Automatic Detection of Cyber Security Events over Social Network Stream

QUENTIN LE SCELLER

A THESIS

IN

THE CONCORDIA INSTITUTE FOR INFORMATION SYSTEMS ENGINEERING

PRESENTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS

FOR THE DEGREE OF

MASTER OF APPLIED SCIENCE IN INFORMATION SYSTEMS SECURITY

CONCORDIA UNIVERSITY

MONTRÉAL, QUÉBEC, CANADA

AUGUST 2017

CONCORDIA UNIVERSITY

School of Graduate Studies

This is to certify that the thesis prepared

By: **Quentin Le Sceller**

Entitled: **Automatic Detection of Cyber Security Events over Social Network Stream**

and submitted in partial fulfillment of the requirements for the degree of

Master of Applied Science in Information Systems Security

complies with the regulations of this University and meets the accepted standards with respect to originality and quality.

Signed by the Final Examining Committee:

_____ Chair
Dr. Jamal Bentahar

_____ External Examiner
Dr. Olga Ormangjieva

_____ Examiner
Dr. Amr Youssef

_____ Supervisor
Dr. Mourad Debbabi

Approved by _____
Chadi Assi, Chair
Concordia Institute for Information Systems Engineering

_____ 2017

_____
Amir Asif, Dean
Faculty of Engineering and Computer Science

Abstract

Automatic Detection of Cyber Security Events over Social Network Stream

Quentin Le Sceller

Everyday, security experts face a growing number of security events that affecting people well-being, their information systems and sometimes the critical infrastructure. The sooner they can detect and understand these threats, the more they can mitigate and forensically investigate them. Therefore, they need to have a situation awareness of the existing security events and their possible effects. However, given the large number of events, it can be difficult for security analysts and researchers to handle this flow of information in an adequate manner and answer the following questions in near real-time: what are the current security events? How long they last? In this thesis, we will try to answer these issues by leveraging social networks that contain a massive amount of valuable information on many topics. However, because of the very high volume, extracting meaningful information can be challenging. For this reason, we propose *SONAR*: an automatic, self-learned framework that can detect, geolocate and categorize cyber security events in near real-time over the Twitter stream. *SONAR* is based on a taxonomy of cyber security events and a set of seed keywords describing type of events that we want to follow in order to start detecting events. Using these seed keywords, it automatically discovers new relevant keywords such as malware names to enhance the range of detection while staying in the same domain. Using a custom taxonomy describing all type of cyber threats, we demonstrate the capabilities of *SONAR* on a dataset of approximately 47.8 million tweets related to cyber security from July 2016 to July 2017. *SONAR* could efficiently and effectively detect, categorize and monitor cyber security related events before getting on the security news, and it could automatically discover new security terminologies with their event. Additionally, *SONAR* is highly scalable and customizable by design; therefore we could adapt *SONAR* framework for virtually any type of events that experts are interested in.

Acknowledgments

I would like to express my sincere gratitude to my supervisor, Dr. Mourad Debbabi, for his resolute recommendations, guidance and assistance from the earliest to the final stages of my research. He has been a great source of motivation throughout my study, and I am profoundly grateful to him.

Last, but definitely not least, I am endlessly grateful to all my family members for their unwavering support and motivation throughout this entire process.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

In this chapter, we start by presenting the motivation of our work. Then, we formally define the problem, our objectives and our research contributions.

1.1 Motivation

In 2015, on average, a zero-day vulnerability was found every week [97] and this number is expected to rise to one-per-day by 2021 [103]. From the perspective of a security practitioner, this might be hard to cope without the proper security awareness tools. Moreover, in the case of a newly discovered vulnerability, the delay between the disclosure and the moment when the security practitioner is aware of it is crucial. For example, attack such as the MongoDB ransomware [110] can be mitigated as soon as the user is aware of the flaw and the possible patch.

In order to learn such information, the analyst has a wide range of sources available: specialized press, tech forums and even specialized communication protocol for the dissemination of cyber threat information [30]. However, social media are the largest and most powerful sources providing a massive source of information about every possible topic. Characteristics of Twitter [53] make it the foremost choice in the case of real-time event detection. Everyday, approximatively 500 millions tweets are sent by 100 million active users [6]. Major events are discussed on social media and often the first reference for such event is on those social networks. In our case, an event is a phenomenon happening during a certain time period which stimulates people to post about it on social media.

For example, denial of service attacks are often first reported by users of the website or service under attack. Users of online social networks witnessing or participating in an event are naturally incentivized to discuss it on social media (e.g., tweeting "I cannot reach X website"). Moreover, Twitter act as a platform where not only cybersecurity media but also mainstream media will tweet about ongoing events and where people can react to such events.

Nonetheless, dealing with document such as tweets for the purpose of detecting cyber security related events poses a wide variety of challenges. First, in order to have a general overview of the current situation, the framework must be able to proceed in real-time and in a timely manner a very high volume data. Hence, all the algorithms used should be fast, scalable and possibly distributed. Secondly, given the nature of tweets, documents retrieved are often unusable: there is a lot of duplicates, off-topic, badly written and incomplete documents. As we can get very valuable information (e.g., information on a data breach), we can also get a lot of irrelevant information (e.g., how I feel today). One of the main challenge lies in correctly filtering the noise from the retrieved content. Finally, the lexical field of cyber security event is growing and constantly changing which makes the tracking of new terminology a hard and time consuming task. Having an autonomous system that leverage social network's data for security event detection would be extremely useful for a security analyst. Moreover, the ability to have an evolving scope of detected cyber security related events represent a substantial advantage.

To address these challenges, we designed and developed a framework called *SONAR* capable of detecting, geolocating, categorizing and monitoring cyber security related events over the Twitter Stream. We then developed a tool using this framework. The framework itself only needs as input a taxonomy along with a list of keywords for each taxon. While other event detection frameworks exist, *SONAR* is unique by its realtime aspect, its highly scalable architecture and its automatic keyword detection: using an algorithm based on word embedding, it automatically discovers new keywords in order to detect new relevant events. Events, which are clusters of tweets, are detected using a sliding window and are described by the initial tweet (also called the first story) and the number of similar documents found. Events are then categorized and displayed on a user interface. SONAR detects events in constant time using locality sensitive hashing (LSH) and can technically process millions of documents per day.

1.2 Problem Definition

In order to define the research we first define the global research problem we aim to solve. Secondly, we observe that From this main problem secondary problems occur.

1.2.1 Initial Subject Definition

Everyday, thousands of cyber security events happen in the world. Being able to follow all of them in real-time is humanly impossible. Moreover, in a lot of cases the delay between the moment a vulnerability is found and the moment the investigator learns about it is critical. In our case, a digital forensic is interested in knowing in real-time what are the current security events happening in the world. *SONAR* aims to fix this problem by giving the investigator a framework to see in real-time the cyber security events. Each event is automatically detected using social network. We choose Twitter because it's the biggest and the easier to access as well as for his uniqueness, meaning how information/news and chatter are tied together. In order to extract as much information from Twitter as possible, we choose to retrieve tweets per keywords. This gives us up to 1% of relevant tweets. However, this technique leads to the second problem.

1.2.2 Automatic Keyword Discovery

How can we ensure that the keywords used by the analyst are relevant and up to date? How can we be sure that the analyst will not miss any events because the vocabulary is not up to date? We need to develop an automatic keyword discovery algorithm that will only use the existing data in Twitter and the previous keywords to discover new relevant terms.

1.3 Objectives

Regarding the previously stated problems, we define here the objectives that we wish to achieve. In particular, this thesis aims at:

- Conducting a comparative study of the state-of-the-art techniques in real-time event detection in Twitter

- Elaborating a framework for cyber security events detection in Twitter
- Designing and implementing the proposed framework
- Validating the proposed approach using several detected events

1.4 Research Contributions

This section list the main contributions of this thesis regarding the objectives stated above. Our contributions in this thesis are the following:

- The design of a novel, automatic, self-learned, cyber security event detection framework from social media feeds
- A self-learned keywords mechanism based on word embedding to discover new relevant keywords over a corpus of short documents
- The development and implementation of this framework
- The evaluation of *SONAR* on 12 months of Twitter data demonstrating the effectiveness of the tool

1.5 Use Cases

SONAR can be used for many different purposes. The following use cases, were tested on real Twitter data for multiple months.

1.5.1 Cybersecurity Events

A cyber security situation awareness tool, provides to the expert a situation awareness capability on what is currently happening in the cyberspace and prioritize their actions toward the new threat. In addition, it is a tool for forensics to observe the chronology of cyber threats on social media and identify potential suspects (for example the first person talking about a newly data breach or malware). Finally, it is a very convenient tool for security teams as it can help them protecting their organization from newly discovered vulnerabilities.

1.5.2 Deceptive Marketing Practices

SONAR can be accommodated to detect deceptive marketing practices. It can identify potential illegal campaign in Twitter and the users promoting it. For example, it can be used to track the promotion of illegal or dangerous drugs advertised in Twitter. More information about this instance of *SONAR* can be found in Section 4.4.1.

1.5.3 Crypto Currencies Related Events

Initially created as a side project, this instance of *SONAR* was developed to detect every events about cryptocurrencies in Twitter. Cryptocurrency such as Bitcoin and Ethereum have large communities behind the project and Twitter is a widely used mean of communication between the participants. In that case, *SONAR* is particularly useful to get important information as soon as possible (possible fork, chain splits, contentious proposals, etc.). See Section 4.4.2 for more information about this instance.

1.5.4 Generic Events

SONAR is, by design, generic and can be adapted to detect any kind of events. As long as enough people discuss an event in a chosen sliding window, the tool will detect it. In this regard, despite the fact that, in this thesis, *SONAR* is accommodated to work only with tweets written in english none of the techniques used are language-specific and so every language is technically supported.

1.6 Thesis Organization

The remainder of the thesis is structured as follows. In Chapter 2, we discuss the related work and background knowledge used for this work. Then, in Chapter 3, we detail the architecture and design of our solution. Finally, in Chapter 4 we implement it and evaluate the performance of *SONAR* before presenting concluding remarks in Chapter 5.

Chapter 2

Related Work and Background

In this chapter, we first describe Twitter as a social network, its unique nature and its specific terminology. We then briefly describe two text mining concepts: text clustering and text classification. Finally, we discuss the current research on the topic of real-time event detection in Twitter.

2.1 Twitter Social Network

Twitter is a social network created by Jack Dorsey, Noah Glass, Biz Stone, and Evan Williams and launched in July 2006. In 2013, it was one of the ten most-visited websites in the world. Twitter, like other social networks such as Facebook, is mostly made out of user generated content. In 2008, Krumm [52] described user generated content as "information, data or media contributed voluntarily by regular people with the purpose of making it available to others in a useful or entertaining way, usually on the web". These type of social networks are particularly interesting for data mining research, however the content can be extremely noisy: posts are often spams, incomplete or written using slang [49] and one of the main challenge for data scientists lies in correctly separating the noise from the desired content.

2.1.1 Mechanism and Specific Terminology

On Twitter, any registered user can post a short message called a *Tweet* (Figure 2.1), this message cannot be more than 140 characters. Files like photos or videos can be added to the tweets and are

not counted is the number of characters. Users can also add URL, which are automatically shortened in order to save space using Twitter's URL shortening service. These hyperlinks usually start with "https://t.co/XX" where XX is a random sequence of characters.

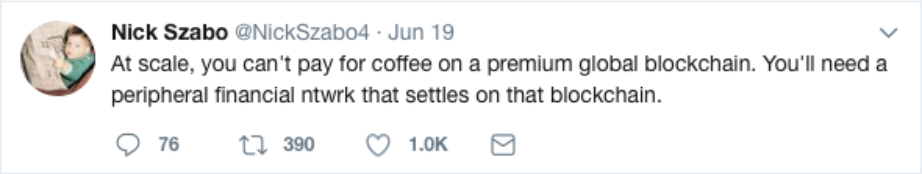

Figure 2.1: Screenshot of a Tweet Containing the Number of Replies, Retweets and Likes

As soon as they register on Twitter, the users are prompted to *follow* accounts they might be interested in (usually classified by category such as sport or high-tech). By following other accounts, the users create their own network (which forms a simple directed graph) and subscribe to the other users' Tweets. At first the connection is unilateral, followed accounts can decide to follow back if they are interested in the content of their follower. Each Twitter account maintain a count of *followers* (the number of people following the account) and *following* (the number of people the account is following), see Figure 2.2 for an example. When they register, Twitter users can write a short description about them and precise where they live.

Figure 2.2: Screenshot of the Twitter Account of Barack Obama

Twitter accounts are by default public and anyone can see a user's post. Users can interact by replying to each other's tweets with the *reply* button or by writing a tweet starting with a "@" and the user screen name. They can also send a private **direct message (DM)** to other accounts which are also constrained within 140 characters.

User can also **mention** other user in their tweet by writing "@" followed by the user screen

name. A mention is constructed the same way as reply, but the Twitter handle is not the first thing in the post. When a Twitter user want to share another user tweet with his followers, he can *retweet* the tweet. In general popularity of a tweet is measured by the number of retweets.

Finally, the last important concept in Twitter is the *hashtag*. A hashtag is a word or a sequence of word preceded by a hash sign (#). Hashtags are used to identify tweets on a specific topic. Twitter automatically uses these hashtags to create geolocated *trending topics*: a list of the most used hashtags per place (see Figure 2.3). This is especially useful to quickly identify which subjects are currently discuss on the social network.

Figure 2.3: Screenshot of the Worldwide Trending Topics in Twitter

Tweets can also be *geolocated*. On mobile client, a user can share his exact location (GPS coordinates) or manually select a place near him. On desktop, users can decide to turn on their location and choose from a predefined list of cities around them.

In January 2017, Twitter had 317 million monthly active users, 500 millions tweets per day with 80% of the twitters tweeting from mobile devices[1].

2.1.2 Working with Twitter Data

Twitter provides several Application Programming Interfaces (APIs) to get tweets:

[1] See `https://www.omnicoreagency.com/twitter-statistics/`.

- A REST (REpresentational State Transfer) API[2] designed to read and write Twitter data. With this API a developer can control his own account and access Twitter Data such as other persons tweets or descriptions. This API also contains the Twitter Search API, allowing "queries against the indices of recent or popular Tweets and behaves similarly to, but not exactly like the Search feature available in Twitter mobile or web clients, such as Twitter.com search". This last API will not retrieve tweets older than 7 days.

- A Streaming API[3] which gives developers access to realtime data in Twitter such as a 1% sample of all the tweets. This API also allows developers to get all the tweets from a user or for a specific set of keywords (up to 1% of the Twitter volume).

- The GNIP API[4] is a pay-per-use API. GNIP, a company purchased by Twitter in 2014, is specialized in social media API aggregation. GNIP API for Twitter is the same as REST and STREAMING APIs without any limitations.

2.1.3 Twitter, a Social Network?

In 2010, Kwak et al. [53], conducted an extensive analysis of the social network: the first quantitative study on Twitter. Among other things, they studied the Twitter follower graph and found that, because of the low rate of reciprocated ties, Twitter is more an information sharing network than a social network. Nowadays, Twitter is mainly constituted of four types of account:

- Bot created by developers for a precise function (e.g., Bitcoin ticker bot will tweet every hour the price of Bitcoin in USD).

- Real accounts owned and maintained by one person.

- Fake accounts, created for different purposes(e.g., spam other people, for a product).

- Company and news media account.

To conclude, people tend to use Twitter as a news feed by following multiple online news media but other Twitter users will only follow "real" users.

[2] See `https://dev.twitter.com/rest/public`.
[3] See `https://dev.twitter.com/streaming/overview`.
[4] See `https://gnip.com`.

2.2 Text Clustering

Text clustering is the task of grouping a set of document in a way that documents from a same group (a cluster) are more similar to each other than those in other clusters. The goal is to find group of similar objects in the data. Usually, the similarity between the objects is measured with the use of a similarity function. Clustering is very useful in the text domain where clusters can be formed from documents in a corpus, sentences and even n-grams. Text clustering can be used for a large number of tasks:

- **Document Organization:** Text clustering is useful for organizing documents into hierarchical categories. Scatter/Gather [33] is a classic example of document browsing system based on clustering, using a hybrid approach (K-means and agglomerative hierarchical clustering).

- **Topic Modelling:** A group of highly used methods to extract the "topics" of a corpus (a group of document). One of the most famous is the latent Dirichlet Allocation (LDA) [21], a generative statistical model that represents documents as a set of topics they belong to.

