ETH
Eidgenössische Technische Hochschule Zürich
Swiss Federal Institute of Technology Zurich

TIK Institut für Technische Informatik und Kommunikationsnetze

Ludovic Amruthalingam

Early Detection of Real-World Events with Twitter

Master Thesis
6th October 2014 to 6th April 2015

Main advisor: Dr. Vincent Lenders (armasuisse)
Advisor: David Gugelmann (ETH Zürich)
Supervisor: Prof. Dr. B. Plattner (ETH Zürich)

Abstract

Twitter is a successful micro-blogging platform where users write about topics which are catching their interest on the moment. Thanks to its real time nature, event detection using Twitter data is a promising and active research area with various interesting applications.
In this work we focus on the detection of plane hijackings, plane crashes and wildfires events. We show that plane hijackings and plane crashes events have similar characteristics which differ greatly from wildfires events in terms of quantity, time span and thus Twitter activity.
We implement a language independent event detection system with two different core algorithms, the first one based on the frequency of tweets and the second on the content of the tweets. Although we test our system on the above specified event types, the same methods can be used for other kind of events with minor changes.
Using sampled Twitter data from October 2013 to October 2014 we achieve faster detection than traditional media like Reuters and BBC for plane hijackings and wildfires. For plane crashes our detectors are faster than BBC but slower than Reuters. To give an idea of our results, we are able to detect:

- 100% of the plane hijackings within 7.5 hours with 70% precision
- 85% of the plane crashes within 18 hours with 100% precision
- 65% of the wildfires within 72 hours with 80% precision

Ora et labora

Travail effectué sous le patronage de Saint Benoît.

Acknowledgments

Je voudrais remercier mes arrières grands-parents Papi et Mamidou, mon grand oncle Ronron, mon grand-père Bon Papa et ma tante Apolline pour leur intercession.

Je voudrais remercier le père Jean de Belleville, Monsieur l'abbé Dähler et les Bénédictines de l'Adoration perpétuelle du Très Saint Sacrement de l'Abbaye Saint Pierre de Mas-Grenier pour leurs prières.

I would like to express my thanks to Dr. Vincent Lenders for his kind guidance, to David Gugelmann for his advices and to Prof. Bernhard Plattner for his supervision.
I am also grateful to Dr. Albert Blarer, Dr. Sean Rooney, Dr. Andreas Schade and to the armasuisse team for their interesting ideas and remarks.

I would like to thank my dear Tatha and Aachi.

Je voudrais remercier mes chères Patty et tatie Marie.

Je voudrais remercier toute ma famille, particulièrement Marc, Artémis et Esther.

Je voudrais remercier l'École polytechnique fédérale de Lausanne, l'Eidgenössische Technische Hochschule de Zürich et la Suisse pour m'avoir offert l'opportunité d'étudier et armasuisse pour avoir organisé ce travail.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

The general goal of this work is to detect events. But what is an event? Up to this date, Philosophy doesn't offer an universal definition as multiple different theories exist. We focus on real-world events so we can start with the simple physical definition of an event: *A phenomenon or occurrence located at a single point in space-time*. But this is too limited... We will see in the section 1.2 Motivation that our horizon is vast and to restrict ourselves to the physical definition would be counterproductive. So we widen the scope of the events to detect. The details are given in the section 2.2.1 Defining an Event, we first need to introduce the subject further to understand the reasons behind our choices.

Before social media, event detection techniques were applied on traditional information sources. The DARPA TIDES program goal is to *enable English-speaking users to access, correlate, and interpret multilingual sources of real-time information and to share the essence of this information with collaborators* [1]. From the TIDES program, spawned in 1997 the Topic Detection and Tracking (TDT) research program with aim to *develop algorithms for discovering and threading together topically related material in streams of data such as newswire and broadcast news in both English and Mandarin Chinese* [2]. Similarly to what we want to do with Twitter the developed techniques allow to automatically extract news of interest and keep track of their evolution. According to [3] we can classify these methods in mainly two types, the document pivot techniques where events are detected by clustering documents based on their textual similarity, and the feature pivot techniques where events are detected by identifying topics with sudden increase in activity. These methods were designed to work with traditional media data i.e. relatively long, structured, relevant and well constructed documents. However we will show in the next paragraph that social media data is completely different and that these methods can't be used directly.

Social media are now very popular and widely spread. Through these services, users interact with each others, create virtual groups and communities, exchange user-generated content. One main innovation is that each user has now become an active source that can create content and share it among the other users whereas with traditional media the number of content producing sources stays limited. The majority of users produce content about recent events which catch their interest but also their current state, what they are experiencing/feeling, what interest them in general. This is very different from traditional medias which aim to report events of interest. One can easily guess that the content produced by users in social media will contain a large proportion of uninformative, irrelevant data. The users aren't working together towards a common goal so it isn't possible to assume that the produced content from different users is structured in any way. Most of the users aren't producing content in a formal systematic organized way, they make mistakes, imprecisions and omissions. Because there are many users then the quantity of data produced is huge so we need methods with low computational cost. From these reasons we see that traditional medias methods aren't suited to detect events with social media data. After stating all these negative points, it is important to think about why detecting events with social media data is still promising. We think that even if only a small

[1] http://www.itl.nist.gov/iad/mig/tests/tdt/

[2] See footnote 1

fraction of the data covers the events we are interested in, the large amount of data makes up for it. Then because every user is a source, the potential coverage and reactiveness of the event reporting capability is better than with traditional media. So there is hope to catch all occurrences of events even faster than what traditional media can achieve.

In this project we use the data of the successful social media platform called Twitter and develop algorithms aiming to *"detect" events* faster than traditional media like BBC or Reuters.
Let it be said here once and for all, the expression *"detect" events* is an abuse of language, a shortcut. In this work, by *"detecting" events* we mean to write that the system detects the events' projections and effects on the Twitter data.
We have the possibility to choose between specified or general events detection. The issue with general events detection is that we are more tracking trending topics than informative events. For example this would place the latest viral joke and the Minsk ceasefire at the same level of importance. So we opt for specified event detection and assume that anyone interested into tracking events occurrences will know what type of events they want to track. This assumption allows to pinpoint and focus on specific event categories.

1.1 Twitter

Twitter is an online social networking service that enables users to send and read short 140-character messages called "tweets" [3]. Starting in 2006 from its original concept of an SMS service to communicate with a small group, Twitter became very popular and in 2014 totaled 255 million monthly active users and 500 million tweets sent per day[4]. Twitter is the typical example of a microblogging platform, aside from writing/reading tweets users can:

- Subscribe to other user's tweets (A "follows" B).
- Repeat a message from another user (A "retweets" B's tweet).
- Reply to a tweet author or mention users in a tweet (A "mention" B).
- Browse Twitter recommendation or through the search option to find interesting tweets and users.

The following aspects of Twitter should be known:

- Hashtags are important elements in a tweets, Twitter uses hashtags to group the similar tweets together, they represent the core entities and keywords in a tweet.
- Users can embed URLs in tweets to provide more content.
- Tweets contain meta-data about the tweet itself and its author. Among others, the meta-data contains the number of retweets of the tweet, the geographic location where it was posted, the timestamp when it was posted, the number of friends/followers of the author, whether the author is giving his geographic location at each tweet, his language, his country, his profile picture, etc..

With its 140 characters limit, Twitter really favors briefness. Users won't be able to elaborate complicated ideas or theories, they can only ask a short question or make a quick statement. Due to their conciseness, most tweets lifespans are extremely short, they represents a fact at a certain point of time. Users will either talk about themselves, their mindsets, their opinions, their topics of interest, or about events which catch their attention. Users also try to tweet about topics which will interest their followers so they are eager to tweet about new information that their followers may not know yet.

[3] http://en.wikipedia.org/wiki/Twitter
[4] http://www.adweek.com/socialtimes/social-media-statistics-2014/499230

1.2 Motivation

In this work we want to detect events using Twitter data, the first step toward the development of techniques to monitor the activity on Twitter. We know that Twitter users reveal their opinion, sentiments, mindset through their tweets. Thanks to its success, Twitter has now a large user base across many countries. Thus in each of these country there is a proportion of the population for which we can track their opinion, sentiments, mindsets, etc. It is clear that for the moment this is strongly biased as the Twitter users aren't randomly sampled in the population but this bias can be evaluated and given time could maybe reduce.

Being able to monitor the activity in Twitter has many applications. For example events similar to the "Arab Spring" could be predicted and perhaps influenced if not thwarted. Many informations usually not reported by the traditional medias could be inferred like crowd agitation, sentiments, interest in a particular topic. This is very useful and could be used for example to evaluate the effect of a political decision. The information extracted from Twitter could also be used to model how crowds react or just correlated with social events, economic events, market stocks...

