Classification and Location Extraction of Harmful Algal Blooms from Microblogs

by

Pranjay Patil

(Under the Direction of Lakshmish Ramaswamy)

ABSTRACT

A well-known fact of the internet age is that online social media is accessed regularly by an increasing number of users. Such platforms enable its users to create, share and spread information with ease in real time. In this research, we explore the possibility of harnessing this information to identify incidents of harmful algal blooms on water bodies. We target the information shared by the users of a popular micro-blogging service known as Twitter. We propose a way to annotate the slew of information obtained from these platforms to create ground truth and for analysis. We develop and test a platform that can extract and separate tweets that report incidents of harmful algal blooms. We apply Machine Learning and Natural Language Processing techniques to identify locations of such reported incidents if mentioned in the body of the tweet. An exploratory quantitative analysis of the collected data is also presented.

INDEX WORDS: Machine Learning, Online Social Media, Blue Green Algae, Algal Blooms

Classification and Location Extraction of Harmful Algal Blooms from Microblogs

by

Pranjay Patil

B.E., Nagpur University, India, 2014

A Thesis Submitted to the Graduate Faculty

of The University of Georgia in Partial Fulfillment

of the

Requirements for the Degree

Master of Science

ATHENS, GEORGIA

2018

Classification and Location Extraction of Harmful Algal Blooms from Microblogs

by

Pranjay Patil

Major Professor:	Lakshmish Ramaswamy
Committee:	Thiab Taha
	Deepak Mishra

Electronic Version Approved:

Suzanne Barbour
Dean of the Graduate School
The University of Georgia
May 2018

DEDICATION

For Mom and Dad. Your constant support has allowed me to sail through this journey with ease.

Acknowledgments

All endeavors become easily achievable with able guidance and support. I would like to take this opportunity to express my gratitude and respect for my adviser Dr. Lakshmish Ramaswamy for his constant support and guidance. I would also like to thank him for two excellent courses that I took under him namely Distributed Systems and Advanced Information Systems. I thoroughly enjoyed learning the contents of both of these courses. Moreover, I would like to appreciate his guidance for international students and helping them succeed. Secondly, I would like to thank Dr. Deepak Mishra for providing valuable insights during this work. His insights allowed fine tuning of the system described here. His guidance has made working on the CyanoTracker project an enjoyable experience. I would like to thank Dr. Thiab Taha for being part my advisory committee and for the support. I would also like to acknowledge and thank Dr. Shuchendra Bhandarkar for his help with the Machine Learning part. I would like to thank Dr. Vinay Boddula for his help from the very beginning of this work. This work would not have achieved its goals without Vinay's help. Finally, I would like to express my deepest appreciation and respect for all the faculty and staff at the Computer Science Department of The University of Georgia. I feel truly blessed to be a part of this esteemed organization.

TABLE OF CONTENTS

Page

Acknowledgments v

LIST OF TABLES viii

LIST OF FIGURES ix

CHAPTER

1 Introduction 1

1.1 Harmful Algal Blooms (HABs) 1

1.2 Twitter and It's Characteristics 2

1.3 Major Contribution 3

2 Background and Related Work 6

2.1 CyanoTracker Project 6

2.2 Related Work 8

3 System Architecture and Overview 17

3.1 Pipeline Architecture 17

3.2 Data Collection 18

3.3 Data Cleaning 19

3.4 Labeling Data for Supervised Learning 20

3.5 Text Classification and Mining Techniques 22

4 Scalable Labeling Using MinHash 24

4.1 The Problem of Labeling 24

4.2 Label Propagation 26

5 Extracting Reports and Location of Blooms 31
5.1 Machine Learning for Classification of CyanoHAB Tweets 31
5.2 Location Extraction 36
6 Empirical Analysis and Experimental Evaluation 40
6.1 Quantitative Data Analysis 40
6.2 Concept Drift 46
6.3 Effectiveness of MinHash Based Labelling 49
6.4 Evaluation of Data Classification 54
6.5 Location Extraction Results 56
7 Conclusion 59
BIBLIOGRAPHY 60

LIST OF TABLES

4.1 Similarity Score and Percent Reduction for 20 Months 29
4.2 Similarity Score and Percent Reduction for Last 4 Months 29
6.1 Total Tweets Collected per Class 41
6.2 Number of Unique Users and Unique User Rate 43
6.3 Number of Reliable Users with Percentage and Rate 44
6.4 Number of Active User and Active User Rate 44
6.5 Percentage of Patron Users per Keyword 45
6.6 Top Words in Profile Description 46
6.7 Accuracy of Label Propagation using Similarity Scores 49
6.8 Number of Clusters for Each Keyword 50
6.9 Number of Outliers for Each Keyword 50
6.10 Classifier Performance 54
6.11 Total Locations Obtained for Each Keyword 58

LIST OF FIGURES

2.1 CyanoTracker Architecture 7
3.1 Architecture of the Pipeline 17
4.1 Determining Similarity Score Threshold 27
6.1 Total Tweets Collected per Month 41
6.2 Total Tweets Collected per Keyword and Class 42
6.3 Tweet Collection by Month 43
6.4 Concept Drift over 24 months of data 48
6.5 Percent of Tweets Assigned to Top 10 Largest Clusters 51
6.6 Number of clusters per threshold 52
6.7 Number of tweets assigned to clusters per threshold 53
6.8 Division of Data 54
6.9 Confusion Matrix for each Classifier 55
6.10 Bloom Locations Obtained from Reports 57

CHAPTER 1

Introduction

1.1 Harmful Algal Blooms (HABs)

Cyanobacterial Blooms have emerged as a major water quality issue in recent years. These blooms are a cause of concern for governments that rely on local water bodies for water supply and revenue from recreational activities. Authorities have to take special measures to treat affected water which creates economic impacts for the local governments [1]. According to the Centers for Disease Control and Prevention (CDC) website [1], algal blooms spread on top of the water and appear to be a foam-like surface. According to CDC, "A harmful algal bloom (HAB) refers to the fast growth of any phytoplankton (cyanobacteria and microalgae) that can cause harm to animals, people, or the local ecology". It further elaborates on the nature of the HABs adding that such algal blooms cover the surface of the water, blocking any sunlight from entering the water. Blooms deplete the oxygen levels in the water. Depleting oxygen levels result in fish kills in large numbers which further harms the local ecology when the plants and animals decay. Moreover, it informs that any organism that consumes fish or other aquatic fauna contaminated by the toxins can fall sick. Thus, HABs are not only harmful to the vegetation and marine life, but also to the animals that get in contact with affected organisms. HABs also have a significant economic impact. One study found that amount lost in revenue can range into millions of US dollars due to losses in fisheries and tourism [2].

[1] https://www.cdc.gov/habs/general.html

Given these impacts, it becomes essential to develop a system that can effectively monitor the lakes and other water bodies for the presence of such harmful blooms. One way to achieve this monitoring is to use sensors which can check the water quality and report the presence of cyanotoxins. Another way is for authorities to conduct regular field trips to collect water samples for measuring water quality. However, due to the limited range of such sensors and the cost involved in the deployment and maintenance, this method of monitoring is inefficient. Moreover, conducting field trips will not scale well due to the time involved and resources requirements [3].

In this thesis, we harness the extensive reach and near real-time nature of social media as a crowdsourcing platform. We target the users of the social networking site Twitter [2] for collection and analysis of data related to CyanoHABs. We expand upon the work by [4] studying the viability of using crowdsourcing in an environmental monitoring system. We expand this work by implementing components for scalable labeling of Twitter data and extracting locations of CyanoHABs.

1.2 Twitter and It's Characteristics

A well-known fact of the day is that never before in the history of humanity, was information sharing and consumption as easy as of today. People share information in the form of text, pictures, videos and a combination of these [5]. Social media services form a major platform for such sharing. The information varies in subject and encompasses topics like opinions, politics, sports, news, entertainment, weather, technology and more [6]. Twitter is a service where users can share limited character posts about a variety of topics; these posts are popularly known as Tweets. As of March 2018, each tweet can be a maximum of 280 English

[2] https://twitter.com/

characters [3]. On this service, people often use hash-tags, @ symbols for referring to other users and RT for indicating a re-tweet [7].

It is All About the Reach

Twitter has proliferated the social media space since it began, mainly owing to the increase in smart-phone use [8]. The number of Twitter users can easily exceed millions and is growing. People are increasingly engaging in the micro-blogging paradigm due to its ease of use and extensive reach [9]. Some governments are opting to alert its citizens of weather and other emergencies via Twitter, Maine state in United States is one example [4]. According to Demirbas, et. al, Twitter plays an essential role in the form of a stream of data. Here, people witnessing an event may post about an incident, and this information quickly spreads over their network. Moreover, people often obtain news via Twitter even before actual news channels started coverage [9]. The follower - following set-up makes it easier for people to engage in the conversation on a topic and thus causing an increased amount of information sharing [7].

1.3 Major Contribution

This work aims to provide social media analysis to CyanoTracker project at The University of Georgia. Joshi [4] presented a quantitative analysis of tweets about Harmful Algal Blooms. They analyzed the data in terms of seasonal variation in the number of tweets, the number of reports of HABs on Twitter, the number of relevant and irrelevant tweets obtained for various HAB related keywords, user characteristics, etc. They also trained a supervised learning classifier to extract reports of algal blooms from tweets. We expand this work by collecting and analyzing Twitter data for years 2015 and 2016.

[3] https://blog.twitter.com/official/en_us/topics/product/2017/Giving-you-more-characters-to-express-yourself.html

[4] http://www.maine.gov/portal/CAS/

Our major contributions are as follows,

- We present a scalable solution for labeling large amounts of textual data.
- We present an empirical study of Twitter data related to Harmful Algal Blooms with regards to the monthly variation of tweets for each keyword, the number of relevant, irrelevant and incident reports obtained from tweets, Twitter user characteristics, etc for the years 2015 and 2016.
- We present a study of Concept Drift in the tweets.
- We train four supervised learning algorithms to extract incident reports from the tweets and analyze their performance.
- We extract locations of algae bloom incidents mentioned in tweets and plot the locations on a map of continental United States.

Thesis Outline

Chapter 2 presents the architecture of CyanoTracker project with other research work related to this thesis.

Chapter 3 gives the detailed architectural overview of our system and the data processing step. It also provides overview of the machine learning component.

In Chapter 4, we propose a novel technique to label tweets into categories based on similarity scores.

In chapter 5, we present a detailed discussion of the Supervised Learning component. We also describe the location extraction component.

Finally, in chapter 6, we present the quantitative data analysis giving us insights into the obtained data. We discuss about a phenomenon known as concept drift. Concept drift tells us whether our models trained once can perform well on future data with similar accuracy.

We present the effectiveness of MinHash based labeling and the discuss the performance of machine learning algorithms. Lastly, we present the locations of algal blooms obtained from our data on the map of continental United States.

CHAPTER 2

Background and Related Work

In this chapter, we present the background of the CyanoTracker project, we also describe the related work where Twitter is used for disaster management and monitoring environment.

2.1 CyanoTracker Project

As mentioned before, Harmful Algae Blooms are detrimental to local ecology and economy. Cyanotracker project [1] at the University of Georgia aims to monitor algae blooms in lakes and other water bodies. It also aims to provide a platform for citizens to contribute news and reports about algal bloom incidents. To this effect, CyanoTracker has released mobile applications on Android and iOS platforms which can be used to share data about algae blooms [10].

As a part of CyanoTracker, sensors have been deployed on various lakes that monitor the water quality and relays this information to CyanoTracker servers. This data is then analyzed to decide whether an algae bloom exists on the lake [11].

Apart from developing a platform to facilitate citizen involvement in reporting HABs, researchers at CyanoTracker project are also involved in developing advanced computational techniques to effectively overcome challenges in cyanobacterial bloom monitoring. In [12] Boddula et al. have presented a system that mitigates the unavailability of remote sensing data due to cloud coverage over water bodies using a "Spatio-Temporal Mining" technique.

[1] http://cyanotracker.uga.edu/

CyanoTracker Architecture

The CyanoTracker architecture has three major parts: *Sensor Cloud, Community Cloud and Data Analytics Cloud*. The architecture by [3] is shown in figure 2.1. The Sensor Cloud and Community Cloud are data collection components that collect and feed data into the Analytics Cloud.

