. In their approach, the following three types of topics were detected through clustering of collected terms from Facebook posts and were analyzed.

1) Disruptive events: the events that cause reaction of Facebook users on a global level such as the death of Amy Winehouse
2) Popular topics: the celebrities or brands that remain popular for an extended period of time such as Harry Potter
3) Daily routines: the common phrases that appear regularly, such as "Happy Birthday" wishes

Twitter tracks the most popular phrases, words and hashtags and shows a list of the top ten items of the moment under the "trends" heading on a left side bar on the user's home page and allows users to customize them. Figure 2.7 shows a screenshot of the Twitter trends list. However, Twitter does not provide additional details such as past trending topics, categories of trend items, user participation, visualizations of tweet geo-locations or the trend's active period.

Twitter is an information spreading social media website and we used its dataset in our system for event detection and trending. To the best of our knowledge, at the moment there is no existing research focusing on event detection followed by trending. In this thesis, we tried to build a less resource intensive system which detects events through cluster-discovery of tweets using a technique suitable for a high speed data stream and then trending the detected events.

Figure 2.7: Twitter trends of the moment

2.2.4 Locality Sensitive Hashing

Allan et al. [9] applied the nearest neighbour approach (k-NN) for clustering news articles that discussed the same event. In their approach, a newly arriving news article is assigned to an existing cluster if it has been previously seen (i.e. similar to a previously analysed news article). Otherwise, a new cluster is created for that news article. The main problem of the nearest neighbour approach is the representation of datasets in high d-dimensional vector space. Social media data is high dimensional in nature and needs to address the dimensionality reduction. For example, the number of distinct features in tweets is huge in a growing Twitter dataset. The speed over linear search in the k-nearest neighbour approach has been significantly improved by an approximate nearest neighbour search [18]. Though there are several indexing data structures proposed for the approximate nearest neighbour search, they do not scale well with high dimension data [46]. Locality sensitive hashing (LSH) is a technique for approximate nearest neighbour search that was first articulated in [2, 9]. Since then, LSH has become a state-of-the-art technique for similarity searches in high dimensional data [13, 18, 47-49]. LSH is capable of processing very large datasets in high dimensional vector space, which provides a high probability of returning a correct or close match to the query point. In LSH, there is always a trade-off between the size of the hash table and the amount of linear search required for finding

the nearest neighbour of the given query point. However, finding the actual nearest neighbour with high probability can be achieved by increasing the number of projections.

Exact Euclidean Locality Sensitive Hashing (E2LSH) was originally proposed by Datar et al. [50], and is another variation of LSH realized in Euclidean space. The idea behind E2LSH is to hash the points by using several p-stable distribution (where $p \geq 0$) based locality sensitive hash functions to achieve two points close enough have a high probability for collision into the same bucket. For instance, p-stable random variables can be generated from two independent variables distributed uniformly over [0, 1]

Yongwei et al. [51] experimented with image retrieval based on E2LSH and query expansion. In their approach, E2LSH was compared with the K-means algorithm and it was confirmed that E2LSH boosts the retrieval accuracy compared to the K-means algorithm at the extra cost of response time due to the query expansion [51].

LSH is comprised of two steps: 1) indexing the data and 2) searching for neighbours of a given query document. Finding the nearest neighbours for the query document is computationally expensive when a given set of data is high-dimensional. Slaney et al. [52] contributed an algorithm to calculate the optimum parameters for nearest-neighbour search using LSH. Their work considers LSH based on p-stable distributions as introduced by Datar et al. [50]. Slaney et al. [52]'s optimization algorithm requires knowledge of five input parameters [52]: 1) the number of points in the data set, 2) an acceptable error probability that misses the exact nearest neighbour, 3) the probability distribution functions (pdfs) for the distance between the query point and its nearest neighbor, 4) the pdfs for the distance between the query point and any random member of the data set and 5) the allowable hamming radius for multiprobe. Given this, their algorithm returns the optimal parameters for an LSH implementation: the quantization width, the number of projections, and the number of tables. Slaney et al. [52] experimented with the optimization parameters for LSH on a wide variety of databases consisting of images to achieve the expected results. Their work could be a good start for finding nearest neighbours in large datasets using LSH. The author concluded that it is difficult to find the best values for LSH parameters especially in large datasets [52]. Similarly, in this work we found it is difficult to determine the best values for the LSH parameters. We explored a number of solutions to find an appropriate set of hashing functions that are mathematically well-defined and fit the constraint of

not colliding dissimilar items into the same bucket. Various experiments were conducted in this research work to compute the best values for the LSH parameters and the result is recorded in Chapter 4.

While different types of approaches have been carried out by researchers towards event detection in social media, applying the LSH technique for event detection in social media [13, 53] is fairly a recent development. Petrovic et al. [13] proposed a method to speedup first story detection task explained in Chapter 1, which could run on social streams such as incoming tweets. The authors attempted to run their system on constant time and space. The method that they applied is based on locality sensitive hashing and it was compared with Allan et al. [9]'s UMass system. In their proposed method, they intersect the vector space into random hyperplanes and define buckets for the spaces which they split up.

We explored in [54], a theoretical approach of using LSH for event detection and trending in multiple social networking sites. In this research work, the LSH technique is used for discovery of tweet clusters to identify events on Twitter. Incremental TF-IDF is used to represent the tweet in a document vector space model. The occurrence of the event is trended over time, geo-locations and cluster sizes by analyzing the tweet clusters. This research work was previously reported in [55].

Chapter 3

Methododology

This chapter describes the methodology used in the proposed framework to detect and trend the event from tweet clusters discovered using LSH.

3.1 The Proposed Framework

Figure 3.1 illustrates the proposed system framework for event detection and trending. The Twitter data corpus is divided into chunks of equal size to achieve a consistent processing time for the cluster discovery process.

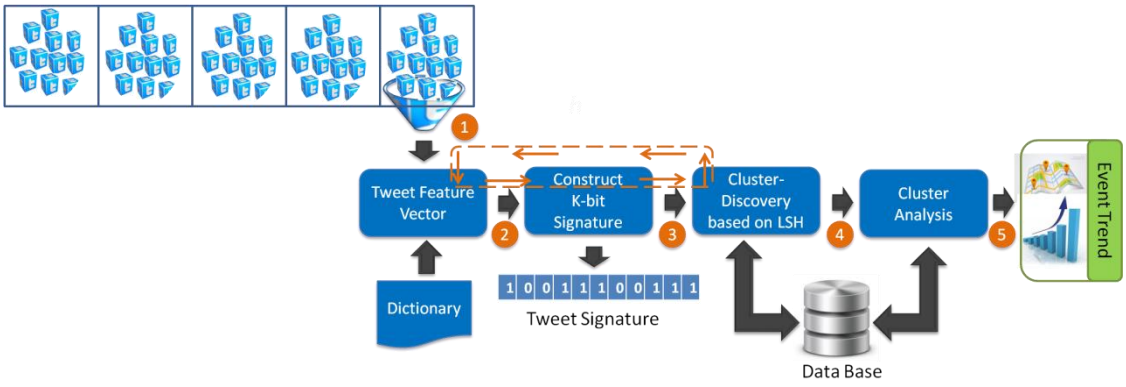

Figure 3.1: The proposed framework for event detection and trending

Each chunk contains no more than *I* number of tweets and is processed individually to find the clusters. In this thesis, the batch processing is implemented for automatic discovery of tweet clusters. The cluster details, consisting of the cluster label and information of individual tweets within that cluster, are stored in a MySQL database which is used later to generate the event trends. Each step of the event detection and trending process is explained below in greater detail:

Step-1:

Each tweet in a chunk is preprocessed and converted into a tweet feature vector mapped with a dictionary as described in Chapter 3 Section 3.1.1.

Step-2:

In order to reduce the dimension of the input vector, each tweet feature vector is further converted into its equivalent K-bit signature using a method proposed by Charikar explained in Chapter 3 Section 3.1.2.

Step-3:

The K-bit tweet signature received from Step 2 is added to its respective cluster using Algorithm-1(lines 10-22) and the weight for each term appearing in that cluster is updated as explained in Chapter 3 Section 3.1.3

Steps 1 through 3 are repeated for all tweets in a chunk. A set of clusters are discovered at the end of Step 3. The sample output for clusters discovered in a small data chunk is shown in Appendix V.

The above procedure is conducted for all tweet chunks. The dictionary is flushed and reconstructed every time a new chunk is processed. The cluster terms are sorted by their weight (including hashtags) and the details of each tweet {user-id, geo-coordinates, timestamp} that appear in that cluster are stored in the database (lines 24-30 of Algorithm-1). The cluster terms are sorted using merge sort (a divide and conquer algorithm that divides the input array into two halves, calls itself for the two halves and merges the two sorted halves) based on their weights in line 25 of Algorithm-1 with the complexity of O(T log T) where *T* is the number of terms in a

cluster. The attributes of each cluster having I_c number of tweets are stored (lines 24-30 in Algorithm-1) in the database with the complexity of $O(R*I_c)$ where *R* is the number of clusters discovered in the chunk. Therefore, the worst case complexity for storing the cluster details is $O(R*(T \log T)*I_c)$.

In this thesis, the minimum information of each cluster is stored in the database for effective use of storage. The table structure used to store the cluster details is included in Appendix III.

Step-4:

Cluster analysis is performed after processing all tweets from a data corpus. The most frequently appearing terms of the cluster are used to label the cluster and are considered as the cluster centroid (line 32 of Algorithm-1). The cluster labels are fetched from the database and analyzed independently. A key search is performed on the cluster labels to find the matching clusters for an event. Therefore, the for-loop (lines 31-34 of Algorithm-1) is repeated to find the matching clusters of an event with the time complexity of O(R) where R is the number of clusters.

Step-5:

The behaviour of the event is shown in terms of its trend using the information provided by the consolidated cluster. The event is trended based on time, geo-locations and cluster size. Algorithm-1(lines 35-36) describes this step.

3.1.1 Tweet Feature Vector

The experiments described within this thesis are conducted on a Twitter dataset containing the event "2011 End Times Prediction"[4, 5]. Figure 3.2 represents the Twitter snapshot for a sample tweet taken from the dataset used in this research work and Figure 3.3 shows an example of tweet in html format and its equivalent after removing the html tags and space. An unprocessed tweet in plain text is referred to as a raw tweet in this thesis.

Figure 3.2: Twitter snapshot for a sample tweet

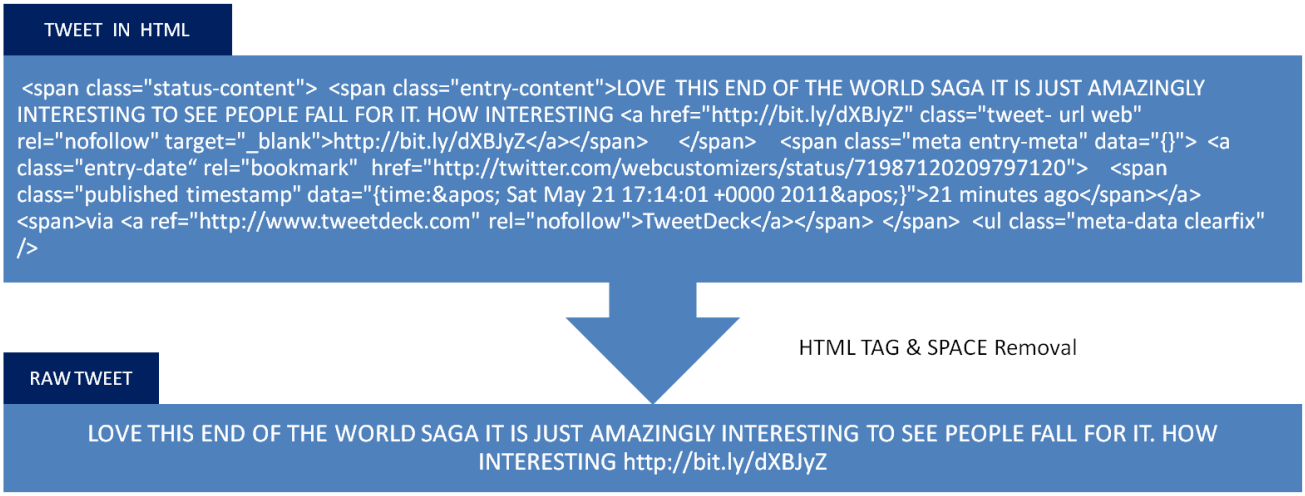

Figure 3.3: Removal of html tag from a tweet

A dataset containing a set of tweets related to the keyword "End%Of%World" is prepared in the format as shown in Table 3.1 and fed into the developed system.

Table 3.1: Input Tweet Format

Tweet Format	**Description**
Tweet-Id	Auto generated id for tweet
User-Id	User of tweet
Geo-Coordinates	Geo location {latitude, longitude}
TimeStamp	Date and time of tweet with UTC time zone
Tweet	Raw tweet

Figure 3.4 represents the system flow diagram for converting a raw tweet into the feature vector, which is later used as input for the clustering process. The creation of a tweet feature vector is comprised of two subtasks: 1) preprocessing a given tweet and 2) preparation of the feature vector for that tweet. A set of features are extracted from each tweet while performing the first subtask. Each feature that appears in a tweet is referred to as a unigram. For simplicity, only the unigrams from tweets are considered in this research work. Each preprocessed tweet is converted to its equivalent feature vector in the second subtask. Both subtasks are explained below in greater detail.

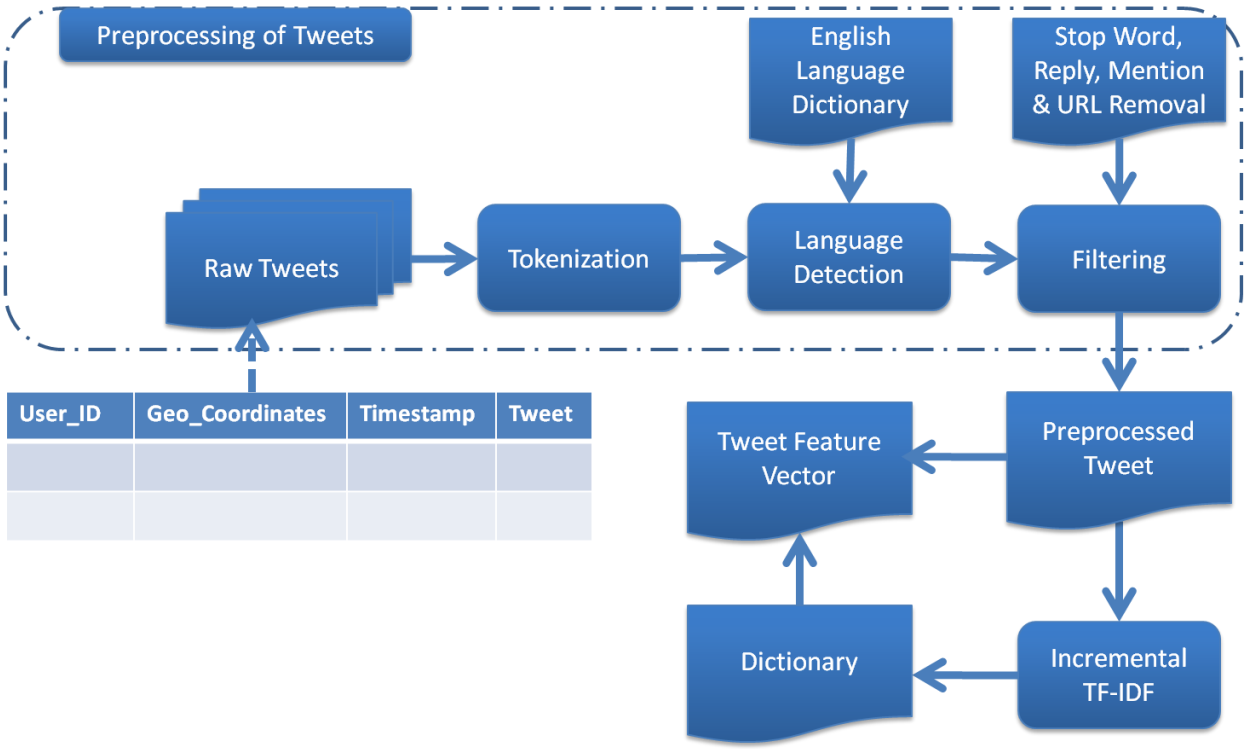

Figure 3.4: System flow for creation of tweet feature vector

3.1.1.1 Preprocessing of Tweets

A considerable level of noise in the data is removed through this subtask. Tokenizing is a process of splitting text into a set of individual terms or tokens. Each tweet is tokenized into a sequence of terms. Only tweets written in the English language are considered for clustering. In this thesis, it is assumed that a minimum of 30% of the words in a given tweet are required to be in English in order to qualify for further processing. In IR, the most commonly used words in a

text document are referred as stopwords. The qualified tweet is checked against a standard stopword list. The Natural Language Tool Kit's stopword list is applied in this process to eliminate terms which carry the least information. For example, articles like 'the', 'a', and 'an' are removed from the tweets. The full list of stopwords used in this research is included in Appendix I. Terms containing only alphanumeric characters are treated as valid tokens. For the purpose of simplification, URLs and the token starting with '@' (i.e. a mention or reply) are removed from the tweets while hashtags are retained in the filtration process. At the end of this process, each tweet is split into a set of features which are included in the vector space model.