- **Corpus Summarization:** Clustering methods can help create summary of the collection such as *word-clusters* [15] or *cluster-digests* [91]

It is worth noting that most of the algorithms used (e.g., k-means) are general purpose algorithms and can be applied to any kind of data. An important number of clustering algorithms exist. However, selection of one of them is based on various parameters such as:

- **Online or Offline:** In the case of an online problem, the algorithm must be able to process each new document without the need to read the whole corpus from the start. In practice, computation time will determine whether the data scientist will use an offline or online algorithm. In the case of stream of documents, online algorithms tend to be more efficient.

- **Size of the corpus:** An important factor in the choice of the algorithm as certain clustering techniques will perform better on large corpus than on smaller ones.

- **Dimensionality of each document**: Another important factor is the size of each document. Documents such as tweets will not be processed the same way large paragraphs are.

- **Size of the lexicon:** The number of unique words that should be used to represent each document. The number of entries in the document vector is directly linked with the number of word in the lexicon. This can cause some troubles when the vocabulary needs to stay large.

Independently of the chosen algorithms the following procedure is almost always applied:

(1) **Tokenization:** The process of parsing each document into smaller units. Bag-of-words and N-grams model are two commonly used tokenization techniques.

(2) **Stemming or lemmatization:** This step is designed to removed similar tokens in order to reduce the computation cost. For example word like "cats" and "cat" carry out the same information.

(3) **Removing stop words and ponctuation:** In order to keep only the relevant tokens, stop-words (e.g., "a" and "about" do not add any information to the document they belong to). Ponctuation is removed as well for the same reason.

(4) **Vectorization:** Preprocessed tokenized documents are vectorized using the term frequency or tf-idf (term frequency-inverse document frequency). Both techniques where used in this thesis.

(5) **Clustering:** The core algorithm, documents are clustered based on the features generated.

(6) **Evaluation:** We evaluate the result of the clustering algorithms and adapt the different parameters if necessary.

In Chapter 3, we will explain how we use the previous methodology in our work.

2.3 Text Classification

Classification is a form of supervised learning where the features of the documents are used to predict the "type" of document. It's the task of taking a document $d_j \in D$ – where D is a set of documents – and assigning a class $c_k \in C$ – where C is a set of predefined classes. Basically, we

are looking for a function γ with the following definition:

$$\gamma : \mathbb{X} \rightarrow \mathbb{C}$$

Where $\mathbb{X}$ is the space of the document and $\mathbb{C}$ the space of all the classes. This γ function is called a *classifier*. Text classification is one of the current hot topic because it can be applied to a lot of different problems such as spam filtering, document categorization and hierarchical categories organization. Popular classification algorithms are Naive-Bayes [119] and Support Vector Machines (SVM) [31]. The previous clustering methodology can be used for classification except that step (5) should be replaced with a classification algorithm.

2.4 Event Detection on Social Network

In this section, we review the research of event detection in social networks. We first investigate the state of the art on event detection in social networks, then we focus on research publications that have been published since since 2009.

2.4.1 State of the Art

With opinion mining, event detection is one of the most studied problem in social network analysis. More specifically, since 2009, approximately 73 papers were published, combining more than 10,000 thousand citations, on the topic of event detection in social networks. Figure 2.4 shows the number of paper and type of paper published every year. 2013 was the year with the most publications on the subject, which is still beeing studied until now.

A total of two surveys have been published in 2012 and 2017. Ten studies on event detection in social networks were published between 2010 and 2015. The first and only corpus of event detected in social networks was published in 2013. Finally, with up to 15 per year, the most published type of paper is framework/algorithm for event detection in social networks. Figure 2.5 classify each paper by type.

In the following subsections, we will detail each category and the work done in each paper.

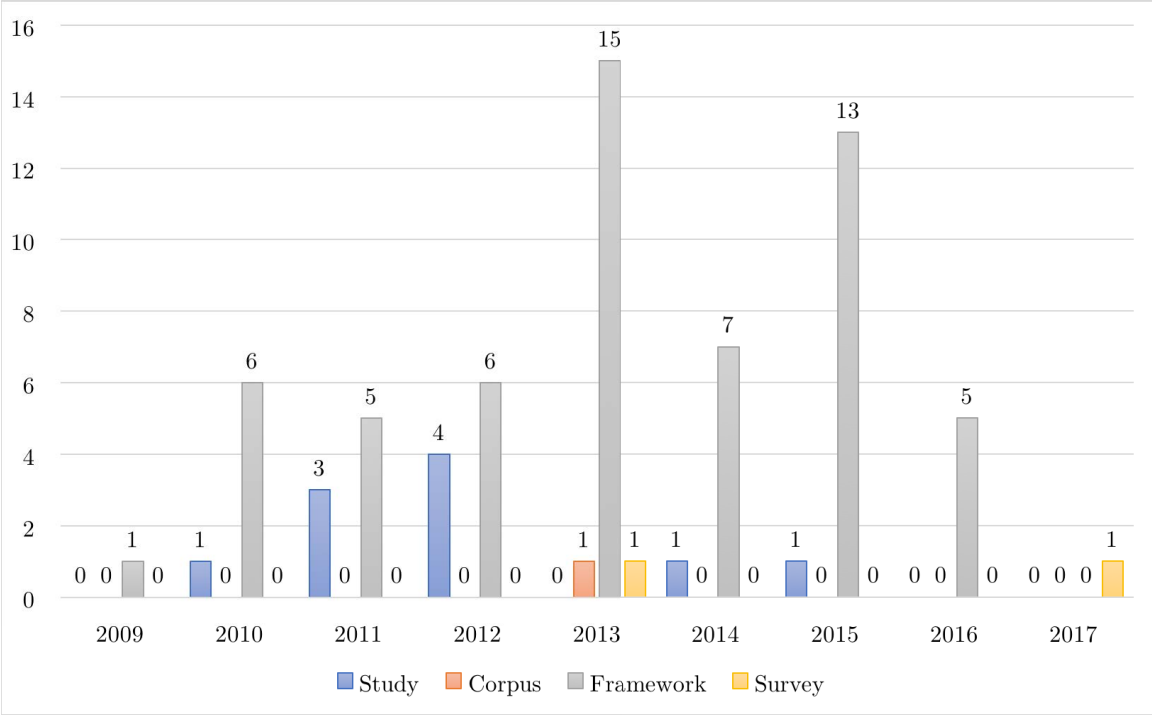

Figure 2.4: Number of Paper on Event Detection Using Social Networks Published per Year

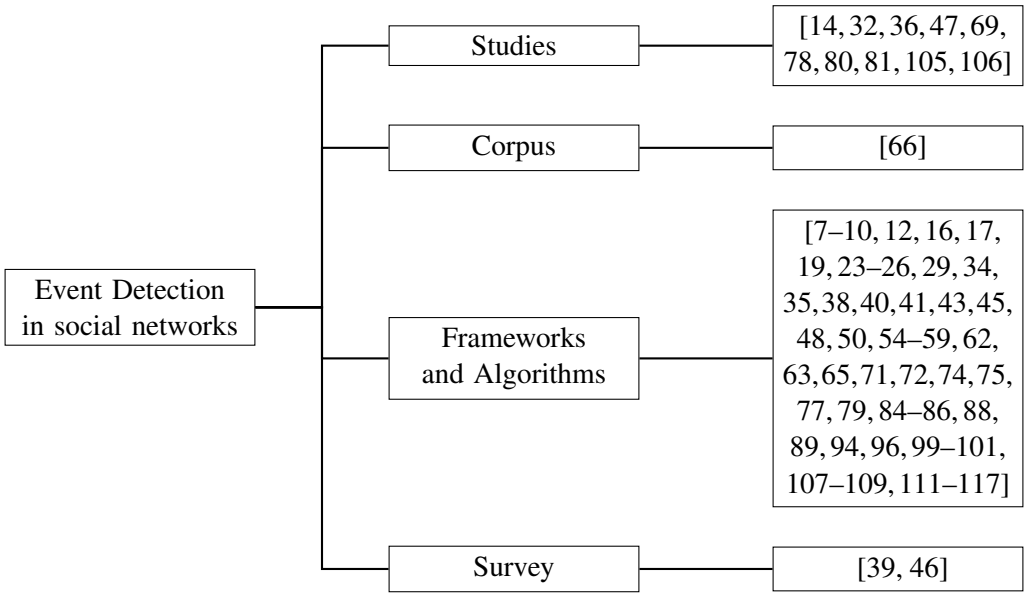

Figure 2.5: High-Level Classification of the Current Literature on Event Detection in Social Networks

2.4.2 Studies

Approximately, 11 studies were published in social networks and real-time event detection. This studies provide critical insights on the validity of the approach. Questions such as "Are social networks really efficient for this task?" or "Are social media more informed than traditional news outlet?" are typical questions answered by these studies. We separate the studies into three categories (see Figure 2.6). The first one contains all the studies that investigate the Twitter response during or after an event. The second one contains studies that compare how information spreads in Twitter compared to traditional news outlet and other social media. The last one contains all the miscellaneous studies.

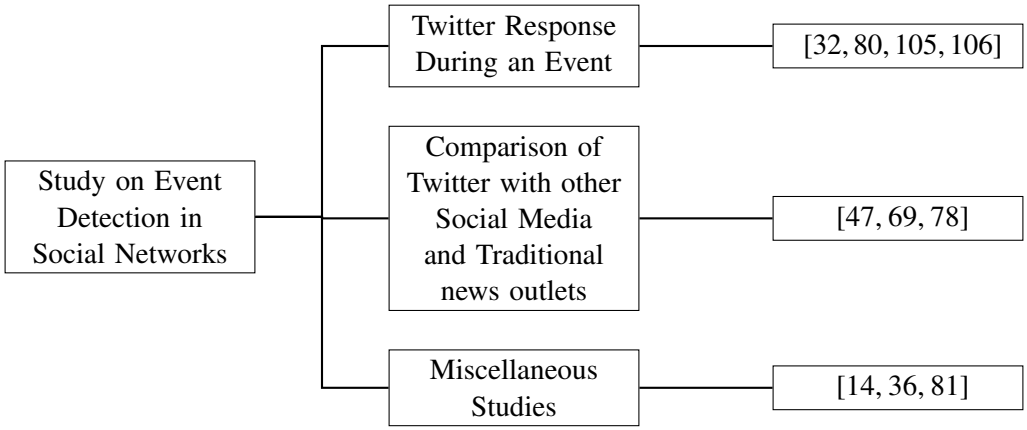

Figure 2.6: Classification of the Current Studies on Event Detection on Social Media

Twitter Response During an Event

One of the first work of this type was done by Vieweg et al. [105] in 2010. They analyze tweets posted during two natural disasters, the Oklahoma Grassfires and the Red River Floods that happened in March and April 2009. This work aims to highlight the use of Twitter posts to improve situational awareness in emergency situations.

In 2013, Crooks et al. [32], analyze the temporal and spatial characteristics of the Twitter activity during a 5.8 magnitude earthquake on August 23, 2011 in the United States. They find that people act as sensors in this type of situation and can be used to complete or enhance other sources of data. In 2015, Preece et al. [80] use this "human as sensors" to demonstrate the usefulness of Twitter

data in situational awareness. Finally, Vis [106] highlights the use of Twitter as a reporting tool by journalists which is an important aspect in this thesis.

Comparison of Twitter with other Social Media and Traditional news outlets

In 2012, Hu et al. [47] published "Breaking News on Twitter". In this paper, they proove for the first time the belief that Twitter broke the news first and that people were convinced of the news before it appeared in traditional news outlets. One year later, in 2013, Petrović et al. [78] find that, when dealing with major news events, Twitter is not leading mainstream media.

In 2014, Osborne et al. [69] compile the first comparative study between Facebook, Google Plus and Twitter regarding breaking news. They find that, while all three social networks carry the same major events, Twitter is most of the time ahead of Facebook and Google Plus. The laters are mostly used for reposting newswire stories.

Miscellaneous Studies

In 2012, Dou et al. [36] summarize and explore the popular tasks in the domain of event detection: New Event Detection, Event Tracking, Event Summarization and Event Association. This work was also one of the first quick survey of the state of the art of event detection in social networks data at that time. The only technical study was done by Quercia et al. [81] in 2012, where they run an extensive evaluation of Labeled Latent Dirichlet Allocation (L-LDA) in the task of supervised topic classification in Twitter. Finally, in 2013, Ruhela et al. [14] published the first comprehensive study on the diffusion of ideas in Twitter.

2.4.3 Surveys

Two surveys were published on event detection in social networks. In 2013, Atefeh and Khreich [39] published a survey of techniques for event detection in Twitter. They classify the publications using three main components: (1) the type of event that needs to be detected (specified or unspecified), (2) the detection method (supervised, unsupervised and hybrid), (3) the detection task (Retrospective Event Detection, i.e. clustering techniques, and New Event Detection). Despite being an extremely interesting survey from a research standpoint, they logically focuse on publications

of 2009 and 2012. Since 2013, several papers (more than 40) were published introducing improved and novel techniques. For example, the work of Gaglio et al. [40], Osborne et al. [71], McMinn, A. J. and Jose, J. M. [65], Liu et al. [61] and Madani et al. [62] are perfect examples of what is being done since 2013.

In 2017, Hasan et al. [62] published "A survey on real-time event detection from the Twitter data stream" which provides an up to date survey on the topic. In contrast to the work conducted by Atefeh et Khreich, they choose to classify the research based on the common traits of their classification method. Moreover, while the work of Atefeh et Khreich focus on 16 papers, Hasan et al. choose to present more than 70 research papers. The classification made by Hasan et al. is interesting and will be partly used in this thesis. However, we notice its lack of precision in the classification of the algorithms used for event detection. In Subsection 2.4.4, we tried as much as possible to categorize as precisely each paper.

2.4.4 Frameworks and Algorithms

Framework and algorithm proposals are the most common type of papers in the field of event detection in social networks. We decided to group algorithms with frameworks as most of the time, a framework is introduce along with a new algorithm. We classified these papers in 7 categories (see Figure 2.7). After presenting the general architecture of the frameworks we will detail the paper of each of these categories.

General Architecture

Almost all real-time event detection system in social networks share the same common high level architecture. Figure 2.8 presents this architecture and the common steps associated. We will see later in this thesis (Chapter 3) how we adapte this model to the problem.

The first step is to retrieve the documents using the social network API, several options can be supplied to apply a first pre-filtering. Researchers can select a language, keywords or a list of users to follow. Once retrieved, the documents need to be process which means they will be tokenize, stop words will be removed and the documents will be possibly term-frequency - inverse document frequency weighted. This pre-processed documents is then used in the event detection algorithm.

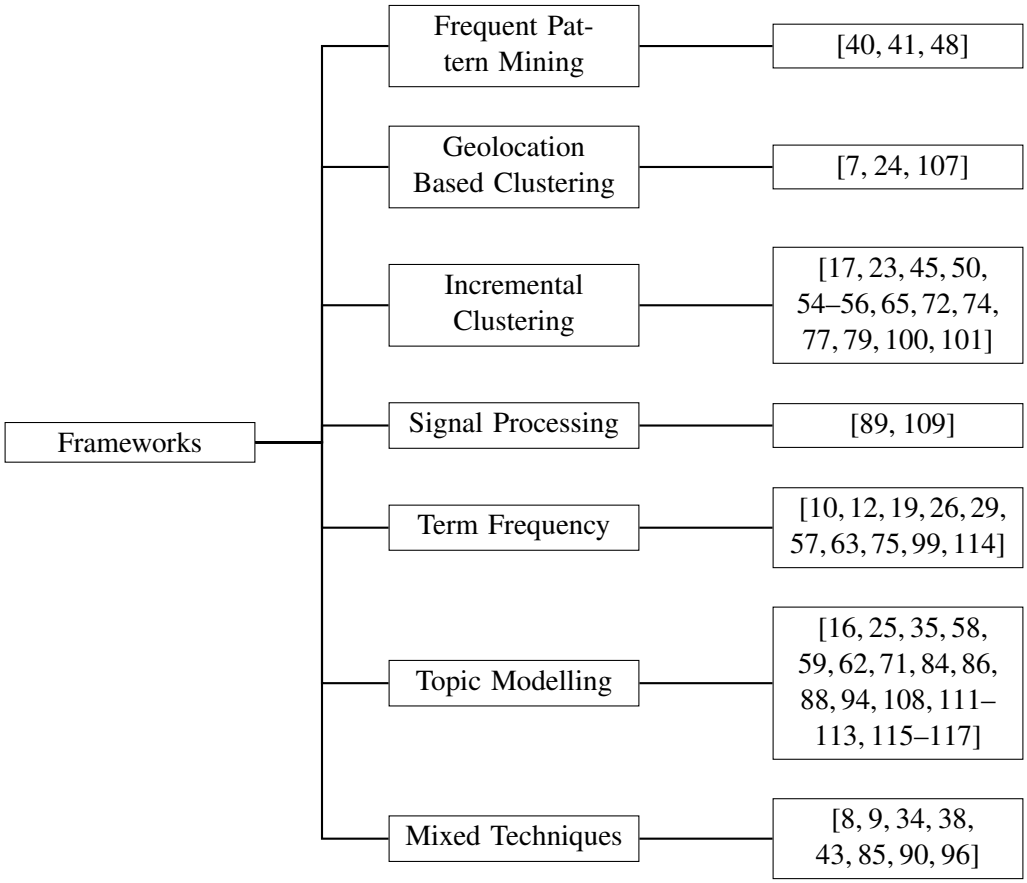

Figure 2.7: Classification of the Framework for Event Detection in Social Networks

This component output usually a list of detected events. Lot of real-time event detection systems rank and/or summarize this list of events before displaying them on an interface.