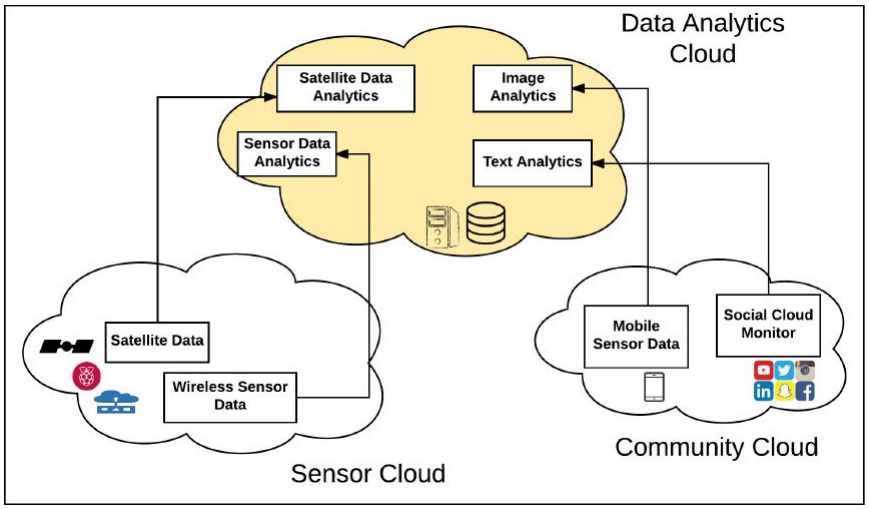


Figure. 2.1: CyanoTracker Architecture

- **Sensor Cloud:** The sensor cloud is a collection of sensors as described in [11]. They are inexpensive monitoring units that are deployed at various water bodies. These units provide measurement of water spectra at regular intervals and transmits the data to Analytics Cloud. This cloud also generates data from various satellites in form of digital images.

- **Community Cloud:** The Community Cloud provides a platform for general public and citizen scientists to take part in the reporting of algal blooms. Participants can volunteer to provide data in the form of images and bloom location using CyanoTracker mobile applications and via website. This cloud also collects data from social networks like Twitter and from news feeds to detect reports of algal blooms.

- **Data Analytics Cloud:** All the data collected from the other clouds are collected at the Analytics clouds. This component integrates the data and uses computational techniques to predict the presence of cyanoHAB and generate alerts.

This thesis contributes to the community cloud component of the architecture by providing data from social media.

2.2 Related Work

In this section, we present the concept of citizen science and some of the research done previously by research community that aligns closely with our goals.

2.2.1 Citizen Science

Citizen Science makes use of the public to accomplish the task of information gathering and reporting [13]. It also allows for cross-continental collection and analysis of information. Interested participants have, in the past, provided huge amounts of data for citizen science projects. Many a time, these amateur participants create the most impact for a project. [14]. Twitter has been used as a way to harness this "Wisdom-of-Crowds" in many systems [9]. Moreover, such crowd-sourced efforts allow mitigating the shortcomings of static sensors. They allow for a more wide range of coverage, thus mitigating the lack of infrastructure. They are energy efficient since we do not need to provide a constant supply of energy to a sensor [9]. Citizen science allows for more varied nature of data collection since they are not bound by sensor capacity. Systems that require human intervention and intelligence benefit greatly from citizen science [9].

Using Twitter data to gather data for our system will be the best option since we can get data not only from the residents, but also from the people visiting a natural water body. It is common knowledge that people visit natural features for recreational purposes and often

share their experience on social media. We thus expect to have a constant, if not increasing number of people sharing their experience of a harmful algal blooms.

2.2.2 Social Media and Environment

Social Media has been used extensively in environmental monitoring.

Sakaki, et. al [5] used Twitter data to implement an early warning system for earthquakes. This system is real time and uses tweets by Twitter users to determine location and time of earthquake. This system is able to act as a monitor for target events by assuming Twitter users as sensors. It acts as a real-time alert and notification system for earthquakes in Japan.

Sakaki, et. al [5] make use of Support Vector Machines (SVM) for classifying tweets into positive and negative class as part of the semantic analysis. Positive meaning a report of earthquake. They claim that tweets can be noisy. For example, a tweet can refer to previous earthquake which is a relevant tweet but not a report. They thus train the classifier to be able to understand such tweets. Sakaki, et. al [5] further report that they used three kinds of features for the classifier. These features used number of words combined with the position of the words in tweet, all the words as features, and words surrounding an event keyword, respectively.

Their system uses probabilistic models for earthquake identification and location estimation [5]. They use time series data for event detection. For each tweet they classify it into positive or negative class. After this, they use a probability threshold to determine event occurrence. If the probability of occurrence is greater than a threshold, they use the GPS location for calculating estimated location. Once an event is confirmed this way, this system can optionally send email to users. Sakai, et. al claim that this system was able to send alerts before JMA (Japan Meteorological Agency) system. On average they sent out alerts within minutes of event occurrence compared to 6 minutes for JMA. They were also able to detect 96% of earthquakes of JMA intensity scale of 3 or more [5].

Zheng et al. [15] have presented a system that can monitor surface water quality using social media. They claim that establishing water quality monitoring networks is expensive. Which in turn affects the number of sites that can be monitored. Thus it restricts the detection of unlawful contamination of water. They further claim that social media based volunteers can be helpful in providing data for gaps left by monitoring networks. According to Zheng et al. [15] "Volunteer Geographic Information (VGI)" can provide a sensing solution for water quality conservation. They argue that data obtained from VGI may not always be accurate as the volunteers are not always trained in performing scientific tasks. Although according to them, studies have found no significant difference in the data provided by volunteers and the data collected in the field. They mention that with proper guidance citizens can become a reliable source of information and that such guidance can be provided through the social media directly to the mobile devices of the volunteers.

In this system [15], the citizens share information in the form of pictures of water bodies. The participants describe the water quality based on color, smell, turbidity, and presence of floating objects. These reports are then shared on a website by the volunteers together with the location. These volunteers are ranked based on their contributions; the top contributors receive awards.

The described system [15] recruits volunteers in two ways. Firstly, university students recruited for this research share the system with their friends. Their friends can then join the system and start contributing reports. Secondly, professional citizens are recruited. These citizens are often associated with environmental organizations. These citizens are tasked with monitoring specific locations.

To test the accuracy of VGI, the data obtained from the participants was validated against data obtained from stations across Yellow River. The researchers found a good correlation between these two data sources. Thus, they claim that VGI can be trusted for environmental data collection given that a large number of participants are actively involved in the process.

They claim that providing incentives for participation can improve environmental monitoring based on citizen science [15].

Kongthon et al. [16] present a case study of Twitter use during Thailand floods of 2011. This study presents interesting insights into the determination of the credibility of Twitter users. The authors claim that Twitter is growing rapidly and the likelihood of spreading of false information is also increasing. It is important to rank the users based on their credibility to create a system that can be used by the authorities to plan and manage disaster relief. The goal of the study is to analyze tweets and user characteristics to improve disaster response.

The authors [16] analyzed the type of information shared by the users over Twitter during the flood. They categorized the tweets into five categories namely "Situation Announcements", "Support Announcement", "Assistance Requests", "Information Requests" and Other. They found that during the event of a flood the most number of tweets were Situation Announcements. They also found that these tweets were posted by local community members thus the information was a first-hand account of the event.

To identify the popularity of the users, the authors [16] suggest using the number of followers as a metric. The authors claim that the number of followers may not reflect the credibility and impact of these users. They found that the users with the highest numbers of followers did not tweet much during the flood period and thus had a low number of re-tweets. They suggest that credibility of a user can be determined by the number of re-tweets to the user's tweets. They add that a large number of re-tweets can be an indicator that people find the user to be a reliable source of information. They conclude that people can follow these credible users during the event of natural disasters and the same can be used to obtain relevant information about an event [16].

2.2.3 Social Media and Disaster Response

In addition to environmental monitoring, social media is also used in various systems as an early warning system and for improving disaster response.

Musaev et al. [17] present a system called LITMUS. This system aims at detecting landslides using a combination of physical sensors and social media. They refer to social media users as "social sensors". The system performs various filtering steps on data from these social sensors and then combines the data obtained from the physical sensors for the same geo-location.

According to Musaev et al. [17], the filtering steps on social sensors are as follows. First, they download data from Twitter based on keywords such as "landslide" and "mudslide". Next, they exclude any negative stop words. Here, negative stop words refer to instances that are not related to a landslide as an environmental phenomenon. After this step, the system extracts the geo-location from the tweet based on the text of the item. They match words present in the text against names of previously known geo-locations. The authors [17] claim that this approach has some challenges like the mentioned location will sometime have a name similar to a verb. They provide examples like "Says" and "Goes" both of these are valid locations. To mitigate this problem, they perform "part-of-speech" tagging before using location identification algorithm. Finally, the system performs machine learning based classification of the tweets. They classify the text into relevant and irrelevant categories. To train this classifier, they used a set of labeled data of confirmed landslides. LITMUS also filters out specific items that contain URLs that are blacklisted by the authors.

According to the authors [17], the LITMUS system detected 42 landslide events; only 11 of these were reported by governmental agencies. The authors manually verified that each incident detected by the system was indeed an actual landslide event. The machine learning classifier performance was also shared by the authors, for Twitter, the classifier had around 0.8 recall and approximately 0.3 precision.

This system aligns closely with the CyanoTracker project. This study provides valuable insight into combining physical sensors and social media for environmental monitoring.

Guy et al. [18] present a system called TED (Twitter Earthquake Detector). This system combines data from scientific earthquake reports with Twitter data to analyze the possibility to create an early earthquake detection system. According to the authors, it takes around 2 to 20 minutes for collecting, analyzing and validating the seismic data, delaying the publication of alerts. But people present in the location of an earthquake are quick to post about the incident on social media.

Authors [18] state that the amount of interest demonstrated by the public is related to the increase in the number of tweets related to hazardous events. The system TED uses social media to capture such events and then potentially alert the users via social media itself. They suggest that this rise in social media activity for a particular incident can be used to inform the authorities of the possibility of an event before sensor data becomes available.

According to Guy et al. [18], the system is designed by extracting Twitter data related to specific keywords like earthquake, tsunami, etc. The system also uses USGS global earthquake stream and saves information like the magnitude, location, time, hypocenter, etc. The system also takes care not to alert the users of minor earthquakes which are not felt by the public. By doing this, the user is more appreciable of any future alert of importance. To determine whether an alert is to be issued, the authors [18] have identified a signification ratio. This ratio is determined based on the number of tweets being generated before and after an event. More tweets after an incident can trigger an alert. The alert informs the public of magnitude, the location of hypocenter, a map of tweets and other essential data. This data provides a first-hand account of the event to the users.

This paper [18] provides valuable insights into the challenges present in Twitter data. Authors suggest that Twitter data may not always be accurate, there is a lack of geo-location information in the tweets. They suggest that not all tweets containing the keywords relate to

earthquake incidents. Although there is a rise in the number of relevant tweets during the event of an earthquake. They conclude by claiming that the tweets about earthquakes correspond to earthquakes that were felt. They further add that the benefit of Twitter is the speed and reach to a large number of users. They claim that a marked increase in Twitter activity following an earthquake in California was available in 20 seconds on Twitter as compared to 3.2 minutes for USGS alert.

Twitter is also used for detecting wildfires. Power et al. [19] describe a system that acts as an emergency management tool for tracking fires. Their goal was to extract information from Twitter about fire events. They collect tweets from Australia and New Zealand. They issue alerts via email if an incident is identified by the system. The authors [19] built an SVM (Support Vector Machines) based classifier to identify tweets that report a fire incident. They determined 17 "root words" after processing the data and assigned intensity levels and a threshold value. If the occurrence of a word is higher than historically observed frequency, then an alert is issued by the system [19]. According to Power et al. [19], this alert is in the form of an email to registered users. For a red alert to be issued, this system requires at least two users to have tweeted about an incident. The email includes the tweets that triggered this alert which helps the users to decide the severity of the event. Moreover, the email provides a way for the users to explore the tweets, gives a location of the Twitter users and topics mentioned in the tweets.

To train the SVM classifier, power et al. [19] selected specific features from the text. These features included the number of words in the tweet, number of user mentions, number of hashtags followed by uni-gram and bi-gram word occurrences. They observed that the SVM classifier had an accuracy of 84.54% which improved the quality of the notifications generated by the system. The system produced a total of 42 warnings during three months of being active. Only 20 corresponded to real fire events. When the filter was applied, this number went down to 21 notifications, improving the notification accuracy to 78%.

Signorini et al. [20] have presented a study of Twitter for monitoring the public activity for swine flu and to track the disease itself. For this task, the authors collected data from Twitter based on search words related to influenza. Since one goal of the study was to capture public sentiment of the disease, the search words included words like "travel", "hygiene" in addition to words like "vaccine", "flu", etc which are disease related words. They processed the data and created a dictionary of the words occurring in the tweets. The authors calculated the daily and weekly usage of these dictionary words.

Based on this technique, Signorini et al. [20] found that the tweets related to H1N1 decreased as the number of cases of disease reports increased. According to the authors, this can be attributed to the fact that the general concern of the public declined as time progressed. Another observation by the study showed that in response to events of reports in media, the number of tweets related to countering influenza showed distinct spikes.

The authors [20] trained a Support Vector Regression model to estimate ILI (Influenza-like Illness) values from the Twitter data. They estimated weekly ILI values by the model using around 1 million tweets. Cross validation was used to measure the accuracy of the model. The model obtained good results and was able to accurately estimate the weekly ILI values as validated using CDC (Centers for Disease Control and Prevention) data. The average error was 0.28% with minimum of 0.04% and standard deviation of 0.23% [20].