To illustrate the above process, consider the following sample tweet:

> LOVE THIS END OF THE WORLD SAGA IT IS JUST AMAZINGLY INTERESTING TO SEE PEOPLE FALL FOR IT. HOW INTERESTING http://bit.ly/dXBJyZ

After the tokenization, language detection and filtration processes, the following set of features are extracted from the above shown tweet:

{'love', 'end', 'world', 'saga', 'amazingly', 'people', 'fall'}

3.1.1.2 Incremental TF-IDF

Mathematically, a vector is quantified using its direction and magnitude. In practice, there are several term weighing schemes that have been used to assign values or magnitude for features appearing in a tweet. Term Frequency-Inverse Document Frequency (TF-IDF) is one of the most fundamental techniques in IR and is used in this thesis to assign the value for a tweet's features. TF-IDF is formally defined as a product of term frequency (TF) and inverse document frequency (IDF) where TF represents the importance of the term in a document and IDF represents the importance of the term in the entire corpus[14, 28]. The TF-IDF weighing scheme assigns weight to term *w* in document *d* using the following formula [56]:

$$\mathrm{tf-idf}(w,d) = \mathrm{tf}(w,d) * \mathrm{idf}(w) \quad (1)$$

$$\mathrm{idf}(w) = \log \frac{N}{\mathrm{df}(w)} \quad (2)$$

where $\mathrm{tf}-\mathrm{idf}(\mathrm{w},\mathrm{d})$ is known as TF-IDF weight. $\mathrm{tf}(\mathrm{w},\mathrm{d})$ is a term frequency that represents the number of times the term *w* occurs in a document *d*. $\mathrm{idf}(\mathrm{w})$ is an inverse document frequency for the term *w* where *N* is the number of documents in a data corpus and $\mathrm{df}(\mathrm{w})$ is the number of documents in which the term *w* occurs atleast once .

Today, social networking sites produce large data corpora and it is difficult to fit them using a document vector space model due to the limitation of memory. Updating the TF-IDF weight of each term in a dictionary is a challenging task due to the growing Twitter data corpus. There are two possible ways to address this problem. Either we could use a static dictionary built from a fixed size data corpus or incrementally update the term weight every time the new document arrives which is known as incremental TF-IDF [10, 30]. A combination of these two approaches is used in this thesis work. To the best of our knowledge, it is the first time that these approaches have been used in combination for Twitter data to solve the aforementioned problem.

The data corpus is divided into chunks of equal size where each chunk has no more than *I* number of tweets. Let us consider the data corpus $\mathrm{D} = \{\mathrm{D}_1, \mathrm{D}_2, \mathrm{D}_3 \ldots \mathrm{Dn}\}$ where $\mathrm{D}_1, \mathrm{D}_2, \mathrm{D}_3 \ldots \mathrm{Dn}$ are tweet chunks. The static dictionary is constructed for each chunk D_I based on incremental TF-IDF. For better utilization of memory, the TF-IDF weight is updated for each term in the dictionary by processing each tweet one at a time in D_I.

Hence, equations (1) and (2) are rewritten and the incremental TF-IDF is defined as below:

$$\mathrm{tf}-\mathrm{idf}_\mathrm{i}(\mathrm{w},\mathrm{t}) = \mathrm{tf}(\mathrm{w},\mathrm{t}) * \mathrm{idf}(\mathrm{w}) \tag{3}$$

$$\mathrm{idf}(\mathrm{w}) = \log \frac{\mathrm{I}_\mathrm{i}}{\mathrm{df}_\mathrm{i}(\mathrm{w})} \tag{4}$$

where $\mathrm{tf}-\mathrm{idf}_\mathrm{i}(\mathrm{w},\mathrm{t})$ is tf-idf for term *w* in tweet *t* from tweet chunk D_i. $\mathrm{idf}(\mathrm{w})$ is an inverse document frequency for the term *w* where I_i is the number of tweets in D_i and $\mathrm{df}_\mathrm{i}(\mathrm{w})$ is the number of tweets from D_i in which the term *w* occurs atleast once.

Finally, tweet features are mapped to the dictionary to generate its tweet feature vector where unit weight is assigned to a dictionary term that appears in the tweet. In summary, a static dictionary for each tweet chunk is constructed using incremental TF-IDF as described above in order to represent a tweet vector in high-dimensional vector space. Appendix II shows the

sample dictionary constructed for a small chunk of the tweet dataset containing 100 tweets. Figure 3.5 shows an example of feature vector representation of tweets. Thus, the worst case time complexity to build a dictionary for *I* number of tweets having *m* terms (line 1 in Algorithm-1) is O(I*m). In this thesis, the feature that occurs only once in a tweet chunk is removed while building the dictionary for that tweet chunk.

Figure 3.5: Example for feature vector representation of tweet

3.1.2 K-bit Signature Representation for Tweet Feature Vector

Event detection is a research topic initiated by Allan [2] during Topic Detection and Tracking research. The objective of the TDT research is to identify events in news streams. However, their proposed approach does not provide a feasible solution to handle the rapidly-growing social media data stream. Indyk and Motwani used LSH technique, an ANNS approach, to detect unseen stories in a stream of broadcast news stories [18]. LSH technique is explained in detail in Chapter 2 Section 2.1.2. The key idea of LSH is to apply the hash functions in such a manner that the probability of collision is much higher for similar tweets than for tweets that are dissimilar (i.e. similar tweets fall into the same bucket). The gap between two dissimilar objects has to be greater in order to prevent the collision of dissimilar objects into the same bucket. In this section, our proposed methodology for cluster discovery of tweets using prefix tree based LSH for the purpose of event detection and trending is discussed.

Charikar proposed a LSH scheme using random hyperplanes for cosine similarity [22]. The family of LSH functions constructed in [22] map high dimensional vectors to low dimensional vectors while preserving similarity between vectors in the original space. In this thesis, Charikar's proposed hash function $h_{\vec{r}}(\vec{u})$ defined in Equation (5) is used to generate a K-bit signature for the tweet feature vector by computing the dot product of feature vector $\vec{u}$ with

m-dimensional random unit vector $\vec{r}$ and retaining the sign of the resulting product [22]. Each dimension in $\vec{r}$ is drawn from gaussian distribution with mean 0 and variance 1.

$$h_{\vec{r}}(\vec{u}) = \begin{cases} 1 & if\ \vec{r}.\vec{u} \geq 0 \\ 0 & if\ \vec{r}.\vec{u} < 0 \end{cases} \quad (5)$$

The K-bit tweet signature reduces the dimension of the original tweet feature vector. Thus, it is a low dimensional vector and it helps to fasten the clustering process. In Charikar's LSH scheme, the similarity between documents is determined using cosine angle distance and is defined below [22]:

$$\cos\big(\theta\ (\vec{u}, \vec{v})\big) = \cos((1 - \Pr[h_{\vec{r}}(\vec{u}) = h_{\vec{r}}(\vec{v})])\pi) \quad (6)$$

where $\theta\ (\vec{u}, \vec{v})$is the cosine angle between $\vec{u}$ and $\vec{v}$ and is proportional to the hamming distance of their signature vectors while preserving the cosine similarity in high dimensional space[22]. $\Pr[h_{\vec{r}}(\vec{u}) = h_{\vec{r}}(\vec{v})]$ is the probability that a random hyperplane separates two vectors which is proportional to the angle (i.e. $\theta\ (\vec{u}, \vec{v})$) between them. The hamming distance is the number of bits that differ between two binary strings. For example, the hamming distance is 2 for the two 3-bit vectors '1<u>00</u>' and '1<u>11</u>'.

A 17-bit tweet signature for a sample tweet as shown below is found by using equation 5. The computation of the K-bit signature for a sample tweet is explained in the case study section in Appendix V. The value for K is set to 17 in this thesis according to the Table 4.1.

Tweet:

LOVE THIS END OF THE WORLD SAGA IT IS JUST AMAZINGLY INTERESTING TO SEE PEOPLE FALL FOR IT. HOW INTERESTING http://bit.ly/dXBJyZ

Signature: bitarray('00000111000101000')

3.1.3 Cluster-Discovery of Tweets using Prefix Tree based LSH

In this thesis, prefix trees are used in the place of buckets in LSH to speed up the process. The detail of prefix tree is described in Chapter 2 Section 2.1.3. We follow the procedure of finding the nearest neighbour of a user as described by Kamath et al. [25]. Algorithm-1[lines 2-

23] is a modified version of an algorithm proposed by Kamath et al. [25] in which the purpose of clustering is to identify newly-formed communities of users from the real-time web. Use of a prefix tree allows a tweet's nearest neighbour (line 10 in Algorithm-1) lookup in O(K) time where K is the length of tweet signature [25].

Let's consider the nearest neighbour of a tweet $\overrightarrow{t}$ is $\overrightarrow{t}_n$, which belongs to cluster C_n and a set of permutation functions $\mathcal{H} = \{\pi_1, \pi_2, \pi_3, \ldots \pi_P\}$, with each permutation function in the following form [24]:

$$\pi(x) = (ax + b) \bmod p \tag{7}$$

where *a*, *b* are chosen randomly and *p* is a prime number. The collection of *P* number of prefix tree P_{tree} is defined where every prefix tree corresponds to a permutation function in $\mathcal{H}$. Hence, *P* times a tweet are permutated and they are stored in P_{tree}.

To compute the nearest neighbour and its cluster for the given tweet, first the signature of the given tweet is calculated using Equation (5). Next, P-times the signature is permutated and its neighbour is found in the prefix tree by iterating the tree starting from its root. At end of the step, possibly *P* number of neighbours with respective cluster are collected for the given tweet. The nearest neighbour of the given tweet is the one which has the smallest hamming distance as given by Equation (6).

In this thesis, the given tweet $\vec{t}$ is assigned to its nearest neighbour tweet's cluster C_n when the similarity between them exceeds the threshold (i.e. the LSH parameter - probability of finding the nearest neighbour with a cosine similarity). Otherwise, a new cluster is created with the given tweet. The cosine similarity is used to compute the similarity between the tweet and its nearest neighbour using the equation (8) and is defined below [56]:

$$\mathrm{sim}(\overrightarrow{t}, \overrightarrow{t_n}) = \cos(\overrightarrow{t}, \overrightarrow{t_n}) = \frac{\overrightarrow{t} \cdot \overrightarrow{t_n}}{|\overrightarrow{t}||\overrightarrow{t_n}|} = \frac{\sum_{i=1}^{M} \mathrm{tf\text{-}idf_i}(w,\vec{t}) * \mathrm{tf\text{-}idf_i}(w,\overrightarrow{t_n})}{\sqrt{\sum_{i=1}^{M} \mathrm{tf\text{-}idf_i^2}(w,\vec{t})} * \sqrt{\sum_{i=1}^{M} \mathrm{tf\text{-}idf_i^2}(w,\overrightarrow{t_n})}} \tag{8}$$

where the numerator is the dot product of two tweet feature vectors and the denominator is the product of the euclidean length of the two feature vectors. The euclidean length of any tweet feature vector $\vec{t}$ with *M* terms is defined as below [56]:

$$|\overrightarrow{t}| = \sqrt{\sum_{i=1}^{M} tf - idf_i^2(w, \overrightarrow{t})} \quad (9)$$

where $tf - idf_i^2(w, \vec{t})$ is the square of TF-IDF of term *w* in a tweet feature vector $\overrightarrow{t}$.

Whenever a new tweet is added into a cluster, the weight for each term appearing in that cluster is updated using the average document frequency method. Selection of the most frequently occurring terms in a cluster is known as the frequency-based feature selection method, which can be used to label the cluster as described in [28]. The cluster labels are further used to find the event clusters by matching their labels with the keywords of the event. In this thesis, the average document frequency method, a category of frequency-based feature selection method, is used to compute the term weight for each term appears in the cluster when a new tweet is added into that cluster. The average document frequency of a term is defined as a number of tweets containing that term over a total number of tweets in a cluster. Therefore, the for-loop is repeated (lines 3-5 in Algorithm-2) to update the weight of each term whenever a new tweet is added into the cluster using the average document frequency with the worst case complexity of O(I*N) where I is the number of tweets having N number of terms.

3.2 Algorithm for Event Detection and Trending

The methods for finding the event from tweet clusters and generating its trend are collectively put together in Algorithm-1. This algorithm is the high level process for event detection and trending from tweet clusters discovered using prefix-tree based LSH. Let us consider the Twitter data corpus $D=\{D_1,D_2,D_3,\ldots D_n\}$ as discussed in Chapter 3 Section 3.1.1.2 where $|D_i| \leq I$ and $D_i = \{t_1,t_2,t_3,\ldots t_I\}$ for each $D_i \in D$. $\{t_1,t_2,t_3,\ldots t_I\}$ is a set of tweets in a chunk D_i.

Algorithm-1: Event Detection and Trending from Tweet Clusters Discovered by using LSH

Given: set of tweets $D_I = \{t_1,t_2,t_3,\ldots t_i\}$ where each tweet in the format of {tweet-id, user-id, geo- coordinates, timestamp, tweet} , Cluster Setting { number of Permutations P, signature length K, probability of finding nearest neighbour with a cosine similarity T}, keywordsInEvent

Output: set of clusters $C_R = \{C_1,C_2,C_3,\ldots C_r\}$, event summary and event trends

*/*Construction of Static Dictionary */*

1. Build dictionary $Dict_i$ for D_i with length m using incremental TF-IDF

/*** Cluster-Discovery Process Begins *****/**

2. Create a set of random vectors in Gaussian distribution
3. for each j in 1 to P
4. $\pi_i \sim \mathcal{H}$ */* Initialize prefix trees using permutation functions $\mathcal{H}$ as described in Section 3.1.3 */*
5. Create prefixtree $P_{tree}[j]$
6. end for
7. for each tweet $t \in D_i$ do:

/ Construct tweet vector using $Dict_I$ as described in section 3.1.1 */*

8. Construct tweet vector using $Dict_i$

/ Create K-bit signature for tweet vector as described in section 3.1.2 */*

9. Create K-bit tweet signature

/ Find the cluster of a tweet t as described in section 3.1.3 */*

10. $t_n \leftarrow$ get nearest neighbour of tweet from prefextree P_{tree}
11. Generate new tweet signature using R
12. for each j in 1 to P
13. Insert tweet signature Into prefextree $P_{tree}[j][\pi_j(t)]$
14. end for
15. if sim(t , t_n) $\geq$ T then
16. if t is not in cluster C_n then:
17. addTweetToCluster(t,C_n)
18. end if
19. else
20. create new cluster C_R += {C}
21. addTweetToCluster(t,C)
22. end if
23. end for

/*** Storing Cluster Details into Database Begins *****/**

24. for each cluster C ∈ C_R and |C| ≠ 1 do: */* |c| ≠ 1 ignores singleton cluster */*
25. Sort(C.cluster_label)
26. *storeClusterToDB*(C.cluster-id,C.cluster_label[hashtags], C.cluster_label[nonhashtags],C.cluster-size)
27. for each t in C do:
28. *storeTweetToDB*(t.cluster-id,t.tweet-id,t.geo-coordinates,t.user-id,t.timestamp, t.tweetvector)
29. end for
30. end for

/*** Cluster Analysis, Event Detection and Trending Begins *****/**

31. for each cluster C fetch from DB do:
32. label ← C.cluster_label[nonhashtags][l] */* select top l-terms from C.cluster_label*/*
33. If label match with KeywordsInEvent then: E += {C}
34. end for
35. Consolidate clusters in E and generate event summary
36. Generate event trend based on time, geo-location, cluster size from tweets of E

Algorithm-2: Procedure to add tweet into cluster

Procedure addTweetToCluster(tweet t, cluster C)

1. C.TweetCount += 1
2. C.tweets += t
3. for each term in t do:
   / Term weigh for cluster label is computed using the average document frequency described in section 3.1.3 */*
4. C.cluster_label[term] = C.tweets[term] / C.TweetCount
5. end for

End Procedure

It is necessary that the tweets in Twitter data corpus D span a specific time interval. Our experiment covers one day of tweets from the dataset explained in Chapter 4. Algorithm-1 is used for all chunks in D to produce the tweet clusters. After processing all tweets in D, the clusters are analysed further for the presence of event and then its trends are generated. A cluster with a single tweet is ignored during cluster analysis due to the fact that its single tweet cannot form the event. The algorithm is evaluated on a set of tweets posted on Twitter in a specific time interval and the experiment is discussed in Chapter 5. The Algorithm-1 is the main contribution of this research study except the lines 2-23 as it has been followed from [25].