In this survey on the event detection framework in social networks, we logically focus on event detection part.

Topic Modeling Based Approaches

Topic modeling techniques are one of the most common approaches for event detection in social networks. A topic model is a statistical model used in text mining to link each document with a mixture of topics. Among the 17 frameworks published using this model we are able to identify 3 different approaches (see Figure 2.9).

(1) **Latent Dirichlet Allocation-Based Models**

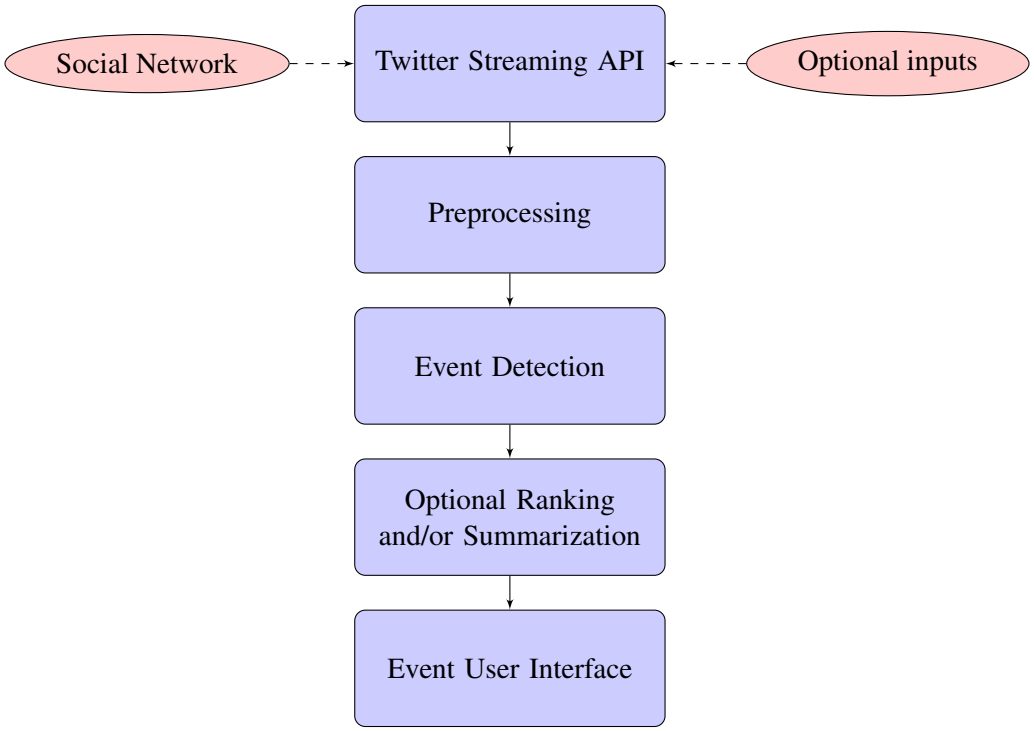

Figure 2.8: Common Architecture of Real-Time Event Detection System using Social Networks

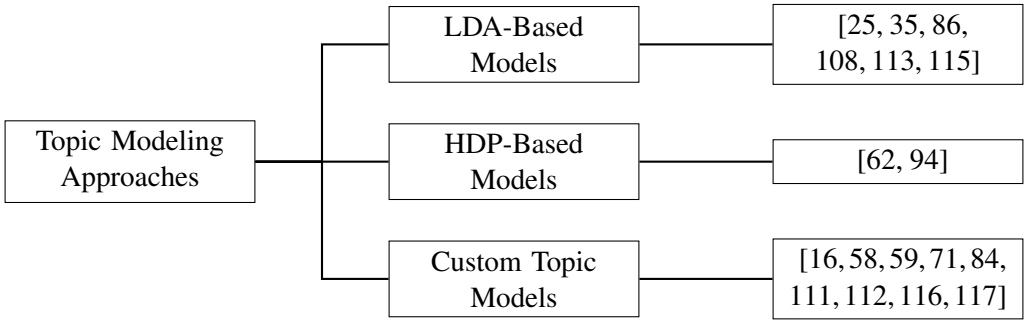

Figure 2.9: Classification of the Topic Modeling Based Approaches for Event Detection in Social Networks

The first work on topic modeling with a LDA-based model is from Wang et al. [108] in 2010. They design a framework where topics are automatically extracted from the Twitter stream of CBS19 Charlottesville and then, with a generalized linear regression model (GLM), are used to predict crime in this city.

In 2011, Zhao et al. [115], designed a model called Twitter-LDA. This model, inspired from LDA, is able to deal with short texts and assumes that each tweet is about a single topic. Then,

they compare the events found with their model with events in the New York Times.

In 2013, Diao et al. [35] proposed a model combining the Recurrent Chinese Restaurant Process and an LDA-based model to find bursty events on the Twitter stream. While their framework is made to be real-time, their evaluation is based on a dataset of tweets from user in Singapour between April 1 and June 30, 2012. Romsaiyud [86] and Cai et al. [25] both designed a complete framework using the geolocation of each Twitter user combined with an algorithm based on LDA to detect and identify events.

Finally, You et al. [113] designed in 2013 a "general and event-related aspects model for Twitter event detection" called GEAM. Similar to LECM [116] but differs in the fact that each tweet is also modeled on general topics in addition to be modeled on event-related aspects such as entities, location and time.

(2) **Hierarchical Dirichlet Process-Based (HDP) Models**

More recently, Shepard [94] and Madani et al. [62] published event detection frameworks in Twitter using Hierarchical Dirichlet Process. One of the biggest drawback of LDA is that the number of topic k is a prerequisite. HDP-Based model are designed to discover this variable automatically. In the case of the Twitter Stream where there is no prior knowledge of the number of topics, this algorithm is very useful. While Madani et al. used HDP as is, Shepard designed a variant of the model using Nonparametric Pachinko Allocation Method (NPAM) [60] to detect nested topics.

(3) **Custom Topic Models**

While LDA remains the most widely used topic model, researchers are creating custom topic models in order to enhance event detection in Twitter. ReDites [71] by Osborne et al., use a model very similar to LDA derived from the Violence Detection Model [16], a weakly supervised Bayesian model, to detect security related events.

TwiCal [84], by Ritter et al., is an "open-domain event extraction and categorization system for Twitter". This system will show the most prominent events mentioned with each day (e.g., June 21 = Summer) using tweets. Similarly, the Latent Event and Category Model (LECM)

[116] propose a latent variable model which models tweet as a distribution over data/time, location of the event occurence, term related to the event and named entities. Additionally, the Location-time constrained topic (LTT) model similarly exploits the geolocation and time of the event.

CLEar (Clairaudient Ear) [111] is another framework making use of a topic model to automatically detect events. The model,TopicSketch [112], is a sketch-based topic model designed to detect topics in real-time. It works by maintaining a new data sketch which is able to detect peak of: frequency of tweets in the stream, frequency of words and pair of words.

Finally, BEE [58] and BEE+ [59] are frameworks to detect bursty events. A bursty event is defined as a topic that is contained in an important number of posts during a certain time-frame. In contrary with most of the previous work, the framework designed by Li et al. was evaluated on Weibo, the Chinese equivalent of Twitter.

Signal Processing Based Approaches

In this category, we grouped all the publications where the framework used signal processing algorithms, such as Kalman filtering, to detect events in social networks. While not being very popular these methods can achieve some promising results.

In 2010, Sakaki et al. [88, 89] (one of the most cited work in this domain at writing time of this thesis) introduced a real-time system to detect earthquake and warn people that are endangered to seek shelters. An idea also used by Crooks and al. [32] in 2013. They consider each Twitter user as a sensor and apply algorithms such as Kaman Filtering and particle filtering to estimate the location of the earthquake. Their framework is designed to capture keywords such as “earthquake” or “typhoon” on the Twitter stream and using the frequency of these tweets and a statistical model, they are able to detect and locate on going earthquakes.

In EDCoW (Event Detection with Clustering of Wavelet-based Signals) [89], Weng et al., present a new system to detect events on the Twitter Stream. Using wavelet theory [102], a signal is created for each word which are then clustered in order to detect bursty terms. While interesting, the wavelet generation algorithm for each word and the correlation calculations are computationally

expensives.

Geolocation Based Approaches

Like signal processing, geolocation based approaches are uncommon methods among the corpus of frameworks we gather. While, limited because only a few percents of the users share their locations, these approaches can be useful for detecting real life events such as natural disasters.

In 2013, Walther et al. [107], designed a framework that cluster tweets in the same area during the same time window. It allows them to detect events such as house fires, traffic jams and demonstrations in real-time. Finally, they train a classifier with more than 41 features (such as tweet count, sentiment, etc.) to assess if the cluster detected is an event or not. This framework is very similar to EvenTweet [7] which contains a similar location-based event detection algorithm. One year later, in GeoScope [24], Budak et al. proposed a very similar approach. Because tracking every location-cluster is computationally expensive, they propose a simplified approach based on a hashtable and a sketch structure.

Frequent Pattern Mining Based Approaches

Pattern mining, a subfield of data mining, consists of designing algorithms that can detect interesting and useful patterns in a database. Pattern mining can be applied to almost any kind of database but is mostly used on corpus.

In 2016, Gaglio et al., created the Twitter Live Detection Framework (TLDF) [41]. They designe a modified Soft Frequent Pattern Mining (SPFM) algorithm [40]. This algorithm regroups related term in a topic after selecting the top-K term using a scoring measure. When evaluated on a dataset along with TwitterMonitor [63] and enBlogue [12], TLDF outperforms them in accuracy (up to 60% improvement).

Finally, Huang et al. [48] presented in 2015 a high utility pattern clustering (HUPC) framework for topic detection on the Twitter stream. High utility pattern mining [87] is a research topic in pattern mining designe to improve traditional frequent pattern mining that takes “only binary databases and identical item importance into consideration”. Patterns are created using the FP-Growth algorithm [44] which are then classified as high utility pattern based on the importance of the terms it

contains. This method outperforms algorithms such as online LDA and SPFM.

Incremental Clustering Approaches

Incremental clustering algorithms are clustering algorithms that process the data usually in one pass and one element at a time. This types of algorithm are particularly useful for online processing since, because of the very high volume, it is no longer possible to keep the data in memory at the same time. In this section, we present each framework (see Figure 2.10) according to the type of incremental clustering they use : density based clustering, leader-follower clustering and sequential nearest-neighbour clustering.

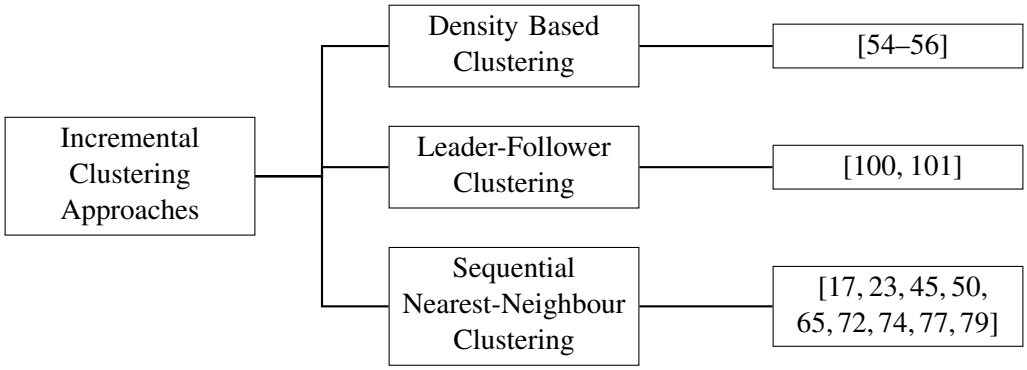

Figure 2.10: Classification of the Incremental Clustering Based Approaches for Event Detection in Social Networks

(1) **Density Based Clustering**

Density-based spatial clustering of applications with noise (DBSCAN) is a data clustering algorithm proposed by Ester et al. [37] in 1996. Given a set of points in space, this density based algorithm clusters the points that are closed together and identifies outliers.

With 3 papers on event detection in social networks, Lee and Chien [54–56] developed an incremental clustering algorithm based on DBSCAN called IncrementalDBSCAN. Using a sliding window, tweets are vectorized using a dynamic term weighting technique then IncrementalDBSCAN automatically groups clusters of tweets into events. In contrary with DBHTE (Density-based Hot Topic Extraction) [56], incrementalDBSCAN does not need the number of topics as prerequisite.

(2) **Leader-Follower Clustering**

This algorithm, introduced by Shah and Zaman [93] in 2010 was initially created for the purpose of community detection in networks.

Unankard and Sharaf [100, 101] propose to adapt this algorithm for detecting emerging event in social networks. Their approach, called Location Sensitive Emerging Event Detection (LSED) detects emerging hotspot events by identifying correlations between the user locations and the event locations. Tweets are clustered when the timestamp of the coming message is greater than the sliding window size. Cosine similarity measure is used to compute the content similarity. Finally, to reduce duplication, cluster top terms are extended with WordNet [68] and compared between each other using Jaccard similarity measure [98].

(3) **Sequential Nearest-Neighbour Clustering**

Sequential nearest-neighbour clustering is the most popular clustering technique for event detection in social networks. It is an efficient, easily understandable and somewhat inexpensive way to cluster online stream. Closely related to nearest neighbor search, the algorithm sequentially considers every points and compare them between each other using a similarity measure such as cosine similarity. Implemented in several frameworks [17,23,45,50,65,72,74,77,79], we present here the three major contributions in the field.

In 2010, Petrović et al. [77], presented a new method to do incremental clustering with application to Twitter. Their approach use locality-sensitive hashing in order to find the nearest neighbors and hence reduce the complexity of the sequential algorithm. Their work is described in details in Section 3.5.3 later in this thesis.

The same year, Phuvipadaw et al. [79] present HotStream: "a method to collect, group, rank and track breaking news in Twitter". Tweets are represented using *tf-idf* and a custom weight is applied on proper nouns. Instead of using classic similarity measure such as cosine distance,

they designed their own similarity formula:

$$\text{sim}(m_1, m_2) = \sum_{t \in m_1} \text{tf}(t, m_2) \times \text{idf}(t) \times \text{boost}(t)$$
$$\text{tf}(t, m) = \frac{\text{count}(t \text{ in } m)}{\text{size}(m)}$$
$$\text{idf}(t) = 1 + log\left(\frac{N}{\text{count}(m \text{ has } t)}\right)$$

where boost(t) is increase for proper nouns and artifacts like hashtags.

Last important contribution in the domain of sequential clustering for event detection in social networks is TwitterNews+ [45]. Most notably, the framework proceed in two stages for the event detection algorithm. First a search module that retrieve tweets containing bursty keywords. Secondly, an EventCluster module which will cluster *tf-idf* weighted tweets using a sequential nearest-neighbour clustering algorithm similar to the one designed by Petrović et al. , first story detection. Information are finally cross correlated between the two stages and displayed on an user interface.

Frequency Based Approaches

Frequency based approaches regroup all the frameworks that use the frequency of appearance of a word/n-gram or any other indicator based on the frequency in the Twitter Stream to detect events. EnBlogue [12], ET [75], Twevent [57] and TwitterMonitor [63] are all frameworks design to detect events or topics using frequent terms or bursty keywords. In this subsection, we highlight three frameworks that use frequency-based approaches for event detection.

TwitterMonitor [63] published by Mathioudakis et al. in 2010 is one the first event detection framework in Twitter that uses a term frequency based approach to detect events. They focus on "bursty keywords", keywords that appear at unusually high rate in the stream. These keywords indicate new emergent topic, they are detected using an algorithm called QueueBurst which is one-pass (particularly useful in real-time system).