Social medias are disruptive technologies, nothing comparable existed at this scale until now. They greatly facilitate the organization and reunion of large amount of people in short time. For example they give opportunity for politically engaged people to influence big crowd which can be particularly threatening for a government in tensed times. The science to influence crowds, rhetorics, is as old as the first men, now it is possible to influence bigger crowds and thus reap bigger effects. Studying them isn't really an option, any country needs to develop this know-how; some already started e.g. America and their attempt to build a Twitter-like social media in Cuba[5].
Possible reluctance to act as "Big Brother" should be swept away. The users already consented to give the social media companies full control over their data and these companies' interests may conflict with countries' will. One could also concern himself with personal privacy questions. This shouldn't be the case because the techniques we aim to develop won't target individuals but crowds and groups behaviors, movements, evolutions, reactions, opinions, etc. To finish with an amusing note, we can think of the social medias as the first testing grounds for the development of the science of phsychohistory in the sci-fi sense of Isaac Asimov's Foundation series...

1.3 Contribution of this Master Thesis

In this work we evaluate the feasibility to use Twitter as an early warning system for real-world events. We implement language independent event detection algorithms and evaluate their performance (precision, recall, delays between events and their detections timestamps) on randomly sampled Twitter data collected from October 2013 to October 2014. Then we compare our detectors' delays with the delay from the traditional medias like BBC and Reuters first reports. To evaluate these performances we established a ground truth i.e. a subset of all real-world events occurrences over the data timespan, for each event types.
As explained in the introduction, we decide to focus on specified event detection. We choose to track wildfires as disaster-type events and plane hijackings and plane crashes. The reason for choosing plane hijackings is that Switzerland was recently concerned by the Ethiopian Airlines ET702 hijacking on the 17th February 2014. This event was reported first on Twitter [6] and much later by traditional medias. There were few plane hijackings over the data span so we choose also plane crashes events which have similar characteristics. As we will show in chapter 2 Challenges, the plane hijackings/crashes events are really different from the wildfires events which allows us to present results of the same detectors on event types with different behavior and characteristics.

[5] http://www.nytimes.com/2014/04/04/world/americas/us-says-it-tried-to-build-a-social-media-site-in-cuba-but-failed.html
[6] https://medium.com/@thatjohn/how-i-broke-the-ethiopian-airlines-et702-hijacking-on-twitter-6c2cc1d2f2e4

1.4 Related Work

Thanks to its success Twitter has a large user base which makes it an interesting laboratory for researchers. Event detection is only a part of the work going on to understand what can be learned from Twitter. To cite a few, Java et al. [1] studied Twitter social network and its user's intentions. Andre et al. [14] analyzed the users tolerance to undesired content and what type of information is valued. The classification of users between informers (information sharing) and meformers (talking about themselves) was proposed by Naaman et al. [12] and between humans, bots and cyborgs (bot-assisted human or human-assisted bot) by Chu et al. [20]. Sharifi et al. [4] proposed a method to create a summary from all the tweets related to an arbitrarily chosen phrase. Tumasjan et al. [2] showed that the political landscape in Germany could be extracted using content analysis on the subset of tweets containing a reference to a party which also reflected the results of the federal elections.
The list could go on; Twitter and its numerous application possibilities have sparked the enthusiasm of a vast research community. Event detection is also a very active field as shown in Atefeh et al. survey [3] where the authors propose a non-exhaustive table listing event detection methods from papers up to year 2012. We complete this table with a selection of more recent papers in table 1.1.

The table 1.1 covers techniques for both specific and general event detection. With general event detection, the algorithms don't know anything about the events they are tracking so they usually extract event candidates (e.g. words, users, geographical locations, etc.) from the tweets and monitor them trying to detect anomalies and sudden bursts in the Twitter activity. As a result these methods can detect any trending topic but cannot really focus on a precise target and will usually be slower compared to specified event detection performances algorithms which have a priori knowledge of the event types characteristics and can be adapted and trained to be more efficient. On the other hand, specified event detection techniques are limited to the event types they were designed for and will perform poorly on other event types if they aren't properly retrained and re-optimized. Another issue with specified event detection techniques is that training often implies the need of labeled data and often results in language dependent models.
We would like to get as much as possible of the best of the two worlds. As we explained in the first part of the Introduction, we will track specified event types because we assume that anyone interested into tracking events occurrences will know what type of events they want to track. But we also want our algorithms to work well on new event types with as little training and required optimization as possible. As a result the techniques we implemented are inspired from papers aiming to track general events which we adapted to work well with specified event types while staying unsupervised and language independent. Simplicity being a quality, it is worth mentioning that the only features used by our system are the tweets' text and additionally for one of the two detectors, the hashtags. Thus the algorithms ideas can be easily improved, built upon and reused as foundation bricks for more complex process.

To the best of our knowledge, the research community didn't attempt yet to detect plane hijackings/crashes and wildfires using Twitter data. Vieweg et al. [16] analyzed microblog posts generated during the April 2009 Oklahoma Grassfires and the March-April 2009 Red River Floods but their goal was to *identify information that may contribute to enhancing situational awareness* and not to detect wildfire events. Plane crashes and plane hijackings are interesting to study because they are popular events generating a large activity in Twitter where users debate about what happened, create theories and relay the latest updates. We chose wildfires as well because their characteristics are completely different (cf section 2.2.2) allowing us to compare how well the same detectors performed on event types incurring very different Twitter responses.

References	Detection techniques	General features	Twitter-specific features	Application
Abdelhaq et al. (2013) [9]	Single-pass clustering on localization of bursty keywords	Word frequency and entropy	Geotags	General localized event detection
McCreadie et al. (2013) [13]	lexical key partitioning, clustering (based on locality sensitive hashing)	Term vector	-	General event detection
Walther et al. (2013) [18]	Spatial clustering, Machine Learning classification	Textual (subjectivity, present tense, sentiment,. . .), counts	Geotags	General localized event detection
You et al. (2013) [19]	Probabilistic graphical model, Gibbs sampling	Named entities	Geotags, hashtags	General event detection
Cheng et al. (2014) [6]	Cluster with space-time permutation model, Latent Dirichlet Allocation for topics	Term vector	Geotags	General event detection
Schubert et al.(2014) [17]	Exponentially weighted floating avg and var, hierarchical clustering with Ward linkage	Word frequency, word co-occurrence frequency	-	General event early detection
Li et al. (2012) [15]	Iteratively refine classification rules, clustering	CDE specific keywords, particularities	Hashtags, mention	Crime and Disaster related Events (CDE)
Ifrim et al. (2014) [10]	Tweet/term filtering, hierarchical clustering	N-gram, term vector	Hashtags	US presidential Elections (2012) events
Chen et al. (2014) [5]	Content filtering, Non-Parametric Heterogeneous Graph Scan	Sentiment, klout score (author influence), term	Hashtags, retweets, mention, replies, geotags	Civil unrest event detection and rare disease outbreak detection
Corney et al. (2014) [7]	Filtering, hierarchical clustering	N-gram	-	Football event

Table 1.1: Event Detection Techniques

1.5 Overview

The report is organized in the following manner. Chapter 2 presents the challenges we face by working with Twitter data, the problems due to the event types nature and the typical issues in event detection. In Chapter 3 we describe the system architecture and the implementation details. Chapter 4 explains our technical setup and how we organized the experiments. Chapter 5 presents the experiments' results and the performance of our system. We conclude in Chapter 6 with our achievements and finally give ideas for potential next research work in chapter 7. The reports then ends with the Appendix section and the Bibliography.

Chapter 2

Challenges

In this chapter, we detail the challenges we face to create an event detector for wildfires, plane hijacking, plane crashes working with Twitter data.

2.1 Twitter Data

The Twitter data is composed by the tweet meta-data which contains the text and many other informations as explained in section 1.1 Twitter. The first thing to note is that we have access to this data only through the Twitter Streaming API which offers just 1% of the total Twitter traffic. The quality of the sampling has been studied by Morstatter et al. [8] and shown to be biased. This is an issue we cannot solve and have to work with.

There are problems due directly to the concept of Twitter. The tweets are very short text (only 140 characters); this creates a problem with traditional natural language (NLP) techniques which usually require longer text and more context to be able to extract interesting informations. We can't just concatenate the tweets together as there is no structure between them, tweets may be in rare cases related (e.g. user transcribes a speech) but otherwise are completely independent. The facts that users make neologisms, new abbreviations, grammatical mistakes, misspelling all the time adds even more to the difficulty. This implies that we can't use NLP techniques without adapting them specifically for Twitter.

Another problem directly linked to Twitter concept is that users talk mainly about uninteresting topics (for event detection). *A San Antonio-based market-research firm Pear Analytics analyzed 2,000 tweets ... and separated them into six categories: Pointless babble 40%, Conversational 38%, Pass-along value 9%, Self-promotion – 6%, Spam 4%, News 4%*[1]. This shows that most of the data is noise which drowns interesting news informations. We need to be able to separate event related tweets from the mass of unimportant tweets.

Thanks to its success, Twitter generates continuously a huge quantity of data; this is good news in terms of potential coverage but an issue for the computational complexity. The algorithms need to stay as simple as possible to be able to handle the incoming flow of tweets. As the data comes in a streaming fashion the algorithms need also to be online. These two points restrict the algorithms we could use, again we see that the methods developed in the TDT program (cf chapter 1 Introduction) and traditional NLP techniques really aren't directly applicable on Twitter data.