Chapter 4

Implementation & Results

The experiments described in this thesis are conducted on a Twitter dataset containing the event "2011 End Times Prediction"[4, 5]. Hereafter, we shall refer to the event as "EndOfWorld" in this thesis report. The event was widely spread over popular news periodicals and social media websites. According to a prediction made by Harold Camping's, judgment day would take place on May 21st, 2011 and followers of Camping claimed that 3% of the world's population would be raptured or ascend to heaven [4-6]. One million publically available tweets posted between 18-May-2011 and 21-May-2011 in html format are extracted using the Twitter search API to serve as this experiment's dataset and they are stored in a MySQL database, from which tweets related to the keyword "End%of%World" are taken to conduct the experiments. The dataset is manually verified for the presence of that event and Figure 4.1 shows four days of tweets containing the same event. The volume of tweets related to an event is generally greater on the day of its occurrence. It is observed that the number of tweets related to the keywords "End%Of%World" were far greater than average on the day of the event (May 21, 2011). Algorithm-1 is applied for event detection and trending on an experimental Twitter dataset containing 1694 tweets published on one day (May 21, 2011). There are no human participants involved, other than the researcher, and the authors of the tweets have been de-identified for this thesis work. Thus, the dataset does not require the approval of the Ryerson Ethics Board (REB) and an electronic confirmation from the REB for the same is attached in Appendix IV.

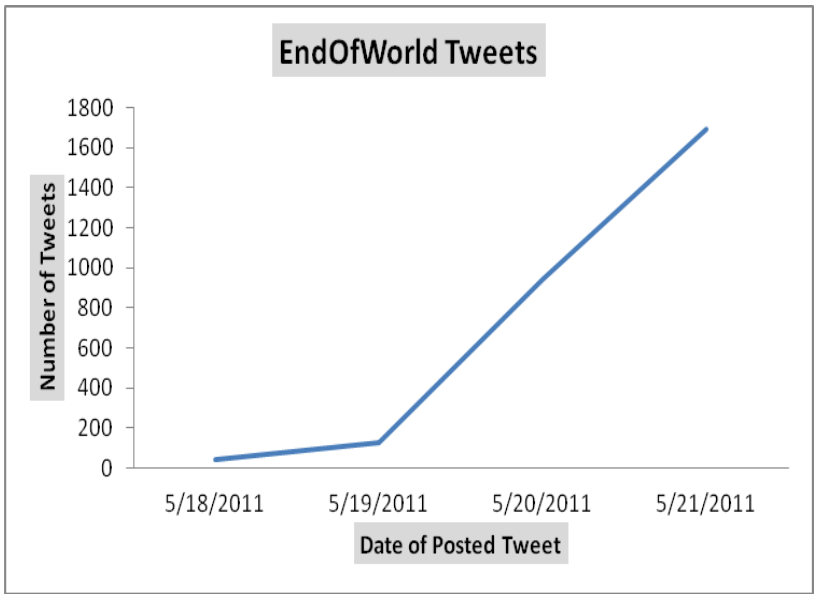

Figure 4.1: Four days of tweets having "EndofWorld" event

This chapter discusses the implementation of the proposed framework explained in Chapter 3, followed by the results of the experiments conducted for cluster-discovery of tweets to accomplish event detection and trending. Clusters discovered by three different techniques are compared: i) LSH, ii) GAAC and iii) K-means. The LSH technique is compared with two previously known clustering algorithms, one of which is a type of flat clustering (K-means) and the other is a type of hierarchical agglomerative clustering (GAAC). Various experiments were conducted in this research work to compute the best values for the LSH parameters. Finally, the clusters are analyzed for the presence of an event and its trends are presented in this chapter.

4.1 Implementation

The software and hardware used for this thesis implementation is discussed in this section. Python programming language is an easy-to-learn interactive, object-oriented and multi-paradigm language [26]. In this thesis, the availability of numerous open source Python libraries are used for the implementation of various components of the system discussed in the previous section. Table 4.1 lists the major libraries used in this research work.

The input file is prepared using a Twitter data corpus containing the event "EndOfWorld" in a format shown in Table 3.1 and is converted into data file having the extension .dat where the

column values are tab separated. The input file is fed into the system described in Chapter 3 Section 3.1. A sample input .dat file is shown in Figure 4.2. The Tweet column in the input file is used for the various clustering techniques.

Table 4.1: List of Major Python Libraries used for System Implementation

Python Library	Description
Gensim	For Term Frequency – Inverse Document Frequency computation[58]
Nltk	The Natural Language Tool Kit [59] is used for, • Stopword filter • Clustering Tweets using K-means and GAAC
Enchant	Spell checker software [60] contains American English language dictionary which is used for tweet language detection
MySQLdb	Python DB API 2.0 complaint interface to access MySQL DB components [61]
Bitarray	Provides object type to represent an array of booleans [62]
Bio.trie	Trie data structure to represent the prefix tree [63]
Time	Time is used to access the time related function [64]
Re	Provides regular expression matching operations [65]
Pytagcloud	Creates simple tag clouds [66]
Pygmaps	Python wrapper for Google Maps Java Script API V3 [67]

```
1  webcustomizers  None    2011-05-21 17:14:01+00:00   LOVE THIS END OF THE WORLD SAGA IT IS JUST AMAZINGLY
   INTERESTING TO SEE PEOPLE FALL FOR IT. HOW INTERESTING http://bit.ly/dXBJyZ
2  webcustomizers  None    2011-05-21 16:13:54+00:00   I LOVE THIS END OF THE WORLD SAGA ON MAY 21ST.IT
   AMAZINGLY INTERESTING TO SEE PEOPLE FALL FOR IT. HOW INTERESTING http://bit.ly/dXBJyZ
3  webcustomizers  None    2011-05-21 15:59:55+00:00   I AM SURE IT IS PAST 6PM SOMEWHERE IN THE WORLD SO
   IT IS NOT THE END OF THE WORLD, FALSE PROPHETS EVERY WHERE http://bit.ly/dXBJyZ
4  joker2070   (u'California, USA', (36.778261, -119.4179324)) 2011-05-21 06:03:21+00:00   @AaronRodgers12
   #endoftheworldconfessions. I really want to see the ring on ur finger n I will bow down in front of u,
   if u sign my ball!
5  thelovetwins    (u'Ohio, USA', (40.4172871, -82.90712300000001))    2011-05-19 06:32:33+00:00
   @nina_mercedez We just got invited to an end of the world party
6  themermaidnyc   (u'Upper West Side, New York, NY, USA', (40.7870106, -73.9753676))  2011-05-20
   20:32:12+00:00   @julz124 Well then you shall receive an oyster po-boy! Excellent choice. We know end of
   the world decisions are hard.
7  jonkelly    None    2011-05-20 16:07:01+00:00   Everything You Need to Survive the Upcoming End of the
   World http://t.co/Dcwf7Sr #thisorthat http://t.co/YuC5HcN
8  realcomedyking  (u'Hartsfield-Jackson Atlanta International Airport (ATL), 6000 N Terminal Pkwy,
   Atlanta, GA 30320, USA', (33.639975, -84.44403199999999))  2011-05-21 05:02:37+00:00
   #EndOfTheWorldConfessions i love God and i love my Mom. If you LOVE your Mom and God, R-T!
9  darlingitsnikki (u'Los Angeles, CA, USA', (34.0522342, -118.2436849))   2011-05-21 03:29:47+00:00   If
   we&apos;re ending, I&apos;m ending at the bar darling! &quot;@Geli: Although a drive-thru Del Taco would
   be a good thing #incaseworldends&quot;
10 lord_voldemort7 (u'Massachusetts, USA', (42.4072107, -71.3824374))  2011-05-21 11:09:17+00:00
   #endoftheworldconfessions I am gossip girl. You know you love me xoxo
```

Figure 4.2: Snapshot of a portion of sample input .dat file

The Twitter data corpus is divided into smaller chunks, each with a maximum size of N number of tweets. The following four tasks are performed on each tweet chunk:

1) Preprocessing tweets
2) Building up the dictionary for tweet features retrieved from preprocessed tweets, followed by the construction of the feature vector
3) K-bit signature generation of the tweet feature vector
4) Performing cluster discovery.

In the process of preprocessing the tweets as explained in Chapter 3 Section 3.1.1.1, the non-alpha numeric characters are removed from the token except the one related to the Twitter hashtag. The hashtag symbol, #, is retained when it appears at the beginning of the term, e.g. #endofconfession. The 're' Python library is used to accomplish this task. The pattern lookup for the alpha numeric characters in the given term is defined using 're' as shown in Table 4.2.

Table 4.2: Alpha numeric Pattern Representation in Python

Alpha numeric Pattern
Pattern = re.compile('[\W_]+')

Next, the stopwords are filtered from each tweet. The Python nltk.corpus stopword module is used for this purpose and a sample list of stopwords is shown in Table 4.3. The complete list of stopwords used in this thesis is shown in Appendix I. The language of the tweet is detected using the Python enchant spell checker library. The American English language dictionary from enchant is used to check the language of each term that appears in the tweet.

Table 4.3: Stopword List

Comma Separated Stopwords
i,me,my,myself,we,our,ours,ourselves,you,your,yours,yourself,yourselves,he,him,his,himself,she,her,hers,etc.,

Gensim [58] is an efficient Natural Language Processing (NLP) software framework which can be used to calculate the normalized TF-IDF scores defined in Equation (3) for the documents in large data corpus. Gensim processes the documents one after another. Thus, it is memory independent and is scalable to any size of data corpus [58]. Gensim is used in this thesis to construct the dictionary on each tweets chunk based on the concept outlined in Chapter 3 Section 3.1.1.2. This dictionary is used further to convert each preprocessed tweet to its feature vector.

The K-bit signature of the tweet feature vector explained in Section 3.1.2 is represented using a Python bitarray data structure. Tweet K-bit signature vector is generated for the purpose of reducing the tweet feature vector length and Algorithm-1 is applied to find its respective cluster as described in Chapter 3 Section 3.2.

Traditional clustering algorithms such as K-means and GAAC explained in Chapter 2 are used in this thesis to compare against the LSH technique used in the proposed Algorithm-1. The next section describes the cluster discovery stage in greater detail.

4.1.1 Cluster-Discovery Stage

Figure 4.3 provides a closer look at the clustering stage. In this thesis, three different clustering algorithms are explored: 1) LSH, 2) K-means and 3) GAAC. The LSH technique is used in the proposed framework explained in Chapter 3 for the cluster discovery process. The cluster module available in a standard Python library (nltk.cluster) is used for implementing K-means and GAAC algorithms. The running time required for each algorithm is calculated using Python time module.

The publically available open source Python code for LSH which was previously used by Kamath et al. [25] has been downloaded from [68] and modified to fit our purpose. Hash tables in LSH are replaced with prefix tree data structures in the proposed cluster-discovery algorithm to speed up the signature lookups while finding the nearest neighbour of tweets. Prefix tree data structure from the Python bio.trie library is used in this thesis to store the tweet signature vectors [63].

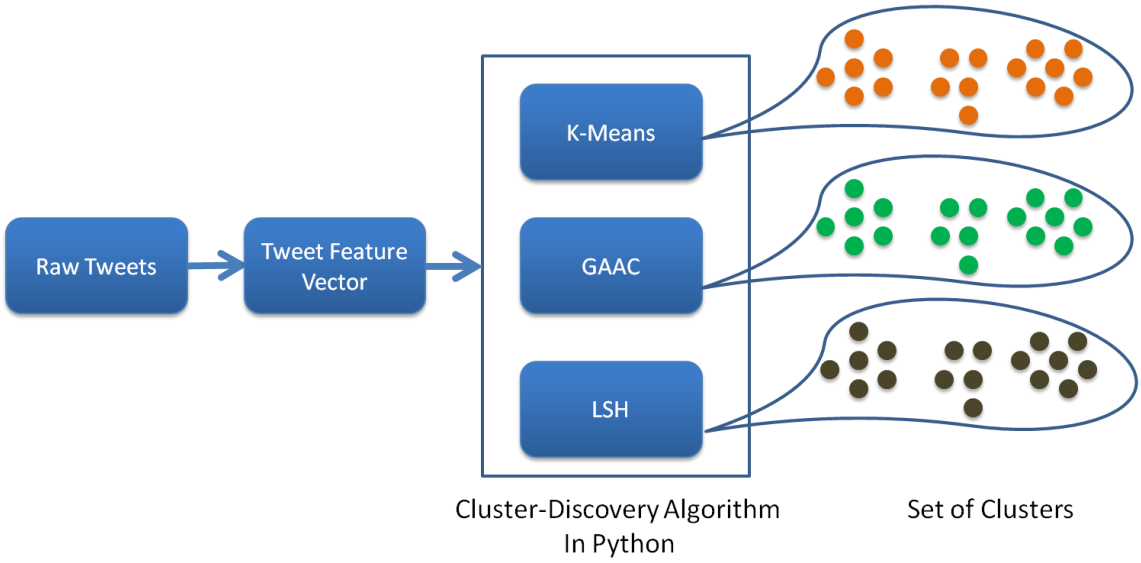

Figure 4.3: Cluster-discovery stage

4.1.2 Cluster Analysis

The attributes of clusters discovered using LSH from each tweets chunk, are stored in a MySQL database for analysis. The Python MySQLdb API 2.0 compliant interface is used to perform MySQL related transactions between Python and the MySQL database. The most frequently appearing terms of the cluster are used to label that cluster. Key search is performed on cluster labels to find the matching clusters for an event. The resultant clusters are combined to form a consolidated cluster from which the event trends are generated. An automatic process for finding the clusters related to the events and generating their trends is employed in this thesis and it is implemented in Python.

Pytagcloud, a Python library, is used to create the tag cloud to represent the cluster related to an event. The most frequently appearing terms including the hashtag of the consolidated cluster participate in the tag cloud as shown in Figure 4.9 for the sample event "EndOfWorld".

The geo location of each tweet from the consolidated cluster is used to construct a Google map to illustrate the geographically distributed tweets of the event as shown in Figure 4.6. The Python wrapper pygmap is used to construct the html form of the Google map to show the tweets based on their location.

The configuration of the systems that we used to implement our proposed framework is shown in Table 4.4.

Table 4.4: System Configuration

Details	System-1	System-2 (HPC)
CPU	Intel Core 2 Duo T8100 @ 2.10GHz	Intel Xeon X7550 @ 2GHz
RAM	2GB	1 TB
Operating System	32-bit Windows Vista SP2	CentOS Release 6.5
Programming Language	Python 2.7	Python 2.6.6
DataBase	MySQL Version 5.6	MySQL Version 5.1

System-1 is used for coding and execution of our experiments. One million publically available tweets are used in this thesis. System-2 is a high performance computing resource from SHARCNET [69], which is used to store these tweets in MySQL database and its size is close to 900MB. The SQL operations are performed on database containing tweets to manually find the event before starting the actual experiments in this research work.