EnBlogue [12], by Alvanaki et al., is based on a three-steps algorithm. First, the algorithm detects seed tags which are high frequency keywords during a sliding window. Then, tag pairs are

created to form candidate topics. Finally, sudden increase in correlations of tag pairs signals that an emergent topic has been detected. In 2015, Zhang et al. [114], proposed a new approach to detect bursty events and predict their future popularity. Each document mc is described as follows:

$$mc_j = (w_{j,1}, w_{j,2}, \ldots, w_{j,l_t})$$

where l_t is the dimension of the set of words used in the time-frame t. The term $w_{j,v}$ represents the weight of the v^{th} vocabulary word in mc_j. This weight is computed using the following formula:

$$w_{j,v} = \left(0.5 + 0.5 \times \frac{tf_{j,v}}{tf_j^{max}}\right) \times au(user(mc_j))$$

where $tf_{j,v}$ is the term frequency of the v^{th} word in mc, tf_j^{max} is the highest term frequency in mc_j and $au(user(mc_j))$ is a metric to measure the influence of a user based on the number of followers.

Mixed Techniques Approaches

Some of the frameworks cannot be classified using the previous categories either because they are combining the previous approaches or are using a completely different method for event detection in social networks.

Most notably, in regards to our work on *SONAR*, Ritter et al. published in 2013 "Weakly Supervised Extraction of Computer Security Events from Twitter" [85]. In this work, they proposed a weakly supervised classifier which can be easily trained on new types of event categories using seed examples.

In 2016, Dang et al. [34] designed a framework for early detection method of emerging topics using a Dynamic Bayesian Network. They found that emerging topics are popular based on two metrics, *attractiveness* and *key-node* (influential users who can cause a large amount of retweets). Using this features, they build a Dynamic Bayesian Network able to calculate the probability of a keyword being an emerging one.

2.4.5 Corpus

In 2013, McMinn et al. [66] proposed a methodology for the creation of an event detection corpus. Using Locality Sensitive Hashing by Petrović et al. [77] and the Cluster Summarization approach proposed by Aggarwal and Subbian [9] with a ground truth from Wikipedia Current Events Portal, they provided a corpus of verified event covering a period of 4 weeks and containing over 120 million tweets. To the best of our knowledge and at the time of writing this thesis, this work is the only publicly accessible corpus of detected events verified with ground truth.

Chapter 3

Architecture and Design

In this chapter, we detail the architecture and design of *SONAR*. We start by presenting a high level overview of the framework, then we presented our taxonomy and how we collected data through the Twitter API. Then, we describe the data preprocessing. Finally, we details our two algorithms: event detection and keywords finder algorithms.

3.1 Approach

In Figure 3.1, we present a high-level overview of the framework architecture. The inner workings of *SONAR* can be broken down into 3 phases. In phase 1, the *stream generator* makes use of Twitter API and a list of keywords to retrieve a continuous stream of tweets (see Section 3.3). This stream of tweets is preprocessed and then stored in a database. Then, in phase 2, the *event detection* algorithm detects events over the Twitter stream (see Section 3.5.3). Events are composed of a set of tweets discussing the same topic during the same timeframe. These events are geolocated, classified and displayed on an intuitive user interface. As we use predefined keywords to create the stream in phase 1, the system is unaware of new keywords such as new malware naming or new type of DDoS attacks. Therefore, in order to get those documents, we created a component that track those new security keywords automatically and add them in the previous list of keywords: the *keyword finder* (see Section 3.6). This component discovers new relevant keywords among the previous tweets and add them to the list of keywords used by the *stream generator* (phase 3). Subsequently, the stream

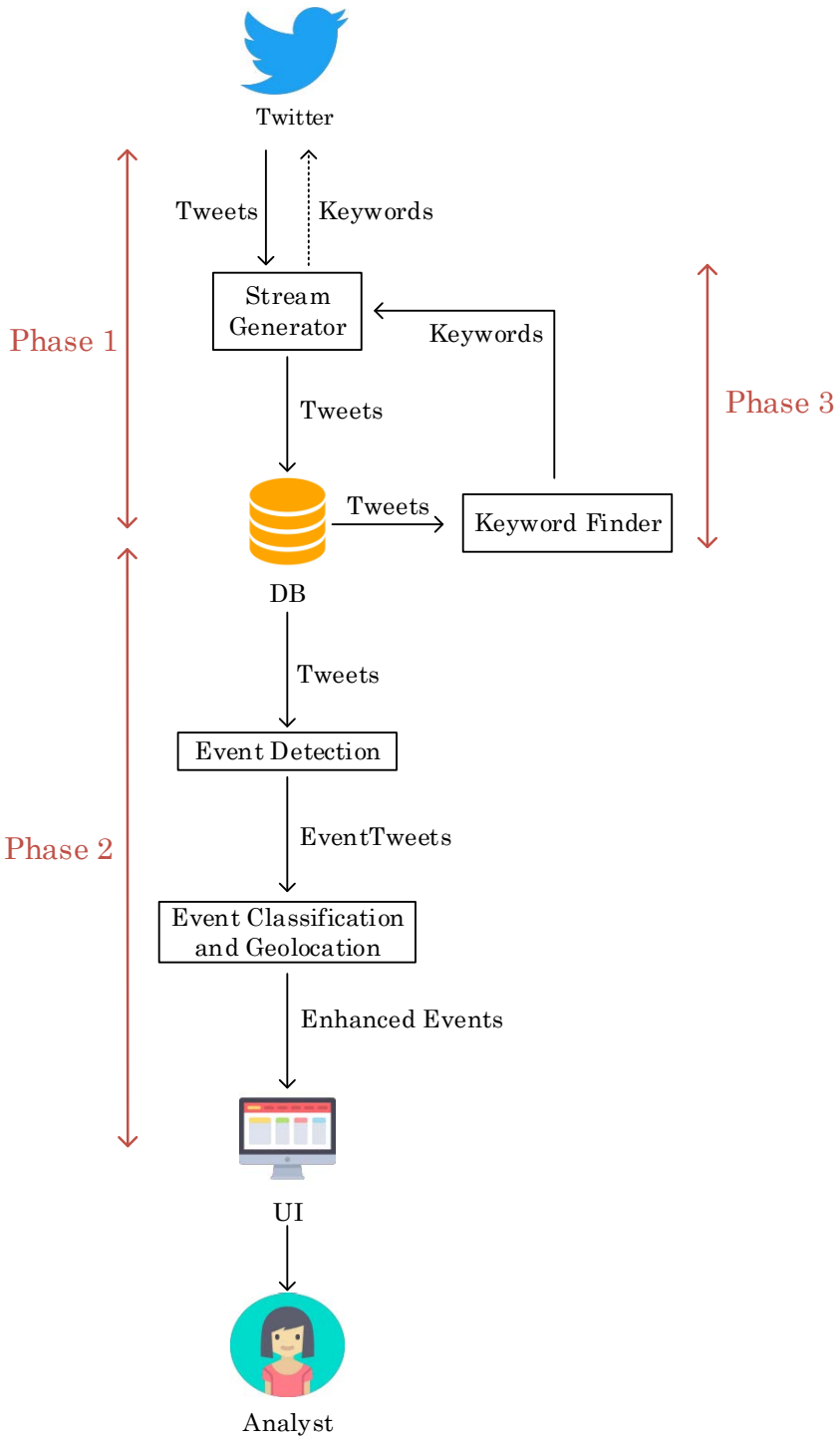

Figure 3.1: System Overview of SONAR

is enriched with tweets containing the discovered keywords. Next section will introduce in details each previous component.

3.2 Taxonomy

There have been several attempts to create cyber security related taxonomies. There are taxonomies for cyber attack in general [27, 95] or on SCADA systems [118], 3G networks [51] and even cyber conflict taxonomy [13]. However, in our case, we were looking for a more general and less specific cyber attack taxonomy. A classification closer to what is discussed on social media: from general account hacking to specific vulnerabilities such as SQL injection. Table 3.1 represents the taxonomy created for *SONAR*. We subdivided it into 5 levels. Level 1 is the main level, the subject of our taxonomy: cyber security. Level 2 contains the six main categories: DoS, Botnet, Malware, Vulnerability, Social Engineering and Data Breach. The taxonomy in its current state can go as deep as level 5 (e.g., "SSDP" flood).

Each taxon, independently of its level, is associated with a set of keywords. Keywords can go from unigram to n-gram. For example, the taxon "APDoS" which stand for "Advanced Persistent Denial of Service" is associated with 4 keywords: "Permanent DoS", "Permanent Denial of Service", "PDoS" and "Phlashing" (a denial of service attack that exploits a vulnerability in network-based firmware updates). The taxon "SQL Injection" is associated with 3 keywords : "SQL Attack", "SQLi" and "SQL Injection". Every tweet containing one or more of the previous keywords will be retrieved. These keywords must be carefully selected in order to false positives: analysts must avoid loosely related (e.g., virus) or imprecise keywords.

The initial keywords provided by the analyst will be called *seed keywords*. Using keyword stemming, we deal with cases such as plural form and "hashtag" form (without space e.g., "denialofservice"). We will see later in the thesis how *SONAR* deals with these keywords.

[1]Flooding Amplification Reflection

Denial of Service	**Botnet**
⋆ Permanent DoS	**Malware**
⋆ Distributed DoS	⋆ Adware
▷ APDoS	⋆ Ransomware
▷ F./A./R.[1]	⋆ Rootkit
· BitTorrent	⋆ Trojan Horse
· CharGen	⋆ Virus
· DNS	⋆ Worm
· ICMP	**Vulnerability**
· Kad	⋆ Backdoor
· Netbios	⋆ Buffer Overflow
· NTP	⋆ Cross Site Scripting
· QOTD	▷ DOM-Based XSS
· Quake Network	▷ Reflected XSS
· SNMP	▷ Stored XSS
· SSDP	⋆ SQL Injection
· Steam	**Social Engineering**
▷ Slow Read	⋆ Phishing
▷ SYN Flood	**Data Breach**

Table 3.1: Cyber Security Taxonomy

3.3 Data Collection

Using Twitter API and the list of seed keywords (in our case, a list of 200 cyber security related keywords from basic unigrams to more complex 6-grams), we query Twitter. The result is a stream of tweets containing at least one of the previous keyword. Using a list of keywords to query the Twitter stream has multiple benefits. First, we avoid a lot of noises: we receive only tweets that contain the relevant keywords hence that are highly probable to be relevant. Secondly, because of the API limitation, we can only get up to 1% of the Twitter stream, which is approximately 1 million tweets per day. With this method, we ensure that we get as most relevant tweets as possible. In our case, we get approximately 200,000 tweets per day which is roughly 2 tweets per second. However this number vary greatly depending on the events occurring. It can go as low as 67,000 tweets per day when nothing significant is happening to almost 500,000 tweets when Wikileaks releases the hacked DNC voicemails on July 27th, 2016.

Finally, the tool used a list of blacklisted n-gram in order to avoid false positives. Tweets containing these forward n-grams will be automatically discarded and not stored in the database.

For example, a document containing the expression “hacked to death” is irrelevant despite the fact that it contains the keyword “hacked”. In the example, the n-gram to blacklist is “hacked to death”. This blacklisted keywords were added after evaluation of the tool. This basic keyword filtering is a quick and simple way to discard irrelevant keywords

3.4 Data Preprocessing

Before being processed by the event detection algorithm, documents need to be preprocessed. Preprocessing is done in 5 steps:

(1) Stop words are removed from the document.

(2) Non-ASCII characters, hashtag symbol and one character words are removed.

(3) Punctuation, except English contractions, is removed.

(4) Tweets are then tokenized using Twokenizer [73].

(5) Tweets are automatically classified using the previous taxonomy: each tweet is mapped with the category of the keyword used to retrieve it. For example, the tweet “Backdoor in x software” is retrieved using the unigram “backdoor”, hence the tweet will be classified as “Cyber Security/Vulnerability/Backdoor”.

This preprocessed documents are then the inputs for the *event detection* algorithm.

3.5 Event Detection

Event detection is a clustering task: a system receive a stream of documents and must organize it in the most appropriate event-based cluster. In our case, we chose the approach used by Petrović et al. [77] which is considered as one of the state-of-the-art technique and used in multiple publications [64, 70].

3.5.1 Problem Statement

Traditionally, documents are represented as vectors where each coordinate is the term frequency (possibly term frequency-inverse document frequency weighted [82]) of a particular term. Similarity between two documents can be computed using various metrics. However, according to the work of Allan et al. in the UMass [11], the cosine similarity outperforms the weighted sum and KL divergence when working on first story detection. The cosine similarity is a well known metric in Information Retrieval and is the inner product between of two normalized vectors. Considering document A and B, the formula is the following:

$$similarity = \cos\theta = \frac{\mathbf{A} \cdot \mathbf{B}}{||\mathbf{A}||\,||\mathbf{B}||} = \frac{\sum_{i=1}^{n} A_i B_i}{\sqrt{\sum_{i=1}^{n} A_i^2}\sqrt{\sum_{i=1}^{n} B_i^2}} \tag{1}$$

We wish to build event related clusters using a constant stream of documents. The classic approach is to compare each new document with the previous one and if the similarity is beyond a chosen threshold, the new document is considered as a first story. Algorithm 1 describes the exact pseudo-code of the UMass system which is the nearest neighbor search.

However, using this approach, the algorithm has a running time of $\mathcal{O}(dN^2)$ where N is the number of document in the corpus and d is the dimensionality of the document. Hence, in the case of large corpus with high dimensionality, finding the first story (in our case the first tweet) in real-time is not feasible.

```
foreach document d in corpus do
    foreach term t in d do
        foreach document d' that contains t do
            update distance(d, d');
        end
    end
    dis_min(d) = min_d'{distance(d, d')};
end
```

Algorithm 1: Classic First Story Detection Approach

3.5.2 Locality-Sensitive Hashing

The problem of the complexity of the nearest neighbor search have been widely studied and Locality Sensitive Hashing (LSH) is a method to reduce it. Petrović et al. used the hashing scheme proposed by Charikar [28]. Using this hashing scheme, the probability of two points p and q colliding considering a random hyperplane vector u is:

$$P_{coll} = P_u[h_u(p) = h_u(q)] = 1 - \frac{\theta(p,q)}{\pi} \tag{2}$$

where $h_u(p) = sign(u \cdot p)$.

With a high number of hyperplanes we can increase the probability of collision with a non-similar point. The size of the candidate set that contains the nearest neighbor can also be decreased by using multiple hash tables. With LSH, clustering can be performed in $\mathcal{O}(1)$ time.

3.5.3 First Story Detection

Algorithm 2 presents the pseudo-code of Petrović et al. First Story Detection algorithm using LSH.

We choose to run the algorithm with 13 hyperplanes and 70 hash tables with a maximum distance of 0.80, which are parameters close to the ones used by Petrović et al.. But instead of running the algorithm every 100,000 tweets, we run it every hour. In order to only keep the cluster that

are significant, SONAR keeps the clusters that satisfy the three following conditions. First, the Shannon entropy (used to measure the amount of information) in the cluster must be more than a fixed value (e.g., 2.7). This will discard clusters that are composed of the same repeated document. Second, clusters must contain more than 10 documents, otherwise we would keep clusters of only one document. Finally, clusters contain document written by different authors. This is to ensure that a small group of person cannot create a cluster considered as an event. However, this approach is not resilient towards Sybil attacks. With all this conditions, we try to dodge spam and keep only the significant events.

---

input: threshold t
foreach *document d in corpus* **do**
    add d to LSH;
    S $\leftarrow$ set of points that collide with d in LSH;
    $dis_{min}(d) \leftarrow 1$;
    foreach *document d' in S* **do**
        $c = distance(d, d')$;
        if $c < dis_{min}(d)$ **then**
            $dis_{min}(d) \leftarrow$ c;
        end
    end
    if $dis_{min}(d) \geq t$ **then**
        compare d to a fixed number of most recent documents as in algorithm 1 and update dis_{min} if necessary;
    end
    assign score dis_{min}(d) to d;
    add d to inverted index;
end

Algorithm 2: Petrović and al. LSH-based Approach

Moreover, SONAR implements an algorithm that will merge similar events happening in more than an hour interval. This algorithm will retrieve the first story (the first tweet of a cluster) of each events in the last hours and compare with the latest events found. This allows us to have events distributed on several days instead of multiple events (for example is the event is not discussed during 1 hour and reappears in the stream after).