Finally, Signorini et al. [20] trained a model to produce real time estimate of the ILI values. Since according to the authors, CDC values are available after 1 to 2 weeks, real time estimation can be valuable. In this study, the model had error of 0.37%, higher than earlier, but was able to approximate the reported ILI data closely.

2.2.4 Preparing Twitter Data for NLP

For using social media posts for analysis, it is important to first prepare the data by removing slang terms and other non-essential text. A study by Imran, et. al [21] presents a research

that demonstrates their technique to prepare data for NLP tasks such as training a classifier. They collected data from various crisis events around the world. They claim that social media data is often erroneous in terms of grammar. Data can contain slang, abbreviations, typing mistakes, etc. To mitigate this, they have presented a system to normalize "out-of-vocabulary" words [21]. Moreover, they trained three classifiers namely Support Vector Machines (SVM), Naive Bayes and Random Forests to test their method. They also trained a word2vec model based on 52 million Twitter messages.

This research by Imran, et. al [21] provides valuable insights into the nature of social media posts. Specifically their commentary on pre-processing is relevant to our work. Imran, et. al [21] present a way to normalize Twitter data. Normalization is a task of removing slang and other errors in the text as described above. They categorize normalization candidates into four categories as namely, (1) abbreviations of single words, (2) abbreviations of multiple words, (3) phonetic substitution and (4) multiple words joined together. This classification gives valuable insight into the kinds of pre-processing challenges involved in social media.

To identify the normalization candidates, Imran, et. al [21] built dictionary containing lexical variations. They found that for many words correct representation can be found by using "one edit-distance change". This technique performs one insertion, one deletion or substitution to achieve correct form. For this purpose, they trained a model based on most frequent words obtained from various corpora. They used Bayes theorem to predict the most probable correct word for an incorrectly typed word and restricting corrections to one edit distance [21]. This paper presented valuable related work to prepare data for NLP tasks.

In the next chapter, we present the overview of our system.

CHAPTER 3

System Architecture and Overview

In this chapter, we present an overview of our system architecture. We describe the steps in the data processing pipeline and provide an overview of the machine learning component of the system.

3.1 Pipeline Architecture

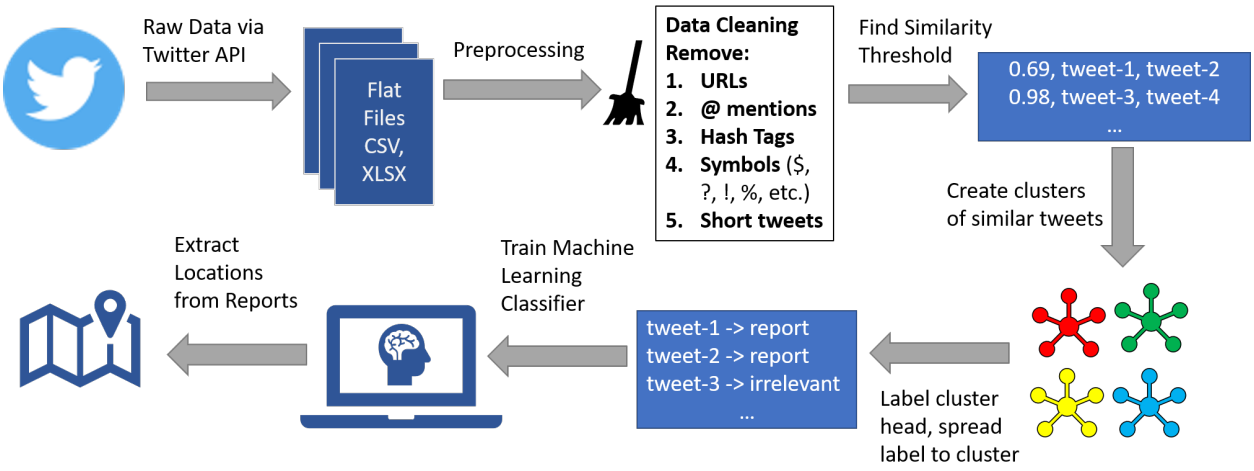


Figure. 3.1: Architecture of the Pipeline

The figure 3.1 shows the architecture of our pipeline. The pipeline begins with collecting data from the Twitter API. The data is stored into flat files like Microsoft Excel and CSV. Before labeling the data, we preprocess it to remove all the URLs, Twitter mentions, Hash-tags, symbols and we also drop the short tweets. We present a detailed description of this pipeline in the sections that follow.

3.2 Data Collection

We collected data using Twitter's REST API [1]. This API can be used to get data that has a specific keyword or a combination of words. The data thus obtained is in JavaScript Object Notation (JSON) format and is composed of the tweet and other meta-data. We parsed this data and stored only the tweet, username and date of the tweet, as required for our analysis. Collecting data from Twitter comes with its own challenges mainly because of the rate limit imposed by Twitter [2]. Due to this rate limit, only a certain number of requests are allowed in a window of time. Thus, the data collection is slow and the effort has to be monitored continuously for proper data collection.

We configured the requests to the Twitter API to return only those tweets that have at least one of the words from the bullet list. Only English language tweets were requested. We will refer to these words as "keywords".

- Algae Bloom
- Algal Bloom
- Blue Green Algae
- Cyanobacteria
- Red Tide
- Toxic Algae

After collecting the data, we parsed it and removed unnecessary meta-data and kept only the fields as described above. We stored the majority of the data in excel files. At this point, the data comprises of unprocessed text in different columns of excel sheet. Each column represents one of tweet, username and date. Note that at this point, a tweet can contain

[1] https://developer.twitter.com/en/docs/api-reference-index
[2] https://developer.twitter.com/en/docs/basics/rate-limiting

hash-tags, URLs, Twitter mentions and occasional emoticons. We need to process and clean these tweets before the next step.

3.3 Data Cleaning

The data cleaning step removes the following from the tweet body:

1. Unicode characters that cannot be converted to ASCII
2. All symbols (e.g. # , ' " ? - \)
3. All URLs
4. All twitter mentions (e.g. @username)

This step also converts all the words to lower case. After this step, the tweet contains only numbers and English alphabet characters. We used a Python package named Openpyxl [3] to read data from excel files. Once processed, we stored the tweets in a comma-separated values file in `<tweet, username, year-month>` format. We obtained six files, one for each keyword mentioned above. We removed any tweets that were six words or less since we observed that any tweet with less than six words does not, in most cases, provide any vital information.

We collected data in two phases. In first phase we got data for 20 months starting from January 2015 to August 2016. In the second phase, we collected and processed data for 4 more months starting from September 2016 to December 2016, adding to a total of 24 months.

After processing the raw data, we move to the most crucial step in data preparation. We devised a way to label a large amount of textual data without having to label all the instances individually. This technique is based on MinHashing as introduced in section 3.4.1.

[3] https://openpyxl.readthedocs.io/en/stable/

3.4 Labeling Data for Supervised Learning

One of the requirements of Supervised Learning is to provide class labels for each tweet in our data. Since we have a large number of tweets, it is time consuming to hand label each tweet. To mitigate this problem, we devised a scalable labeling approach. In this approach we use MinHashing to find similar tweets. We then create clusters of the similar tweets and label one tweet from each cluster. Since all other tweets are similar to this one tweet, we can assign the same label to each tweet in the cluster. The section 3.4.1 explains the MinHash technique in detail.

3.4.1 MinHashing

We used MinHashing based similarity score for calculating the similarity of two tweets as presented by Leskovec, et. al [22]. One point to note is that our goal is to compare the tweets on the basis of the words used and not the meaning of the tweet. Such document comparison is used in detecting Plagiarism, duplicate detection for web pages and more [22].

There are two ways to compare two instances of text, one using Jaccard Similarity and other using MinHashing [22]. But, it has been found that using Jaccard Similarity on a large number of documents is computationally expensive and time consuming. MinHasing offers a faster alternative to Jaccard Similarity. It has been proved that MinHash based similarity score is a good approximation of similarity and is either close or equal to Jaccard Similarity score [22].

According to Leskovec, et. al, MinHash algorithm works by creating MinHash signatures of the input text. Each of these signatures is fixed in length. The signature is created by using the MinHash algorithm. We create the MinHash signature for each of the input texts and then compare the signature to obtain the similarity score. Comparing the signatures is faster

than comparing the intersection to obtain Jaccard Similarity due to the small size of the signatures.

Leskovec et. al explain the MinHash Algorithm as follows. To create a signature for an input, MinHash uses a set of hashing functions. The output of these hash function is an integer. MinHashing works by applying each hash function to all of the words in the input and generating an integer. It selects the minimum hash value generated and this value becomes the first component in the signature. Similarly, outputs of other hash functions become part of the signature. This process is repeated for all inputs. Thus, in the end we end up with MinHash signatures for each input. These signatures are then compared to calculate the similarity score [22].

k - SHINGLING

A variation of MinHash algorithm uses k-shingles of the input document [22]. According to Leskovec et. al, shingling is a technique where combinations of words appearing in the input are used in addition to single words to create the input set for MinHashing. The variable k determines the number of words to combine to form a shingle. For example, consider the text `I want to travel the world`. In this case if $k = 2$, then the shingles formed would be, `"I want", "want to", "to travel", "travel the", "the world"`. This technique is found to improve the identification of similar documents [22].

However, this approach adds an overhead to decide the optimal k which according to [22] depends on the dataset. Using a sub-optimal k can reduce the performance of the algorithm [22]. Thus, it requires experimentation with different values of k. Moreover, we believe that computing shingles also adds an overhead into the algorithm. Given a large number of tweets, computing k-shingles for each and every combination of word in the tweet will certainly

require more computation compared to using a bag of words approach. Finally, storing k-shingles requires more memory and on large datasets this introduces memory overhead [22]. Thus, using k-shingling presents a trade off between efficiency and accuracy.

To summarize MinHash algorithm we can say that,

- MinHashing provides a faster alternative to Jaccard Similarity.
- The similarity scores provided my MinHash algorithm are comparable to Jaccard Similarity.

Chapter 4 explains the details of scalable labeling using MinHash technique.

3.5 Text Classification and Mining Techniques

As described in the pipeline architecture diagram in figure 3.1, we use machine learning to separate tweets into two categories *Report and Irrelevant*. This section we describes the motivation for using text classification techniques to extract algal bloom reports from Twitter data. The major challenge lies in the fact that searching for an algae bloom report on Twitter would be equivalent to searching for a needle in a hack stack.

Twitter users tweet about a variety of topics. Extracting tweets related to algae bloom reports takes more than just searching Twitter using a keyword. The reason being the fact that people not only tweet about algae blooms incidents but also about algae bloom as a phenomenon. For example, a person might speak about the danger to sea life due to an algae bloom. This tweet will be a part of the search but does not report any incidents of algae bloom. Applying machine learning and natural language processing to the data can help us solve this problem. We trained a total of four supervised learning algorithms namely,

- Naive Bayes
- Random Forests

- Gradient Tree Boosting
- Logistic Regression with Naive Bayes Features

Once we have generated the training data using MinHashing, we proceed to train Supervised Learning algorithms. Each of these algorithms were trained on randomly selected 80% of all the data. We then tested the performance of these trained models on the rest of the 20% of the data. Chapter 5 describes these classification techniques in detail.

CHAPTER 4

Scalable Labeling Using MinHash

We saw the data collection and cleaning steps in chapter 3. In this chapter, we discuss the technique we use to label the tweets obtained from Twitter.

4.1 The Problem of Labeling

We collected data for 24 months as described in chapter 3. A total of 120300 tweets remained after processing. Labeling this data manually is time-consuming and inefficient. There were two options to label this entire data-set. One was to use some form of unsupervised learning technique. Second, we could use a semi-supervised approach to label this data. Since the machine learning-based classifier would learn from this data, it was important that the labels be accurate. According to Allahyari et al. [23] unsupervised algorithms are used to find the hidden structure in data. These algorithms generally try to identify unknown relationships between data instances. Algorithms like clustering can group similar text documents but are probabilistic and not strictly limited to clustering based on a well defined topic [23]. Since our goal was to classify our data into three well defined categories, we decided to go for a semi-supervised approach. We found a unique semi-supervised approach where a labeled subset of tweets can be used to label the remaining tweets.

This technique is based on MinHashing algorithm [22] as explained in section 3.4.1. In this technique, we compare two instances of text and generate a similarity score. The score ranges from [0.0, 1.0]. The higher the score, the more "similar" the tweets. We found after some experimentation that most tweets with higher similarity score conveyed similar information.

For example, consider the following tweets, the MinHash score is 0.92, which means the tweets are highly similar,

Tweet-1: `uga researchers identify name toxic cyanobacteria killing american bald eagles`

Tweet-2: `researchers identify name toxic cyanobacteria killing american bald eagles`

The only difference is the word "uga" in Tweet-1. These tweets are posted by different users and are categorized as retweets.