[1] http://en.wikipedia.org/wiki/Twitter

2.2 Events

2.2.1 Defining an Event

As explained in the chapter 1 Introduction, defining the term event isn't simple because there is no universal definition. With section 1.2 Motivation in mind, we extend the basic physical definition of an event. We aren't interested only in the time/space information of an event but also by its evolution in the users discussions. As an example let us consider the mh370 crash: over the course of the investigation, mh370 started from a plane crash, became a plane hijacking event and finally reverted to a plane crash. The cause of the accident changed over time as well as the crash location. The users discussed about the official news releases, created theories, gave their opinions... If we restricted ourselves to the physical definition, we would lose all these informations about the users perception of the mh370 event.

We also extend our event definition to tolerate past event (e.g. 9/11/2001) and false rumors (e.g. on the 24/3/2014 a confusing boat picture was mistaken for a crashed plane). For past event the reason is that if the crowd talks about them, it means that the topic still matters for the users and as it is linked to a past real-world event, this information should be known and monitored. For false rumors, the reason is that we wish to track the crowd belief. It is also interesting to see how fast the users start trusting the crowd knowledge and how easily they are deceived.
In this way we extend our tolerance on what the event detector should report which results in more alerts that needs to be handled properly. Our event definition doesn't include viral joke (e.g. a popular joke about watching a plane hijacking movie before taking a flight) because they are only noise and uninformative.

To recapitulate, during this work we will consider as event the set containing events in the sense of the physical definition, new information about an event which already occurred (e.g. after a news release), reminiscence of users about a past event (9/11) and false rumors.
We don't consider as event the set containing viral jokes because they aren't related in any ways to a real-word event or to a real belief of the users that an event is happening.

2.2.2 Event Types Specific Characteristics

Each of the chosen event types has its own behavior, particularities and differences that we will present in this section. These differences makes it more difficult to create an algorithm which works well for all the event types.
Plane hijackings are relatively rare, during the year from October 2013 to October 2014, there were "only" 4 occurrences. When a plane hijacking happens it usually becomes a major event discussed worldwide. The physical duration of such an event is relatively short, it never lasts more than a day and the physical location is usually clear. From this description we can already guess that the plane hijacking event type "signal" should have little noise and highly recognizable spikes.

Plane crashes happen relatively more often than plane hijackings. These events don't always take a large importance in the discussions as many of the occurrences concern small private planes or relatively remote countries. The usual physical duration is also short, they never last more than a day and their physical location is also clear. We see from this description that plane crashes events behaviors are similar to plane hijackings events, a bit more noisier because of the greater number of occurrences.

Wildfires events on the other hand are very different from plane hijackings and plane crashes events. They are very frequent during the year, they can last from days to months and their locations can move and often isn't clear in the tweets. For example in 2014, 5620 wildfires happened in California alone[2]. This makes it very difficult for a given wildfire to understand when it started and where. The intuition is that the signal for wildfire events will be much wilder and noisier than for plane hijackings and plane crashes.

[2] http://en.wikipedia.org/wiki/2014_California_wildfires

2.2.3 Past Events

People still discuss about past events like 9/11/2001 or the day the music died[3] on the 3/2/1959 on their anniversaries each year. As we explained in section 2.2.1, they are interesting to track but they create an issue when the system tries to match them with recorded known event occurrences. The reason is that we don't know which past events will be of interest for the users. They are also hard to recognize because the users don't disclose much details about the events as they are considered common knowledge. It may happen that new informations related to an event is released months after the event occurrence leading to an event detection. In this case the detection is related to a past event and the system should be able to match the detection to the original event but this proves difficult as we explain in section 2.2.6 Events Detections Aggregation.

2.2.4 False Rumors

False rumors are tracked as real events because it is interesting to check how quickly they spread and how well they are accepted by the crowd. But it is very difficult to recognize if a detection is a false rumor. This ability is very important as it was demonstrated on the 23/04/2013 when the White House Twitter account was hacked and the false rumor of a bombing injuring president Obama was released. As a result the Dow Jones index and the S&P 500 crashed shortly. Such false rumors need to be recognized and treated as false information but it is hard using only Twitter data.

2.2.5 Viral Jokes

Viral jokes are the false positives we have to filter out from our detections. They usually take the form of a very popular sentence like *when ur squad couldn't hijack the plane...* which can be taken as event related information by an algorithm. They are totally uninformative and often lead to the belief that an event is happening when it isn't. These tweets are really liked by the users and usually have a higher than normal retweet count and favorite count. This illustrates the fact that users use Twitter as a recreational activity and gives insight on how we can diminish the viral jokes impact.

2.2.6 Events Detections Aggregation

The algorithms will often raise duplicates detections about the same event when the related Twitter activity lasts long enough. One hard challenge aside from triggering event detections is to automatically make sense of the detections in order to differentiate or aggregate them. A detection consists of a set of tweets which are supposed to be related to an event. From this set of tweet, we need to extract the core elements of the event, usually the location and the involved entities, and use them to differentiate the current detection from the others. This process isn't straightforward because we have to work in an online fashion (the detections come one after the other), we never know how many different events occurred and this number increases over time. For these reasons we can't use typical clustering algorithms.

Event differentiation proves especially difficult for wildfires as they can often happen in a similar location (e.g. two wildfires both in the state of California) or exactly the same location but in different point of times which can be confusing as wildfires can have a long duration. We need to be able to understand whether the detection is a new fire which started at the same place or the same fire which lasted since the first detection.

Concurrent events are also difficult to handle; the system needs to be able to separate the event related tweets and match them to their corresponding events. It wouldn't be an issue for a human to read the tweets and understand what events are actually occurring but if we want the system to analyze itself the detections it is a much harder problem. Still this capability is useful and worth looking into because it saves manual work, for example the system could track the

[3] http://en.wikipedia.org/wiki/The_Day_the_Music_Died

evolution of an event and then automatically generate a summary presenting its main aspects. This challenge is already a future work topic but we will propose a first solution attempt for the first step which is aggregating the duplicates detections together.

2.3 Language Independent Detection

Twitter has users all over the world using different languages but we want to be able to detect events anywhere so our methods should be language independent. This restricts the methods we can use in our detector algorithms to language independent techniques like n-grams, bag of words, etc. For practical reason in this work we test the methods only using English tweets but the developed methods would work with any other languages with quasi no changes. To improve our detections aggregation algorithm results we apply stemming and stop words removal but we use Lucene which is multilingual and this step can be skipped if needed.

2.4 Absence of Standard Dataset

There is no standard event detection dataset for plane hijackings, plane crashes and wildfires. This implies that we cannot test our algorithms performance against other results from the community and that we have to create our own ground truth (known real-world events which physically occurred over the data timespan) dataset to be able to evaluate our performance. We also can't use supervised machine learning techniques as there is no labeled dataset available and we can't label a dataset ourselves otherwise the trained models would become language dependent as the labeled dataset would be in English only.

2.5 Semantics

An important part of the work is to separate the event related tweets from the mass of tweets. This is done by choosing for each language and for each event type a set of keywords best describing the event type. For example for the wildfires event type and the English language, we choose the keywords "wildfire", "forest" and "fire". This choice is complex because depending on the sentence the keyword may mean something else *e.g. Spread like wildfire....* Adding to the difficulty, we need to choose for performance reason the smallest set of keywords which still best represents the event type. If we miss one keyword, event related tweets will be ignored but in the opposite case, unrelated tweets will be taken into account possibly leading to a false positive event detection. Semantics plays an important role as well during the analysis of the detections where we have to understand clearly what are the relevant entities (involved parties, location, etc.) so that we can match them together with their related event.

2.6 Efficient Algorithms

The Twitter traffic is huge which implies that the algorithms need to stay computationally simple and be fast enough to be able to support very high streaming load.
The algorithms need to be efficient also in terms of the detection speed to achieve the goal of this work of being faster than traditional media. The delay between an event occurrence and the very first tweet talking about it cannot be reduced but the extra time taken by the detector should be minimized as much as possible. A human is able to tell that something is happening directly after reading the very first tweet, we attempt to achieve this performance with our detectors.
Then comes the traditional precision and recall trade-off; our algorithm should detect as much events which are reported on Twitter as possible while having an acceptable false positive rate.

2.7 Performance Evaluation

Because of the absence of a standard dataset, we have to build the dataset with the ground truth (known real-world events which physically occurred over the data timespan) and the social

media delays. This dataset isn't complete as there may be events which really occurred but aren't present in the ground truth in which case if a detection occurs, will be ignored.
From this dataset we should then evaluate the precision, recall and delays of our detectors. The difficulty is to automate the performance evaluation. We have to match as much detections as possible automatically to the known events without making mistakes. This process isn't straightforward, many detections which aren't false positives remain unmatched and require manual work to correct. For example it is possible that the first detection of an event remains unmatched but a second detection is assigned to the real-world event resulting in a wrong delay value. To correct such mistakes, there is no other solution than going through the experiment's logs.

Chapter 3

Design and Implementation

In this chapter we present the design and implementation of our event detection system for Twitter data. It is composed of several modules which can be independently changed. We describe two different alternatives for the detection algorithms for which we will compare the performances.

3.1 Flow Chart

The system is composed of four independent modules which were all implemented using the Java programming language. The next sections detail the modules individually.