3.6 Keywords Finder

Earlier in this thesis, we discussed the advantages of using a list of keywords to query Twitter instead of directly listening to the raw stream. However, this approach presents some limitations. First of all, there is no way for an analyst to be sure that he did not forget a keyword at the initialization and hence missing all the events containing the specific word. Next, an analyst will insert in *SONAR* the seed keywords based on his previously acquired knowledge, which might eventually falls short within a few months time. For example, a ransomware is a new type of malware and let's imagine that the analyst is not aware of it and consequently do not insert in the list of seed keywords. SONAR is capable of retrieving tweets containing the keyword "malware" but not the ones containing "ransomware". However, in multiple tweets the word "ransomware" appears with the keyword "malware": our *keyword finder* algorithm will discover this relation and suggest the keyword "ransomware" to the analyst. In this subsection, we describe the algorithm that *SONAR* uses to discover automatically new keywords.

3.6.1 Word Embeddings

The term "word embeddings" was coined in 2003 by Bengio et al [18], but was eventually popularized by Mikolov et al. with *word2vec* [67], a toolkit that "provides an efficient implementation of the continuous bag-of-words and skip-gram architectures for computing vector representations of words". Word embeddings refer to dense representations of words in a low-dimensional vector space. Unsupervised learned word embedding models have shown to be highly effective in natural language processing (e.g., discover semantic relation between words embeddings).

In our case, we would like to discover new keywords, which are semantically close to the *seed keywords*. For example, we are interested by the category "Ransomware" and we would like to get automatically new names of ransomware such as TeslaCrypt or CryptoLocker using the initial tweets.

While *word2vec* is historically the first widely used word embeddings model, we choose to use GloVe [76] by Pennington et al. (2014) as it was reported to outperform the previous in terms of

accuracy and running time [83, 92].

GloVe

Let us consider that X is the matrix of word-word co-occurrence counts. We denote by X_{ij} the number of times i occurs with the context word j and $P_{ij} = P(j|i) = X_{ij}/X_j$ the probability that the word j appears in the context word i. Pennington et al. started with the following idea, the ratios of co-occurrence probabilities (i.e. P_{ik}/P_{jk}) is more accurate to spot relevant word from irrelevant word. Hence the following starting point:

$$F(w_i, w_j, \tilde{w_k}) = \frac{P_{ik}}{P_{jk}} \tag{3}$$

where $w \in \mathbb{R}^d$ are word vectors and $\tilde{w} \in \mathbb{R}^d$ is a context word vectors of dimension d.

Finally, the model is defined by the following function:

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w_j} + b_i + \tilde{b_j} - \log X_{ij})^2 \tag{4}$$

where b is the bias and f is a weighting function which has the following properties:

(1) $f(0) = 0$

(2) $\forall (x, y) \in \mathbb{R}_+^{*\,2}, x < y, f(x) \leq f(y)$

(3) f(x) must be "relatively small for large values of x".

In that case, Pennington et al. found that the following function suits well:

$$f(x) = \begin{cases} (x/x_{max})^\alpha \text{ if } x < x_{max} \\ 1 \text{ otherwise} \end{cases} \tag{5}$$

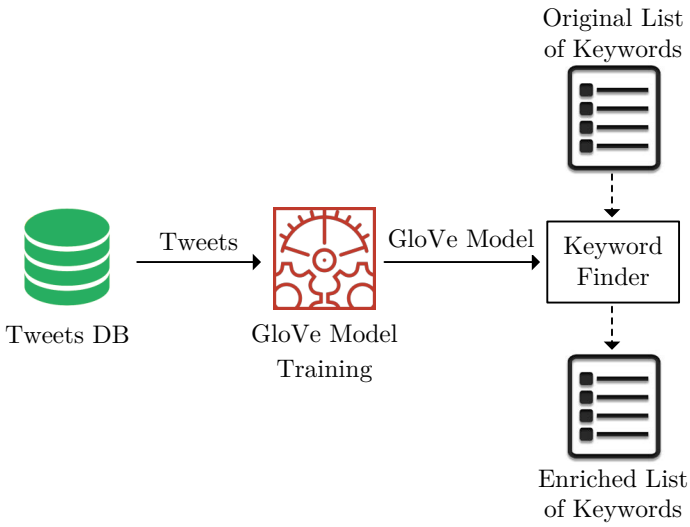

Figure 3.2: Architecture of the Keyword Finder Component

Algorithm

Ultimately, our goal is to find keywords that have a close semantic relationship with the *seed keyword* of a given category and at the same time represent solely this category. For example, the word "Linux" may have a close semantic relationship with the "Malware" category but does not represent it well. To achieve this, we designed an algorithm that used language processing techniques and the previous word embedding model.

In Figure 3.2, we present the high-level architecture of our keyword finder component. We train a GloVe model on a large corpus of preprocessed tweets. This gives us a trained model that will be one of the input of our *keyword finder* algorithm. Let $C = \{C_1, C_2, \ldots, C_n\}$ be a set of top level categories (e.g., "Denial of Service", "Malware", etc.) from the taxonomy, where n is the total number of categories. Let $K_n = \{K_{n,1}, \ldots, K_{n,i}\}$ be the set of i unigram seed keywords that belongs to category n and its respective subcategories (for example the category "Denial of Service" and the subcategories "DDoS", "APDoS", etc.). In this case, $K_{1,1}$ could be the keyword "DDoS"'. Consequently, each top level category is represented by a set of seed keywords K_n. Using the previously trained GloVe model, we look for the nearest neighbors of each keyword of

each category. The nearest neighbors are found using the model and the cosine similarity between the candidate keyword vector and the *seed keyword* vector. To restrict the number of candidates, we defined a threshold α for the cosine similarity (i.e. only keywords that are significantly close are selected). Each discovered nearest neighbor is assigned to the category of the seed keyword used to retrieved it. Hence we form, $M_n = M_{n,1}, \ldots, M_{n,j}$, a set of j candidate discovered unigram for category n. Each M_n is then considered as a document. Finally, to compute the list of each keyword to add for each category we leverage the so-called *Term Frequency-Inverse Document Frequency*.

Term frequency-inverse document frequency (*tf-idf*) is a famous technique adopted in the field of *natural language processing*. The latter computes vectors of inputted text documents by considering both the frequency in the individual documents and in the whole set. Let $D = \{d_1, d_2, \ldots, d_n\}$ be a set of text documents, where n is the number of documents, and let $d = \{w_1, w_2, \ldots, w_m\}$ be a document, where m is the number of words in d. The *tf-idf* of a word w and document d is the product of *term frequency* of w in d and the *inverse document frequency* of w, as shown in Formula 6. The *term frequency* (Formula 7) is the occurrence number of w in d. Finally, the *inverse document frequency* of w (Formula 8) represents the number of documents n divided by the number of documents that contain w in the logarithmic form. The computation of *tf-idf* is very scalable, which suites our needs (Section 4.3).

$$\textit{tf-idf}(w, d) = \textit{tf}(w, d) \times \textit{idf}(w) \tag{6}$$

$$\textit{tf}(w, d) = |w \in d,\ d = \{w_1, w_2, \ldots, w_n\} : w = w_i| \tag{7}$$

$$\textit{idf}(w) = log \frac{|D|}{1 + |d : w \in d|} \tag{8}$$

Using *tf-idf* and our documents M_n (which are sets of candidate discovered keywords for category n), we keep only the keywords that have a high term frequency in M_n and a low document frequency across all the M_n. In other words, we keep only candidate keyword that are category specific. We also define a threshold β in order to keep only highly relevant keywords. Algorithm

3 describes the inner working of the KeywordFinder component. Results of this algorithm using a model trained with a month of tweets are available in Section 4.3.

---

input: threshold α, threshold β, trained GloVe model, list of seed keywords per category
output: a set of keywords per category L

---

foreach *category i in set of category C* **do**
  foreach *seed keyword s in category i* **do**
    $T_i \leftarrow$ nearest neighbors of s with threshold $> \alpha$
  end
end
vectors $\leftarrow$ TF-IDF(T)
foreach *category i in C* **do**
  foreach *keyword k in* T_i **do**
    $L_i \leftarrow$ k, if vector(k, T_i) $> \beta$
  end
end

Algorithm 3: KeywordFinder Algorithm

Chapter 4

Implementation and Evaluation

In this chapter, we first describe the implementation of *SONAR* designed to detect cyber-security related events. Then, we evaluate the algorithms and the events detected. Finally, we describe what we call the *SONAR Reports* and two custom-made implementations.

4.1 Implementation

This section presents the implementation of *SONAR* in details. Figure 4.1 displays a technical and simplified overview. Scalability and performance are two important objectives in the design of *SONAR* as the framework. It must be well equipped to handle any type of event from the least popular (few documents per seconds) to the most popular (hundred of documents per second). *SONAR* is built upon open source frameworks:

- Apache Zookeeper [4] for the coordination between the components.
- Apache Kafka [2] for the streaming platform.
- Apache Spark [3] for the computation.
- Cassandra [1] for the storage and query of data.
- ELK Stack [5] (Elasticsearch, Logstash and Kibana) for the storage and consultation of the data.

Each choice of framework is the result of a thoughtful analysis:

- Apache Kafka is highly scalable distributed streaming platform capable of handling millions of messages per second. We designed SONAR with the idea in mind that we might have access to more than 1% of the Twitter feed hence handling millions of tweets per day.
- Apache Spark is a fast engine for big data processing, highly scalable and efficient engine with several machine learning libraries integrated.
- Apache Cassandra is one the fastest database. As *SONAR* works in real-time, we needs a robust solution that can handle a high number of queries per second.

Data Collection. Tweets are collected using Java Twitter4J[1] library directly linked with Kafka. The Spark cluster receives the stream of tweets and with Spark Streaming, tweets are categorized and stored them into an Elasticsearch index and simultaneously in Cassandra database.

Event Detection or first story detection (FDS) is done with a Java implementation of the algorithm of Petrović et al. on Spark. Events are categorized according to the taxonomy and geomapped using Google Map Geocoding API. Each tweet is geolocated using the GPS coordinates when available or the location of the user. This allows us to geographically locate where an event is most discussed.

Keyword Refinement is done using the GloVe implementation available on GitHub[2]. The model is trained using the previous algorithm and a python/R script regularly looks on a regular basis for new relevant keywords.

SONAR is designed to be distributed over multiple servers for maximum efficiency. When a particular component is overloaded, the maintainer do not have to change the entire system in order to speed up the framework, he just needs to add another node to the slow component.

[1] `http://twitter4j.org`
[2] `https://github.com/stanfordnlp/GloVe`

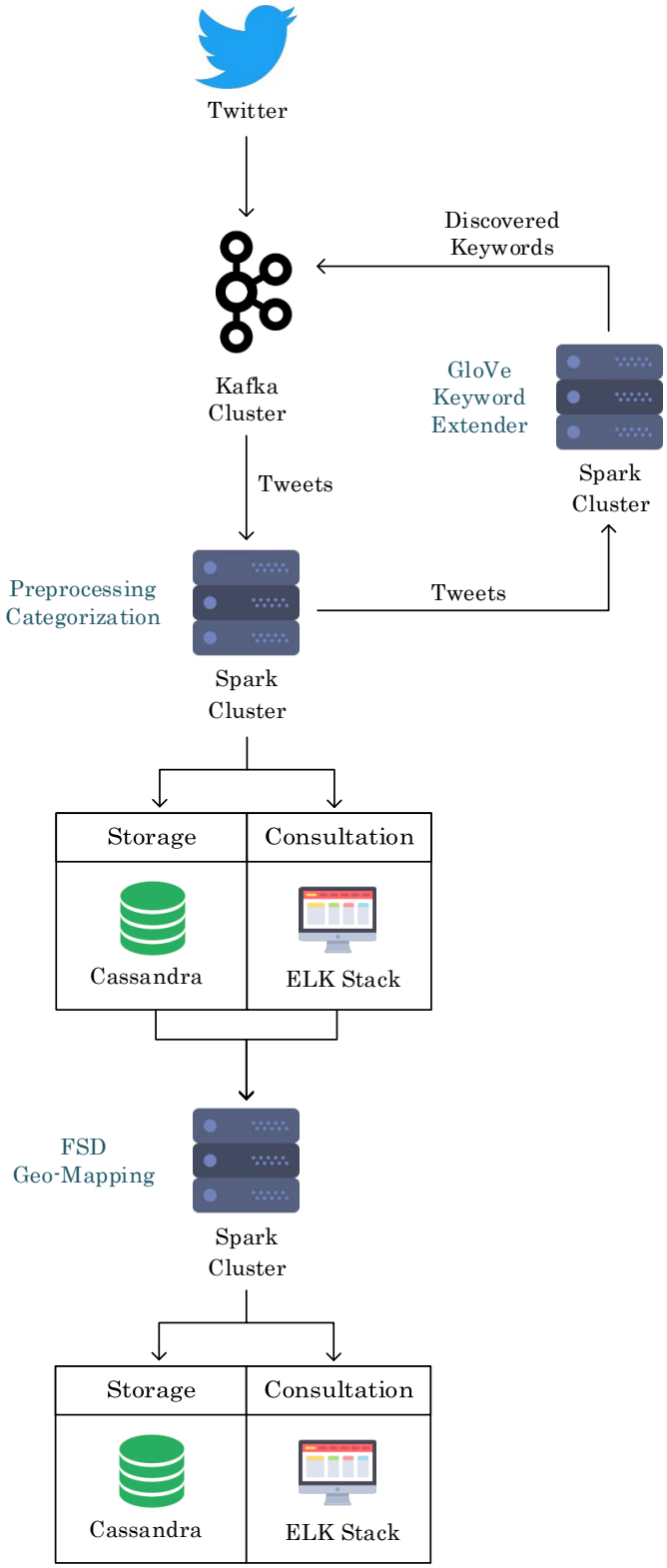

Figure 4.1: Technical Overview of SONAR

4.2 Algorithms Evaluation

This section evaluate three different components of SONAR. First, we analyze the speed of our implementation of Petrović et al. First Story Detection Algorithm, then we demonstrate the result of the keyword refinement component. Finally, we present a few automatically discovered events by SONAR and show how the tool knowledgeably summarize the information.

4.2.1 First Story Detection

We ran the test on a Linux virtual machine with 8GB of RAM and 4 core of an Intel i7-4790. Parameters are the followings: 13 hyperplanes and 70 hash tables with a maximum distance of 0.80. In that evalutation, we are not directly interested in performances but in the fact that, according to Petrović et al., the algorithm must run in constant time. Figure 4.2 shows that our Spark implementation do run in constant time with approximately 60 milliseconds for 100 documents (we can actually expect faster results on a real Spark cluster).

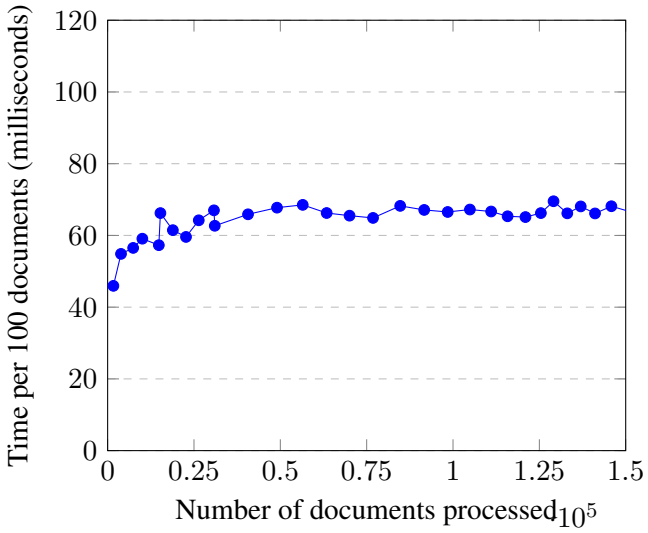

Figure 4.2: Processing Time per 100 Documents

Consequently, we are able to detect in near real-time 759 events per month on average. This number can go up to 1328 in December 2016 (see Figure 4.3).