The following tweets have a MinHash score of 0.04,

Tweet-1: `army corps to reduce lake flows fueling florida algae bloom`

Tweet-2: `the red is described as an algae bloom on a nuku alofa beach maybe a consequence of the eruption but is it sure`

Thus, we can see that for two dissimilar tweets, the MinHash score is very low.

Another reason to choose MinHash algorithm is due to the nature of the dataset. All of the tweets fall into one of the following types,

- Original Tweet - A tweet posted for the first time or one without any retweets.
- Retweets - A tweet posted by multiple users without any change.
- Similar Tweets - A Retweet with some modification.

We observed that many tweets were either re-tweets or modified re-tweets and thus, two tweets that have high similarity score should carry the same label. Due to these reasons we decided to pursue the technique as described in the section 4.2.

k-means Clustering

An option to cluster data is to use an unsupervised algorithm. We experimented using k-means clustering algorithm [23]. In this algorithm, we create k number of clusters where k is a user-specified parameter. Here, each cluster ideally is formed of similar tweets [23]. Elbow method is widely used to determine the optimal value of k [24]. In elbow method, the k-means algorithm is applied to the data for different values of k and a graph is plotted for the percentage variance against the number of clusters [24]. The percentage variance is expected to drop at a higher rate for small number of clusters and then stabilize for some value of k. This value of k is suggested to be the optimal value for clustering [24].

In our experiments, we could not obtain an optimal k value. Moreover, for larger datasets we often ran into "out of memory errors" for k sizes of 2000 and over. Our experimental setup had 8 GB of memory on a cloud machine. We hypothesize that this could be a characteristic of our data and k-means might not be a suitable clustering algorithm in our case.

4.2 Label Propagation

Label propagation, as the name identifies, is a technique were we propagate the label for one manually labeled tweet to multiple tweets. The steps are as follows,

1. Decide score threshold
2. Form similarity clusters
3. Label cluster heads manually
4. Propagate label to entire cluster

4.2.1 Deciding Score Threshold

In this section we describe the steps to decide the score threshold. The goal is to determine a MinHash score which will act as the threshold for forming clusters. This threshold will be used as described in figure 4.1. Score threshold determines which tweets fall into the same cluster. As described earlier, MinHash Algorithm provides a score in the range of 0.0 to 1.0. We need to determine the best score, for each keyword, that can ideally,

(a) Create clusters where all tweets have the same meaning.

(b) Leave the least number of tweets unassigned.

We decided the score thresholds heuristically for each keyword as explained later in this section.

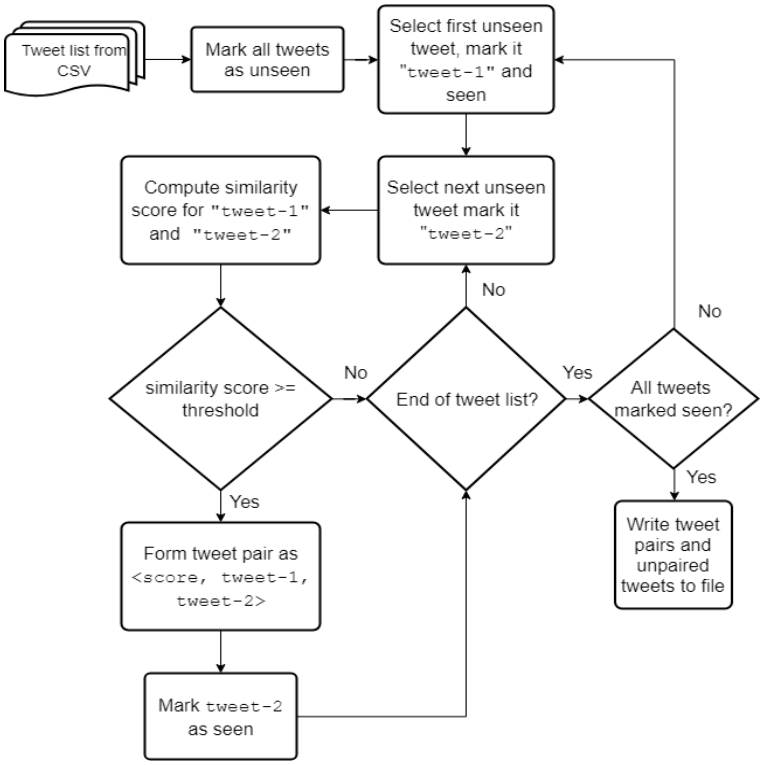


Figure. 4.1: Determining Similarity Score Threshold

The steps to determine similarity threshold are as follows,

(1) Start with 0.3 as the base threshold. Create tweet pairs as described in figure 4.1. The output is of the format `<score, tweet-1, tweet-2>`.

(2) Observe the tweet pairs formed in the current output for up to 0.7 similarity score and determine whether the paired tweets represent the same category of information. We found that any pair above similarity score 0.7 was highly similar and thus if lower score forms satisfactory pairs, then higher score will follow suit.

(3) If satisfactory pairs are not obtained, then increment the score threshold by 0.05 and repeat from step 2.

Following these steps we obtained similarity thresholds for each keyword. The table 4.1 and table 4.2 lists the thresholds obtained for each keyword and the reduction in the number of tweets to be labeled. As we can see, we have obtained a reasonable reduction in the number of tweets that require manual labeling after creating clusters.

Forming Clusters

The goal of this step is to create clusters of similar tweets. The cluster formation stage consumes the file output of the earlier stage. In this step, we select "`tweet-1`" from the output and mark it as a cluster head. All the tweets paired up with this tweet form a cluster. We repeat this step until all tweets paired with a cluster head are put in the cluster. After this, all tweets left unpaired are collected and made cluster heads. The only difference is that in this case, the cluster size zero.

Manual Labeling and Propagating Labels

In this step, we manually labeled the cluster heads. After every tweet is labeled, we just replicate the label for all tweets in the cluster. Thus, following this technique, we can label a

considerable number of tweets by just labeling a few tweets that represent an entire cluster. The table 4.1 and 4.2 describes the total reduction in number of tweets to be labeled manually. We discuss the accuracy of this method in section 6.3.

Score Thresholds and Reduction in Labeling Workload

Table. 4.1: Similarity Score and Percent Reduction for 20 Months

Keyword	Similarity Score Threshold	Percent Reduction in # of tweets
Algae Bloom	0.55	68.66
Algal Bloom	0.55	59.62
Blue Green Algae	0.65	41.20
Cyanobacteria	0.50	35.99
Red Tide	0.55	45.12
Toxic Algae	0.50	72.12

Table. 4.2: Similarity Score and Percent Reduction for Last 4 Months

Keyword	Similarity Score Threshold	Percent Reduction in # of tweets
Algae Bloom	0.50	47.37
Algal Bloom	0.40	54.11
Blue Green Algae	0.50	43.91
Cyanobacteria	0.45	26.30
Red Tide	0.40	47.48
Toxic Algae	0.45	67.88

We can observe that the least reduction is for Cyanobacteria, this can be due to the fact that Cyanobacteria keyword is highly specific to HABs and does not have many retweets. Thus, less number of retweets means more unique tweets to label. The term Toxic Algae shows highest percent reduction in number of tweets to be labeled, again this could be due to large number of retweets since this is a layman term to refer to HABs.

4.2.2 Limitations of Current Approach

As described in this chapter, we remove a tweet from the tweet list once it is paired and assigned to a cluster. This means that not all tweets are compared with all other tweets. It is possible that a tweet appearing earlier in the list may have higher similarity score to a tweet appearing later in the list. These two tweets are not compared when the earlier tweet is paired and removed from list. Thus, the tweets appearing earlier in the list determine the clusters and if the list is shuffled then different clusters will be formed. We hypothesize that this will not affect the label accuracy of the tweet because although a tweet was paired with another tweet with comparatively lower similarity score, the score was still above the threshold determined for the data set. Thus, although a tweet can potentially fall into different clusters based on initial tweets in the list, it will still obtain the same label albeit through a different cluster.

CHAPTER 5

Extracting Reports and Location of Blooms

In this chapter, we describe the implementation details of the machine learning and location extraction components. The goal of the machine learning component as described in section 3.5 is to separate, without human intervention, the tweets which contain a report of an algae bloom. Once we have obtained the data from Twitter and labeled it using the techniques mentioned in chapter 4, we move to the next step in the pipeline.

5.1 Machine Learning for Classification of CyanoHAB Tweets

In this section, we discuss the details of the machine learning models that we trained to separate algae bloom reports from irrelevant and non-reporting tweets. We experimented with four machine learning algorithms to study their efficiency for our task. We trained the classifiers on labeled data obtained from earlier steps. We then measured the performance of the model on unseen data.

5.1.1 Preparation of Training Data

One important step before training any model is to prepare the data for machine learning. This step includes removal of any unwanted characters from text, adjusting white space between words and creating a numerical representation of the text. Before feeding data to the algorithm, we need to convert it to a format that can be consumed by the algorithm. All the algorithms we experimented with need numerical representation of text. TF-IDF vectors are one of the possible forms of input.

Creating TF-IDF Vectors

We used the scikit - learn's [1] inbuilt `TfidfVectorizer` [2] to convert the text to TF-IDF vectors. This package allows us to tweak various parameters while creating TF-IDF vectors. These parameters are crucial in determining the efficiency of our model. We can think of TF-IDF vectors as a way of representing data. Changes in this representation reflects in the ability of our model to classify tweets. Let us take a look at some of the most important parameters.

- `ngram_range`

  This parameter is probably the most important one to tune the performance of our model. It defines the number of words to use while creating the vectors. Since we are working with textual data, it is important that we grasp the context of the words. Treating each words individually as a feature will fail to grab much of the context of tweet. For example, consider the words "alert" and "issued". They do not convey any information individually, but when taken together, we can say that some alert has been issued. For our models, we found that using 2 words worked the best (`bi-grams`). So we ended up with features such as "advisory issued", "bloom found", "public warned", etc for tweets marked as reports. We also kept the most frequent uni-grams.

- `min_df`

  This parameter controls the terms that will be used in the vector by specifying the minimum frequency in terms of proportion of the entire corpus if float or exact number if specified as an integer. This is used to remove terms that rarely occur in our corpus of text. For our dataset, `min_df = 2` worked best.

[1] http://scikit-learn.org/stable/

[2] http://scikit-learn.org/stable/modules/generated/sklearn.feature_extraction.text.TfidfVectorizer.html

- `max_df`

  This parameter defines the maximum frequency to use when creating the TF-IDF vector. Similar to `min_df`, this parameter is used to define an upper bound for the frequency. If defined as a float then it is the portion of the entire corpus or if defined as an integer, then it is the exact number of times the document can appear. This is useful in controlling the documents that appear too frequently, in our case words increased in count due to re-tweets can be controlled. For our dataset, `max_df = 0.9` worked best.

The TF-IDF vectors were then fed to each machine learning algorithm together with the ground truth labels.

5.1.2 The Classifiers

We trained 4 algorithms namely Naive Bayes, Gradient Tree Boosting, Random Forests, and Logistic Regression with Naive Bayes Features. This section describes the models in detail.

Naive Bayes

According to Zhang [25], Naive Bayes is a supervised learning algorithm. Naive Bayes assumes that all the features in the training set are independent of each other. Naive Bayes assigns a class to an instance based on a probabilistic model determined as follows,

$$p(c|F) = \frac{p(F|c)p(c)}{p(F)}$$

Here, $P(c|F)$ is the probability that an instance with features F belongs to c.

$p(F|c)$ is the probability of F given class c.

$p(c)$ is the probability of class c.

$p(F)$ is the probability of F.

Based on this model, Naive Bayes assigns one class to an instance for which the probability is the highest [25]. In our case, Naive Bayes had the fastest learning ability with performance comparable to other classifiers. For this research, we used the following hyper-parameters for Naive Bayes. All these parameter are for scikit-learn [26].

- `alpha`: The smoothing parameter, this was set to 1.0.
- `class_prior = True`, this parameter controls whether to learn class prior probabilities.

Gradient Tree Boosting

This is an ensemble learning technique. In Gradient Tree Boosting, a set of Decision Trees are trained sequentially. In each iteration of learning, the algorithm tries to minimize the classification error produced by the previous iteration. The model in the current iteration is able to fit the data better than the previous iteration [27]. Having control over the learning rate is helpful in preventing over-fitting. This implementation is also made for performance. In our case, using 100 estimators required training time comparable to 18 estimators in Random Forests [28]. We used the following hyper-parameters for Gradient Tree Boosting,

- `learning_rate=0.6, n_estimators=50`, the learning rate determines how quickly the algorithm learns. Higher learning rate may cause the algorithm to never find global optima, lower learning rate will cause the algorithm to run for long duration without much improvement in accuracy.

Random Forests

Breiman [29] explains the Random Forests as follows. Random Forests builds multiple Decision Trees internally and trains them on a sample of training instances. These samples are selected randomly with replacement. Each Decision Tree is trained on a randomly selected subset of features from the feature vector. Once the desired number of trees are trained, every

new instance is classified based on voting mechanism. Each trained tree in the ensemble classifies the new instance to a class. The majority assigned class is selected as the final label for the instance. Each of these decision trees are independent with very low correlation [29]. In our problem, we found that this classifier gave the least number of false positives i.e instances classified incorrectly as incident reports.