4.2 Results

The performance metrics used in this thesis for cluster evaluation are discussed in this section, followed by the results of the experiments conducted for event detection and trending through the cluster-discovery process.

4.2.1 Performance Metrics

The tweets need to be pre-categorized in order to evaluate the clusters produced by different techniques. This can be accomplished by manually categorizing the tweets or by using a technique for automatic categorization. Manual categorization would be an extremely time consuming process, especially on large social media datasets. Automatic categorization would be economically effective as it requires less time. The Hierarchical Agglomerative clustering technique has recently been used to automatically categorize the documents in [57]. GAAC clustering of documents in one language (English) is used as the "gold standard" and are compared with clusters produced from another language (Bulgarian) in [57]. In these

experiments, the similarity between two documents is computed using cosine distance. Similarly, this thesis work attempts to use the clusters produced using GAAC as gold standard clusters and comparing them with the clusters discovered using LSH and K-means. Estimating the number of clusters in a given data set is generally unreliable as the nature of the data is unknown. However, this thesis work attempts to determine the value for the number of clusters by processing the documents using LSH and then, using this to set the value for the number of clusters 'K' in the K-means algorithm and the value for the dendrogram depth 'c' in GAAC in order to perform the clustering process.

The quality of the clusters discovered by LSH and K-means are compared against GAAC. Purity and Normalized Mutual Information (NMI) are external evaluation criterion for cluster quality [28]. As explained earlier, the GAAC technique is considered as a golden truth clustering class in cluster evaluation. Let us consider the results of clusters $K=\{C_1,C_2,C_3,\ldots,C_n\}$ and the golden truth of clusters $\Omega=\{W_1,W_2,W_3,\ldots,W_w\}$ for 'N' number of tweets. Tweets in predicted clusters are labeled according to GAAC and quality is measured as follows:

Purity: To calculate the purity, each cluster is assigned to the class which is most frequent in the cluster, and then the purity of the cluster is measured as below [28]:

$$\text{Purity}(K, \Omega) = \frac{1}{N} \sum_n \max_w |C_n \cap W_w| \quad (10)$$

Normalized Mutual Information (NMI): High purity can be achieved when the number of clusters is large in which each document gets its own cluster. Thus, we use the NMI measure to avoid a tradeoff between the quality and the number of clusters. NMI is a normalized mutual information score which ranges between 0 (no mutual information) and 1 (perfect correlation) and is computed using mutual information (MI) and entropy.

Entropy is a measure of uncertainty for a partition set. For results clusters (K) and golden truth clusters (Ω), their entropy is defined by [28] as follows:

$$H(K) = -\sum_n \frac{|C_n|}{N} \log \frac{|C_n|}{N} \quad (11)$$

Where, $\frac{|C_n|}{N}$ is the probability that a tweet picked at random falls into a cluster$|C_n|$, likewise forΩ:

$$H(\Omega) = -\sum_w \frac{|W_w|}{N} \log \frac{|W_w|}{N} \quad (12)$$

The mutual information (MI) between K and Ω is calculated by [28]:

$$MI(K, \Omega) = \sum_n \sum_w \frac{|C_n \cap W_w|}{N} \log \frac{N|C_n \cap W_w|}{|C_n||W_w|} \quad (13)$$

Where $\frac{|C_n \cap W_w|}{N}$ is the probability that a tweet picked at random falls into both $C_n$and W_w.

Finally, NMI is formally defined as [28]:

$$NMI(K, \Omega) = \frac{MI(K,\Omega)}{[H(K) + H(\Omega)]/2} \quad (14)$$

In this research, the following metrics are used to measure the performance of clustering algorithms in addition to purity and NMI.

1. Running time: The amount of time required for a clustering technique to discover the tweet clusters from a given Twitter dataset. It measures the speed of the clustering technique.
2. Cluster Size: It is measured by counting the number of tweets in each cluster. The size of the cluster is proportional to the popularity of the event.

In this thesis, purity and NMI are used to find the values for LSH parameters and the results of the experiments are discussed in the next section

4.2.2 Evaluation of LSH Parameters

The experiments are repeated in a chunk of tweets containing 100 tweets with varying choices of LSH parameters: number of Permutations P, signature length K and the probability of finding nearest neighbour with a cosine similarity T. The experimental result for the effect of LSH parameters on purity and NMI with T=0.005 is reported in Table-4.1. We also conducted the experiments by setting different values for T with an interval of 0.1 over the range of 0.005 to 0.605. During experiments, it is observed that the increase in the value of T identifies a large number of singleton clusters (i.e. a cluster with a single tweet). The highest purity can be achieved for larger number of singleton clusters. However, attaining the highest possible value for NMI can avoid the trade off between the quality and number of clusters discovered in a given dataset. According to Table 4.5, the proposed algorithm yields better result in terms of NMI when P is 23, K is 17 and T=0.005.

Table 4.5: Experimental Results of LSH Parameters in a Chunk of Tweets

K / P	13		17		19	
	Purity	**NMI**	**Purity**	**NMI**	**Purity**	**NMI**
11	0.6771	0.5542	0.7396	0.6072	0.7396	0.5684
13	0.7083	0.5707	0.7604	0.6235	0.6875	0.5600
17	0.7708	0.6344	0.8021	0.6170	0.7292	0.5738
19	0.7500	0.5730	0.6875	0.5755	0.7604	0.6467
23	0.6875	0.5375	**0.7813**	**0.6830**	0.7708	0.6118
29	0.7396	0.6321	0.7813	0.6314	0.7708	0.6231
31	0.7396	0.5960	0.7604	0.6025	0.7813	0.6292
37	0.7292	0.5843	0.7708	0.6118	0.7708	0.6118
41	0.7500	0.6179	0.7188	0.5430	0.7917	0.6553
43	0.7708	0.6333	0.7708	0.6118	0.7188	0.5770
47	0.7396	0.5817	0.7813	0.6383	0.7604	0.5832
53	0.7292	0.6025	0.7708	0.6118	0.7708	0.6118
59	0.7188	0.5828	0.8125	0.6370	0.7292	0.5644
61	0.7396	0.5910	0.7708	0.6118	0.7604	0.5783
67	0.7708	0.6118	0.7917	0.6289	0.7708	0.6118
71	0.7708	0.6132	0.7708	0.6118	0.7708	0.6118
73	0.7500	0.5808	0.7500	0.6108	0.7292	0.5644
79	0.7708	0.6214	0.7708	0.6181	0.7604	0.5938

Increasing the values of P and K will usually have a significant impact on the running time required for the proposed algorithm. However, according to Figure 4.5, the proposed algorithm still requires a smaller constant running time to process the tweets compared to K-means, with the final set values of P, K and T. Henceforth, the experiment uses LSH values P=23, K= 17 and T=0.005 for the cluster discovery process.

4.2.3 Comparison of Clustering Algorithms

The selection of clustering algorithm can be effective on clusters' quality. Therefore, we experimented with different clustering algorithms to find an algorithm which performs well for the cluster discovery of tweets for the purpose of event detection in social media. Two well known clustering algorithms, K-means and GAAC from the Python NLTK library, are used on the experimental dataset. The Python library lacks a LSH algorithm which can be used for direct application of event detection in social media data. Hence, we acquired Python-based LSH from [68] and modified it to fit our proposed algorithm for event detection and trending. In short, K-means and GAAC are used from a pre-built Python library and the LSH is used from [68], which was modified for the purpose of event detection and trending. Tweet clusters discovered by using the LSH technique and K-means are compared against those discovered using GAAC. Purity and NMI are external evaluation criterion for cluster quality [28]. The GAAC technique is used as the golden truth clustering class.

In this thesis, the resultant clusters are those retrieved from the proposed algorithm and K-means and the golden truth clusters are those discovered using GAAC. Tweets in resultant clusters are labeled according to the golden truth cluster labels and their quality is measured using purity and NMI. In this thesis, the cluster evaluation is performed in a chunk containing 100 tweets. The LSH technique found 8 clusters in that chunk after setting its parameters P=23, K= 17 and T=0.005 as shown in Table 4.5. The same number of clusters is discovered using GAAC and K-means algorithms. Figure 4.4 shows comparison between the quality of the clusters discovered using K-means and the proposed algorithm against hierarchical GAAC. From the experiments, we found that using LSH attained 12.5% and 16.6% higher purity and NMI respectively than using K-means.

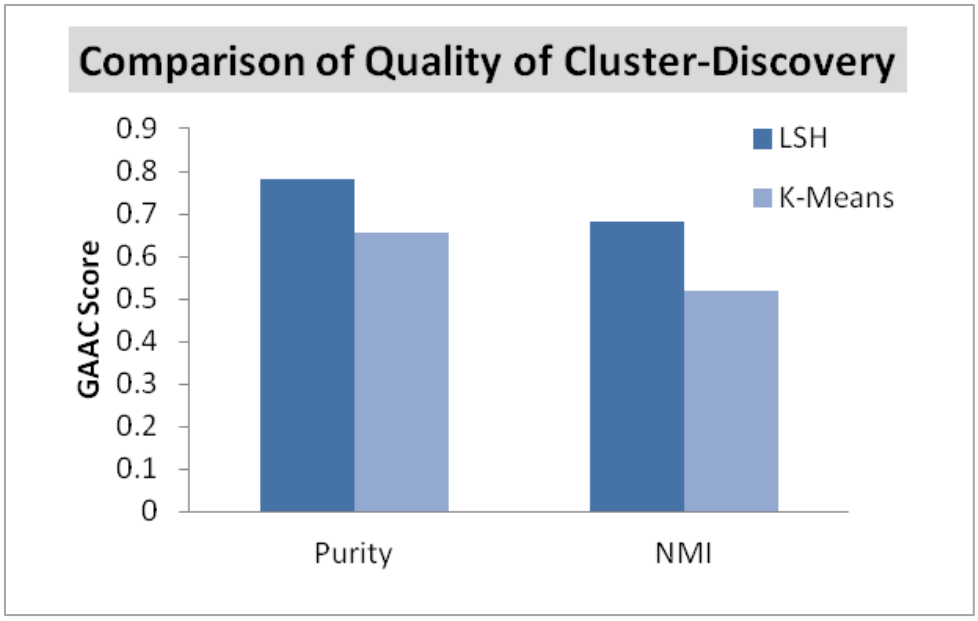

Figure 4.4: Quality of cluster-discovery algorithms for a chunk of tweets

Another important aspect of the performance of clustering algorithms is measured through the running time required for each of them. It is an important property that a clustering algorithm performs well in terms of both the quality of the cluster it produces and the time required to discover the clusters.

Figure 4.5 shows the running time required for the LSH technique and K-means to discover the clusters from the experimental dataset containing 1694 tweets. LSH took only 12.99% of the average running time required for K-means and consistently about 0.3 second to the complete entire cluster discovery process. It is clearly observed that there is a significant difference in the running time required for LSH compared to K-means while maintaining the quality of clusters. The LSH technique is achieved this consistent running time due to the fact that a given Twitter dataset is processed in chunks of equal size and the prefix tree is used to speed up the process of finding the nearest neighbour tweet. In this research work, the running time require for GAAC is not recorded due to the known fact that the hierarchical clustering suffers higher time and space complexities in a high dimensional large dataset [29].

The experimental results demonstrate that the proposed algorithm based on LSH technique is a better choice for social media data to detect high quality clusters in a shorter time period compared to the traditional K-means. It helps for early detection of events in social media website.

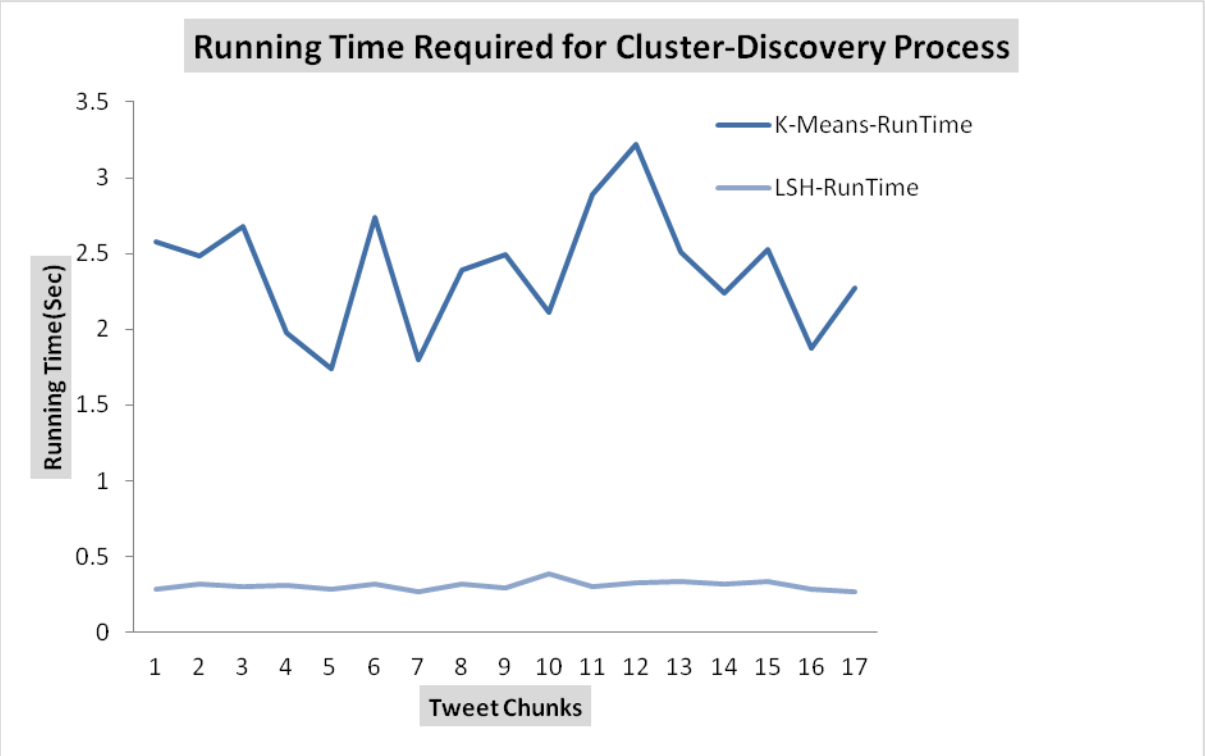

Figure 4.5: Running time required for LSH and K-means in an experimental dataset

4.2.4 Cluster Analysis for Event Detection and Trending

The cluster retrieved using the LSH technique is analyzed for event detection and trending. Frequency-based feature selection is applied to build the list of frequently occurring terms for each cluster. The most frequent terms are considered as cluster centroid, and are used to label the cluster. Key search is performed on these cluster labels to find matching clusters pertaining to the event "EndOfWorld" which are then consolidated to a single cluster. We found 22 clusters for a given experiment dataset which are related to the event "EndOfWorld" and they are consolidated into a single cluster. Figure 4.6 shows the size of each cluster related to the event spanning over a time period of one day, May 21, 2011. According to Figure 4.6, it is observed that 16 out of 22 clusters have more than 30% of tweets of a chunk and it conveys that many Twitter users discussed the same topic. Event trending is performed using the cluster that belongs to the event and is discussed further in this section.