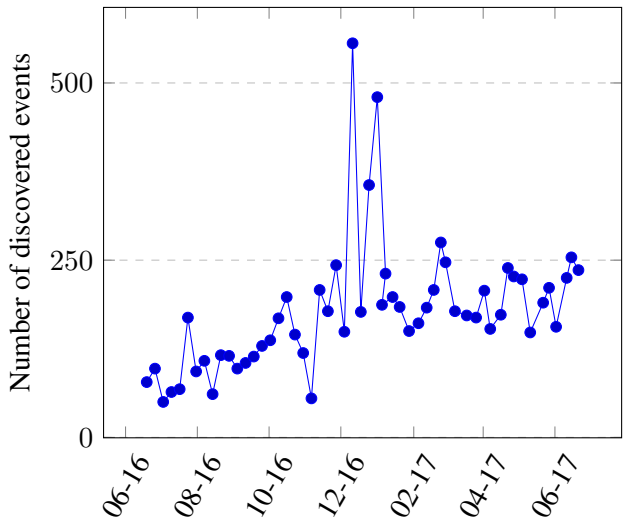

Figure 4.3: Number of Events Discovered Each Month from June 20 2016 to July 1 2017

4.2.2 Keywords Finder

In this section, we present the results of the *Keyword Refinement* component trained on different periods. As it is the most time consuming operation of SONAR, we run it on a bi-weekly basis (representing approximately 3 million tweets) but it can be run every week. Results for the month of October and December 2016 are available in table 2, result of December 2016 are written in italicized text. We choose high values of α (semantically close nearest neighbors) and β (discovered keywords which are unique to each category) in order to avoid false positives. Before integrating this new keywords as seed keywords, the analyst must verify them manually. Keywords discovered in October 2016 such as “locky” for “Malware” category, “mirai” for “Botnet” category and “ddosattack” for “Denial of Service” category can be safely incorporated into the seeds keywords. However, keywords such as “dyn”, “scam” and “iot” should be dismissed as they can possibly retrieve extraneous tweets in the future. “Scripting” and “spyware” keywords were discovered in December 2016 and are highly relevant to their category.

It’s worth noting that at some point the system might receive more tweets that it can proceed because of a too large number of seed keywords. In that case, maintainers should upgrade their system by adding a new server in the Spark cluster.

Category	Keywords
Social Engineering	scam
Malware	cybercrime, locky, *spyware*
Botnet	mirai, iot, *iotsecurity*
Denial of Service	ddosattack, dyn, *cyberattack*
Vulnerability	*scripting*

Table 4.1: Sample of Discovered Keywords for October 2016 and *December 2016*

4.3 Events Evaluation

In this section, we evaluate the relevance of the discovered events then we analyze four significant events that happened between September 2016 and July 2017.

4.3.1 Relevance of the Discovered Events

Not all the events detected by *SONAR* might be relevant for a security practitioner. Numerous events are about hacked accounts of public figures. Also an important part of the discovered are parts of misinformation campaign (especially during the US presidential election). One of the most type of detected events are hacking of the Twitter account of public figures, this happens regularly and it is often a widely discuss topic on the social network., however this kind of events are most of the time not relevant for us.

Table 4.2 presents the notable events from January 25 to February 1. During this week, 100 events were automatically discovered, 25 were considered as highly relevant. In general, 1 of 4 detected event is a relevant cyber security events. We also noticed that some events, despite being already detected previously appeared again in *SONAR* (mainly about the "russian hacking in the US election").

While can we check with other sources that a detected events is really happening, it's really hard to be sure that we did miss any events. We found that we can miss small events such as vulnerabilities that are not critical but we did not miss any big events (major attacks) from July 2016 to July 2017.

General Events
Data Breach Database Site “LeakedSource” Goes Offline After Alleged Police Raid
WWE Social Media Accounts Hacked By OurMine
Cardinals fined $2 million, must send two draft picks to Astros as hacking penalty
Spanish Police Claim to Have Arrested Phineas Fisher - Hacking Team Hacker
Uber Pays Hacker US $9,000 for Partner Firm’s Bug
Denial of Service Events
Sonic Customers Offline As Local Internet Provider Hit With DDoS Attack
Malware Events
Disk-nuking malware takes out Saudi Arabiangear.
Data-Stealing Ransomware App Found in Official Google Play Store
Rootnik Android malware variant designed to frustrate researchers
The Nuke HTTP bot Malware offered for sale on a Dark Web forum
38% of Android VPN Apps on GooglePlay Store Plagued with Malware .
Ransomware Hacks Lock Hotel Guests in Their Rooms
Ransomware Killed 70% of Washington DC CCTV Ahead of Inauguration
Police Department Loses Years Worth of Evidence in Ransomware Incident
Newly Discovered Banking Malware Creates Fresh Threat to Users
Malicious Office files using fileless UAC bypass to drop KEYBASE malware
Vulnerability Events
CVE-2017-3797 Vulnerability in Cisco WebEx Meetings Server
WordPress4.7.2UpdateFixesXSS,SQLInjectionBugs
CVE-2017-3422 Vulnerability in Oracle One-to-One Oracle E-Business Suite
Netgear Exploit Found in 31 Models Lets Hackers Turn Your Router Into a Botnet
Social Engineering Events
Phishers new social engineering trick: PDF attachments with malicious links
Data Breach Events
180,000 Members of an Underground Adult Website Have Been Leaked Online

Table 4.2: Notable Events Discovered from January 25 to February 1

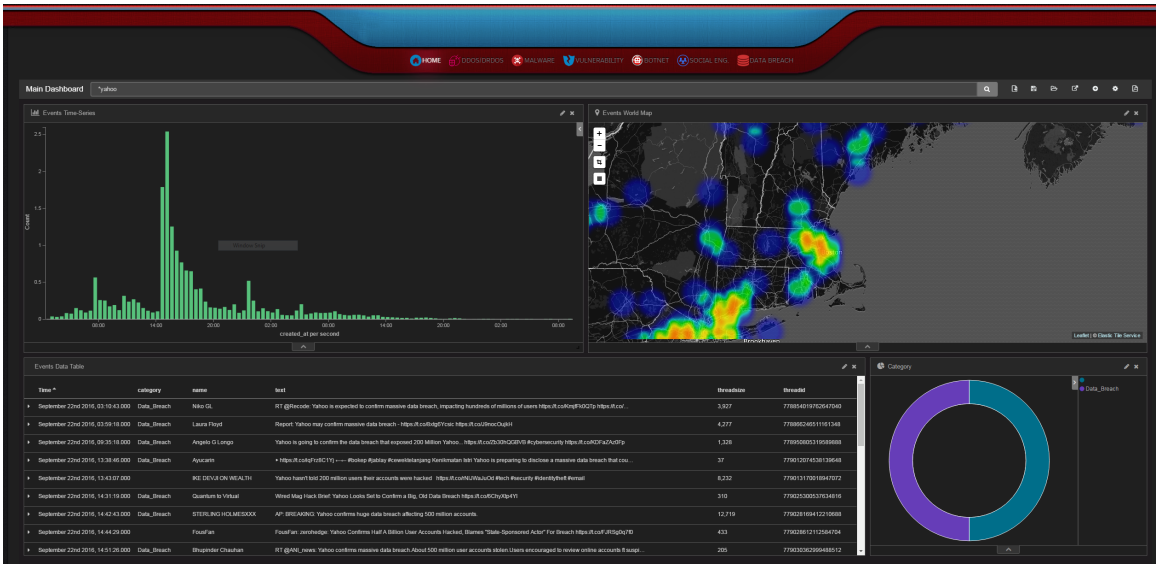

Figure 4.4: Sonar User Interface Displaying All the Events Linked to the Yahoo Data Breach

4.3.2 The Yahoo Data Breach

The "Yahoo data breach"[3] reported in September 2016 is considered as one of the largest data breach in the history of internet. 500 millions of Yahoo accounts were compromised and data such as names, email addresses, telephone numbers, encrypted or unencrypted security questions and answers, dates of birth, and encrypted passwords were leaked. On September 22 2016, SONAR captured this event, all its different steps and its propagation on the network.

Figure 4.4 shows what the user interface display when looking at the Yahoo Data Breach event. We were able to get all the different milestones of this event on September 22 at time displayed:

(1) Around 1 EST, people started talking about the tweet from RECODE "Yahoo is expected to confirm massive data breach, impacting hundreds of millions of users...".

(2) Few hours later, at 4 EST, one another step is the event is reached "Yahoo may confirm massive data breach".

(3) At 2:42, a tweet from Associated Press is highly retweeted " BREAKING: Yahoo confirms huge data breach affecting 500 million accounts."

[3]See `https://en.wikipedia.org/wiki/Yahoo!_data_breaches`

(4) We learn after that Yahoo blames a “State-Sponsored Actor” and two days later, we learn that the company is being sued.

SONAR successfully captures the different states of the event. Because of the number of people talking about the event at the same time, an analyst could be aware of the situation as soon at the news is propagated on Twitter.

4.3.3 Dyn Denial of Service Attack

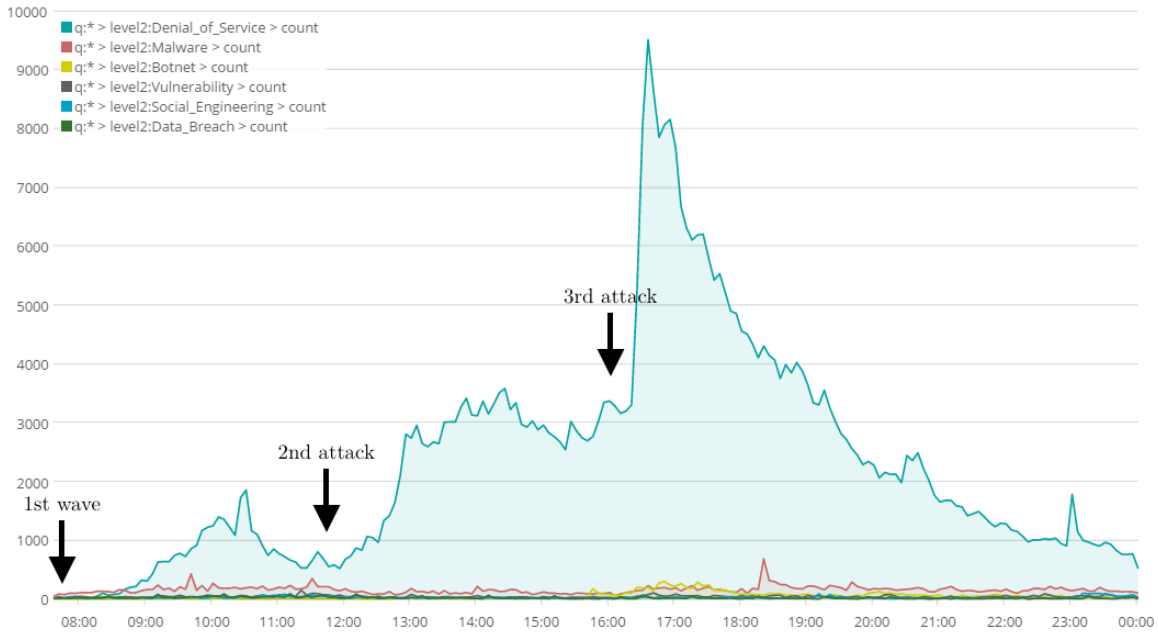

Figure 4.5: Screenshot of SONAR UI on October 21, 2016

The 2016 Dyn Denial of Service Attack[4] is a "cyberattack that took place on October 21, 2016, and involved multiple distributed denial-of-service attacks (DDoS attacks) targeting systems operated by Domain Name System (DNS) provider Dyn, which caused major Internet platforms and services to be unavailable to large swathes of users in Europe and North America." This attack can be broken down in three attacks. First attack was from 7 am to 9:20 am EST. A second attack started at 11:52 am and a third attack was reported at 4pm.

At 8 am EST time, we started receiving an abnormal number of tweets labeled as "Denial of Service" category (see Figure 4.5. 30 minutes later, we start receiving the first event. People reports having trouble accessing services such as Github and Spotify. Using SONAR, we can see directly the three attacks against DYN on that day using the total number of tweets.

[4]See `https://en.wikipedia.org/wiki/2016_Dyn_cyberattack`

4.3.4 Google Doc Phishing Attack

The Google Doc phishing attack was one of the most sophisticated phishing attack in 2017. On May 3 2017, people started receiving "an emailed invitation from someone they may know, takes them to a real Google sign-in screen, then asks them to continue to Google Docs. But this grants permissions to a (malicious) third-party web app that's simply been named Google Docs, which gives phishers access to your email and address book." [104]

The propagation of this event is very interesting because victims will report it first on Twitter and before social media. In our case, we detected the social engineering attacks with a tweet from user "andy birkholz" (see Figure 4.6) at 14:31:48.

Figure 4.6: Screenshot of the First Tweet About May 3 Google Doc Phishing Attack

At this time, the extend of the attack was not clearly identified. However, after few hours, we found out that with more than 20,000 thousands tweets on May 3, with a peak of 1.6 tweets per second on the subject from users all over the world, it was a relatively important phishing campaign (See Figure 4.7 for a sample of the biggest tweet threads.). With *SONAR*, the analyst can be aware of this phishing attack before any news media.

Time	name	text	threadsize
May 3rd 2017, 15:04:26.000	William Griffith	RT @theGunrun: ALERT: Huge Google Docs Phishing Scam is live - https://t.co/JES6bNv8FU DO NOT OPEN GOOGLE DOCS EMAILS	8,617
May 3rd 2017, 15:01:25.000	MaryAnn Johanson	Google Docs phishing emails apparently exploding all over right now. They look like this. Do not click on the link. https://t.co/6Jwqc8RXSa	7,121
May 3rd 2017, 15:47:46.000	#NoBanNoWall#NoTrump	RT @RoyalHoeliness: More info on the Google docs phishing scam >>	2,328

Figure 4.7: Screenshot of the Three Biggest Tweet Threads on the Google Doc Phishing Attack

4.3.5 WannaCry Ransomware

The WannaCry ransomware attack[5] was a malware attack in May 12 2017. The ransomware used an exploit in WIndows' Server Block Message (SMB) called EternalBlue. The malware encrypt the filesystems and ask for a ransom between $300 and $600. Within a day more than 230,000 computers in over 150 countries were infected. Very similarly to the previous events, infection was first reported by simple user and then by news media. Figure 4.8 shows the number of tweets per second about the ransomware on May 12. As we can see, Twitter users started to talk about it around 8AM EST before the information going into the mainstream media.

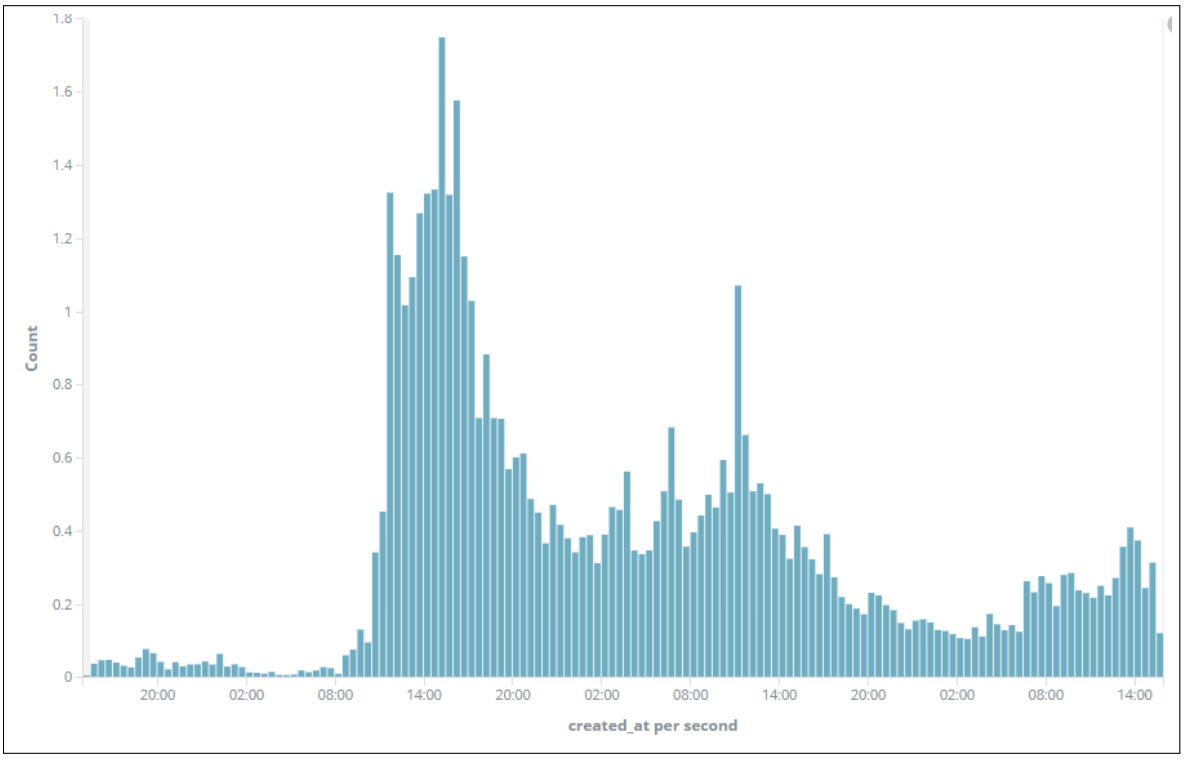

Figure 4.8: Timeseries of the Tweet About the WannaCry Ransomware

4.4 Custom-Made Implementation

In this section, we present two custom-made implementation of *SONAR* that we built this year.