We used the following hyper-parameter for scikit-learn's Random Forests implementation [30],

- `n_estimators = 18`, the number of decision trees to build internally.

Logistic Regression with Naive Bayes Features

One of the classifiers that we studied during this research was presented here [31]. This section describes the details of the same.

This classifier is inspired by a work of Wang et. al [32]. Wang, et. al found that using a variation of Naive Bayes features for Support Vector Machines performed well for short text classification problem. Since most of the tweets obtained for our research are short texts, we decided to study the performance of this method. Instead of using SVM as described in [32], we used Logistic Regression because we found that in our case it is faster to train than SVM. Logistic Regression is a supervised learning algorithm. It calculates the probability of an instance to fall into a class instead of directly assigning a label to the instance [33].

The method presented by [31] works as follows, instead of training Logistic Regression algorithm on TF-IDF feature vectors directly, we first process the TF-IDF vectors using Naive Bayes. We obtain a new set of features after this step. We then train a Logistic Regression model on these newly obtained features. We used bi-grams while creating the TF-IDF vectors.

This classifier performed the best overall in terms of classification accuracy. Although it had more false positives than Random Forests, the number was not too high.

The hyper-parameter used was,

- `C = 1.0`, the regularization parameter for Logistic Regression used internally.

In section 6.4, we discuss the performance of these classifiers. We found that Naive Bayes based Logistic Regression works better than other classifiers in identifying reports.

5.2 Location Extraction

Once we have obtained the incidents reports from the data, the next step is to obtain the location of the bloom. This step is crucial since we can perform insightful analysis on the obtained locations. For example, we will in one of the following sections, plot the obtained locations on a map. Here, we will see a pattern in the usage of keywords across the continental United States.

Obtaining Locations

To obtain the locations from the tweets, there were two ways. One was to use the location tag present in the tweet meta-data. If a person tweets from a lake about a bloom and their location services is active on the device, then Twitter will provide this information in downloaded meta-data. However, the problem with this approach is that our data did not have many tweets with location information. People often disallow location tagging for protecting their privacy. Moreover, the given location information may not necessarily reflect the actual location of the bloom, since a user can tweet a long time after visiting the place of the HAB.

Another way, which is more complicated, involves the use of a Natural Language Processing technique called Named Entity Recognition (NER). According to [34], Named Entity Recognition is a process of identifying words in text that refer to people, organizations, places, etc. They are often nouns or pronouns and carry useful information about a topic that the text conveys. Identification of Named Entities require the use of complicated computational techniques [34]. In our system we use Named Entity Recognition to extract words from tweets that refer to places.

We used a software provided by Institut fur Angewandte Informatik e.V.[3] for performing NER. We used the software DBpedia Spotlight [35]. In the section 5.2.1 we provide the information on the usage of DBpedia Spotlight.

5.2.1 DBpedia Spotlight

DBpedia Spotlight is a tool for annotating contents of a text document [35]. Annotation can identify words or phrases in the document that refer to articles in DBpedia [36]. According to Mendes et al. [35], DBpedia Spotlight performs a series of text processing tasks to identify candidate words that can fall into various "resource types". A resource type can be organization, person, place, etc. Currently DBpedia Spotlight can identify 272 classes. These classes can have a hierarchy such that for example, place can be subdivided into `PopulatedPlace`, `BodyOfWater`, etc. It allows annotation to be restricted to any combination of these classes. According to Mendes et al. [35], a candidate for annotation is a word or group of words that have been identified as a possible DBpedia resource. Once all the candidates are identified, the software performs disambiguation of the resource type to be applied. For example, if a candidate is the word `Apple` then the software decides whether it refers to an organization or a fruit. This is done using the words surrounding the candidate to obtain the context in

[3] http://www.dbpedia-spotlight.org/

which the word is used. Finally, the identified candidates are filtered based on user specification [35]. For our use, we restricted the candidates to water bodies. The tasks are described as follows,

- `Spotting`: Extract named entity candidates from the input.
- `Candidate Selection`: Use DBpedia for figuring out meaning for an entity.
- `Disambiguation:` If several candidates are found for a named entity, then select the best candidate.
- `Filtering`: Remove entities that do not meet user requirements.

DBpedia Spotlight provides a web service to perform annotations. This can be done locally by running a server and interacting with the web service programmatically. We installed this software locally and made GET request to the annotation web service with parameters that define our requirements. The annotation tool provides a way to fine tune the annotation task by using headers in the GET request. For our purpose, we wanted to extract locations from tweets but wanted to filter out any location that was not a water body. Thus, we used the following parameters for filtering out the locations found.

```
"types":"DBpedia:BodyOfWater,
Freebase:/geography/body_of_water,
Freebase:/geography/lake,
Freebase:/geography/lake_type,
Freebase:/geography/river,
Freebase:/geography/waterfall,
Schema:BodyOfWater,
Schema:Park"
```

This filter was used for all keywords except Red Tide. We found very few locations using the above filter, thus for Red Tide we used filter: `DBpedia:PopulatedPlace`. This filter selects all locations. We present the locations obtained in section 6.5.

CHAPTER 6

Empirical Analysis and Experimental Evaluation

In this chapter, we present the findings of our research. We present a quantitative analysis of the data followed by the experimental evaluation results of salable labeling technique described in chapter 4 and Machine Learning as outlined in chapter 5 with the locations obtained from this data.

6.1 Quantitative Data Analysis

First, we discuss the properties of the data collected. These results provide valuable insight into the seasonal pattern observed in algae blooms. Warm weather is conducive to harmful algal blooms [3]. The seasonal variation demonstrated by the number of tweets collected each month reinforces the seasonal variation of algae blooms due to weather changes. We also present the number of reports, relevant and irrelevant tweets obtained.

Number of Tweets Collected by Month

As mentioned before, we collected data for 24 months. Figure 6.1 indicates the number of tweets collected every month.

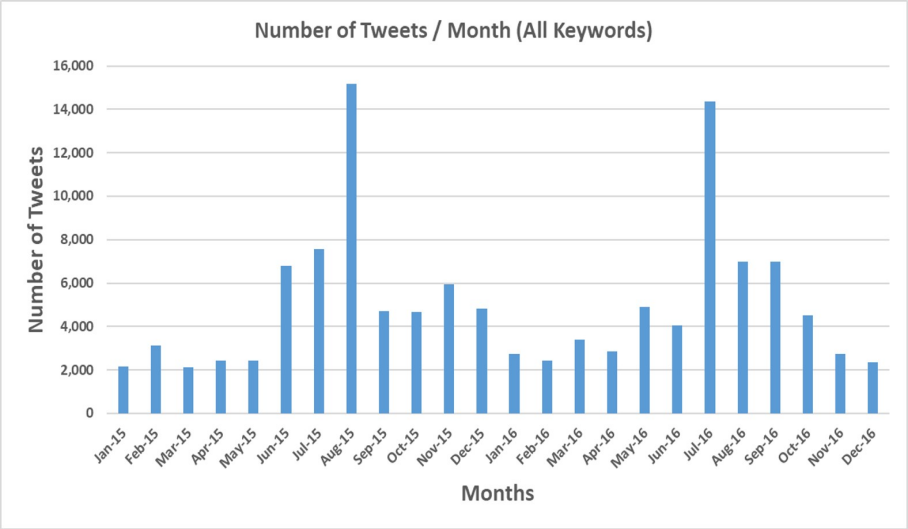


Figure. 6.1: Total Tweets Collected per Month

According to the pattern demonstrated by figure 6.1, the number of tweets collected increases from January up to July / August and then decreases as winter starts. This is expected since summer months usually see high bloom activity.

Total Tweets by Keyword and Class

Table 6.1 gives the number of Irrelevant, Relevant and Reporting tweets for each keyword.

Table. 6.1: Total Tweets Collected per Class

Keyword	Advisory	Relevant	Irrelevant	Total
Algae Bloom	8493	14185	2307	24985
Algal Bloom	4374	4837	495	9706
Blue Green Algae	5425	10244	2581	18250
Cyanobacteria	1644	7686	646	9976
Red Tide	2964	5077	17954	25995
Toxic Algae	7855	22127	1406	31388

The figure 6.2 shows the data from table 6.1 as a graph.

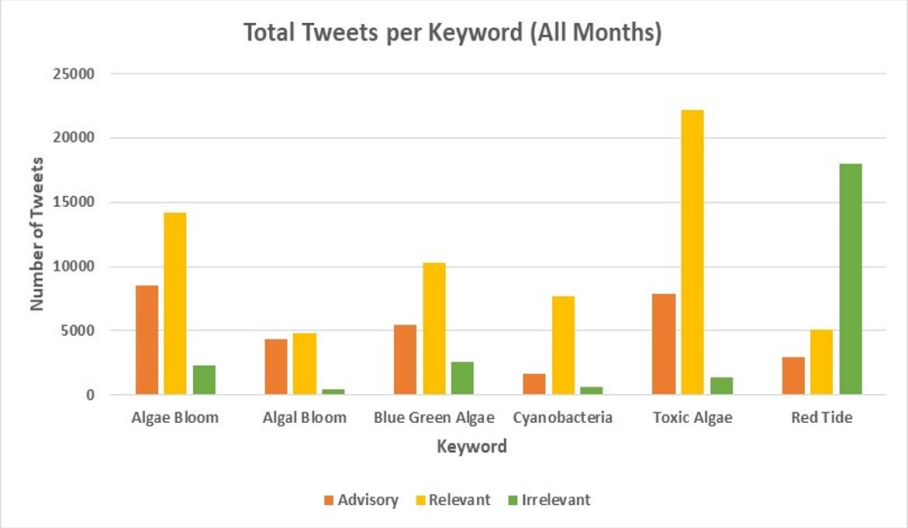


Figure. 6.2: Total Tweets Collected per Keyword and Class

Figure 6.2 clearly shows high number of irrelevant tweets and the least number of reports for red tide. We can infer that red tide tweets mostly fall into topics that are not related to harmful algal blooms. Toxic Algae on the other hand shows the most number of relevant tweets. This is expected since toxic algae is a specific term that refers to harmful algal blooms. Moreover, toxic algae and algae bloom both show large number of reports, this could be due to the fact that Twitter users use non-scientific terms to refer to algae blooms.

Tweets by Month

Figure 6.3 shows the total tweets collected each month.

As we can observe, all keywords show a distinct spike in number of tweets around summer months. For cyanobacteria, this spike is not that distinct. We believe that it is due to the fact that cyanobacteria is used by Twitter users to refer to both blooms and as a phenomenon.

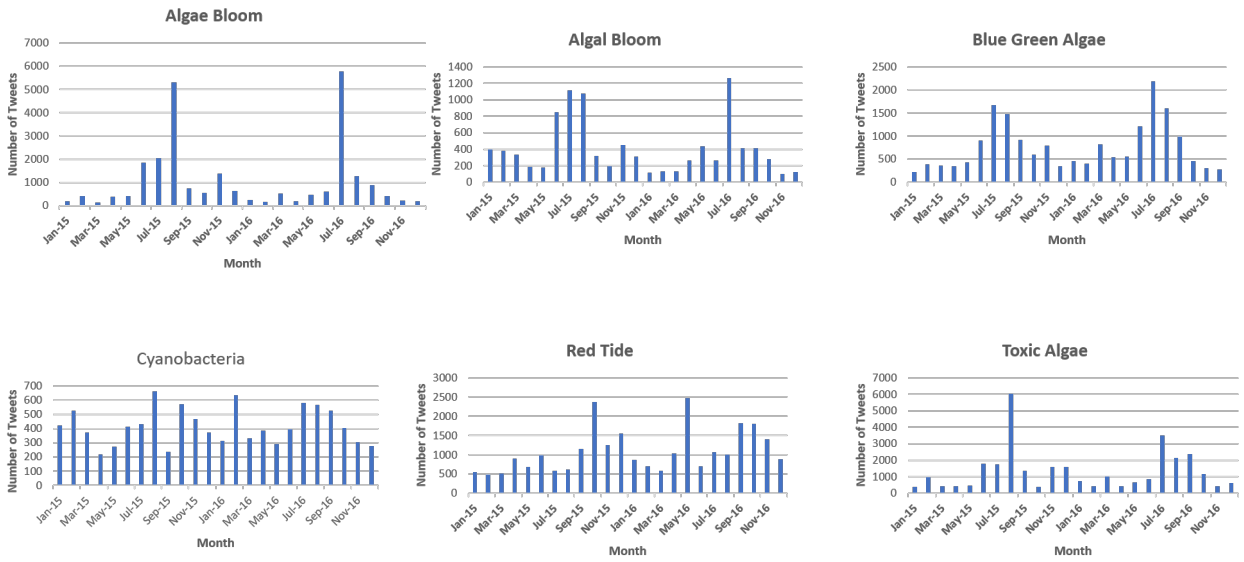


Figure. 6.3: Tweet Collection by Month

Unique Users

Table 6.2 shows the number of unique users per keyword. This shows the number of users who tweeted at least once. It also shows the rate of distinct users, the rate shows the average number of unique users tweeting each month.