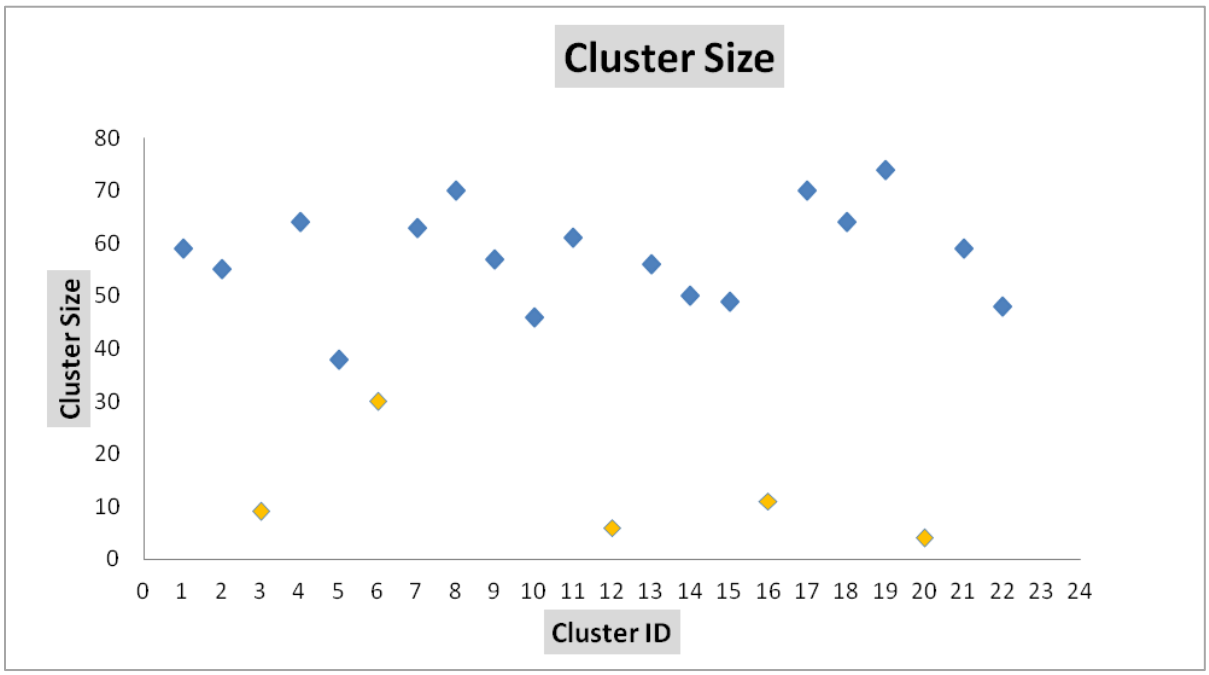

Figure 4.6: Size of each event cluster discovered from an experimental dataset using LSH

4.2.4.1 Event Trending based on Tweet Timestamp

The behaviour of the event over time is studied in this section. The timestamps of tweets from the consolidated cluster containing 1043 tweets are analysed to trend the hourly distribution of tweets spanning over a day, which is shown in Figure 4.7. In this thesis, the tweet timestamp is converted to Coordinated Universal Time (UTC) format as it is a standard format by which the world regulates the clock's time. According to Figure 4.7, the maximum number of tweets was posted between 16th and 17th hour of that day. It was predicted that the world would end on May 21, 2011 at 6PM. According to the Figure 4.7, we can see that the number of tweets is increasing until the 16th hour of the day after which it began to diminish. This type of trending graph illustrates the popularity of the event over time.

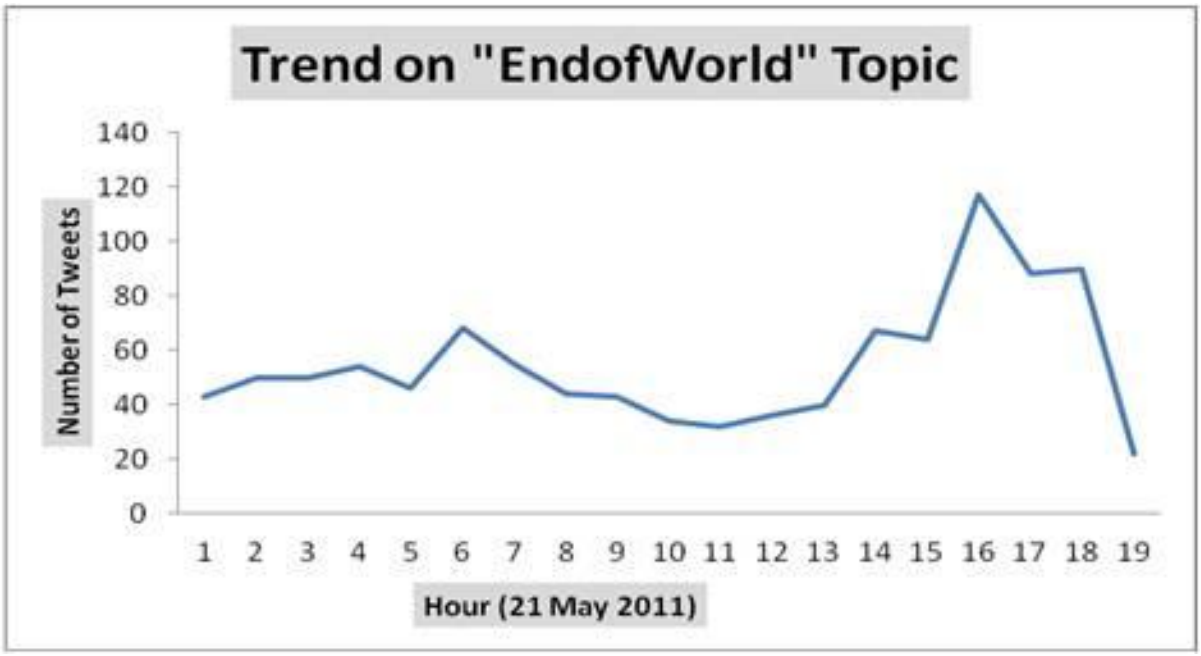

Figure 4.7: Trending event tweets based on time from an experimental dataset

4.2.4.2 Event Trending based on Tweet Geo-Locations

We aim to correlate geographical location with the way people perceive different topics. The geo-coordinates of tweets from the experimental dataset are used to infer their location information. Figure 4.8 shows the Google map for geographically distributed tweets of consolidated clusters which provides an idea on the type of event. Tweets specific to a topic tweeted all over the world have a global effect. From Figure 4.8, it can be observed that the "EndOfWorld" event had a global effect, which clearly indicates that many Twitter users around the world participated in the discussion of that event and majority of the tweets were tweeted in the North American region. From experimental study, it is observed that trending the event by geographical location helps to understand the global impact of the significant events across the world.

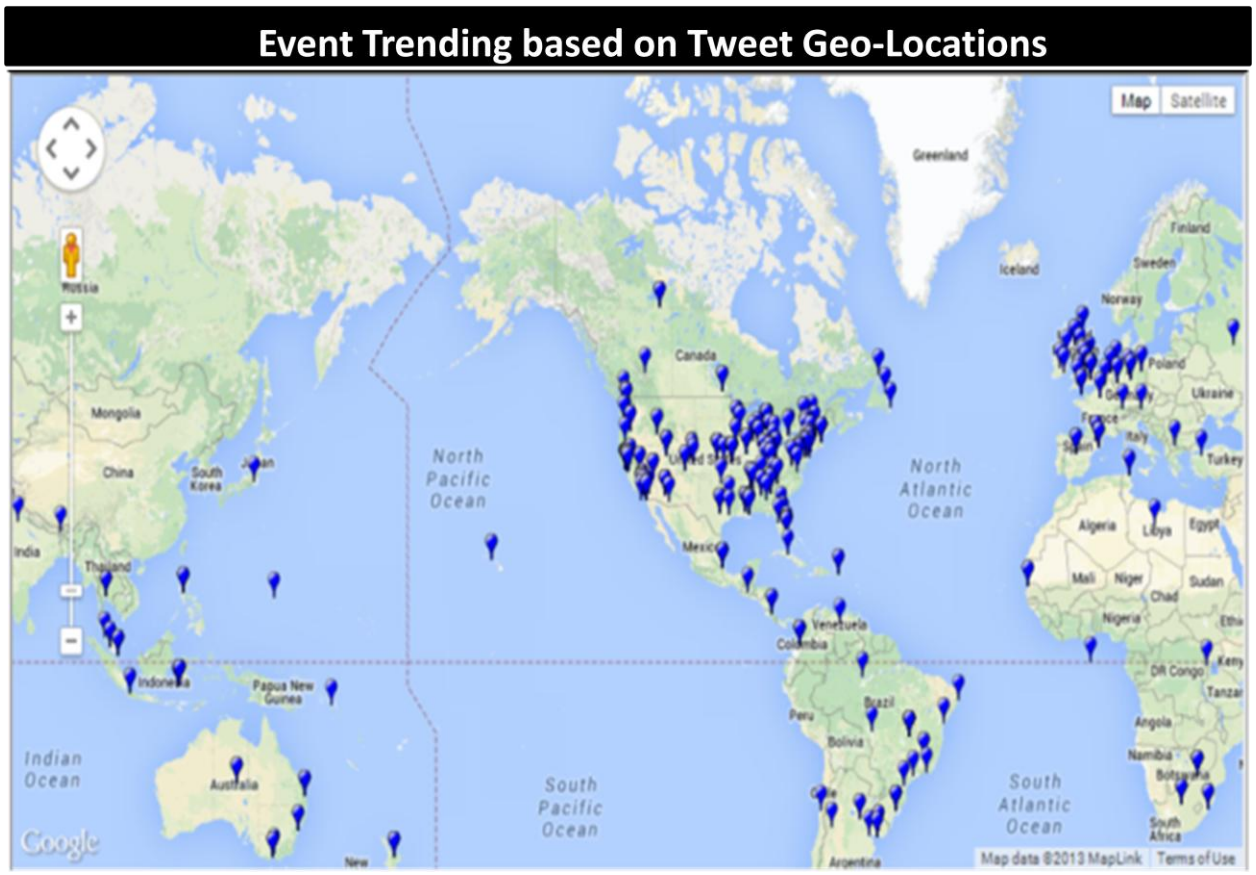

Figure 4.8: Trending event tweets based on geo-location from an experimental dataset

4.2.4.3 Consolidated Event Cluster Details

The event summary, including the number of unique users and the number of tweets in the consolidated cluster, is reported in Figure 4.9. A tag cloud or word cloud is a visual representation for text data which gives greater prominence to words that appear more frequently in data source. In this thesis, a set of highly weighted terms, the most frequently appearing terms from the consolidated cluster is used to form a tag cloud. The term weight for each term appearing in the cluster is assigned by computing its average document frequency as described in lines 3-5 of Algorithm-2. Table 4.6 shows the sample terms, including hashtags, in the consolidate cluster with their weights. For example, the cumulative average document frequency of the term ‘world’ is 17.0672 and the hashtag '#endoftheworldconfessions' is 2.7301 which are computed from 22 clusters containing 1043 tweets. The importance of terms is displayed through positive correlation with the font size. As shown in Figure 4.9, the keywords ‘world’, ‘end’, ‘#endoftheworldconfession’, ‘#rapture’ etc. are frequently appearing unigrams related to the

"EndOfWorld" event. The identified event is not spam as it is found as an existing wikipage in Wikipedia (http://en.wikipedia.org/wiki/2011_end_times_prediction). From our sample of a single day's tweets, we see 1002 unique Twitter users tweeted 1043 tweets about the "EndOfWorld" topic.

Table 4.6: Sample Tokens

Token Type	Token Weight
Terms	('world', 17.0672), ('end', 16.0144), ('itaposs', 2.8231), ('iaposm', 2.0447), ('time', 1.3120), ('rapture', 1.2651),('amp', 1.2409), ('tomorrow', 1.2279),('day', 0.9119)
Hashtags	('#endoftheworldconfessions', 2.7302), ('#rapture', 1.2795), ('#endoftheworld', 1.1715), ('#iftheworldendsonsaturday', 0.7065), ('#theworldends', 0.2669)

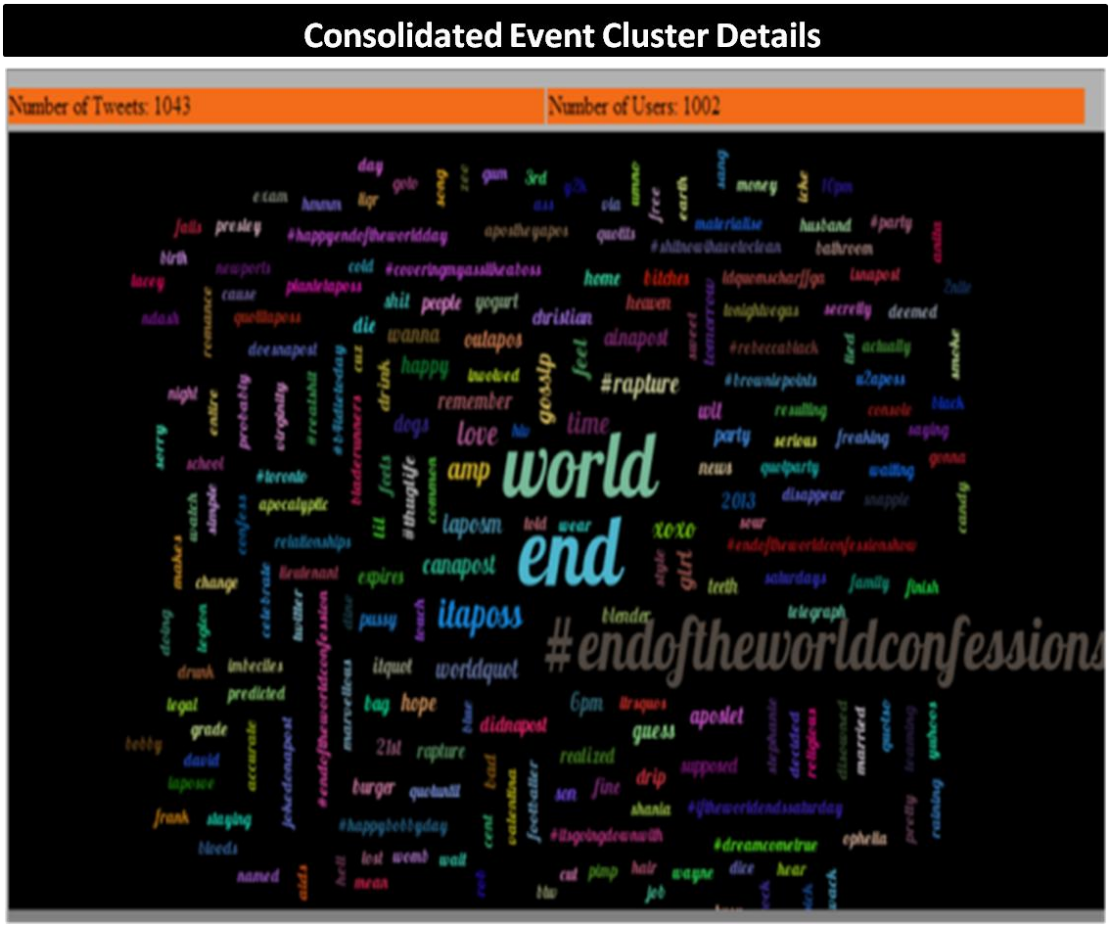

Figure 4.9: Consolidated event cluster details

From Figure 4.6 to Figure 4.8, the behaviour of the event is illustrated by tracking the clusters' size, time and geo-locations. Trending details are stored in a MySQL database in order to maintain their history. The proposed framework in this thesis is not only describes an approach to find the event in social media but also demonstrates an effective way for trending them.

4.2.5 Dimensionality Reduction

Tweets are posted in unstructured language with many typos and grammatical errors which increase the growth in dimension of tweet vectors. Preparation of the tweet feature vectors in high dimensional vector space is one of the key challenges in this research. Figure 4.10 shows the linear growth in dimension of unique features encountered in the experimental Twitter data corpus.