[5]See `https://en.wikipedia.org/wiki/WannaCry_ransomware_attack/`.

4.4.1 *SONAR* for Deceptive Marketing Practices

Beginning December 2016, we deployed a version of *SONAR* designed to detect deceptive marketing on Twitter. We created a list of categories and keywords (see Table 4.3) for the complete list of keyword) to monitor such events.

Health	⋆ Skin Care
⋆ Weight Loss Category	▷ Anti-Aging
▷ Risk Free Trial	▷ Wrinkle removal
▷ Free Sample	▷ Retinolla
▷ Green Coffee	▷ Stemologica
▷ Garcinia Cambogia	▷ LumaEssence
▷ Forskolin	▷ Nugenix
▷ Slim Phen	▷ Hydroluxe
⋆ Muscle strength	▷ Revoluxe
▷ Testo Fuel	▷Lorelia
▷ Nitro Tech	▷ Abella Mayfair
▷ Hydro Muscle Max	▷ Nuvella
▷ Xtreme Exo	▷ Chantel
▷ TestX Core	**Financial**
▷ HL12	▷ Work at home
⋆ Teeth Whitening	▷ Cash Loophole
▷ WhiteLight	▷ Casino Secret
▷ Blizzard White	▷ Risk Free Money
▷ Action Pro White	▷ Get Rich Fast

Table 4.3: Deceptive Marketing Taxonomy

With this taxonomy and the keywords associated we deployed *SONAR*. We were able to detect approximately 340 events per month (except on February 2017, were the tool was mostly under maintenance). Figure 4.9 shows how many events were detected per month.

In this case, the term event refers to marketing campaign being run on Twitter, for example we were able to detect hundred of fake accounts (most probably bot) tweeting about “green coffee” for a few hours. This bot tries to sell their products by replying to other people tweets and most of the time the Twitter platform ban them. Finally, we designed an interface based on Kibana for this instance of *SONAR*.

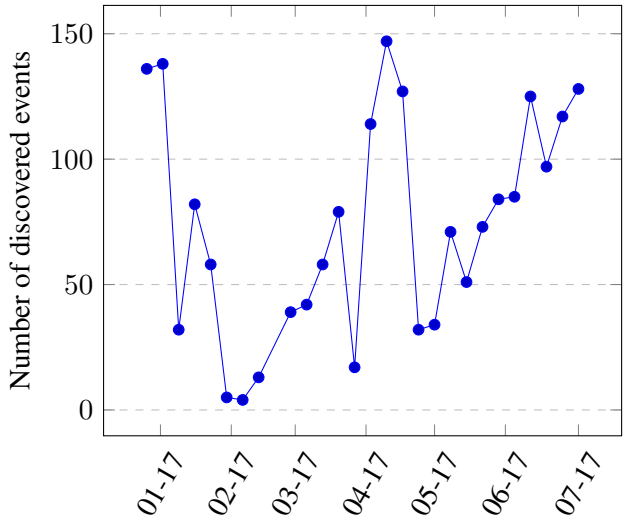

Figure 4.9: Number of Events Found from January 2017 to July 2017

4.4.2 *SONAR* for Cryptocurrencies Events

This implementation of *SONAR* was deployed from September to December 2016 and was mostly designed to detect events regarding cryptocurrencies in particular: Bitcoin, Ethereum, Monero, Litecoin and Zcash.

We used this database of tweets and automatically discovered events in our course project INSE 6180 (Security and Privacy Implications of Data Mining): "Predicting Bitcoin Price With Twitter". The principle was that Bitcoin users sometimes use Twitter to express their opinion. More importantly, core developers of Bitcoin tweet regularly and can also influence other Bitcoin users. We wanted to see if we could predict the Bitcoin price fluctuation using Twitter. First, we tried to use a linear model, a Bivariate Granger Causality Analysis [42], with a disappointing performance. We then tried to use a non linear model: a neural network. We choose to use a Long Short-Term Memory (LSTM) network with a Nonlinear Autoregressive Exogenous Model (NARX) [20]. We were able to predict the upward and downward trend of the Bitcoin price with an accuracy up to 83% over a two months period. This work was heavily inspired by the work of Bollen et al. in 2010 [22].

Chapter 5

Conclusion and Future Work

Event detection in social networks is still a promising research field. Being able to detect topics and events directly from the affected persons is the fastest way to obtain hindsights. With the progress in topic modeling and memory bound techniques such as sequential nearest neigbour search, event detection techniques are progressing every year.

In this research initiative, we expose *SONAR* an automatic, self-learned framework that can detect, geolocate, categorize cyber security events in near real-time over the Twitter Stream. We presented the state of the art on event detection in Twitter. We analyzed the existing publications and we found that despite an important number of research projects there is still room for an improved real-time event detection system. We proposed the architecture of a dynamic and heavily scalable real-time event detection system. This system leverages existing event detection algorithms with our custom-made cyber security taxonomy. This system was implemented using the state-of-the-art open source softwares currently available. Consequently, we proposed an automated and self learned keyword detection algorithm which works using word embeddings techniques (Global Vectors) and widely used text mining methods such as term frequency-inverse document frequency (tf-idf). Finally, we evaluated the performance of our algorithms and the accuracy of the events detected in the past 12 months. During this time period, major events such as the Yahoo data breach, Dyn denial of service attack, the WannaCry Ransomware, the NotPetya Ransomware and the U.S. election happened. This makes our dataset a great resource for future work directly related to *SONAR* or even other cyber security research.

Subsequently, we found that most of the times social networks and traditional news media are on par in terms of reporting speed. However, for big events such as denial of service attack or ransomware attack, we are able to detect and locate the events before it appears in the traditional news. In term of scalability, we discovered that maintaining a running instance of *SONAR* can consume a lot of disk and memory (as the algorithms use it) and can probably be enhanced in that way (e.g., by removing old data or improving the algorithms).

SONAR can be greatly enhanced in its current form. First, filtering non relevant events can be done using a classifier and a list of labeled cyber security events detected the past year. This would allows us to filter automatically spammy or "non-interesting" events. Moreover, we are currently working on a event grading system to more efficiently display the important events that need the analyst attention. This last technique could technically preserve the generic nature of *SONAR* by leveraging the *reputation* of each user. For future work, we would like to add a drill down capability to the framework sourced from others news sources (e.g., malware feed). We would like to identify the target and the source of an event. Finally, this database of events could help us build a model to predict future events.

Bibliography

[1] Apache cassandra: https://cassandra.apache.org. Last visited August 16, 2017.

[2] Apache kafka: https://kafka.apache.org. Last visited August 16, 2017.

[3] Apache spark: https://spark.apache.org. Last visited August 16, 2017.

[4] Apache zookeeper: https://zookeeper.apache.org. Last visited August 16, 2017.

[5] Elk stack: http://www.elastic.co. Last visited August 16, 2017.

[6] New tweets per second record, and how! volume https://blog.twitter.com/2013/new-tweets-per-second-record-and-how, August 2013.

[7] Hamed Abdelhaq, Christian Sengstock, and Michael Gertz. Eventweet: Online localized event detection from twitter. *PVLDB*, 6(12):1326–1329, 2013.

[8] Mariam Adedoyin-Olowe, Mohamed Medhat Gaber, Carlos J. Martín-Dancausa, Frederic T. Stahl, and João Bártolo Gomes. A rule dynamics approach to event detection in twitter with its application to sports and politics. *Expert Syst. Appl.*, 55:351–360, 2016.

[9] Charu C. Aggarwal and Karthik Subbian. Event detection in social streams. In *Proceedings of the Twelfth SIAM International Conference on Data Mining, Anaheim, California, USA, April 26-28, 2012.*, pages 624–635, 2012.

[10] Luca Maria Aiello, Georgios Petkos, Carlos J. Martín, David Corney, Symeon Papadopoulos, Ryan Skraba, Ayse Göker, Ioannis Kompatsiaris, and Alejandro Jaimes. Sensing trending topics in twitter. *IEEE Trans. Multimedia*, 15(6):1268–1282, 2013.

[11] James Allan, Victor Lavrenko, Daniella Malin, and Russell Swan. Detections, bounds, and timelines: Umass and tdt-3. In *Proceedings of Topic Detection and Tracking Workshop*, pages 164–174, 2000.

[12] Foteini Alvanaki, Sebastian Michel, Krithi Ramamritham, and Gerhard Weikum. Enblogue: emergent topic detection in web 2.0 streams. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2011, Athens, Greece, June 12-16, 2011*, pages 1271–1274, 2011.

[13] S. D. Applegate and A. Stavrou. Towards a cyber conflict taxonomy. In *2013 5th International Conference on Cyber Conflict (CYCON 2013)*, pages 1–18, June 2013.

[14] Sebastien Ardon, Amitabha Bagchi, Anirban Mahanti, Amit Ruhela, Aaditeshwar Seth, Rudra Mohan Tripathy, and Sipat Triukose. Spatio-temporal and events based analysis of topic popularity in twitter. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 219–228, 2013.

[15] L. Douglas Baker and Andrew Kachites McCallum. Distributional clustering of words for text classification. In *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '98, pages 96–103, New York, NY, USA, 1998. ACM.

[16] Amparo Elizabeth Cano Basave, Yulan He, Kang Liu, and Jun Zhao. A weakly supervised bayesian model for violence detection in social media. In *Sixth International Joint Conference on Natural Language Processing, IJCNLP 2013, Nagoya, Japan, October 14-18, 2013*, pages 109–117, 2013.

[17] Hila Becker, Mor Naaman, and Luis Gravano. Beyond trending topics: Real-world event identification on twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*, 2011.

[18] Yoshua Bengio, Réjean Ducharme, Pascal Vincent, and Christian Jauvin. A neural probabilistic language model. In *Advances in Neural Information Processing Systems 13 (NIPS'00)*, pages 933–938, 2001.

[19] James Benhardus and Jugal Kalita. Streaming trend detection in twitter. *IJWBC*, 9(1):122–139, 2013.

[20] Stephen A Billings. *Nonlinear system identification: NARMAX methods in the time, frequency, and spatio-temporal domains*. John Wiley & Sons, 2013.

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

[22] Johan Bollen, Huina Mao, and Xiao-Jun Zeng. Twitter mood predicts the stock market. *CoRR*, abs/1010.3003, 2010.

[23] Cedric De Boom, Steven Van Canneyt, and Bart Dhoedt. Semantics-driven event clustering in twitter feeds. In *Proceedings of the the 5th Workshop on Making Sense of Microposts co-located with the 24th International World Wide Web Conference (WWW 2015), Florence, Italy, May 18th, 2015.*, pages 2–9, 2015.

[24] Ceren Budak, Theodore Georgiou, Divyakant Agrawal, and Amr El Abbadi. Geoscope: Online detection of geo-correlated information trends in social networks. *PVLDB*, 7(4):229–240, 2013.

[25] HongYun Cai, Yang Yang, Xuefei Li, and Zi Huang. What are popular: Exploring twitter features for event detection, tracking and visualization. In *Proceedings of the 23rd Annual ACM Conference on Multimedia Conference, MM '15, Brisbane, Australia, October 26 - 30, 2015*, pages 89–98, 2015.

[26] Mario Cataldi, Luigi Di Caro, and Claudio Schifanella. Emerging topic detection on twitter based on temporal and social terms evaluation. In *Proceedings of the Tenth International Workshop on Multimedia Data Mining*, MDMKDD '10, pages 4:1–4:10, New York, NY, USA, 2010. ACM.

[27] James J. Cebula and Lisa R. Young. A taxonomy of operational cyber security risks. Technical report, Carnegie Mellon University, 2010.

[28] Moses S. Charikar. Similarity estimation techniques from rounding algorithms. In *Proceedings of the Thiry-fourth Annual ACM Symposium on Theory of Computing*, STOC '02, pages 380–388, New York, NY, USA, 2002. ACM.

[29] Flavio Chierichetti, Jon M. Kleinberg, Ravi Kumar, Mohammad Mahdian, and Sandeep Pandey. Event detection via communication pattern analysis. In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014.*, 2014.

[30] Julie Connolly, Mark Davidson, and Charles Schmidt. The trusted automated exchange of indicator information. Technical report, The MITRE Corporation, 2014.

[31] Corinna Cortes and Vladimir Vapnik. Support-vector networks. *Machine learning*, 20(3):273–297, 1995.

[32] Andrew Crooks, Arie Croitoru, Anthony Stefanidis, and Jacek Radzikowski. #earthquake: Twitter as a distributed sensor system. *Trans. GIS*, 17(1):124–147, 2013.

[33] Douglass R. Cutting, David R. Karger, Jan O. Pedersen, and John W. Tukey. Scatter/gather: A cluster-based approach to browsing large document collections. In *Proceedings of the 15th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '92, pages 318–329, New York, NY, USA, 1992. ACM.

[34] Qi Dang, Feng Gao, and Yadong Zhou. Early detection method for emerging topics based on dynamic bayesian networks in micro-blogging networks. *Expert Syst. Appl.*, 57:285–295, 2016.

[35] Qiming Diao and Jing Jiang. A unified model for topics, events and users on twitter. In *Proceedings of the 2013 Conference on Empirical Methods in Natural Language Processing, EMNLP 2013, 18-21 October 2013, Grand Hyatt Seattle, Seattle, Washington, USA, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1869–1879, 2013.

[36] Wenwen Dou, K Wang, William Ribarsky, and Michelle Zhou. Event detection in social media data. In *IEEE VisWeek Workshop on Interactive Visual Text Analytics-Task Driven Analytics of Social Media Content*, pages 971–980, 2012.

[37] Martin Ester, Hans-Peter Kriegel, Jörg Sander, and Xiaowei Xu. A density-based algorithm for discovering clusters a density-based algorithm for discovering clusters in large spatial databases with noise. In *Proceedings of the Second International Conference on Knowledge Discovery and Data Mining*, KDD'96, pages 226–231. AAAI Press, 1996.

[38] Yixiang Fang, Haijun Zhang, Yunming Ye, and Xutao Li. Detecting hot topics from twitter: A multiview approach. *J. Information Science*, 40(5):578–593, 2014.

[39] Atefeh Farzindar and Wael Khreich. A survey of techniques for event detection in twitter. *Computational Intelligence*, 31(1):132–164, 2015.

[40] Salvatore Gaglio, Giuseppe Lo Re, and Marco Morana. Real-time detection of twitter social events from the user's perspective. In *2015 IEEE International Conference on Communications, ICC 2015, London, United Kingdom, June 8-12, 2015*, pages 1207–1212, 2015.

[41] Salvatore Gaglio, Giuseppe Lo Re, and Marco Morana. A framework for real-time twitter data analysis. *Computer Communications*, 73:236–242, 2016.

[42] C. W. J. Granger. Investigating causal relations by econometric models and cross-spectral methods. *Econometrica*, 37, 1969.

[43] Adrien Guille and Cécile Favre. Event detection, tracking, and visualization in twitter: a mention-anomaly-based approach. *Social Netw. Analys. Mining*, 5(1):18:1–18:18, 2015.

[44] Jiawei Han, Jian Pei, and Yiwen Yin. Mining frequent patterns without candidate generation. *SIGMOD Rec.*, 29(2):1–12, May 2000.

[45] Mahmud Hasan, Mehmet A. Orgun, and Rolf Schwitter. Twitternews+: A framework for real time event detection from the twitter data stream. In *Social Informatics - 8th International Conference, SocInfo 2016, Bellevue, WA, USA, November 11-14, 2016, Proceedings, Part I*, pages 224–239, 2016.

[46] Mahmud Hasan, Mehmet A Orgun, and Rolf Schwitter. A survey on real-time event detection from the twitter data stream. *Journal of Information Science*, pages 1–21, 2017.

[47] Mengdie Hu, Shixia Liu, Furu Wei, Yingcai Wu, John T. Stasko, and Kwan-Liu Ma. Breaking news on twitter. In *CHI Conference on Human Factors in Computing Systems, CHI '12, Austin, TX, USA - May 05 - 10, 2012*, pages 2751–2754, 2012.

[48] Jiajia Huang, Min Peng, and Hua Wang. Topic detection from large scale of microblog stream with high utility pattern clustering. In *Proceedings of the 8th Workshop on Ph.D. Workshop in Information and Knowledge Management, PIKM 2015, Melbourne, Australia, October 19, 2015*, pages 3–10, 2015.

[49] Alan Jackoway, Hanan Samet, and Jagan Sankaranarayanan. Identification of live news events using twitter. In *Proceedings of the 3rd ACM SIGSPATIAL International Workshop on Location-Based Social Networks*, LBSN '11, pages 25–32, New York, NY, USA, 2011. ACM.