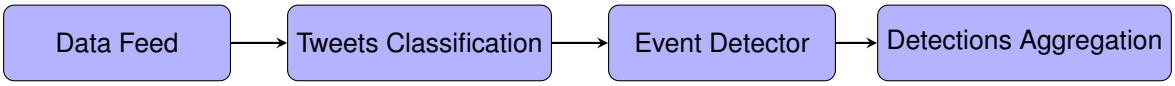


Figure 3.1: System Flow Chart

3.2 Data Feed

The data feed module retrieves the flow of tweets and canalizes it to the classification module. In the current implementation the data feed reads the tweets which are stored in avro format (*Apache Avro is a data serialization system which relies on JSON schema*[1]). To extract the tweets from the avro format, the data feed uses an hierarchy of classes which were automatically generated[2] from the avro schema using the avro-tools JAR from Apache.
We use this setup to be able to test our system on Twitter data from October 2013 to October 2014. To use the system online with fresh Twitter data, the only modification would be to modify this module and link it to the Twitter stream through the Twitter Streaming API.

The Java classes and packages of interest:

- The data feed is represented by the AvroTweetFeed class in the dataFeeder package.
- The generated classes used to deserialize the tweets are located in package ch.armasuisse.twitterdemo.avro.v2.

3.3 Classifying Event Related Tweets

Once we retrieved the tweets, we need to separate the event related tweets from the mass of uninformative tweets. We have to do this classification without any available labeled data, while

[1] https://avro.apache.org/docs/1.7.7/index.html
[2] https://avro.apache.org/docs/1.7.7/gettingstartedjava.html

staying language independent and as efficiently as possible because we have to be able to process the Twitter stream of tweets (more than a thousand tweets per second, cf. chapter 1 Introduction).

We use a keyword based approach which has the merit of being Twitter friendly as Twitter's interface allows to subscribe to tweets containing specified keywords. For each event types of interest we choose a set of keywords in every tracked language, which best represent the concept of the event type. In our case for the English language we choose:

- For plane hijackings, the keywords "plane" and "hijack"
- For plane crashes, the keywords "plane" and "crash"
- For wildfires, the keywords "wildfire", "forest" and "fire"

We chose these keywords because we made the assumption that if a tweet contained any two of them, it was highly likely that this tweet was related to the corresponding event type. There are other ways to select the keyword, for example by taking the words with top frequency from a text defining or reporting about the event type.
We are able to evaluate the quality of the classification as we will show in section 5.1 by computing the correlation of the obtained event related tweet counts with the real world events ground truth time series. We tested several combination of keywords and picked the one giving the highest correlation while containing as few keywords as possible. Having few keywords is important because we observed that the running time of the experiments increased notably even with a single additional keyword.
The classification is then done using algorithm 1.

Algorithm 1 Decide if a tweet is related to an event

procedure ISRELATEDTOEVENT($tweet$, $type$)
  $score = 0$
  for each keyword w in $type.getKewords()$ **do**
    if $tweet$ contains w **then**
      $score = score + 1$
    end if
  end for
  Return $score \geq MIN\_SCORE$
end procedure

In our system we choose $MIN\_SCORE = 2$ i.e. the tweet contains all keywords for plane hijackings/crashes and 2 out of 3 for wildfires. We apply this procedure to the stream of tweets, one tweet after the other. The resulting stream of event related tweets are then passed as input to the event detector module as they are classified.
This method is simple but powerful. It has the advantage of being language independent and also very easy to extend for other event types. For example if we want to start tracking events related to the Ukraine crisis we could set keywords such as "Ukraine", "Minsk", "Donetsk", etc.

The Java class holding the event types' keywords and taking care of the classification of the tweets is the EventType class in the Event package.

3.4 Event Detectors

The following algorithms use as input only the event related tweets resulting from the classification module. They decide at any point in time whether or not an event detection alert should be triggered.

3.4.1 Frequency based Event Detector (Detector A)

This detector was proposed by Zhao [21] as *Threshold-Based Event Detection* for events related to National League Football matches i.e. fumble, interception, touchdown, etc. The time scope of a NFL match amounts to a few hours and the detection delays to seconds but we show in the chapter 5 Results that the detector, after parameter tuning, performs well also for real-world events. We decided to call this detector frequency-based because it doesn't focus on the content of the tweets but on the quantity.

The general idea of this detector is to monitor the activity of Twitter regarding an event type and trigger detection alert if certain thresholds are breached. The main aspects are explained by the following points:

- To estimate the Twitter activity, we count the number of events related tweets and store this information in a list of time windows shifted in time.
- At each timestep (our implementation uses 15 minutes) we check each time window; if the ratio of the count from the first half of the time window with the count from the second half is below a chosen threshold (we use 0.5 i.e. the number of tweets at least doubles between the first half of the window and its second half) and if the total amount of tweets in the time window is above a chosen threshold (we use 50 tweets) then the detector triggers an alert.
- The size of the time window is an important parameter; too long would delay the detection longer than necessary but too short would miss events for which the activity grows only slowly. This issue is addressed by the use of dynamic sized time windows. Basically each time window has several possible sizes (in our system they range from one hour to ten hours); when the end of a time window is reached, the algorithm tries to expand it if this is possible and if it isn't, discards the time window.
- When an alert is triggered, it means that the Twitter activity around the event type is high because the community is discovering the event and debating about it. It is very probable that in the short future the detector will have many other occasions to trigger an alert for the same event with often the exact same informations. In order not to get submerged by duplicates detection we use a filter which, once the first detection is triggered, filters out detections related to the same event type during a chosen period of time (in our implementation 24 hours).

The procedure has to be done for each event types, it is summarized in algorithm 2.

Algorithm 2 Frequency based detector's procedure

procedure TIMESTEPUPDATE
At each timestep T
Create new time window and add it to the list of time windows
Count all event related tweets which occurred in the timestep
Add obtained count to all active time windows
For each time window, check if condition are met to trigger detection alert
Update all time windows and prepare them for next timestep
end procedure

The evident shortcoming of this detector is that it can't differentiate concurrent events of the same type because it focuses only on the Twitter activity around the event type concept and not about the events themselves. It can happen for example that the same detection contains tweets related to two different plane crashes.
Also it cannot detect events of the same type separated in time by less than the delay during which detections are filtered out in order to prevent duplicate detection flooding (24 hours in our system).
The detector must count a minimum number of event related tweets before it can trigger an alert. In our case it needs at least 50 tweets which is a lot when we think that a human could

understand that something is happening from the first 2-3 tweets. This could incur further detection delays if the event isn't very popular and the tweets are coming slowly in time.

But this isn't as bad as it looks, we will see in the Results chapter that the detector performs well enough and that the actual delay caused by waiting for 50 tweets isn't high at least for the event types we track. It is true that the detector doesn't use well the content of the tweets but this can be a strength when we focus on an event type concept which isn't very precise because it grasps the tweets together without separating them too strictly. It is less precise but gives more of an overview. As this detector focuses on the activity in Twitter, we can follow the crowd interest in a subject, which corresponds well to the motivation of this work.

The Java classes and packages of interest are:

- The time window logic is located in package timeWindow, the TimeWindow class represents a single time window and the TimeWindowManager class manages the list of active time windows.
- The main detector logic is coded in the FrequencyBasedEventDetector class in the analyzer package.

3.4.2 Content based Event Detector (Detector B)

This algorithm was created by using some ideas developed in a paper by Krstajic et al. [11] focusing on general event detection.
The goal of this detector is to improve over the shortcomings of the frequency based detector. The content based detector uses the content of the tweets, it is able to detect concurrent events and requires fewer tweets before triggering a detection.

The general idea of the content based detector is to extract the core entities from each tweet and cluster the tweets with similar entities together. Once a cluster's size reaches a threshold, the detector triggers an alert. The extraction of the core entities is the central aspect of the algorithm, it should be precise and fast enough to handle the Twitter stream and again, language independent. The main aspects of the detector are explained in the following points:

- We use directly the hashtags of the tweets as their core entities best representing the tweet's content. As we explained in section 1.1 Twitter, the hashtags are chosen by the users to help Twitter reference their tweets. With this knowledge, the users have a strong incentive to choose well the hashtags because it enables them to reach a larger audience, so they usually pick the core concepts/entities of their tweets as hashtags.
- A cluster of tweets is represented by a set containing all its tweets' hashtags.
- Once the hashtags are extracted from a tweet, the detector checks all active clusters' sets for a non-empty intersection. There are three possible cases:
  - No cluster contains any of the tweet's hashtags; in this case we create a new cluster containing the tweet with the same set of hashtags.
  - A single cluster gives a non-empty intersection; in this case we add the tweet to this cluster and update its hashtags set.
  - Many clusters have non-empty intersections; in this case we merge all these clusters along with the tweet into a single cluster for which we recompute the hashtags set.
- If a cluster's size reaches a threshold (our system uses 5-10 tweets) and if this cluster didn't already trigger a detection, then the detector raise an alert.
- We delete clusters in case their set contains too many hashtags (we allow a maximum of 10 hashtags) because we assume that these clusters have become too general, not specific enough to represent an event. We also delete clusters when their last update (i.e. the last tweet added) is too old (we use a limit of 24 hours) because we assume that their topics aren't active enough to represent active events which just occurred.