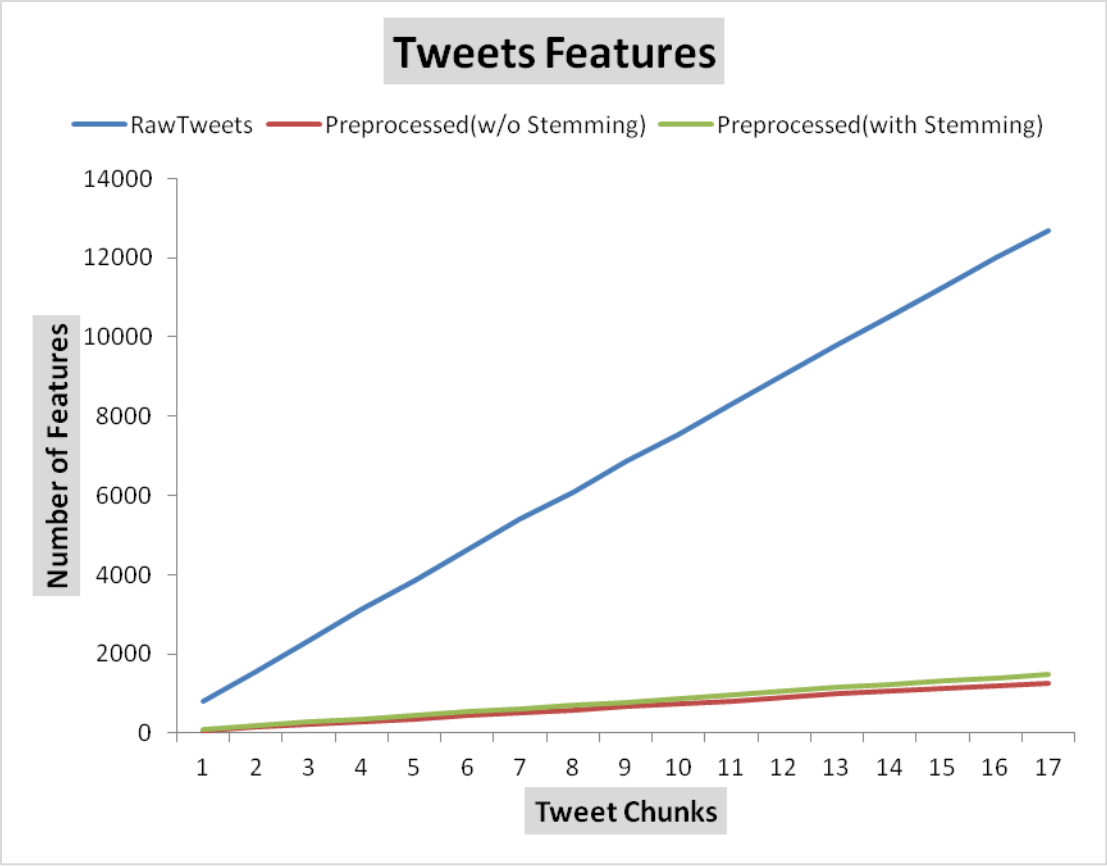

Figure 4.10: Linear growth of tweets' features for each chunk

The stemming is a process of removing the common morphological endings of each token [28]. For example, a stemmer changes the term "discovering" to "discover". The stemmer does not show any significant impact when the good cluster technique is applied [70]. According to the experimental result shown in Figure 4.10, it is observed that the dimensions have not been significantly reduced using stemming process. Thus, stemming is not engaged in this thesis when the tweet feature vector is prepared. The preprocessing task in the proposed framework is produced the less features compared to the number of features appears in the original given experimental Twitter dataset. We applied Charikar's family of hash functions to reduce the tweet feature vector into K-bit signature array of much smaller dimension. The traditional clustering algorithm does not have the flexibility of reducing the feature vector's dimension. Thus, the proposed framework is easily adoptable for social media dataset.

Chapter 5

Conclusions & Future Work

Twitter produces high dimensional data which often contains noisy, unstructured and incorrect texts messages. The Twitter dataset used in this thesis is divided into multiple chunks. This study deals the problem of creating a tweet feature vector in high dimensional data by using a static dictionary constructed for each chunk using an incremental TF-IDF technique. Charikar's family of hash functions are applied to reduce high dimensional tweet feature vectors into K-bit signature array of much smaller dimension.

Locality sensitive hashing, state-of-art technique is suitable for analysing the large-scale social media data because of its capability to fast processing of real-time data. In this thesis, each chunk goes through the process of discovering the tweet clusters using the LSH technique. Clusters are labeled using frequency-based feature selection method and their details are stored in a MySQL database. A novel part of the proposed framework in this thesis is to find an interesting event by matching its keywords on cluster labels and trending it based on time, geo-locations and cluster size. This study explored the various sets of LSH parameter: number of Permutations P, signature length K and the probability of finding nearest neighbour with a cosine similarity T. This study found that a set of LSH parameters P=23, K= 17 and T=0.005 produced the highest quality of clusters in terms of NMI and hence, as a result, they are retained for clustering the tweets in all chunks. GAAC is used as the golden-truth clustering technique in this research. The quality of clusters discovered using LSH and K-means is compared with those discovered using GAAC and it is observed that LSH attained 12.5% and 16.6% of purity and NMI respectively greater than K-means.

Since the proposed methodology leverages the LSH technique to find the tweet clusters, it has a significant advantage that the number of clusters need not be supplied in advance. Therefore, it acts as an unsupervised model. The prefix tree data structure is used in this study to improve the speed of LSH technique by prefix matching of tweet signature vector to find its nearest neighbour.

The runtime required for cluster discovery process is significantly improved by using the prefix tree data structure in LSH technique. LSH took only 12.99% of the average running time required for K-means and was consistently completed in approximately about 0.3 seconds during the complete cluster discovery process. It is also observed that LSH based cluster-discovery algorithm outperforms K-means while maintaining the quality of clusters.

The contributions of this study are outlined in detail as follows:

- It proposed a novel framework to find an interesting event by matching its keywords on cluster labels and trending it based on time, geo-locations and cluster size by using the existing methods that were not used before for the purpose of event detection and trending.
- Creating tweet feature vectors map with a dictionary is a challenging task for a growing Twitter data corpus as it increases the dimension of vector size and it is difficult to fit on a document vector space model due to the limitation of memory. The problem of creating the feature vector in high dimensional data is solved by using a static dictionary constructed using an incremental TF-IDF technique.
- This research work examined and confirmed that the LSH technique requires less running time for the cluster discovery process compared to a K-means clustering algorithm for an experimental Twitter dataset
- The proposed cluster-discovery process using LSH was sped up by using a prefix tree data structure while maintaining a better quality of clusters compare to a traditional K-means clustering algorithm.
- An approach for selecting the value for LSH parameters such as number of Permutations , signature length and the probability of finding nearest neighbour with a cosine similarity

is clearly shown in this thesis work. It could be made available to other users who are doing similar research.

The limitations of this research work are as follows:

- The proposed LSH technique is compared with other traditional algorithms based on quality of clusters (Purity & NMI) retrieved and time required for discovering the tweet clusters. Since the problem of event detection is dealt through clustering techniques where each cluster containing a set of tweets represents an event, we constrain our thesis performance metrics to the two mentioned above and they are not evaluated using other metrics such as precision and recall.
- Social media has a large volume of data containing both non-trivial and spurious events. An additional task would have been performed to find the truly interesting events by filtering the spurious events from a set of detected events with the help of a Wikipedia like knowledgebase database. It is not been included in this thesis due to the time constraints.

In future work, we would like to include the following tasks:

- The appearance of event may differ from one social media website to another. The set of attributes defining an event might be changed for each social media website as it records the data on its own way. For example, Facebook allows posting of images. The attribute set can be increased while we analyze the same event across multiple social media websites. This would help us to explore the events in greater depth and result in the generation of more useful information. Henceforth, we would like to enhance our approach with additional attributes and apply it on multiple social media websites to see the effect of the same events appearing on different social media platforms and record their attributes.
- Only tweets in English language are considered for the clustering in this thesis which can be extended in future to support multiple languages.
- During implementation, event detection and trending was performed for a specific event. However, future research can be done to highlight the list of all events presence in a

given dataset. Finding possible events by analyzing all tweet clusters discovered using LSH can also be an opportunity to return the list of events ordered by cluster-size.

- Currently event trending was displayed based on time, geo-location and cluster size. However, the proposed methodology can be extended to develop a more sophisticated trending system to generate event trends for user queried keywords. For example, Google Trends provides the time series graphs of search terms queried by users across the globe using the Google search engine.

Appendix I

List of Stopwords

a	beings	face	Higher	many	opening	see	though	why
about	best	faces	Highest	may	Opens	seem	thought	will
above	better	fact	Him	me	Or	seemed	thoughts	with
across	between	facts	Himself	member	Order	seeming	three	within
after	big	far	His	members	ordered	seems	through	without
again	both	felt	How	men	ordering	sees	thus	work
against	but	few	however	might	orders	several	to	worked
all	by	find	i	more	Other	shall	today	working
almost	c	finds	if	most	Others	she	together	works
alone	came	first	important	mostly	Our	should	too	would
along	can	for	in	mr	Out	show	took	x
already	cannot	four	interest	mrs	Over	showed	toward	y
also	case	from	interested	much	P	showing	turn	year
although	cases	full	interesting	must	Part	shows	turned	years
always	certain	fully	interests	my	Parted	side	turning	yet
among	certainly	further	into	myself	parting	sides	turns	you
an	clear	furthered	is	n	Parts	since	two	young
and	clearly	furthering	it	necessary	per	small	u	younger
another	come	furthers	its	need	perhaps	smaller	under	youngest
any	could	g	itself	needed	place	smallest	until	your
anybody	d	Gave	j	needing	places	so	up	yours
anyone	did	general	just	needs	point	some	upon	z
anything	differ	generally	k	never	pointed	somebody	us	
anywhere	different	get	keep	new	pointing	someone	use	
are	differently	gets	keeps	newer	points	something	used	
area	do	give	kind	newest	possible	somewhere	uses	
areas	does	given	knew	next	present	state	v	
around	done	gives	know	no	presented	states	very	
as	down	go	known	nobody	presenting	still	w	
ask	downed	going	knows	non	presents	such	want	
asked	downing	good	l	noone	problem	sure	wanted	
asking	downs	goods	large	not	problems	t	wanting	
asks	during	got	largely	nothing	put	take	wants	
at	e	great	last	Now	puts	taken	was	
away	each	Greater	later	nowhere	q	than	way	
b	early	Greatest	latest	number	quite	that	ways	

List of Stopwords (Continue)

back	either	group	least	numbers	r	the	we
backed	ended	grouped	less	o	rather	their	well
backing	ending	grouping	let	of	really	them	wells
backs	ends	groups	lets	off	right	then	went
be	enough	h	like	often	room	there	were
became	even	had	likely	old	rooms	therefore	what
because	evenly	has	long	older	s	these	when
become	ever	have	longer	oldest	said	they	where
becomes	every	having	longest	on	same	thing	whether
been	everybody	he	m	once	saw	things	which
before	everyone	her	made	one	say	think	while
began	everything	here	make	only	says	thinks	who
behind	everywhere	herself	making	open	Second	this	whole
being	F	high	man	opened	Seconds	those	whose

Appendix II

Dictionary for Sample of 100 tweets

Iaposm	6pm	Sing	weaposll
#endoftheworldconfessions	People	Itaposs	finale
End	Finally	Confession	results
Predicted	Haha	Mean	saga
Night	Day	Hit	amazingly
#iftheworldendsonsaturday	Gonna	Forward	clean
Didnapost	Love	Fall	clocks
Timezones	Wearing	Camping	fails
Females	Amp	2011	deliver
Breakfast	#endoftheworld	Youaposre	light
Sun	Lol	Iaposve	son
Waiting	Poor	Rapture	expires
Hours	Mom	Worry	yogurt
Believe	World	Canapost	2013
Maybe	Saturday	Joke	#thuglife
Wine	Iaposd	Weekend	ainapost
Feel	Drunk	Looking	
Thataposs	Free	Weigh	
Life	Apparently	Residents	
#rapture	Donapost	Prediction	

Appendix III

List of MySQL tables used for Event Detection and Trending

Table Name: ***cluster_tweet***

Table Structure:

Field	Data Type
ClusterId	varchar(1500)
TweetId	int(11)
Geo	varchar(1500)
Name	varchar(1500)
Timestamp	varchar(50)

Table Name: ***cluster_hashtags***

Table Structure:

Field	Data Type
ClusterId	varchar(1500)
Hashtags	Text
Wgt	Text

Table Name: ***cluster_singletons***

Table Structure:

Field	Data Type
ClusterId	varchar(1500)
Singletons	text
Wgt	text

Appendix IV

Ethics Application - No REB Review Required

rebchair@ryerson.ca <rebchair@ryerson.ca> Tue, Sep 16, 2014 at 11:16 AM

To: kshakira@ryerson.ca

Re: REB 2014-306 Cluster-Discovery of Twitter Messages for Event Detection and Trending

Dear Shakira Banu Kaleel,

The Research Ethics Board has determined that your protocol does not require its review.

Thank you for submitting an application for ethics review and approval for the above noted project. Based on the information provided the project falls within Article 2.2 of the federal guidelines governing research ethics as the research relies exclusively on publicly available information and there is no reasonable expectation of privacy regarding this information, and so does not require Research Ethics Board review.
If you have any questions regarding your submission or the review process, please do not hesitate to get in touch with the Research Ethics Board (contact information below).
Record respecting or associated with a research ethics application submitted to Ryerson University.
Yours sincerely,
Lynn Lavallée, Ph.D.
Chair, Research Ethics Board
Associate Professor
Ryerson University EPH-200C
350 Victoria St., Toronto, ON
(416)979-5000 ext. 4791
lavallee@ryerson.ca
rebchair@ryerson.ca
http://www.ryerson.ca/research

Toni Fletcher, MA
Research Ethics Co-Ordinator
Office of Research Services
Ryerson University
(416)979-5000 ext. 7112
toni.fletcher@ryerson.ca
http://www.ryerson.ca/research

Appendix V

A Case Study Illustrating Cluster-Discovery of Twitter Messages

In this section, we use an example to illustrate the cluster-discovery process. Table 4.3 illustrates the sample of 100 tweets.

Table A-5.1: Sample Twitter Dataset

ID	Tweet
0	LOVE THIS END OF THE WORLD SAGA IT IS JUST AMAZINGLY INTERESTING TO SEE PEOPLE FALL FOR IT. HOW INTERESTING http://bit.ly/dXBJyZ
1	I LOVE THIS END OF THE WORLD SAGA ON MAY 21ST.IT AMAZINGLY INTERESTING TO SEE PEOPLE FALL FOR IT. HOW INTERESTING http://bit.ly/dXBJyZ
2	I AM SURE IT IS PAST 6PM SOMEWHERE IN THE WORLD SO IT IS NOT THE END OF THE WORLD, FALSE PROPHETS EVERY WHERE http://bit.ly/dXBJyZ
3	@AaronRodgers12 #endoftheworldconfessions. I really want to see the ring on ur finger n I will bow down in front of u, if u sign my ball!
4	#EndOfTheWorldConfessions i love God and i love my Mom. If you LOVE your Mom and God, R-T!
5	If we're ending, I'm ending at the bar darling! "@Geli: Although a drive-thru Del Taco would be a good thing #incaseworldends"
6	#endoftheworldconfessions I am gossip girl. You know you love me xoxo
7	#iftheworldendsonsaturday at least I spent the last day sippin martini's with my favorite girls, Got 2 kiss Turbo, ... http://tmi.me/aroCW
8	the only end of the world confession im waiting for is from Tommy from Martin. I got $300 saying that nigga never had a job!
9	@nicolebelisle tomorrow night... #dancingtiltheworldends
10	Haha yes! RT @RyanSeacrest: #iftheworldendsonsaturday we'll have no @oprah finale, no @americanidol results, no (cont) http://tl.gd/ajbl5g
11	Didn't someone say today was suppose to be the end of the world...ha!
12	May 21, 2011, & Other #JudgmentDays That Have Come & Gone: http://abcn.ws/lG14eq #judgement #endoftheworld #ItsNotTheEndOfTheWorld
13	It must be the end of the world. I’m Waiting for a train at the Bedford stop after being in williamsburg and... http://tumblr.com/x3t2m4u3v2
14	#iftheworldendsonsaturday we'll have no @oprah finale, no @americanidol results, no @ladygaga album? That's just poor scheduling
15	Text from mom: "Some bible guru predicted 2 mor o @ 8pm 2 b the end of the world just in case know we luv u and did the best we could"
16	Who's Justin Beiber? “@2Spotter: Totally not ready for the world to end today..I haven't seen Justin Beiber in concert yet...#EndOfTheWorld”
17	The sun is shining, birds are chirping... What a beautiful day for the end of the world. Bring it on, #rapture.
18	Oh so the end of the world is set to Americas watch? That's a coincidence
19	Tonight's plan: Berries and wine in blender. Blender contents in drip bag. Begin drip. Wait for end of world. Empty bag. Repeat.
20	Y'all more worried about the end of the world than living your life right..