Table. 6.2: Number of Unique Users and Unique User Rate

Keyword	Unique Users	Unique User Rate
Algae Bloom	14097	587.37
Algal Bloom	5883	245.12
Blue Green Algae	9287	386.95
Cyanobacteria	4975	207.29
Red Tide	16092	670.50
Toxic Algae	17342	722.58

Reliable Users

A reliable user is one who has tweeted only incident reports and relevant tweets. The rate shows average number of reliable users each month.

Table. 6.3: Number of Reliable Users with Percentage and Rate

Keyword	Reliable Users	Reliable User Percentage	Reliable User Rate
Algae Bloom	12941	91.79	944.91
Algal Bloom	5625	95.61	383.79
Blue Green Algae	8182	88.10	652.87
Cyanobacteria	4701	94.49	388.75
Red Tide	5749	35.72	335.04
Toxic Algae	16716	96.39	1249.25

As shown in table 6.3, the reliable user percentage is least for red tide. This shows that very few users tweet relevant content when referring to red tide. Whereas Toxic Algae is a specific term used by general public to refer to HABs and thus shows highest percent reliability.

Active Users

Active users are those who have tweeted at least twice in 24 months. The rate of active users shows the average number of active users per month.

Table. 6.4: Number of Active User and Active User Rate

Keyword	Active Users	Active User Rate
Algae Bloom	3250	135.41
Algal Bloom	1171	48.79
Blue Green Algae	1919	79.95
Cyanobacteria	1110	46.25
Red Tide	799	33.29
Toxic Algae	4318	179.91

Table 6.4 shows that the most active users are tweeting using Toxic Algae keyword. Again, this could be a result of the general popularity of the word among citizens. Also, Red tide

shows least popularity most likely because it is used largely to mention subjects irrelevant to HABs.

Patron User Percentage

Patron user percentage is calculated by dividing active user rate by reliable user rate. This gives us the keyword giving maximum information.

Table. 6.5: Percentage of Patron Users per Keyword

Keyword	Patron %
Algae Bloom	25.11
Algal Bloom	20.81
Blue Green Algae	23.45
Cyanobacteria	23.61
Red Tide	13.89
Toxic Algae	25.83

From table 6.5, we can observe that, for Red Tide, the Patron % is the least. This means that layman terms provide least information. Moreover, scientific terms like blue green algae and cyanobacteria show high information gain. Form this, we can infer that highly scientific terms may be being used by researchers or domain experts.

Top Profile Words

We used the Twitter API to download profile descriptions of the users in our data-set. We then calculated the top ten words found in the profile descriptions for each keyword. The table 6.6 presents these words.

As we can observe from this table, domain specific terms such as Blue Green Algae and Cyanobacteria show profile words such as “news”, “research”, “health”, “water”, “biology” which are relevant to HABs. Whereas Red Tide being a terms used in wide variety of topics,

Table. 6.6: Top Words in Profile Description

Rank	Algae Bloom	Algal Bloom	Blue Green Algae	Cyanobacteria	Red Tide	Toxic Algae
1	news	news	news	science	news	news
2	world	science	health	news	love	world
3	love	water	water	research	world	love
4	social	world	love	water	life	breaking
5	life	love	local	university	follow	life
6	media	tweets	life	tweets	like	water
7	follow	environmental	media	biology	breaking	science
8	breaking	health	reporter	new	latest	media
9	science	follow	follow	world	twitter	health
10	writer	twitter	animal	life	music	follow

have words like “love”, “life”, “follow” and “music” which are irrelevant to harmful algae blooms.

6.2 Concept Drift

In this section, we present an interesting study. The goal of this study is to identify the concept drift in the collected data. This section explains concept drift and our methodology to calculate the same.

Textual data collected over long periods of time can demonstrate concept drift. Concept Drift means that data collected for some concept may change in meaning, usage or sentence construction as language evolves. As an extreme example, consider English used during the medieval times to the modern times, a machine learning model trained on old English will not work efficiently on the later. The above is an extreme example, but it certainly presents a problem for machine learning based systems since the rate of concept drift may be higher for one subject compared to other. We wanted to study the concept drift in our data and

how fast the language used by Twitter users is evolving for the subject of harmful algae blooms.

To quantify the concept drift, we decided to use the bag of words approach. In this experiment, we first removed all the stop words from the collected data. We then created a set of all words used in the tweets for each month. Let these sets be $S1, S2, S3, ...S24$, for 24 months of data. We then calculated the Jaccard Coefficient [22] for these months according to the following formula, If $JC(M_i, M_j)$ is the Jaccard Coefficient for month i and month j then,

$$JC(M_i, M_j) = \frac{M_i \cap M_j}{M_i \cup M_j}$$

As we can see, we take an intersection of the two sets M_i and M_j and divide the union of the same. Here, the numerator is the number of words common to both sets and denominator is sum of all of the unique words in both sets. Thus, if in given two months, the same words were used, then the Jaccard Coefficient will be equal to 1. Similarly the dissimilar the sets, the lower the Jaccard Coefficient.

We calculated the Jaccard Coefficients over an increasing monthly distance over 24 months. We first calculated the Jaccard Coefficient of months that are one month apart, for example, January 2015 - February 2015, February 2015 - March 2015, up to November 2016 - December 2016. Then two months apart like, July 2015 - September 2015, etc. We calculated the Jaccard Coefficient for months up to 23 months apart. We then calculated the average Jaccard Coefficient for each bracket of the monthly gap. The figure 6.4 shows the results.

The figure 6.4 shows the Jaccard Coefficient for 24 months period. The $x - axis$ is monthly gap and $y - axis$ is average Jaccard Coefficient.

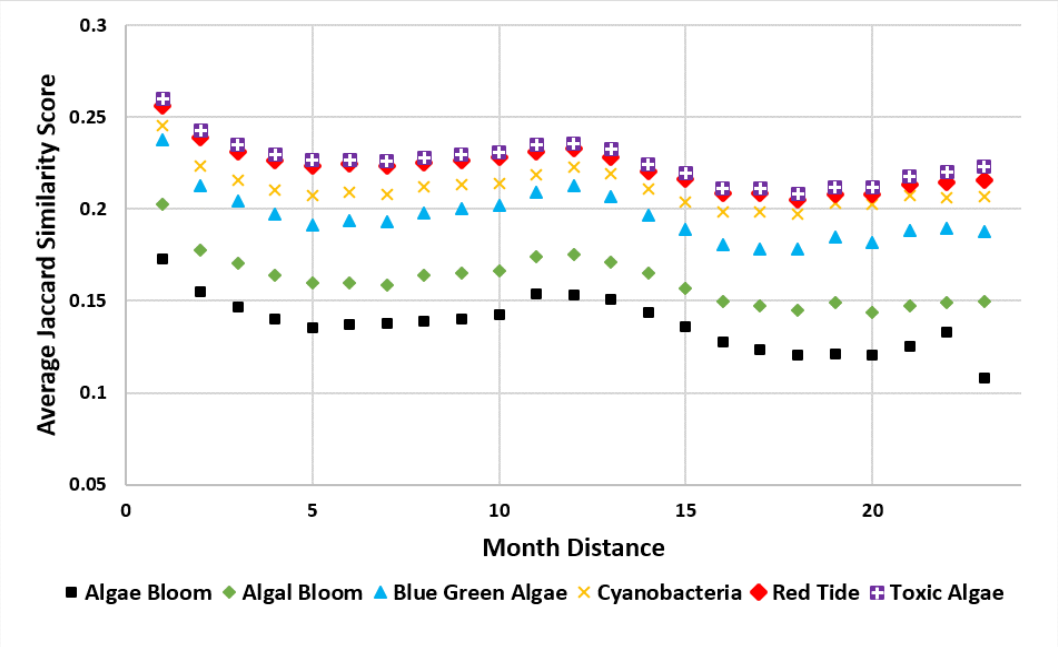


Figure. 6.4: Concept Drift over 24 months of data

As we can observe, the chart displays a distinct cyclic pattern for each keyword. Each pattern decreases around 5 to 6 month difference and then rises up to 12 month gap. The Jaccard Coefficient starts to decrease again till the next 6 months and then stars to increase till 23 months gap.

This phenomenon can be interpreted as follows, a monthly distance of around 10 to 12 month show near similar context or use of words. Observe the 5 month and 10, 11 and 12 month gap of Cyanobacteria, they are very close at around 0.2 average Jaccard Coefficient. The above shows that people tweeting about Cyanobacteria use similar words over 10 to 12 months of period. This can be attributed to the seasonal nature of harmful algal blooms. For example, in a gap of 5 to 10 months, people might be referring to Cyanobacteria as a natural phenomenon and over 10 to 12 months of distance, they might be reporting bloom incidents as they occur at nearly the same time each year.

Note that, although the Jaccard Coefficient increases over 12 months cycles, it never reaches the same score as when it began. That is, if we plot a trend line for the patterns, it will show a decreasing trend. Since this result is obtained for 24 months of data, it is not sufficient to conclusively state whether concept drift exists in our Twitter data. A conclusive statement can be made about concept drift only after this experiment is performed over a larger data set covering more number of years.

6.3 Effectiveness of MinHash Based Labelling

In this section, we present the accuracy of the technique described in chapter 4. The table 6.7 lists the accuracy of using MinHash algorithm based label propagation.

To determine the effectiveness of our technique, we randomly selected 20% tweets from each cluster and manually labeled them without referring to the label inherited from the cluster head.

We found the percent accuracy for each keyword as described in table 6.7.

Table. 6.7: Accuracy of Label Propagation using Similarity Scores

Keyword	Accuracy (%)
Algae Bloom	87.89
Algal Bloom	83.73
Blue Green Algae	90.49
Cyanobacteria	94.14
Red Tide	88.71
Toxic Algae	94.24

We can observe that the accuracy is more than 90% for three keywords and more than 85% for two keywords. This accuracy can be considered good given the fact that we had to label almost less than 53% tweets on an average across all keywords. We hypothesize that toxic algae shows the highest accuracy due to the fact that it has large number of re-tweets in

the data. This is corroborated by the fact that in table 4.1, Toxic Algae has highest percent reduction in tweet count pointing to the fact that many tweets are similar.

6.3.1 Analysis of the Clusters

In this section, we present an analysis of the clusters formed for scalable labeling as described in the chapter 4.

The table 6.8 gives the number of clusters formed for each keyword. Note that since we collected data in two phases, we performed cluster formation twice, separately for each keyword.

Table. 6.8: Number of Clusters for Each Keyword

Keyword	Number of Clusters (20 Months)	Number of Clusters (4 months)
Algae Bloom	2244	890
Algal Bloom	1011	419
Blue Green Algae	1819	1143
Cyanobacteria	694	1113
Red Tide	1881	3111
Toxic Algae	704	1460

The table 6.9 shows the number of outliers for each keyword. The outliers are those tweets that were not found similar to other tweets and thus not assigned to any cluster.

Table. 6.9: Number of Outliers for Each Keyword

Keyword	Number of Outliers (20 Months)	Number of Outliers (4 months)
Algae Bloom	5067	731
Algal Bloom	2540	324
Blue Green Algae	7687	900
Cyanobacteria	4725	989
Red Tide	9143	2529
Toxic Algae	1264	1081

Red Tide shows the most number of outliers. This can be attributed to that fact that tweets using the Red Tide keyword vary significantly from each other and that there might not be

many re-tweets. This can be validated with the results in figure 6.5. Red Tide has one of the least number of tweets assigned to top cluster.

Percent of Tweets Assigned to Each Cluster

The figure 6.5 presents the percent of tweets assigned to the most popular ten clusters for each keyword.