Algorithm 3 Content based detector's procedure

procedure PROCESSTWEET
For each event related tweet
Extract tweet's hashtags
Try assign to existing cluster, else create new cluster
Check updated cluster if condition is met to trigger detection alert
Remove too old or too general clusters
end procedure

The procedure is done for each event types, it is summarized in algorithm 3.

The shortcomings of this detector come mainly from the fact that we rely on hashtags to represent the core entities of the tweets. There are some cases when this isn't true but the main problem happens when the hashtags used by the users aren't popular and aren't used by other users. This causes the tweet to be ignored even when they could be relevant to an event. It is also possible that we miss the very first tweets talking about an event because their authors didn't choose the right hashtags which are being reused by the other users.

This detector detects events quickly because it requires few tweets to trigger a detection. It performs better on events specific enough so that users talking about the event use the same hashtags in their tweets. When this isn't the case i.e. for more general concepts, it has more trouble and may miss events.

The content based detector logic is coded in the ContentBasedEventDetector Java class from the analyzer package and in the EventCandidate Java class from the event package.

3.5 Detections Aggregation

This module implements a first attempt at solving the detections clustering problem which goes beyond the scope of this work as explained in section 2.2.6 Events Detections Aggregation.
We define a new detection as the set of tweets which triggered the event alert and an aggregate representative as the set of tweets obtained from the union of all the detections in the aggregate. We develop a solution based on the computation of a similarity score between the new detection and all the aggregate representatives. The similarity score is computed using algorithm 4.

Algorithm 4 Similarity of new detection D and aggregate representative C

procedure SIMILARITYSCORE(D, C)
$score = 0$
for each word n_C in $C.getTopKngramTerms()$ **do**
for each word n_D in $D.getTopKngramTerms()$ **do**
if Jaro-Winkler$(n_D, n_C) \geq MIN_STRING_SIMILARITY$ **then**
$score = score + 1$
end if
end for
end for
Return $score$
end procedure

For each set, the top K n-grams are extracted from the concatenated tweets with $K = 5$ and $n = 2$. In our experiments we remove stop words and apply stemming before extracting the n-grams to improve the quality of the aggregation. This preprocessing is done with lucene library, a multilingual library, and need slight adaptation before working with other languages than English. We prefer to work with the top K n-grams' terms (i.e. the individual words) because it is more difficult to compute the effective similarity between two groups of words than between two words.

The similarity between two strings is computed using the Jaro-Winkler distance[3] and we set $MIN\_STRING\_SIMILARITY = 0.8$.
Using this similarity score function we aggregate the new detections with algorithm 5.

Algorithm 5 Aggregate new detection D

procedure AGGREGATE(D)
  $max\_score = 0$
  for C in $allAggregateRepresentatives()$ **do**
    $max\_score = max(max\_score, similarityScore(D, C))$
  end for
  if $max\_score \geq MIN\_SIMILARITY$ **then**
    Aggregate D with corresponding aggregate representative
  else
    Establish D as a new aggregate representative
  end if
end procedure

In our system we take $MIN\_SIMILARITY = 3$. Every aggregate representatives ideally corresponds to an individual event which really occurred. If the new detection isn't similar to any of them, we assume that it relates to a new event.

The Java class taking care of the aggregation of the detections is the DetectedEventsManager class in the Event package.

[3] http://en.wikipedia.org/wiki/Jaro%E2%80%93Winkler_distance

Chapter 4

Experiment Design

The experiments we ran aimed at finding the best configuration parameters for the different modules and find out how fast the system was able to report an event occurrence. The analysis (graphs, features, etc.) of the data obtained from the experiments were realized using the R programming language.

4.1 Available Data

We run the experiments on Twitter data i.e. tweets and their meta-data, from the 1st October 2013 to the 7th October 2014 (around 850GB). The data was collected by armasuisse through the Twitter streaming API without any filters so we have around 1% of the total Twitter traffic of this time period. The tweets were serialized in avro format[1], the avro schema is available in Appendix C.

4.2 Technical Configuration

The experiments were all run on a laptop running Windows 7 Enterprise, through eclipse and Java 8, with an Intel(R) Core(TM) i7-3520M CPU @ 2.90GHz processor, 16 GB RAM and the data stored on SSD Samsung 850 Pro.

We used the following Java libraries to implement the system:

- avro tools 1.7.7 from Apache to read avro format
- JRI 0.5-0 to call R scripts from Java
- lucene 3.6.2 to extract n-grams
- opencsv 3.3 to write to CSV files
- SimMetrics 1.6.2 for the Jaro-Winkler distance between two strings

We also used the following R libraries to analyze the results of the experiments:

- ggplot2 0.9.3.1 for plotting the graphs
- scales 0.2.4 for plotting the graphs
- Hmisc 3.15-0 for plotting cumulative distribution functions
- qcc 2.6 for computing EWMA
- plyr 1.8.1 for data preprocessing
- chron 2.3-45 for manipulating the date and times

[1] https://avro.apache.org/docs/1.7.7/index.html

- caret 6.0-41 for training models
- stats 2.3.15 for fitting models
- fpc 2.1-9 for clustering

4.3 Ground Truth

Before evaluating the performance of our system, we first have to establish the ground truth i.e. the list of real-world known events which occurred between October 2013 and October 2014 and their timestamps. To gather these informations we used lists available in Wikipedia (list of plane hijackings[2], list of plane crashes[3], list of wildfires[4]), browsed google, BBC, Reuters and completed the resulting lists with new events we discovered after running the detectors on the data.

Using this method we collected 43 important events: 5 hijackings, 18 crashes and 20 wildfires. It is important to note that this list isn't complete. At some point we stopped adding events we discovered in order to keep the same ground truth for all experiments. The list of known events with the URLs to the sources used are available in Appendix A

To be able to compare the delays with traditional media, we searched for each of these events, their first mention in articles from BBC and reports from Reuters. This data isn't organized to be easily accessible (we had to browse their website with their limited search options and google) plus it is only possible to get an article/report latest update's timestamp. So the gathered timestamps don't represent perfectly the very first mention of the event by BBC and Reuters but still it allows us to see how the system compares with traditional media. The list of known events' traditional media detections' timestamps is available in Appendix B.

4.4 Classifier Performance Evaluation

We run the system up to the classifier module over the data. During the experiments, we compute the daily and hourly event related tweet counts for each event type with and without retweets. Using these counts we can create graphs to visualize the twitter event related activity.

To evaluate the quality of the classification, we create a time series from the ground truth known events timestamps and correlate it with the measured event related tweet counts. We also compute the lagged correlation i.e. the correlation of the known events time series with the shifted event related tweets counts vector. This gives us insight on the time lag between the occurrence of an event and the reaction it creates in the twitter data.

To filter out noise, in case it has an effect on the correlation, we apply an exponential weighted moving average filter (EWMA, *a type of infinite impulse response filter that applies weighting factors which decrease exponentially*[5]) on top of the counts data. From Wikipedia: *that is we compute for each count* x_i *the observation* $z_i = \lambda * \bar{x_i} + (1 - \lambda) * z_{i-1}$ *by computing the rational subgroup average,* $\bar{x_i}$ (in our case there is always only one measure for the tweet count so averaging is trivial), *and then combining that new subgroup average with the running average of all preceding observations,* z_{i-1} *with* λ *the weight given to the most recent rational subgroup mean* (i.e. the most recent count). To be sure to pick the filter giving the best possible results we vary the λ parameter between maximum filter ($\lambda = 0$, i.e. always taking the successive average of the counts, maximum dependence on past data) and no filter ($\lambda = 1$, i.e. no filter, no dependence on past data).

[2] http://en.wikipedia.org/wiki/List_of_aircraft_hijackings
[3] http://en.wikipedia.org/wiki/List_of_accidents_and_incidents_involving_commercial_aircraft
[4] http://en.wikipedia.org/wiki/List_of_wildfires
[5] http://en.wikipedia.org/wiki/Moving_average#Exponential_moving_average and http://en.wikipedia.org/wiki/EWMA_chart

As this analysis is specific and only needed for these experiments, we create a separate Java class called TweetFluxAnalyzer inside the analyzer package. We use this class to browse the data and compute the daily/hourly event related tweets counts.

4.5 Detector Performance Evaluation

We run the system up to the detector module over the data without retweets (cf section 5.2) and then analyze the list of detections triggered by the detector. This process is repeated so that we can find the best parameters values i.e. the ratio and minimum tweet count threshold for the frequency based detector and the minimum tweet count threshold for the content based detector.
For the frequency detector when varying one of the two parameters, the second parameter is set to the value giving the best performance. The adaptive time window sizes are set to 1, 5 and 10 hours. We don't test other sizes because we already cover thanks to the adaptive time window technique a large enough range. To evaluate how well the detectors perform we make the following analysis.