ID	Tweet
21	@Photos_in_Chile Sediment on the bottom of a bottle does not consitut the end of the world.Great shots at this point is up to your camera
22	Can't think of a rapture joke? It's not the end of the world. ;)
23	#endoftheworldconfessions Not looking for a record contract but I can sing. The one person to confirm this is Stacy Brown. lol #justaskher
24	Lawd let me get off these #endoftheworldconfessions because I'm actually done confessing I think I'm just "talking" now.... lol
25	#endoftheworldconfessions I struggle with who I want to be fashion wise. Edgy? Conservative? Bohemian? So I just go by my mood that day!
26	Last #endoftheworldconfessions tweet wasn't really a confession was it? lol I meant to say I hate looking at them! But I LOVE yall though...
27	Best one yet ? RT @Tinasintunes: If you can't think of a rapture joke, don't worry, its not the end of the world.
28	jerry west joins the GS @warriors ?!?...maybe there's something to this end of the world talk.
29	About 3.5 hours until the end of the world starts in Kiritimati
30	Saturday's end-of-world is the third predicted for May http://bit.ly/mQrsPi #lifestyle #news
31	Billboards in Chile predict end of the world: Santiago, Chile (Reuters) - Billboard ads in Santiago warn passers... http://bit.ly/kYWLw7
32	Looking forward to a weekend full of flying and sailing... oh and the end of the world. #fb
33	@carolhanna i'm doing Economics. I don't finish till thursday #endoftheworld
34	#endoftheworldconfessions and #myraptureplaylist are both trending. Really people?
35	I don't know about you, but I am about to attend a Rapture Party! #iftheworldendsonsaturday #peoplearesilly #thisisnews?
36	If the end of the world doesn't happen tomorrow...don't worry. It's not the end of the world. LOL! I'm so funny.
37	End of the world. (@ 1050 Lenox Park Pool w/ @marlenawbz) [pic]: http://4sq.com/k9uuSD
38	#rapture tweets are oddly reminiscent of govt shutdown tweets. Lets hope this is the last end of the world this year...
39	#iftheworldendsonsaturday I get a chance to wear my Air Jordan #SpaceJams finally lol
40	#iftheworldendsonsaturday then @TheReal_NuNu has to go back home lol
41	#iftheworldendsonsaturday I'ma drop my Mixtape Next Friday lol
42	#endoftheworldconfessions i dont wanna die , theres my confession , and i still have to meet @justinbieber and fall in love (L)
43	@rickwine what would you recommend as the best Niagara wine to sip during the rapture at 6:00 pm today a end of world send off, wine blog...
44	if this was the end i`d tat @SDOTB `s logo on my ass #endoftheworldconfessions ;-p haha
45	RT @pastormark: “It's the end of the world as we know it and I feel fine.” - REM // ME too! =)
46	Top 10 End of The World Prophecies http://ti.me/kAgcio
47	@LittleMomaaay 12am na. haha boring ngaa tae end of the world naa dw ah. hahahaha 6pm dw xD
48	Just think, #iftheworldendsonsaturday we will finally get to find out if clouds are fluffy or itchy.
49	i thought the end of the world was supposed to be at 6AM today... stayed up for nothing..
50	So are we kiwis going to fake a loud end of the world for those in later timezones or what? #raptureisnexttogullibleinthedictionary
51	I'm in trouble #iftheworldends. I love me some Jesus but I think I've cursed the Phillies too much for my life to be right.
52	#iftheworldendsonsaturday alot of babies are gonna be born in February
53	#iftheworldendsonsaturday I bet Rebecca Black aint looking forward to the weekend this time

ID	Tweet
54	Mr camping: I know you feel absolutely gutted that you're an intrrnational laughing stock but don't worry: it's not the end of the world.
55	#iftheworldendsonsaturday , I'll have an extra three hours , yay for Vancouver over Ottawa :)
56	End of the world prediction fails to deliver as clocks hit 6 p.m. http://goo.gl/fb/5XdZx
57	End of the world prediction fails to deliver as clocks hit 6 p.m. http://goo.gl/fb/xYnPV
58	I love how the wikipedia page says "This article documents a current event." #EndOfTheWorld
59	"He's the heartbeat to this team" -@dwadeofficial on Udonis Haslem. #endoftheworldconfessions I'd hit it!
60	#endoftheworldconfessions @LodyLucci be producin some of the hottest shit hands
61	#endoftheworldconfessions I love @Chris_Broussard
62	:D “@GirlBug: @fashion_noble Apparently end of the world ...some biblical reference. Timezones make it a bit confusing to prepare :P”
63	#endoftheworldconfessions I was drunk last night.
64	So are is the fat woman going to sing? Are pigs going to fly? I mean what's gonna be the official start of the end of the world
65	Saturday Rockpile: It's Not the End of the World http://FANpeeps.com/-fVeb (Via @ROCKIESpeeps) #mlb #rockies
66	In honor of it being the end of the world and all, my husband made me breakfast!
67	RT @aecaldwell so apparently tomorrow's the end of the world? where have i been? ohhhh, that's right, participating in real life
68	SLAM! FIGHT! BRIGHT LIGHT! FEEL PRETTY PSYCHED! IT END OF WORLD AS DRUNK HULK KNOW IT! AND DRUNK HULK FEEL FINE!
69	Your LAST night b4 rapture. WTF. And more movies made into musicals. It really is the end of the world.
70	When does the market close? RT @Vaneska: Fresh soft-shell crab from St. Lawrence! Maybe it really is the end of the world.
71	End of the world, or just another Saturday? N.J. residents weigh in on predicted May 21 rapture http://bit.ly/k4MLsa
72	In light of #endoftheworld prediction for later today I've started tidying the garden and am wearing clean underwear - sufficient?
73	I'm at Rapture day, End of the World! (TIMES Square, New York) w/ 44 others http://4sq.com/jpKn8E
74	#iftheworldendsonsaturday then 25% off Modern Dino Apparel & a FREE gift goes to waste! 24HR SALE!! Use Code:ALIVENKICKEN @ checkout! <3MD
75	End of the world, or just another Saturday? N.J. residents weigh in on predicted May 21 rapture: From rabbis to ... http://bit.ly/k4MLsa
76	@CSNStucky is REALLY a White Sox fan. #endoftheworldconfessions
77	#EndofTheWorldConfessions ....the world ain't ending son, my yogurt expires in 2013 ... #ThugLife
78	i've never understood the concept behind a "converted- hoe" lol I mean, u still a hoe wit modifications #endoftheworldconfessions
79	really tho, #iwannaslap the dude that created this madness today. let's see if he predicted that. #endoftheworldconfessions
80	“@josephFAZ: I'm done fuckin with all females..... Unless there rocking a Gucci fanny pack.... #endoftheworldshit @KtSmokesHydro” :) ? u
81	somebody somewhere is having an END OF THE WORLD PARTY...doin like those folks did in Independence Day. smh, poor lil tink tinks!
82	Hoping everyone is having a pleasant end of the world, I am going to go shopping for end of the world sales... ^MC
83	@God I know your busy with the end of the world and all but can Fresno get some sun please.
84	I really don't believe in the end of the world, but I'm not bothering to clean anything this weekend just in case #rapture

ID	Tweet
85	#sincetheworlddidntend I'm gonna eat this amazing breakfast
86	end of the world....what would i like to say.....im really not a drug addict
87	By the 26th song of the elementary school talent show, I was praying for the end of the world, believe me.
88	GUYS- If you're involved with a female, cut it off, end it, tell her she sucks & become free and single today! DO IT! #EndOfTheGirlWorld
89	(1/2) #MyManicDepression is telling me that I just want today to be the end of the world ONLY for females in relationships, which means...
90	RT @andersoncooper: The ridiculist RT @heighmichael: @andersoncooper So how will you be celebrating the end of the world?
91	@DerekBrunsonMMA Ha! I didn't come up with #iftheworldendsonsaturday. I'd prefer #IfOprahEndsTheWorldWhenSheEndsHerShow #Oprahcolypse
92	@SEXY_FLEXY #iftheworldendsonsaturday I would have wanted to make out with @SEXY_FLEXY. Ha!
93	#rapture Epic Fu - End of the World http://www.youtube.com/watch?v=FuQRmtBq-ls
94	#EndofTheWorldConfessions i'm not wearing any pants
95	#EndofTheWorldConfessions ....the world ain't ending son, my yogurt expires in 2013 ... #ThugLife
96	What China lookin like? #iftheworldendsonsaturday they should all be dead already.
97	MAY 21, 2011 END OF THE WORLD PREDICTION OF HAROLD CAMPING FAILED, DECEMBER 21, 2012 NEXT http://bit.ly/moJ8UZ
98	... I'm still here! Y'all good?? #endoftheworldheadcount <3
99	Shuttle crew to take closxe-up look at damaged tile: NASA ordered Endeavour's crew to take an unusual close-up lo... http://bit.ly/jAL8GX

The dictionary of the sample tweets of Table A-5.1 is built using incremental TF-IDF and is shown in Appendix II based on which the dictionary size is 76.

The cluster of each tweet is identified by applying the processes detailed in Steps 1-3 described in Chapter 3 Section 3.1. In this section, the result of each step for the below sample tweet is explained in greater detail.

ID: 0

Tweet:

LOVE THIS END OF THE WORLD SAGA IT IS JUST AMAZINGLY INTERESTING TO SEE PEOPLE FALL FOR IT. HOW INTERESTING http://bit.ly/dXBJyZ

The language detection score for the above tweet is 0.904762 and is allowed for further processing.

The list of stopwords is included in Appendix I. The set of features are extracted from the above tweet after removing the stopwords and URL is shown below:

Tweet:

LOVE ~~THIS~~ END ~~OF~~ ~~THE~~ WORLD SAGA ~~IT~~ ~~IS~~ ~~JUST~~ AMAZINGLY ~~INTERESTING~~ ~~TO SEE~~ PEOPLE FALL ~~FOR IT.~~ ~~HOW~~ ~~INTERESTING~~ ~~http://bit.ly/dXBJyZ~~

The resultant tweet features:

{'end', 'love', 'people', 'saga', 'amazingly', 'fall', 'world'}

The tweet feature vector is generated in Step-1 of Chapter 3 Section 3.1 and the output for the above shown tweet is:

Tweet Feature Vector: [('end', 1), ('love', 1), ('people', 1), ('saga', 1), ('amazingly', 1), ('fall', 1), 'world', 1)]

The K-bit signature for each tweet is computed in Step-2 of Chapter 3 Section 3.1. The 17-bit signature for the above tweet is shown below and the details of the computational step are explained further:

Signature: bitarray('00000111000101000')

Let us consider the following to compute the first bit of 17-bit tweet signature:

- Hash function: Floor[(a.x+b)/p] where random values for a,b and prime number for p are chosen as p=79, a=5 and b=59.
- The m-dimensional random vector, such that each dimension is drawn from gaussian distribution with mean 0 and variance 1 where m=79.

{0: 0.2880236091883922, 1: -0.024169848777495763, 2: 0.06279525312432852,
3: 0.032659590828312046, 4: -0.006691224328133619, 5: 0.04233210769558055,
6: -0.039400575417235295, 7: -0.017233564028582254, 8: -0.042220130349593606,
9: -0.08936045966315218, 10: 0.03914398108603037, 11: -0.07170234223525442,
12: 0.09052813845817041, 13: -0.05163204658465051, 14: -0.012173734906141308,
15: 0.07716699520790953, 16: -0.176148028187156, 17: 0.16965152250075582,
18: 0.14062873024197217, 19: 0.04802977871813637, 20: 0.09673641305445048,
21: 0.00937918747699141, 22: 0.13363123596088147, 23: -0.029300111055126414,
24: -0.06330658698981516, 25: 0.11891207737106015, 26: -0.11329259753050239,
27: -0.020671579110751685, 28: -0.11206602186878074, 29: 0.06760472660148267,
30: 0.08210657243962496, 31: 0.14351065177709188, 32: -0.09698354786870844,
33: -0.16721212842905664, 34: 0.06387871905214342, 35: 0.012413428905722482,

36: -0.013938422603104137, 37: -0.27270768355981306, 38: 0.06547092066390159,
39: -0.04384727536710721, 40: 0.020250286761466025, 41: 0.1392514890251581,
42: -0.045389597113516394, 43: 0.048822140304740615, 44: -0.04988234916434659,
45: 0.0559406180831005, 46: 0.01710501404458232, 47: -0.05918321771964802,
48: -0.11328770353979561, 49: 0.045794651282216346, 50: 0.01759761463660221,
51: -0.006306336213121755, 52: -0.16880331441932298, 53: 0.11196002830143635,
54: -0.10863179982413235, 55: -0.12478786157562766, 56: -0.02052468253224801,
57: -0.07887330829565463, 58: -0.25578677852373516, 59: 0.26920276957040257,
60: -0.09621689790726412, 61: 0.018468532673647203, 62: 0.1597463387800809,
63: 0.05315651816714169, 64: 0.2231175194768114, 65: -0.09317827157524601,
66: 0.011942088550919664, 67: -0.08721818066925108, 68: -0.07379730703896488,
69: 0.00727145566805136, 70: -0.15740847486894377, 71: -0.07446819704174572,
72: -0.1031555914359891, 73: -0.1580989543702044, 74: 0.1900883673188416,
75: -0.10981057345479606, 76: 0.204655041479839, 77: -0.07135366814819553,
78: -0.18273103559191536}

- The tweet feature vector is mapped with a dictionary is shown below. For example, the dictionary index for a feature ‘amazingly’ is 64.

{64: 1, 33: 1, 2: 1, 46: 1, 21: 1, 26: 1, 63: 1}

The hash function is applied on mapped vector as shown below:

x:64, a:5, b:59, p:79 → 63

x:33, a:5, b:59, p:79 → 66

x:2, a:5, b:59, p:79 → 69

x:46, a:5, b:59, p:79 → 52

x:21, a:5, b:59, p:79 → 6

x:26, a:5, b:59, p:79 →31

x:63, a:5, b:59, p:79 →58

Now, the above values are mapped to Gaussian distribution and equation (5) from Chapter 3 Section 3.1.2 is applied to compute the sum as follows:

Total = SUM(0.0531565181671, 0.0119420885509, 0.00727145566805, -0.168803314419, -0.0394005754172, 0.143510651777, -0.255786778524)

= **- 0.248109954197**

Since the returned value is less than 0, the bit is set to 0. The above concept is extended to the remaining bits for the 17-bit signature.

The following table shows the parameters applied to the sample tweets shown in Table A-5.1. The values for LSH parameters such as number of permutations, signature length and probability of finding nearest neighbour with a cosine similarity are assigned based on the result shown in Table 4.5.

Table A-5.2: List of LSH Parameters

Parameter Name	Value
Number of Permutations	23
Signature length	17
Probability of finding nearest neighbour with a cosine similarity	0.005
Dictionary Size	76

Table A-5.3 lists the clusters discovered using the proposed Algorithm-1. Cluster_4 from Table A-5.3 is a singleton cluster and is ignored for the purpose of event detection and trending.

Table A-5.3: List of Clusters

Cluster-ID	Tweet-ID
Cluster_0	{0, 1, 2, 4, 6, 8, 11, 13, 15, 16, 17, 18, 19, 20, 21, 22, 23, 26, 27, 28, 29, 30, 31, 32, 36, 37, 38, 43, 44, 45, 46, 47, 49, 50, 54, 56, 57, 62, 64, 65, 66, 67, 68, 69, 70, 71, 73, 75, 81, 82, 83, 84, 86, 87, 88, 89, 90, 93, 97}
Cluster_1	{3, 25, 34, 42, 59, 60, 61, 63, 76, 77, 78, 79, 94, 95}
Cluster_2	{5, 24, 33, 51, 80, 85}
Cluster_3	{7, 10, 14, 35, 39, 40, 41, 48, 52, 53, 55, 74, 91, 92, 96}
Cluster_4	{9}
Cluster_5	{12, 58, 72}

Table A-5.4 shows the top 20 highly weighted terms of each cluster on which the search related to keywords of event is performed. From the sample data, it is found that cluster_0 belongs to the event “EndOfWorld” after matching with keywords of the event and Table A-5.5 shows the resultant cluster related to the event.