[50] Shakira Banu Kaleel and Abdolreza Abhari. Cluster-discovery of twitter messages for event detection and trending. *J. Comput. Science*, 6:47–57, 2015.

[51] Kameswari Kotapati, Peng Liu, Yan Sun, and Thomas F. LaPorta. *A Taxonomy of Cyber Attacks on 3G Networks*, pages 631–633. Springer Berlin Heidelberg, Berlin, Heidelberg, 2005.

[52] John Krumm, Nigel Davies, and Chandra Narayanaswami. User-generated content. *IEEE Pervasive Computing*, 7(4):10–11, October 2008.

[53] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, pages 591–600, 2010.

[54] Chung-Hong Lee. Mining spatio-temporal information on microblogging streams using a density-based online clustering method. *Expert Syst. Appl.*, 39(10):9623–9641, 2012.

[55] Chung-Hong Lee and Tzan-Feng Chien. Leveraging microblogging big data with a modified density-based clustering approach for event awareness and topic ranking. *J. Information Science*, 39(4):523–543, 2013.

[56] Chung-Hong Lee, Tzan-Feng Chien, and Hsin-Chang Yang. An automatic topic ranking approach for event detection on microblogging messages. In *Proceedings of the IEEE International Conference on Systems, Man and Cybernetics, Anchorage, Alaska, USA, October 9-12, 2011*, pages 1358–1363, 2011.

[57] Chenliang Li, Aixin Sun, and Anwitaman Datta. Twevent: segment-based event detection from tweets. In *21st ACM International Conference on Information and Knowledge Management, CIKM'12, Maui, HI, USA, October 29 - November 02, 2012*, pages 155–164, 2012.

[58] Jianxin Li, Zhenying Tai, Richong Zhang, Weiren Yu, and Lu Liu. Online bursty event detection from microblog. In *Proceedings of the 7th IEEE/ACM International Conference on Utility and Cloud Computing, UCC 2014, London, United Kingdom, December 8-11, 2014*, pages 865–870, 2014.

[59] Jianxin Li, Jianfeng Wen, Zhenying Tai, Richong Zhang, and Weiren Yu. Bursty event detection from microblog: a distributed and incremental approach. *Concurrency and Computation: Practice and Experience*, 28(11):3115–3130, 2016.

[60] Wei Li, David M. Blei, and Andrew McCallum. Nonparametric bayes pachinko allocation. *CoRR*, abs/1206.5270, 2012.

[61] Xiaomo Liu, Armineh Nourbakhsh, Quanzhi Li, Rui Fang, and Sameena Shah. Real-time rumor debunking on twitter. In *Proceedings of the 24th ACM International Conference on Information and Knowledge Management, CIKM 2015, Melbourne, VIC, Australia, October 19 - 23, 2015*, pages 1867–1870, 2015.

[62] Amina Madani, Omar Boussaid, and Djamel Eddine Zegour. Real-time trending topics detection and description from twitter content. *Social Netw. Analys. Mining*, 5(1):59:1–59:13, 2015.

[63] Michael Mathioudakis and Nick Koudas. Twittermonitor: trend detection over the twitter stream. In *Proceedings of the ACM SIGMOD International Conference on Management of Data, SIGMOD 2010, Indianapolis, Indiana, USA, June 6-10, 2010*, pages 1155–1158, 2010.

[64] Andrew J. McMinn, Yashar Moshfeghi, and Joemon M. Jose. Building a large-scale corpus for evaluating event detection on twitter. In *Proceedings of the 22Nd ACM International Conference on Information & Knowledge Management*, pages 409–418, 2013.

[65] Andrew James McMinn and Joemon M. Jose. Real-time entity-based event detection for twitter. In *Experimental IR Meets Multilinguality, Multimodality, and Interaction - 6th International Conference of the CLEF Association, CLEF 2015, Toulouse, France, September 8-11, 2015, Proceedings*, pages 65–77, 2015.

[66] Andrew James McMinn, Yashar Moshfeghi, and Joemon M. Jose. Building a large-scale corpus for evaluating event detection on twitter. In *22nd ACM International Conference on Information and Knowledge Management, CIKM'13, San Francisco, CA, USA, October 27 - November 1, 2013*, pages 409–418, 2013.

[67] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg S Corrado, and Jeff Dean. Distributed representations of words and phrases and their compositionality. In C. J. C. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K. Q. Weinberger, editors, *Advances in Neural Information Processing Systems 26*, pages 3111–3119. Curran Associates, Inc., 2013.

[68] George A. Miller. WORDNET: A lexical database for english. In *Human Language Technology, Proceedings of a Workshop held at Plainsboro, New Jerey, USA, March 8-11, 1994*, 1994.

[69] Miles Osborne and Mark Dredze. Facebook, twitter and google plus for breaking news: Is there a winner? In *Proceedings of the Eighth International Conference on Weblogs and Social Media, ICWSM 2014, Ann Arbor, Michigan, USA, June 1-4, 2014.*, 2014.

[70] Miles Osborne, Sean Moran, Richard Mccreadie, Alexander Von Lunen, Martin Sykora, Elizabeth Cano, Neil Ireson, Craig Macdonald, Iadh Ounis, Yulan He, Tom Jackson, Fabio

Ciravegna, and Ann O'Brien. Real-time detection, tracking, and monitoring of automatically discovered events in social media. In *52nd Annual Meeting of the Association for Computational Linguistics: System Demonstrations*, pages 37–42, June 2014.

[71] Miles Osborne, Sean Moran, Richard McCreadie, Alexander von Lünen, Martin D. Sykora, Amparo Elizabeth Cano, Neil Ireson, Craig Macdonald, Iadh Ounis, Yulan He, Tom Jackson, Fabio Ciravegna, and Ann O'Brien. Real-time detection, tracking, and monitoring of automatically discovered events in social media. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics, ACL 2014, June 22-27, 2014, Baltimore, MD, USA, System Demonstrations*, pages 37–42, 2014.

[72] Miles Osborne, Saa Petrovi, Richard Mccreadie, Craig Macdonald, and Iadh Ounis. Bieber no more: First story detection using twitter and wikipedia. In *In SIGIR 2012 Workshop on Time-aware Information Access*, 2012.

[73] Olutobi Owoputi, Brendan O'Connor, Chris Dyer, Kevin Gimpel, Nathan Schneider, and Noah A. Smith. Improved part-of-speech tagging for online conversational text with word clusters. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 9-14, 2013, Westin Peachtree Plaza Hotel, Atlanta, Georgia, USA*, pages 380–390, 2013.

[74] Ozer Ozdikis, Pinar Senkul, and Halit Oguztuzun. Semantic expansion of tweet contents for enhanced event detection in twitter. In *Proceedings of the 2012 International Conference on Advances in Social Networks Analysis and Mining (ASONAM 2012)*, ASONAM '12, pages 20–24, Washington, DC, USA, 2012. IEEE Computer Society.

[75] Ruchi Parikh and Kamalakar Karlapalem. ET: events from tweets. In *22nd International World Wide Web Conference, WWW '13, Rio de Janeiro, Brazil, May 13-17, 2013, Companion Volume*, pages 613–620, 2013.

[76] Jeffrey Pennington, Richard Socher, and Christopher D. Manning. Glove: Global vectors for word representation. In *Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, 2014.

[77] Sasa Petrovic, Miles Osborne, and Victor Lavrenko. Streaming first story detection with application to twitter. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 2-4, 2010, Los Angeles, California, USA*, pages 181–189, 2010.

[78] Sasa Petrovic, Miles Osborne, Richard McCreadie, Craig Macdonald, Iadh Ounis, and Luke Shrimpton. Can twitter replace newswire for breaking news? In *Proceedings of the Seventh International Conference on Weblogs and Social Media, ICWSM 2013, Cambridge, Massachusetts, USA, July 8-11, 2013.*, 2013.

[79] Swit Phuvipadawat and Tsuyoshi Murata. Breaking news detection and tracking in twitter. In *Proceedings of the 2010 IEEE/WIC/ACM International Conference on Web Intelligence and International Conference on Intelligent Agent Technology - Workshops, Toronto, Canada, August 31 - September 3, 2010*, pages 120–123, 2010.

[80] Alun Preece, Will Webberley, and Dave Braines. Tasking the tweeters: Obtaining actionable information from human sensors. In *SPIE Defense+ Security*, pages 946402–946402. International Society for Optics and Photonics, 2015.

[81] Daniele Quercia, Harry Askham, and Jon Crowcroft. Tweetlda: supervised topic classification and link prediction in twitter. In *Web Science 2012, WebSci '12, Evanston, IL, USA - June 22 - 24, 2012*, pages 247–250, 2012.

[82] Anand Rajaraman and Jeffrey David Ullman. *Mining of Massive Datasets*. Cambridge University Press, New York, NY, USA, 2011.

[83] Radim Řehůřek. Making sense of word2vec, 2014. `https://rare-technologies.com/making-sense-of-word2vec/`.

[84] Alan Ritter, Mausam, Oren Etzioni, and Sam Clark. Open domain event extraction from twitter. In *The 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '12, Beijing, China, August 12-16, 2012*, pages 1104–1112, 2012.

[85] Alan Ritter, Evan Wright, William Casey, and Tom M. Mitchell. Weakly supervised extraction of computer security events from twitter. In *Proceedings of the 24th International Conference on World Wide Web, WWW 2015, Florence, Italy, May 18-22, 2015*, pages 896–905, 2015.

[86] Walisa Romsaiyud. Detecting emergency events and geo-location awareness from twitter streams. In *The International Conference on E-Technologies and Business on the Web (EBW2013)*, pages 22–27, 2013.

[87] Heungmo Ryang and Unil Yun. High utility pattern mining over data streams with sliding window technique. *Expert Systems with Applications*, 57:214 – 231, 2016.

[88] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web, WWW 2010, Raleigh, North Carolina, USA, April 26-30, 2010*, pages 851–860, 2010.

[89] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Tweet analysis for real-time event detection and earthquake reporting system development. *IEEE Trans. Knowl. Data Eng.*, 25(4):919–931, 2013.

[90] Hassan Sayyadi, Matthew Hurst, and Alexey Maykov. Event detection and tracking in social streams. In *Proceedings of the Third International Conference on Weblogs and Social Media, ICWSM 2009, San Jose, California, USA, May 17-20, 2009*, 2009.

[91] Hinrich Schütze and Craig Silverstein. Projections for efficient document clustering. In *Proceedings of the 20th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR '97, pages 74–81, New York, NY, USA, 1997. ACM.

[92] Dmitriy Selivanov. Glove vs word2vec revisited, 2015. `http://dsnotes.com/post/glove-enwiki/`.

[93] Devavrat Shah and Tauhid Zaman. Community detection in networks: The leader-follower algorithm. *arXiv preprint arXiv:1011.0774*, 2010.

[94] David Shepard. Nonparametric bayes pachinko allocation for super-event detection in twitter. In *TENCON 2014-2014 IEEE Region 10 Conference*, pages 1–5. IEEE, 2014.

[95] Chris Simmons, Charles Ellis, Sajjan Shiva, Dipankar Dasgupta, and Qishi Wu. Avoidit: A cyber attack taxonomy. Technical report, University of Memphis, 2009.

[96] Giovanni Stilo and Paola Velardi. Efficient temporal mining of micro-blog texts and its application to event discovery. *Data Min. Knowl. Discov.*, 30(2):372–402, 2016.

[97] Symantec. Internet security threat report, 2016. `https://www.symantec.com/content/dam/symantec/docs/reports/istr-21-2016-en.pdf`.

[98] Pang-Ning Tan, Michael Steinbach, and Vipin Kumar. *Introduction to Data Mining, (First Edition)*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 2005.

[99] Nicholas A. Thapen, Donal Stephen Simmie, and Chris Hankin. The early bird catches the term: combining twitter and news data for event detection and situational awareness. *J. Biomedical Semantics*, 7:61, 2016.

[100] Sayan Unankard, Xue Li, and Mohamed A. Sharaf. Location-based emerging event detection in social networks. In *Web Technologies and Applications - 15th Asia-Pacific Web Conference, APWeb 2013, Sydney, Australia, April 4-6, 2013. Proceedings*, pages 280–291, 2013.

[101] Sayan Unankard, Xue Li, and Mohamed A. Sharaf. Emerging event detection in social networks with location sensitivity. *World Wide Web*, 18(5):1393–1417, 2015.

[102] Clemens Valens. A really friendly guide to wavelets. 1999.

[103] Cybersecurity Ventures. Zero day report, 2016. `http://cybersecurityventures.com/zero-day-vulnerabilities-attacks-exploits-report-2017/`.

[104] The Verge. Google docs users hit with sophisticated phishing attack, 2017. https://www.theverge.com/2017/5/3/15534768/google-docs-phishing-attack-share-this-document-with-you-spam.

[105] Sarah Vieweg, Amanda Lee Hughes, Kate Starbird, and Leysia Palen. Microblogging during two natural hazards events: what twitter may contribute to situational awareness. In *Proceedings of the 28th International Conference on Human Factors in Computing Systems, CHI 2010, Atlanta, Georgia, USA, April 10-15, 2010*, pages 1079–1088, 2010.

[106] Farida Vis. Twitter as a reporting tool for breaking news: Journalists tweeting the 2011 uk riots. *Digital journalism*, 1(1):27–47, 2013.

[107] Maximilian Walther and Michael Kaisser. Geo-spatial event detection in the twitter stream. In *Advances in Information Retrieval - 35th European Conference on IR Research, ECIR 2013, Moscow, Russia, March 24-27, 2013. Proceedings*, pages 356–367, 2013.

[108] Xiaofeng Wang, Matthew S. Gerber, and Donald E. Brown. Automatic crime prediction using events extracted from twitter posts. In *Social Computing, Behavioral - Cultural Modeling and Prediction - 5th International Conference, SBP 2012, College Park, MD, USA, April 3-5, 2012. Proceedings*, pages 231–238, 2012.

[109] Jianshu Weng and Bu-Sung Lee. Event detection in twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*, 2011.

[110] Computer World. Mongodb ransomware attacks and lessons learned, 2016. http://www.computerworld.com/article/3157766/linux/mongodb-ransomware-attacks-and-lessons-learned.html.

[111] Runquan Xie, Feida Zhu, Hui Ma, Wei Xie, and Chen Lin. Clear: A real-time online observatory for bursty and viral events. *PVLDB*, 7(13):1637–1640, 2014.

[112] Wei Xie, Feida Zhu, Jing Jiang, Ee-Peng Lim, and Ke Wang. Topicsketch: Real-time bursty topic detection from twitter. *IEEE Trans. Knowl. Data Eng.*, 28(8):2216–2229, 2016.

[113] Yue You, Guangyan Huang, Jian Cao, Enhong Chen, Jing He, Yanchun Zhang, and Liang Hu. GEAM: A general and event-related aspects model for twitter event detection. In *Web Information Systems Engineering - WISE 2013 - 14th International Conference, Nanjing, China, October 13-15, 2013, Proceedings, Part II*, pages 319–332, 2013.

[114] Xiaoming Zhang, Xiaoming Chen, Yan Chen, Senzhang Wang, Zhoujun Li, and Jiali Xia. Event detection and popularity prediction in microblogging. *Neurocomputing*, 149:1469–1480, 2015.

[115] Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. Comparing twitter and traditional media using topic models. In *Advances in Information Retrieval - 33rd European Conference on IR Research, ECIR 2011, Dublin, Ireland, April 18-21, 2011. Proceedings*, pages 338–349, 2011.

[116] Deyu Zhou, Liangyu Chen, and Yulan He. An unsupervised framework of exploring events on twitter: Filtering, extraction and categorization. In *Proceedings of the Twenty-Ninth AAAI Conference on Artificial Intelligence, January 25-30, 2015, Austin, Texas, USA.*, pages 2468–2475, 2015.

[117] Xiangmin Zhou and Lei Chen. Event detection over twitter social media streams. *VLDB J.*, 23(3):381–400, 2014.

[118] Bonnie Zhu, Anthony Joseph, and Shankar Sastry. A taxonomy of cyber attacks on scada systems. In *Proceedings of The 2011 IEEE International Conference on Internet of Things (iThings'11)*, pages pp. 380–388, 2011.

[119] Andrzej Zolnierek and Bartlomiej Rubacha. The empirical study of the naive bayes classifier in the case of markov chain recognition task. In *Computer Recognition Systems, Proceedings of the 4th International Conference on Computer Recognition Systems, CORES'05, May 22-25, 2005, Rydzyna Castle, Poland*, pages 329–336, 2005.