On the basis of the ground truth, we compute the recall, precision (see figure 4.1) and delays values. We then compare the observed delays with traditional media's first reports' delays. The comparison is made over the same set of detected events otherwise it wouldn't make sense.
It isn't feasible to match all the detections with the ground truth manually so we implement the following system which is similar to our detection aggregation module from section 3.5 Detections Aggregation:

- For each known event from the ground truth, we create a set of distinctive keywords.
- For each detection, we concatenate the related tweets together, remove stop words, apply stemming and finally extract the top K n-grams.
- For each detection we try matching its n-grams with the known events keywords. If the maximum matching score is above a threshold (we use a threshold of 3 matches) we match the detection with the corresponding known event otherwise we leave it as unrecognized to be manually matched afterward.

This method leaves quite a few unrecognized detections which then need to be manually classified. For practical reasons, we are testing the performance of the system on English tweets.
We use the following formulas for recall and precision:

$$\text{Recall} = \frac{\text{\# detected known events}}{\text{total \# known events}} \text{ and Precision} = \frac{\text{\# detected known events}}{\text{\# detected known events} + \text{\# false positives}}$$

The experiments resulting false positive counts and detection counts are available in Appendix D.

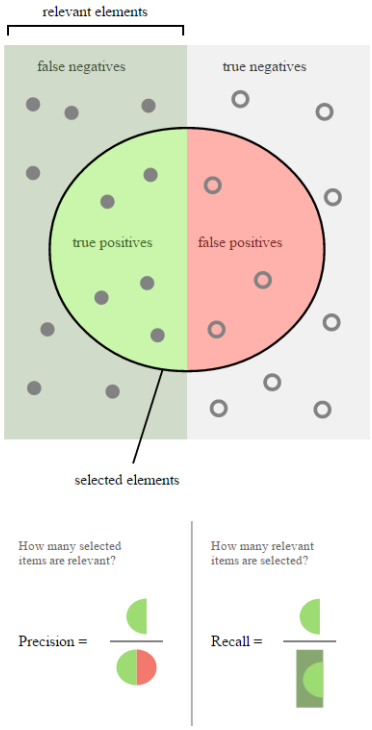


Figure 4.1: Wikipedia precision and recall definition

4.6 Detections Aggregation Performance Evaluation

We run the whole system with the same settings as in section 4.5 and then use the triggered detections as input to the detection aggregation module. As this module is outside the scope of the master thesis and mainly because of the lack of time, we couldn't check all the logs manually and verify individually the aggregates representatives. Still in order to get an idea of the results of the algorithm, we compare the resulting number of aggregates with the number of false positives and detected events and discuss on the conditions under which the algorithm works best.

Chapter 5

Results

In this chapter we present for each module the results of the experiments described in chapter 4 Experiment Design.

5.1 Classification

In this section we explain the results of the experiments on the classification module. For each event type, we start by discussing the output of the classifier then its correlation with the ground truth time series after applying the exponential weighted moving average filter to reduce the noise. The main points to remember are:

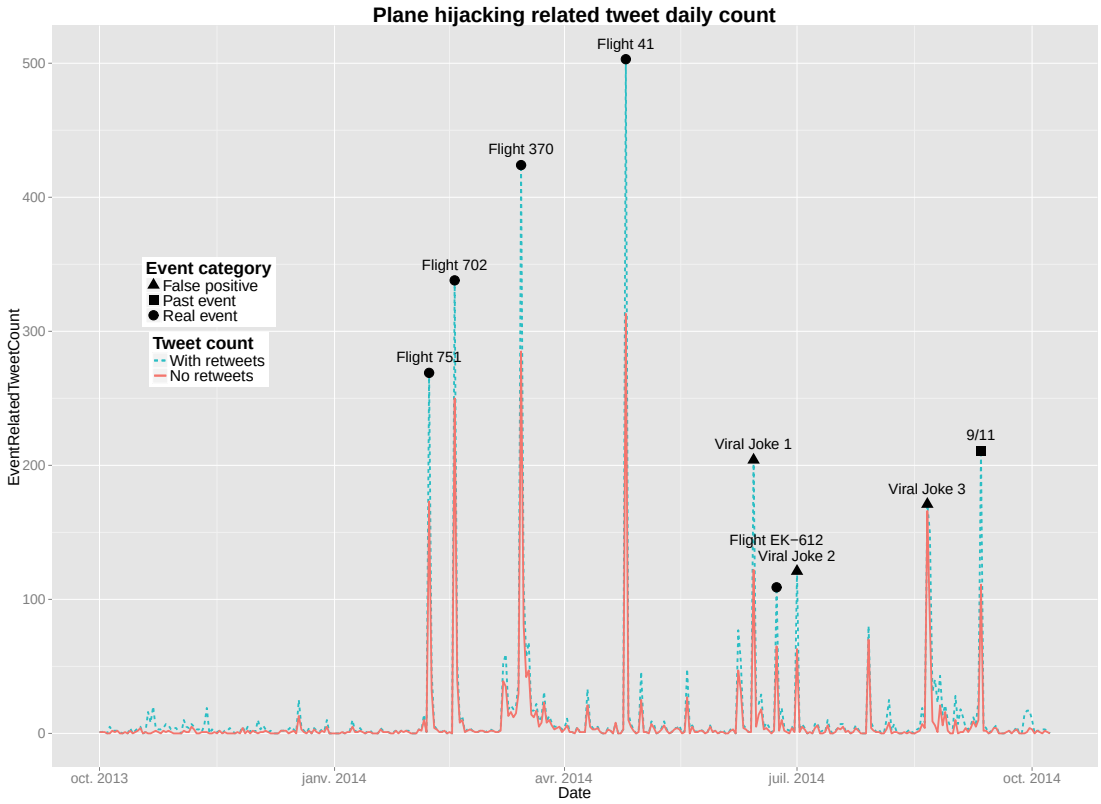
- Classification works well with plane hijackings and crashes but is noisier with wildfires.
- The daily counts data correlates better with the ground truth time series than the hourly counts data and taking the lagged correlation (i.e. the correlation between the shifted count vector and the ground truth) can improve the results.
- Removing retweets from the count data doesn't change much the correlation with the ground truth. We see in the graphs that the behavior of the data stays similar which strongly suggests that it can be beneficial for the frequency based detector to ignore retweets.

5.1.1 Plane Hijackings

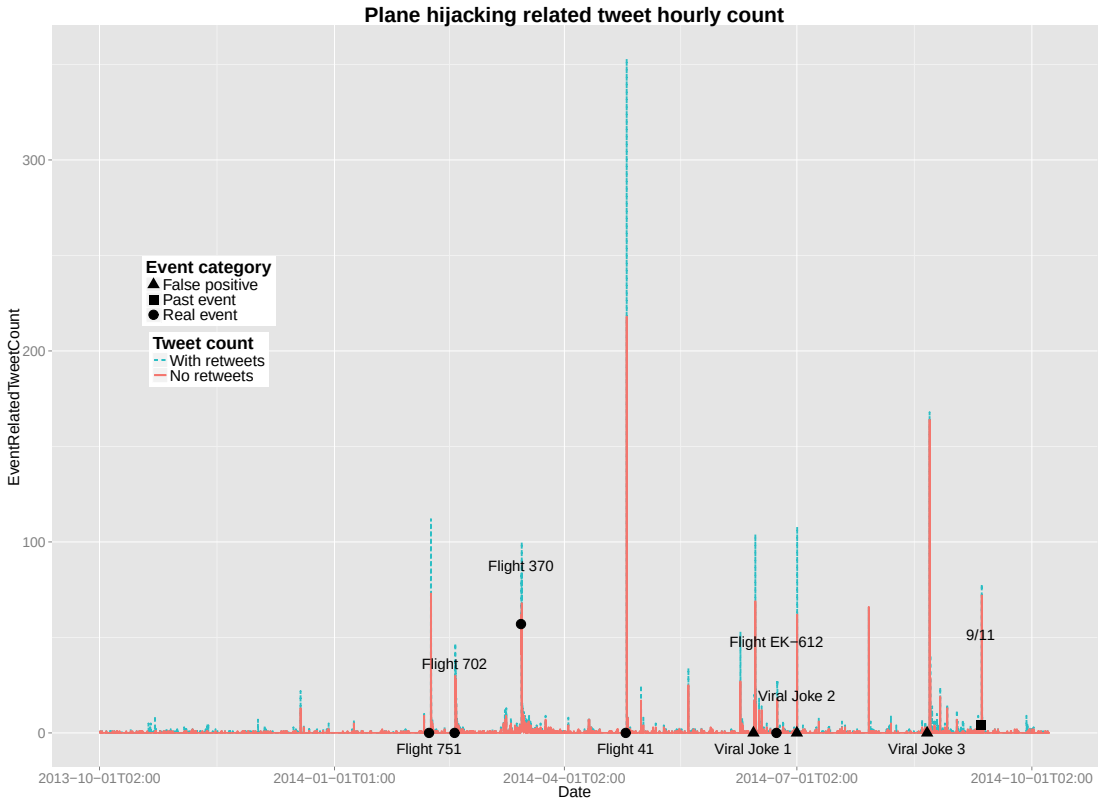
Data Visualization

The plane hijacking related tweets daily and hourly counts are presented in figure 5.1.
In general we can see that except for the spikes, the usual amount of event related tweets stays small which means that the classification module is working properly. The spikes are well separated from the rest of the data and we see the activity ceases cleanly in a few days. This represents the user interest in the event decaying quickly as nothing new happens about the event. This illustrates that plane hijacking events physically happen over a short period of time. The majority of the spikes are related to known events or to one past event (9/11) but the rest corresponds to viral jokes (e.g. joke about watching a plane hijacking movie before taking a flight) which will be considered as false positive should the detectors raise an alert. If the classifier was perfect, it would have filtered out the viral jokes.