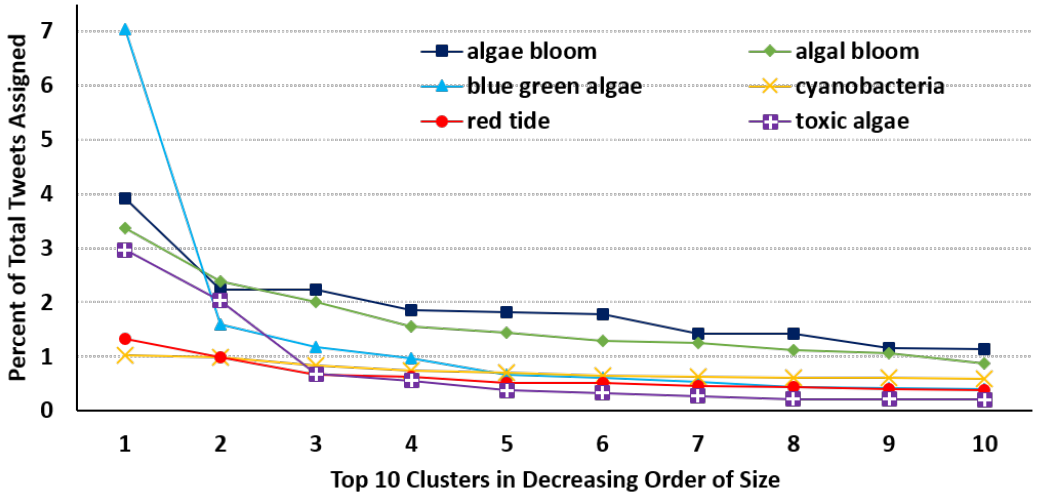


Figure. 6.5: Percent of Tweets Assigned to Top 10 Largest Clusters

As we can observe, the highest percent of tweets assigned within all keywords is around 7% for Blue Green Algae. This can be a result of large number of retweets for a particular tweet. Since the retweets have a high similarity score, they will all fall into one cluster. Moreover, we can observe that cyanobacteria had the least number of tweets assigned to the top cluster. This could be attributed to the fact that there might not have been many retweets for cyanobacteria keyword. If we observe the active user rate for cyanobacteria from table 6.4, we can see that it has the second lowest rate of active users. Thus we can conclude that not many users tweeting about cyanobacteria actively retweet other users tweets. Interestingly, the two lowest active user rates show the similar percent of tweets assigned to top cluster.

Moreover, Blue Green Algae with highest rate of Active Users shows highest percent of tweets assigned to top cluster.

6.3.2 Threshold Sensitivity Analysis

In this section, we present the analysis of the sensitivity of the score threshold. The goal is to determine the effect of small changes in score threshold on the number of clusters formed and the amount of data assigned to a cluster.

For this experiment, we applied the scalable labeling algorithm mentioned in section 4.2 to each keyword. We performed 9 iterations and increased the score threshold by 0.05 in each iteration starting from 0.30 to 0.70. The graphs 6.6 and 6.7 describe the effect of increasing the score threshold on each keyword.

Effect of Threshold on Number of Clusters

As demonstrated by graph 6.6, as the threshold increases, the number of clusters also increase for all keywords except Red Tide and Cyanobacteria. In this experiment, we counted only those clusters that have at least one tweet assigned.

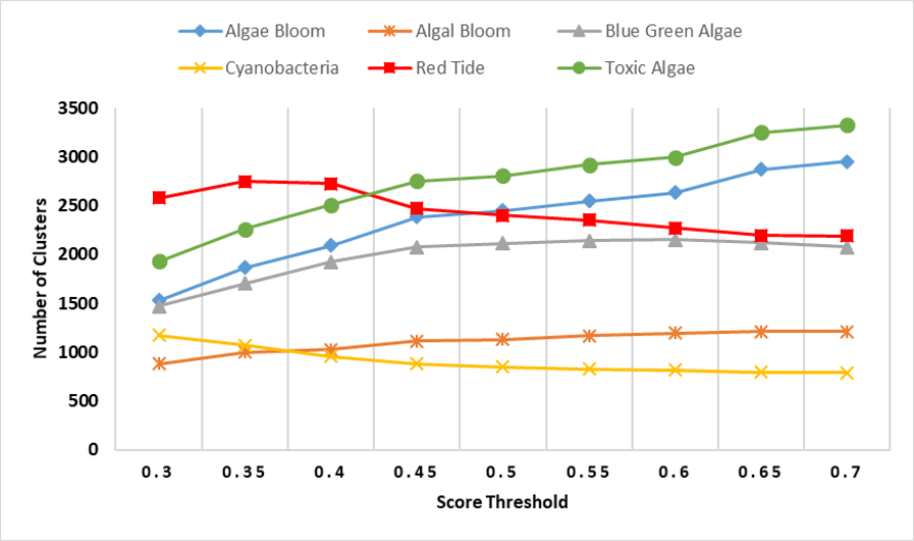


Figure. 6.6: Number of clusters per threshold

The increase in number of clusters indicate that less number of tweets are assigned to a cluster and more tweets are forming their own clusters. In case of Red Tide and Cyanobacteria, a decreasing number of clusters indicate that no tweets are similar enough to be assigned to a cluster and large number of tweets are remaining unassigned. This is also an indicator of large number of unique tweets. Such behaviour can be expected from Red Tide and Cyanobacteria as they fall on the opposite end of the spectrum. Red Tide being a layman term referring to variety of topics and Cyanobacteria being a highly specific term, giving rise to large number of unique tweets.

Effect of Threshold on Number of Tweets Assigned to a Cluster

According to graph 6.7, the number of tweets assigned to a cluster decrease as the threshold increases.

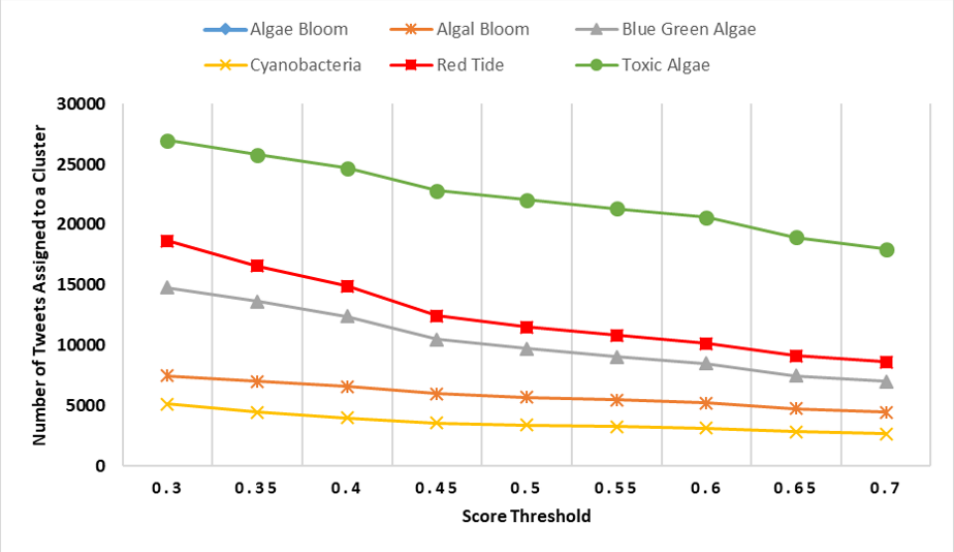


Figure. 6.7: Number of tweets assigned to clusters per threshold

This behaviour is expected because a larger threshold forms less number of tweet pairs thus less number of tweets get assigned to a cluster. However, the effect of threshold variation is different for different datasets. The datasets for keywords with comparatively more number of overall tweets show higher sensitivity to threshold. A slight variation of threshold causes

a rapid decline in number of clustered tweets. Whereas, keywords like Cyanobacteria and Algal Bloom with relatively low number of tweets show lower rate of decline. Thus, we can hypothesize that the sensitivity of the similarity score threshold on the clustering is a characteristic of individual dataset. Every dataset is expected to show a decline in number of clustered tweets but the rate of decline may vary for each dataset.

6.4 Evaluation of Data Classification

In this section, we present the performance of each of the classifiers. We also present the confusion matrix which presents a visual representation of the classification results. For this data-set we split the training and testing in 80:20 ratio.

The figure 6.8 shows the division of data into train and test sets together with the number of irrelevant and reporting tweets.

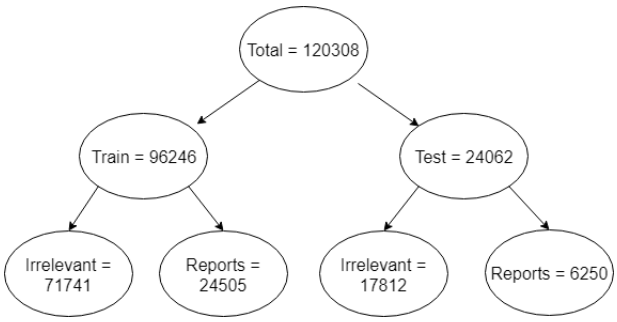


Figure. 6.8: Division of Data

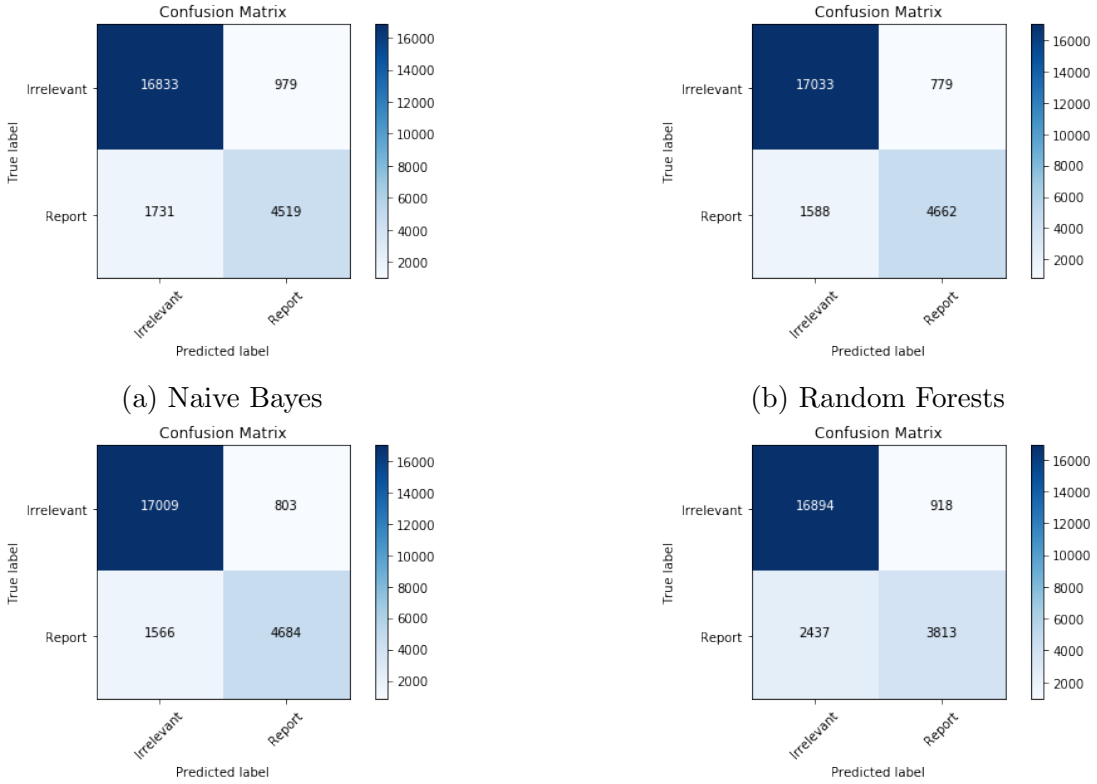
The table 6.10 provides the result of testing classifier performance on test data in terms of classification accuracy.

Table. 6.10: Classifier Performance

Classifier	Accuracy (%)
Gradient Tree Boosting	86.05
Naive Bayes	88.73
Random Forests	90.16
Logistic Regression with NB Features	90.15

Confusion Matrix

The confusion matrix for each classifier is presented in the figure 6.9. The confusion matrix gives the number of instances classified in each class vs actual class. More number of instances in the diagonal of the matrix means better classification. Our goal in this classification was to reduce the number of False Positives i.e. Irrelevant tweets classified as reports. Since a tweet talking about algae bloom but not reporting should not be used as an indicator of a bloom incident. This can cause incorrect location to be marked as having algal bloom, thus we wanted to avoid such cases.



(a) Naive Bayes

(b) Random Forests

(c) Logistic Regression with NB Features

(d) Gradient Tree Boosting

Figure. 6.9: Confusion Matrix for each Classifier

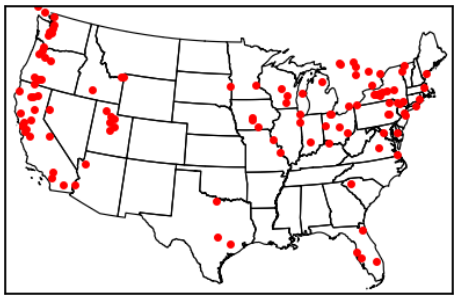
We can observe that Random Forests really shines in avoiding false positives. Although the correctly classified instances of reports are less, still they are very close to the highest number of (4662 vs 4684) correctly classified reports.

6.5 Location Extraction Results

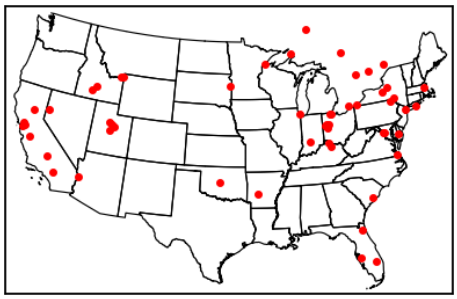
In this section, we present the locations obtained from the incident reports in our data. If we observe the locations, we find interesting insights that reinforce the information available about HABs. First, we can observe that all keywords show maximum density of locations around North-East, entire west coast and Florida State. No keyword shows any significant activity in the Mid-West and Central America. Red Tide shows interesting phenomenon were the blooms appear in large numbers around Florida and Gulf of Mexico in coastal regions. Cyanobacteria shows the least density and number of reported incident which can be attributed to the fact that in our data-set Cyanobacteria incident reports are much lesser than the other keywords. Algal Bloom and Algae Bloom show an overlapping set of locations largely because these are essentially the same keywords. Algal bloom albeit has lesser locations which can again be attributed to the fact that it has almost 51% less reports than Algae Bloom.