Table A-5.4: Top 20 Weighted Terms of Clusters

Cluster-ID	Collection of Unigrams
Cluster_0	{end,world,love,rapture,predicted,itaposs,feel,mom,day,iaposm,saturday,looking,lol,prediction,worry,donapost,wine,saga,weekend,clocks,6pm}
Cluster_1	{love,expires,yogurt,2013,world,son,ainapost,quotconverted,fashion,dont,iapos d,people,hands,confession,sign,understood,wit,lol,sox,ring}
Cluster_2	{iaposm,love,doing,weaposre,cursed,lol,females,breakfast,rocking,quotgeli,thursday,actually,till,amazing,darling,phillies,confessing,life,finish,unless}
Cluster_3	{lol,results,finally,finale,weaposll,lookin,cont,iaposd,extra,rapture,fluffy,moder n,prefer,dead,sale,born,codealivenkicken,ottawa,amp,scheduling}
Cluster_5	{amp,gone,love,prediction,eventquot,documents,wikipedia,current,2011,wearin g,garden,quotthis,started,sufficient,iaposve,tidying,article,underwear,light,clean }

Table A-5.5: Highly weighted Terms of Cluster Related to Event

Cluster-ID	Collection of Unigrams
Cluster_0	{end,world,love,rapture,predicted,itaposs,feel,mom,day,iaposm,saturday,looking,lol,prediction,worry,donapost,wine,saga,weekend,clocks,6pm,#endoftheworld confessions,#rapture}

Bibliography

[1] M. R. Morris, J. Teevan and K. Panovich, "What do people ask their social networks, and why?: A survey study of status message and behavior," in *Proceedings of the SIGCHI Conference on Human Factors in Computing Systems,* Atlanta, Georgia, USA, 2010, pp. 1739-1748.

[2] J. Allan, "Topic detection and tracking," in , J. Allan, Ed. Norwell, MA, USA: Kluwer Academic Publishers, 2002, pp. 1-16.

[3] J. G. Fiscus and G. R. Doddington, "Topic detection and tracking evaluation overview," in *Topic Detection and Tracking*, J. Allan, Ed. Norwell, Massachusetts: Kluwer Academic Publishers, 2002, pp. 17-32.

[4] New York Magazine (May 11,2011), *A Conversation with Harold Camping, Prophesier of Judgment Day,* http://nymag.com/daily/intel/2011/05/a_conversation_with_harold_cam.html, Last visited: Dec 15, 2014

[5] Goffard, Christopher (21 May 2011), *Harold Camping is at the heart of a mediapocalypse*, http://articles.latimes.com/2011/may/21/local/la-me-rapture-20110521, Last visited: Dec 15, 2014

[6] Family Radio (May 21, 2011), *The end of the world is here! Holy god will bring judgement day on May 21,2011*, http://web.archive.org/web/20110608223300/http://www.familyradio.com/graphical/literature/judgment/judgment.html, Last visited: Dec 15, 2014

[7] J. Lin and D. Ryaboy, "Scaling big data mining infrastructure: the twitter experience," *SIGKDD Explor.Newsl.*, vol. 14, pp. 6-19, apr, 2013.

[8] G. Mishne, J. Dalton, Z. Li, A. Sharma and J. Lin, "Fast data in the era of big data: Twitter's real-time related query suggestion architecture," in *Proceedings of the 2013 ACM SIGMOD International Conference on Management of Data*, New York, USA, 2013, pp. 1147-1158.

[9] J. Allan, V. Lavrenko, D. Malin and R. Swan, "Detections, bounds, and timelines: UMass and TDT-3," in *Proceedings of Topic Detection and Tracking Workshop,* 2000, pp. 167-174.

[10] Y. Yang, T. Pierce and J. Carbonell, "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,* Melbourne, Australia, 1998, pp. 28-36.

[11] H. S. Packer, S. Samangooei, J. S. Hare, N. Gibbins and P. H. Lewis, "Event detection using twitter and structured semantic query expansion," in *Proceedings of the 1st International Workshop on Multimodal Crowd Sensing*, Maui, Hawaii, USA, 2012, pp. 7-14.

[12] T. Sakaki, M. Okazaki and Y. Matsuo, "Earthquake shakes twitter users: Real-time event detection by social sensors," in *Proceedings of the 19th International Conference on World Wide Web*, Raleigh, North Carolina, USA, 2010, pp. 851-860.

[13] S. Petrovic, M. Osborne and V. Lavrenko, "Streaming first story detection with application to twitter," in *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, Los Angeles, California, 2010, pp. 181-189.

[14] G. Salton and M. J. McGill, *Introduction to Modern Information Retrieval*. New York, NY, USA: McGraw-Hill, Inc, 1986.

[15] d. m. boyd and N. B. Ellison, "Social Network Sites: Definition, History, and Scholarship," *Journal of Computer-Mediated Communication*, vol. 13, pp. 210-230, 2007.

[16] Karen Wickre (Mar 21 2013), *Celebrating #Twitter7*, https://blog.twitter.com/2013/celebrating-twitter7, Last visited: Dec 15, 2014

[17] Twitter.(2014), *REST API v1.1 Resources*, https://dev.twitter.com/docs/api/1.1, Last visited: Dec 15, 2014

[18] P. Indyk and R. Motwani, "Approximate nearest neighbors: Towards removing the curse of dimensionality," in *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing*, Dallas, Texas, USA, 1998, pp. 604-613.

[19] K. L. Clarkson, "An algorithm for approximate closest-point queries," in *Proceedings of the Tenth Annual Symposium on Computational Geometry*, Stony Brook, New York, USA, 1994, pp. 160-164.

[20] A. Broder, "On the resemblance and containment of documents," in *Proceedings of the Compression and Complexity of Sequences 1997*, 1997, pp. 21.

[21] Hao Li (Jul 2010). *LSH: Theory and Application. JDL Seminar on Large-Scale Indexing and Search, July, 2010*. Available: http://www.haoli.me/uploads/9/0/6/7/9067538/jdl-lsh_seminar_lh.pdf, Last visited: Dec 15, 2014

[22] M. S. Charikar, "Similarity estimation techniques from rounding algorithms," in *Proceedings of the Thiry-Fourth Annual ACM Symposium on Theory of Computing*, Montreal, Quebec, Canada, 2002, pp. 380-388.

[23] B. Van Durme and A. Lall, "Online generation of locality sensitive hash signatures," in *Proceedings of the ACL 2010 Conference Short Papers*, Uppsala, Sweden, 2010, pp. 231-235.

[24] D. Ravichandran, P. Pantel and E. Hovy, "Randomized algorithms and NLP: Using locality sensitive hash function for high speed noun clustering," in *Proceedings of the 43rd Annual Meeting on Association for Computational Linguistics,* Ann Arbor, Michigan, 2005, pp. 622-629.

[25] K. Y. Kamath and J. Caverlee, "Content-based crowd retrieval on the real-time web," in *Proceedings of the 21st ACM International Conference on Information and Knowledge Management,* Maui, Hawaii, USA, 2012, pp. 195-204.

[26] M. T. Goodrich, R. Tamassia and M. H. Goldwasser, *Data Structures and Algorithms in Python.* Wiley Publishing, 2013, ISBN:1118290275 9781118290279

[27] E. Fredkin, "Trie Memory," *Communication of the ACM,* vol. 3, pp. 490-499, Sep 1960.

[28] C. D. Manning, P. Raghavan and H. Schutze, *Introduction to Information Retrieval.* New York, NY, USA: Cambridge University Press, 2008.

[29] D. R. Cutting, D. R. Karger, J. O. Pedersen and J. W. Tukey, "Scatter/gather: A cluster-based approach to browsing large document collections," in *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval,* Copenhagen, Denmark, 1992, pp. 318-329.

[30] T. Brants, F. Chen and A. Farahat, "A system for new event detection," in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Informaion Retrieval,* Toronto, Canada, 2003, pp. 330-337.

[31] Facebook. (2014). *Quickstart for Graph API.* Available: https://developers.facebook.com/docs/graph-api/quickstart/, Last visited: Dec 15, 2014

[32] J. Bollen, H. Mao and X. Zeng, "Twitter mood predicts the stock market," *Journal of Computational Science,* vol. 2, pp. 1-8, 2011.

[33] H. Choi and H. Varian, "Predicting the Present with Google Trends," *Economic Record,* vol. 88, pp. 2-9, 2012.

[34] T. Preis, H. S. Moat and H. E. Stanley, "Quantifying trading behavior in financial markets using Google Trends," *Scientific Reports,* vol. 3, 2013.

[35] R. Parikh and K. Karlapalem, "ET: Events from tweets," in *Proceedings of the 22nd International Conference on World Wide Web Companion,* Rio de Janeiro, Brazil, 2013, pp. 613-620.

[36] B. Sriram, D. Fuhry, E. Demir, H. Ferhatosmanoglu and M. Demirbas, "Short text classification in twitter to improve information filtering," in *Proceedings of the 33rd International ACM SIGIR Conference on Research and Development in Information Retrieval,* Geneva, Switzerland, 2010, pp. 841-842.

[37] J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman and J. Sperling, "TwitterStand: News in tweets," in *Proceedings of the 17th ACM SIGSPATIAL International Conference on Advances in Geographic Information Systems*, Seattle, Washington, 2009, pp. 42-51.

[38] O. Ozdikis, P. Senkul and H. Oguztuzun, "Semantic expansion of hashtags for enhanced event detection in twitter," in *Proceedings of the 1st International Workshop on Online Social Systems*, Istanbul, Turkey, 2012.

[39] A. Ahmed, Q. Ho, J. Eisenstein, E. Xing, A. J. Smola and C. H. Teo, "Unified analysis of streaming news," in *Proceedings of the 20th International Conference on World Wide Web*, Hyderabad, India, 2011, pp. 267-276.

[40] Max Planck Institute for Informatics (2014) *YAGO2s: A High-Quality Knowledge Base*. Available: http://www.mpi-inf.mpg.de/yago-naga/yago/, Last visited: Dec 15, 2014

[41] J. Hoffart, F. M. Suchanek, K. Berberich, E. Lewis-Kelham, G. de Melo and G. Weikum, "YAGO2: Exploring and querying world knowledge in time, space, context, and many languages," in *Proceedings of the 20th International Conference Companion on World Wide Web*, Hyderabad, India, 2011, pp. 229-232.

[42] S. Yardi and D. Boyd, "Tweeting from the Town Square: Measuring Geographic Local Networks," in *Proceedings of the 4th International AAAI Conference on Weblogs and Social Media*, Washington, 2010.

[43] M. Naaman, H. Becker and L. Gravano, "Hip and Trendy: Characterizing Emerging Trends on Twitter," in *Journal of the American Society for Information Science and Technology.*, vol. 62, pp. 902-918, May, 2011.

[44] H. Kwak, C. Lee, H. Park and S. Moon, "What is twitter, a social network or a news media?" in *Proceedings of the 19th International Conference on World Wide Web*, Raleigh, North Carolina, USA, 2010, pp. 591-600.

[45] I. P. Cvijikj and F. Michahelles, "Monitoring trends on facebook," in *Proceedings of the 2011 IEEE Ninth International Conference on Dependable, Autonomic and Secure Computing*, 2011, pp. 895-902.

[46] R. Weber, H. Schek and S. Blott, "A quantitative analysis and performance study for similarity-search methods in high-dimensional spaces," in *Proceedings of the 24rd International Conference on very Large Data Bases*, 1998, pp. 194-205.

[47] A. Dasgupta, R. Kumar and T. Sarlos, "Fast locality-sensitive hashing," in *Proceedings of the 17th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, San Diego, California, USA, 2011, pp. 1073-1081.

[48] S. Petrovic, "Real-time event detection in massive streams," Ph.D. dissertation, Institute for Language, Cognition and Computation, School of Informatics, University of Edinburgh, UK, 2012.

[49] P. Indyk, R. Motwani, P. Raghavan and S. Vempala, "Locality-preserving hashing in multidimensional spaces," in *Proceedings of the Twenty-Ninth Annual ACM Symposium on Theory of Computing*, El Paso, Texas, USA, 1997, pp. 618-625.

[50] M. Datar, N. Immorlica, P. Indyk and V. S. Mirrokni, "Locality-sensitive hashing scheme based on p-stable distributions," in *Proceedings of the Twentieth Annual Symposium on Computational Geometry*, Brooklyn, New York, USA, 2004, pp. 253-262.

[51] Z. Yongwei, L. Bicheng and G. Haolin, "Bag-of-Visual-Words Based Object Retrieval with E2LSH and Query Expansion," in *Instrumentation, Measurement, Circuits and Systems*, vol. 127, pp. 713-725, 2012.

[52] M. Slaney and M. Casey, "Locality-Sensitive Hashing for Finding Nearest Neighbors," *Signal Processing Magazine, IEEE*, vol. 25, pp. 128-131, Mar 2008.

[53] B. Sharifi, M. Hutton and J. K. Kalita, "Experiments in microblog summarization," in *Proceedings of the 2010 IEEE Second International Conference on Social Computing*, Minneapolis, MN, USA, 2010, pp. 49-56.

[54] S. B. Kaleel, M. AlMeshary and A. Abhari, "Event detection and trending in multiple social networking sites," in *Proceedings of the 16th Communications & Networking Symposium*, San Diego, California, 2013, pp. 5:1-5:5.

[55] S. B. Kaleel and A. Abhari, "Cluster-discovery of Twitter messages for event detection and trending," *Journal of Computational Science*, vol. 6, pp. 47, 2015.

[56] M. A. Russell, *Mining the Social Web: Analyzing Data from Facebook, Twitter, LinkedIn, and Other Social Media Sites.* O'Reilly Media Inc, 2011.

[57] L. Uden, L. S. L. Wang, J. M. C. Rodrguez, H. Yang and I. Ting, *The 8th International Conference on Knowledge Management in Organizations: Social and Big Data Computing for Knowledge Management.* Springer Publishing Company, Incorporated, 2013.

[58] R. Řehůřek and P. Sojka, "Software framework for topic modelling with large corpora," in *Proceedings of LREC 2010 Workshop New Challenges for NLP Frameworks*, Valletta, Malta, 2010, pp. 46-50.

[59] NLTK Project (2014), *Natural Language Toolkit*, https://www.nltk.org, Last visited: Dec 15, 2014

[60] Python (2014), *Pyenchant 1.6.5*, https://pypi.python.org/pypi/pyenchant/1.6.5, Last visited: Dec 15, 2014

[61] Python (2014). *Index of Packages*, https://pypi.python.org/pypi/MySQL-python, Last visited: Dec 15, 2014

[62] Python (2014), *bitarray 0.8.1*, https://pypi.python.org/pypi/bitarray/0.8.1, Last visited: Dec 15, 2014

[63] Biopython (2014), *Biopython: Introduction*, http://biopython.org/wiki/Main_Page, Last visited: Dec 15, 2014

[64] Python (2014), *15.3. time-Time access and conversions*, https://docs.python.org/2/library/time.html, Last visited: Dec 15, 2014

[65] Python (2014), *re-Regular expression operations*, https://docs.python.org/2/library/re.html, Last visited: Dec 15, 2014

[66] Python (2014), *pytagcloud 0.3.5*, https://pypi.python.org/pypi/pytagcloud/0.3.5, Last visited: Dec 15, 2014

[67] Python (2014), *pygmap 0.1.1*, https://pypi.python.org/pypi/pygmaps/0.1.1, Last visited: Dec 15, 2014

[68] Github (2011), *Kykamth (Krishna Kamath) - Github*, https://github.com/kykamath, Last visited: Dec 15, 2014

[69] SHARCNET (2014), *SHARCNET*, https://www.sharcnet.ca/my/front/, Last visited: Dec 15, 2014

[70] S. Goorha and L. Ungar, "Discovery of significant emerging trends," in *Proceedings of the 16th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, Washington, DC, USA, 2010, pp. 57-64.