When we compare the hourly and daily data, we see that the general data behavior is the same. The hourly graph is much more vertically compressed because the counts are distributed over the 24 hours of each day which makes the data very volatile in the smaller counts area. Still we see the events spikes in the hourly data are well separated from the rest meaning that the activity was concentrated over some hours. In general we will see that the hourly graphs are harder to analyze than the daily graphs because they are much more crowded and concentrated in the low count area.



(a) Daily counts



(b) Hourly counts

Figure 5.1: Plane hijacking related tweet counts

We compute the retweet proportion for the daily data by taking the mean of the daily average and get 40.27% (hourly data has 44.95% but it contains many zero counts which we ignore to compute the mean). The behavior of the count data without retweets is very similar to the count data with retweet. Removing the retweets doesn't change the Twitter measured activity drastically, the spikes are reduced by less than half their maximum count. This is good news for the frequency based detector because ignoring retweets will make its execution faster without creating an issue as long as we adapt the minimum tweet count threshold in consequence.

Classification Evaluation - Correlation with Ground Truth

In order to filter potential noise we apply EWMA to the plane hijacking related tweets classification and correlates the results with the ground truth time series. The graphs from figure 5.2 show this correlation against the variation of the EWMA lambda parameter with $\lambda = 0$ meaning maximum filtering i.e. full dependence on past data and $\lambda = 1$ no filtering i.e. no dependence on past data. The graphs from figure 5.3 show the daily and hourly lag values giving the highest correlation against the variation of the EWMA lambda parameter.

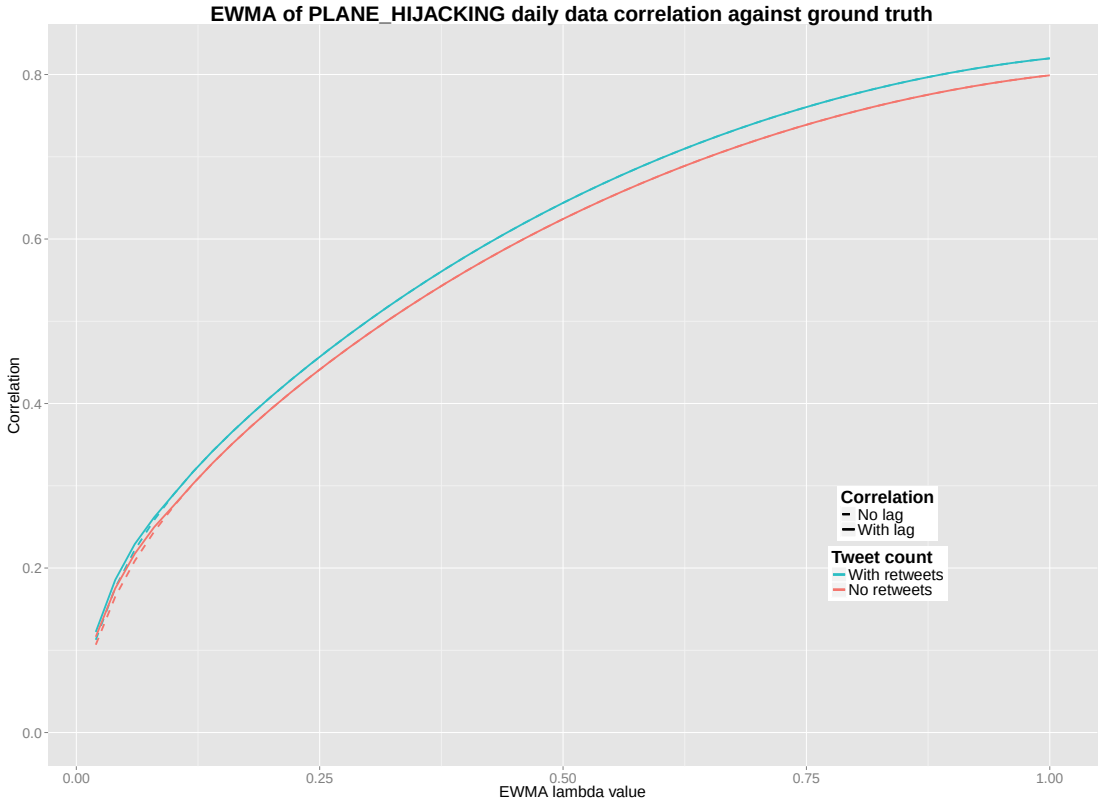
We see that the daily correlation outperforms the hourly correlation probably because the hourly data is more noisy and doesn't react in the very next hours after the event occurs. We have to note that the hourly ground truth time series is very sparse compared to the daily time series and thus the noise has a bigger effect on the correlation which is also a reason why it decreases like this.

We reach a high correlation value of around 0.8 with the daily data which is a good results considering that the classifier created mistakes for several viral jokes. We see that EWMA filtering isn't necessary because we reach the highest correlation if $\lambda = 1$ (i.e. when we don't filter) meaning that the noise doesn't have a strong impact.

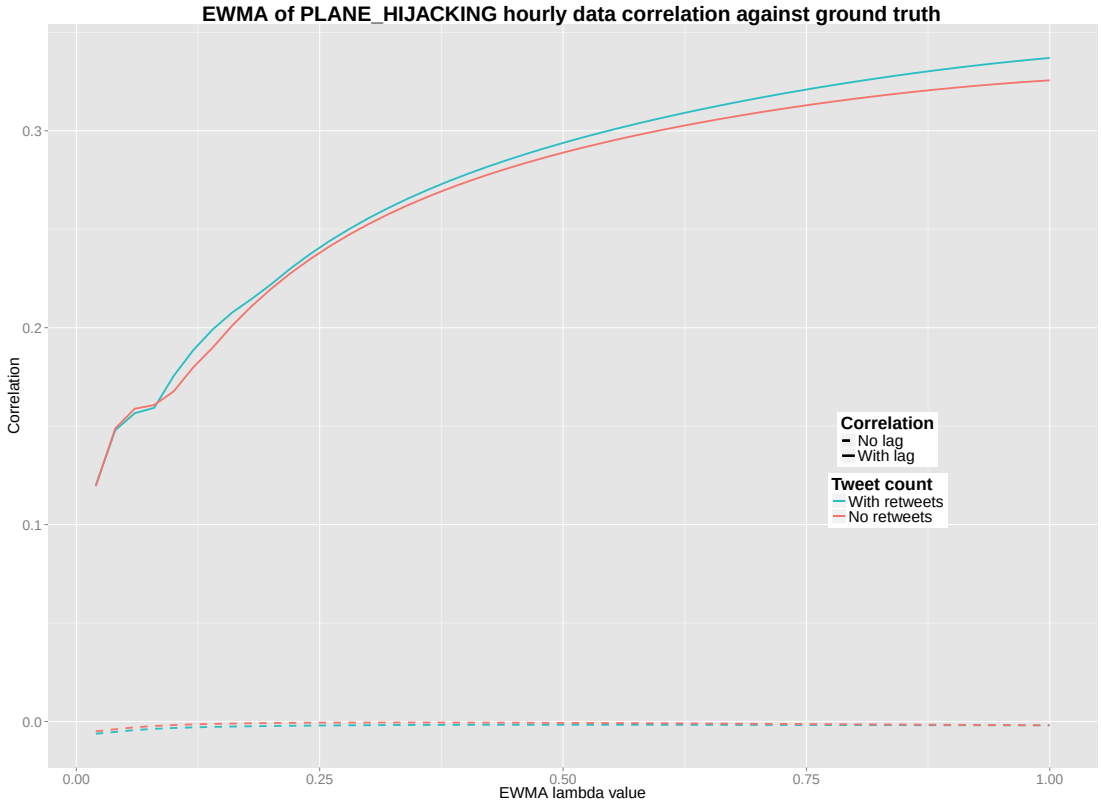
Regarding the lagged correlation we see for the daily data that the highest values are obtained with no lag (zero value) so the dashed and solid curve are overlapping. We can deduce from this that the Twitter data reacts on the same day as when the plane hijacking events occur. For the hourly lag, the correlation is way better when the lag has value 8 suggesting that twitter data reacts usually 8 hours after an event occur.

Removing retweets gives a little worser correlation to the ground truth but there isn't a very big difference. We see also that there is no impact on the lag values which stay the same regardless of whether we include retweets or not meaning the retweets delays don't differ from other tweets.

As we reach a high correlation value we can consider that the classification is doing well for plane hijackings events. In the next section 5.3 we will be able to compare the intuitions given by this analysis with the detectors results.