Let us take a look at the locations obtained from DBPedia Spotlight. We used the geoPy package for python [1] for converting location name obtained from DBpedia Spotlight to location coordinates. The figure 6.10 shows the locations obtained for each keyword.

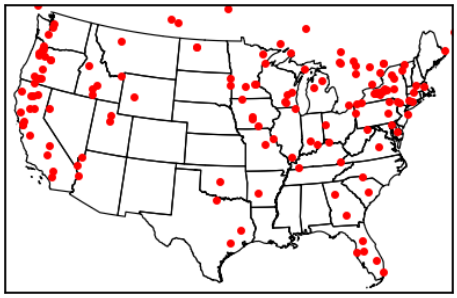
[1] https://github.com/geopy/geopy



(a) Algae Bloom

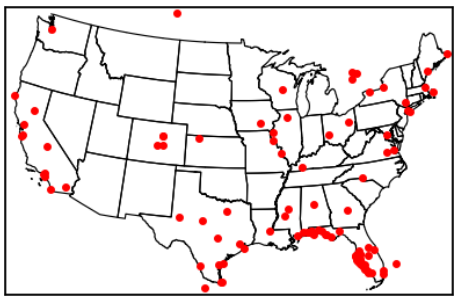


(b) Algal Bloom

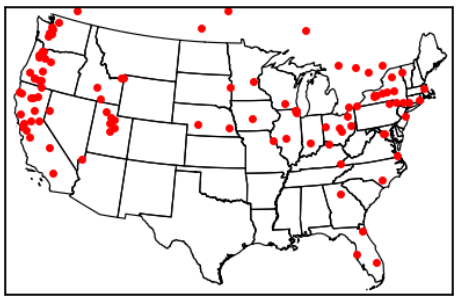


(c) Blue Green Algae

(d) Cyanobacteria



(e) Red Tide



(f) Toxic Algae

Figure. 6.10: Bloom Locations Obtained from Reports

The table 6.11 gives the exact number of locations obtained for each keyword. Note that many of these locations fall outside of the continental United States and thus are not visible on the maps in figure 6.10.

Table. 6.11: Total Locations Obtained for Each Keyword

Keyword	Number of Unique Locations
Algae Bloom	147
Algal Bloom	81
Blue Green Algae	207
Cyanobacteria	74
Red Tide	227
Toxic Algae	129

As we can observe and expect, the more popular or layman keywords like Toxic Algae and Red Tide have large number of locations reported followed by Algal Bloom. Blue Green Algae although not strictly a scientific term nor a layman term shows significant number of location reports. We believe that this phenomenon can be attributed to the fact that Blue Green Algae reports mention water body name more than other keywords.

CHAPTER 7

Conclusion

Cyanobacterial blooms are a major environmental issue. Effective monitoring and tracking are vital for taking preventive actions to protect the valuable flora and fauna of a region. Social media is an effective tool for analyzing algal blooms both by virtue of it's wide reach and near real-time nature. In this thesis, we discussed the environmental and economical effects of Cyanobacterial blooms. We described the architecture of the CyanoTracker project for effective monitoring of CyanoHABs. We covered three major goals for social media based monitoring of Harmful Algal Blooms. Firstly, we presented a technique to label large number of tweets with high accuracy using MinHash algorithm. Secondly, we presented detailed analysis of the collected data. We saw the seasonal nature of social media activity which lines-up with the seasonal nature of algae blooms. This analysis also demonstrated how certain Twitter keywords are more relevant to algae blooms. We saw the nature of users that tend to use scientific terms over those used by non domain experts. We presented the study of concept drift which demonstrated the evolving nature of natural language which points to the fact that machine learning models when applied to textual data from social media need to be retrained to adapt to newer terms and sentence construction. Finally, we presented the locations extracted from the reporting tweets. These locations gave an insight into the nature of algae blooms and regions affected most by a certain kind of bloom.

BIBLIOGRAPHY

[1] D. S. Francy, J. L. Graham, E. A. Stelzer, C. D. Ecker, A. M. G. Brady, P. Struffolino, and K. A. Loftin, "Water quality, cyanobacteria, and environmental factors and their relations to microcystin concentrations for use in predictive models at ohio lake erie and inland lake recreational sites, 2013-14," US Geological Survey, Tech. Rep., 2015.

[2] P. Hoagland, D. M. Anderson, Y. Kaoru, and A. W. White, "The economic effects of harmful algal blooms in the united states: Estimates, assessment issues, and information needs," *Estuaries*, vol. 25, no. 4, pp. 819–837, 2002. [Online]. Available: http://www.jstor.org/stable/1353035

[3] V. K. Boddula, "Cyber-social-physical approaches for effective detection of cyanobacterial blooms," Ph.D. dissertation, The University of Georgia, 2017.

[4] A. R. Joshi, "Study of microblog activity on cyanobacterial harmful algal blooms," Master's thesis, The University of Georgia, 2015.

[5] 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*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 851–860. [Online]. Available: http://doi.acm.org/10.1145/1772690.1772777

[6] S. Asur and B. A. Huberman, "Predicting the future with social media," *CoRR*, vol. abs/1003.5699, 2010. [Online]. Available: http://arxiv.org/abs/1003.5699

[7] 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*, ser.

WWW '10. New York, NY, USA: ACM, 2010, pp. 591–600. [Online]. Available: http://doi.acm.org/10.1145/1772690.1772751

[8] A. Java, X. Song, T. Finin, and B. Tseng, "Why we twitter: Understanding microblogging usage and communities," in *Proceedings of the 9th WebKDD and 1st SNA-KDD 2007 Workshop on Web Mining and Social Network Analysis*, ser. WebKDD/SNA-KDD '07. New York, NY, USA: ACM, 2007, pp. 56–65. [Online]. Available: http://doi.acm.org/10.1145/1348549.1348556

[9] M. Demirbas, M. A. Bayir, C. G. Akcora, Y. S. Yilmaz, and H. Ferhatosmanoglu, "Crowd-sourced sensing and collaboration using twitter," in *2010 IEEE International Symposium on "A World of Wireless, Mobile and Multimedia Networks" (WoWMoM)*, June 2010, pp. 1–9.

[10] M. D. Scott, L. Ramaswamy, and V. Lawson, "Cyanotracker: A citizen science project for reporting harmful algal blooms," in *Collaboration and Internet Computing (CIC), 2016 IEEE 2nd International Conference on*. IEEE, 2016, pp. 391–397.

[11] V. Boddula, L. Ramaswamy, and D. Mishra, "Cyanosense: A wireless remote sensor system using raspberry-pi and arduino with application to algal bloom," in *AI & Mobile Services (AIMS), 2017 IEEE International Conference on*. IEEE, 2017, pp. 85–88.

[12] ——, "A spatio-temporal mining approach for enhancing satellite data availability: A case study on blue green algae," in *Big Data (BigData Congress), 2017 IEEE International Congress on*. IEEE, 2017, pp. 216–223.

[13] R. Bonney, C. B. Cooper, J. Dickinson, S. Kelling, T. Phillips, K. V. Rosenberg, and J. Shirk, "Citizen science: A developing tool for expanding science knowledge and scientific literacy," *BioScience*, vol. 59, no. 11, pp. 977–984, 2009. [Online]. Available: +http://dx.doi.org/10.1525/bio.2009.59.11.9

[14] B. L. Sullivan, C. L. Wood, M. J. Iliff, R. E. Bonney, D. Fink, and S. Kelling, "ebird: A citizen-based bird observation network in the biological sciences," *Biological Conservation*, vol. 142, no. 10, pp. 2282 – 2292, 2009. [Online]. Available: http://www.sciencedirect.com/science/article/pii/S000632070900216X

[15] H. Zheng, H. Yang, D. Long, and J. Hua, "Monitoring surface water quality using social media in the context of citizen science," *Hydrology and Earth System Sciences*, vol. 21, no. 2, p. 949, 2017.

[16] A. Kongthon, C. Haruechaiyasak, J. Pailai, and S. Kongyoung, "The role of twitter during a natural disaster: Case study of 2011 thai flood," in *Technology Management for Emerging Technologies (PICMET), 2012 Proceedings of PICMET'12:*. IEEE, 2012, pp. 2227–2232.

[17] A. Musaev, D. Wang, and C. Pu, "Litmus: Landslide detection by integrating multiple sources." in *ISCRAM*, 2014.

[18] M. Guy, P. Earle, C. Ostrum, K. Gruchalla, and S. Horvath, "Integration and dissemination of citizen reported and seismically derived earthquake information via social network technologies," in *International Symposium on Intelligent Data Analysis*. Springer, 2010, pp. 42–53.

[19] R. Power, B. Robinson, and D. Ratcliffe, "Finding fires with twitter," in *Proceedings of the Australasian Language Technology Association Workshop 2013 (ALTA 2013)*, 2013, pp. 80–89.

[20] A. Signorini, A. M. Segre, and P. M. Polgreen, "The use of twitter to track levels of disease activity and public concern in the us during the influenza a h1n1 pandemic," *PloS one*, vol. 6, no. 5, p. e19467, 2011.

[21] M. Imran, P. Mitra, and C. Castillo, "Twitter as a lifeline: Human-annotated twitter corpora for NLP of crisis-related messages," *CoRR*, vol. abs/1605.05894, 2016. [Online].

Available: http://arxiv.org/abs/1605.05894

[22] J. Leskovec, A. Rajaraman, and J. D. Ullman, *Mining of Massive Datasets*, 2nd ed. Cambridge University Press, 2014.

[23] M. Allahyari, S. Pouriyeh, M. Assefi, S. Safaei, E. D. Trippe, J. B. Gutierrez, and K. Kochut, "A brief survey of text mining: Classification, clustering and extraction techniques," *arXiv preprint arXiv:1707.02919*, 2017.

[24] T. S. Madhulatha, "An overview on clustering methods," *CoRR*, vol. abs/1205.1117, 2012. [Online]. Available: http://arxiv.org/abs/1205.1117

[25] H. Zhang, "The optimality of naive bayes," *AA*, vol. 1, no. 2, p. 3, 2004.

[26] "1.9. Naive Bayes scikit-learn 0.19.1 documentation," http://scikit-learn.org/stable/modules/naive_bayes.html, 2018, [Online; accessed 2-April-2018].

[27] J. H. Friedman, "Greedy function approximation: A gradient boosting machine," *The Annals of Statistics*, vol. 29, no. 5, pp. 1189–1232, 2001. [Online]. Available: http://www.jstor.org/stable/2699986

[28] "1.11. Ensemble methods scikit-learn 0.19.1 documentation," http://scikit-learn.org/stable/modules/ensemble.html#gradient-tree-boosting, 2018, [Online; accessed 2-April-2018].

[29] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, Oct 2001. [Online]. Available: https://doi.org/10.1023/A:1010933404324

[30] "1.11. Ensemble methods scikit-learn 0.19.1 documentation," http://scikit-learn.org/stable/modules/ensemble.html#random-forests, 2018, [Online; accessed 2-April-2018].

[31] J. Howard, "NB-SVM strong linear baseline," https://www.kaggle.com/jhoward/nb-svm-strong-linear-baseline, 2018, [Online; accessed 18-January-2018].

[32] S. Wang and C. D. Manning, "Baselines and bigrams: Simple, good sentiment and topic classification," in *Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics: Short Papers - Volume 2*, ser. ACL '12. Stroudsburg, PA, USA: Association for Computational Linguistics, 2012, pp. 90–94. [Online]. Available: http://dl.acm.org/citation.cfm?id=2390665.2390688

[33] A. Agresti, *Logistic regression*. Wiley Online Library, 2002.

[34] B. Mohit, *Named Entity Recognition*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2014, pp. 221–245. [Online]. Available: https://doi.org/10.1007/978-3-642-45358-8_7

[35] J. Daiber, M. Jakob, C. Hokamp, and P. N. Mendes, "Improving efficiency and accuracy in multilingual entity extraction," in *Proceedings of the 9th International Conference on Semantic Systems (I-Semantics)*, 2013.

[36] J. Lehmann, R. Isele, M. Jakob, A. Jentzsch, D. Kontokostas, P. N. Mendes, S. Hellmann, M. Morsey, P. van Kleef, S. Auer, and C. Bizer, "Dbpedia - A large-scale, multilingual knowledge base extracted from wikipedia," *Semantic Web*, vol. 6, no. 2, pp. 167–195, 2015. [Online]. Available: https://doi.org/10.3233/SW-140134