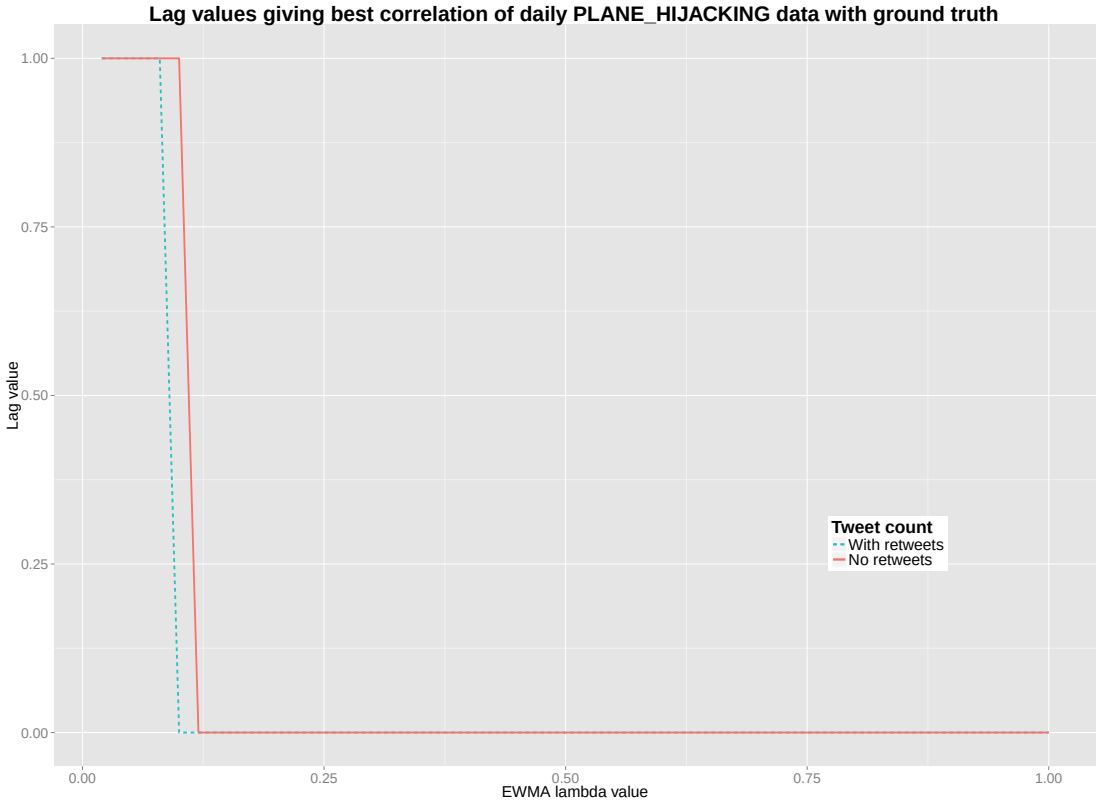


(a) Daily data correlation

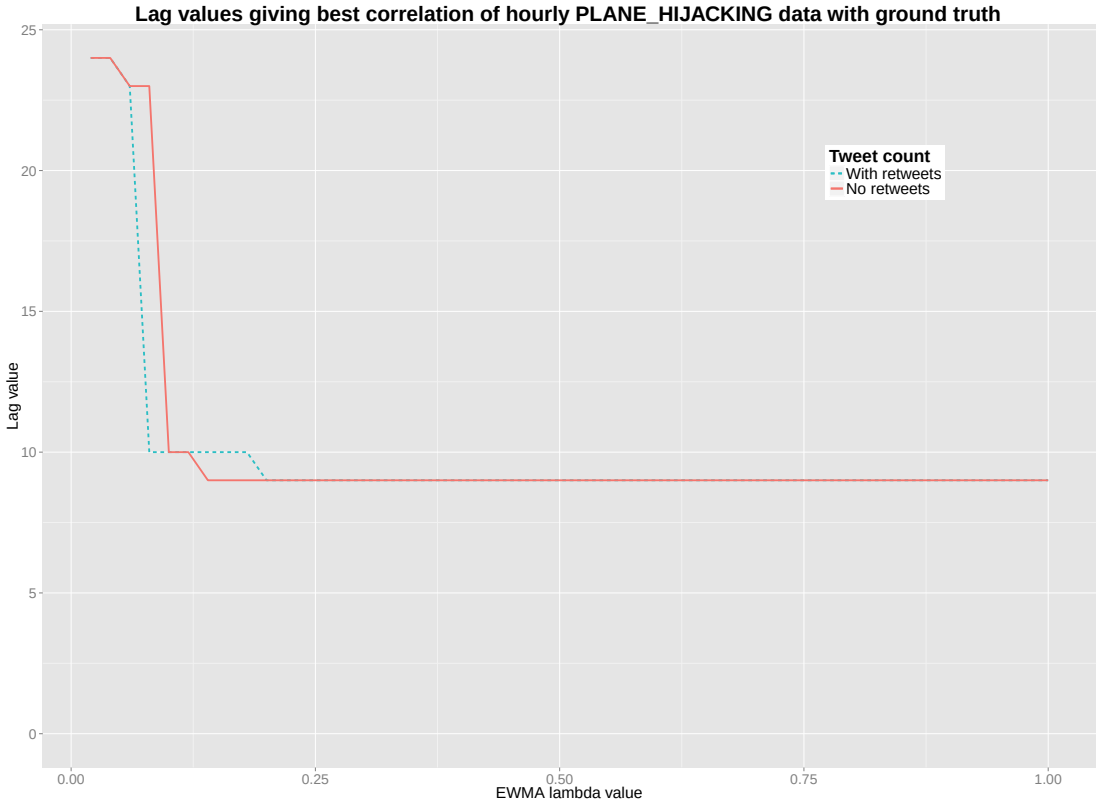


(b) Hourly data correlation

Figure 5.2: Plane hijacking data correlation with ground truth



(a) Daily lag values



(b) Hourly lag values

Figure 5.3: Plane hijacking lag values giving highest correlation with the ground truth

5.1.2 Plane Crashes

Data Visualization

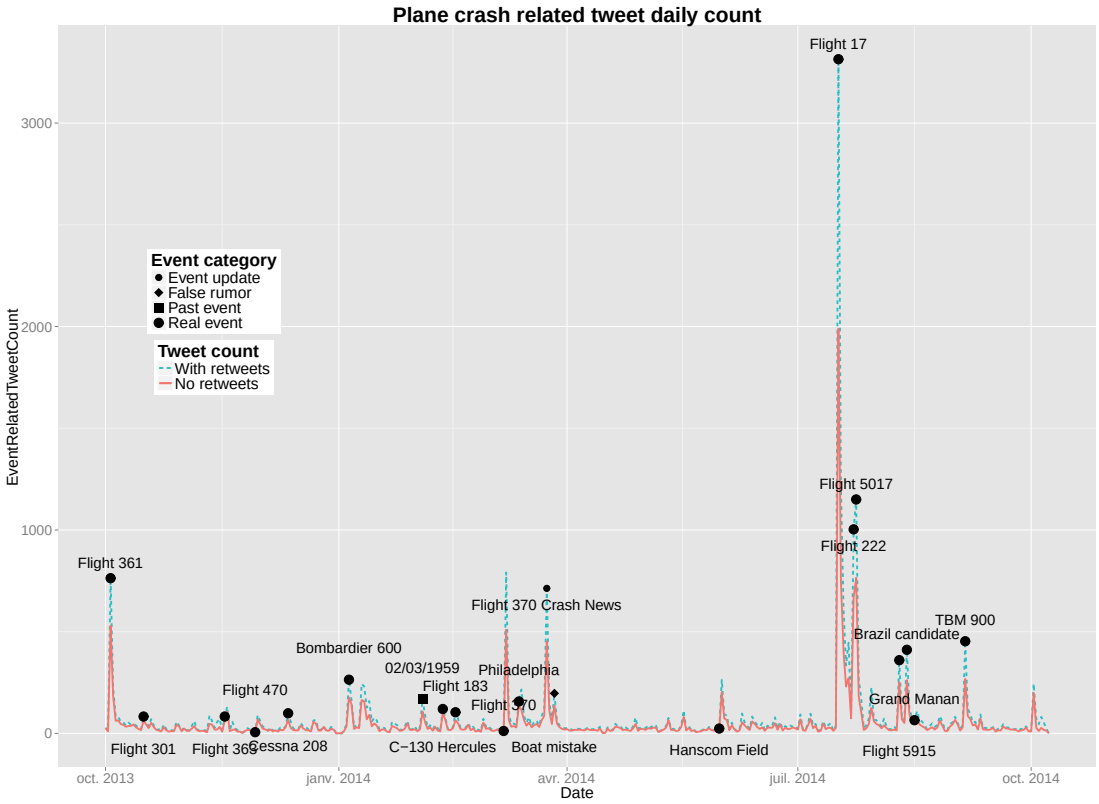
The plane crash related tweets daily and hourly counts are presented in figure 5.4.

The first difference to note is the change of scale from the plane hijackings graphs. There are many more known event occurrences and the Twitter activity is much higher as shown by the relatively high counts. Like with the plane hijackings graph, the activity around an event is short lived and returns to lower values after the spikes which is explained by the fact that plane crashes physically happen over a relatively short period of time. The few events with small response from Twitter data are small scale crashes involving private jets or few passengers.

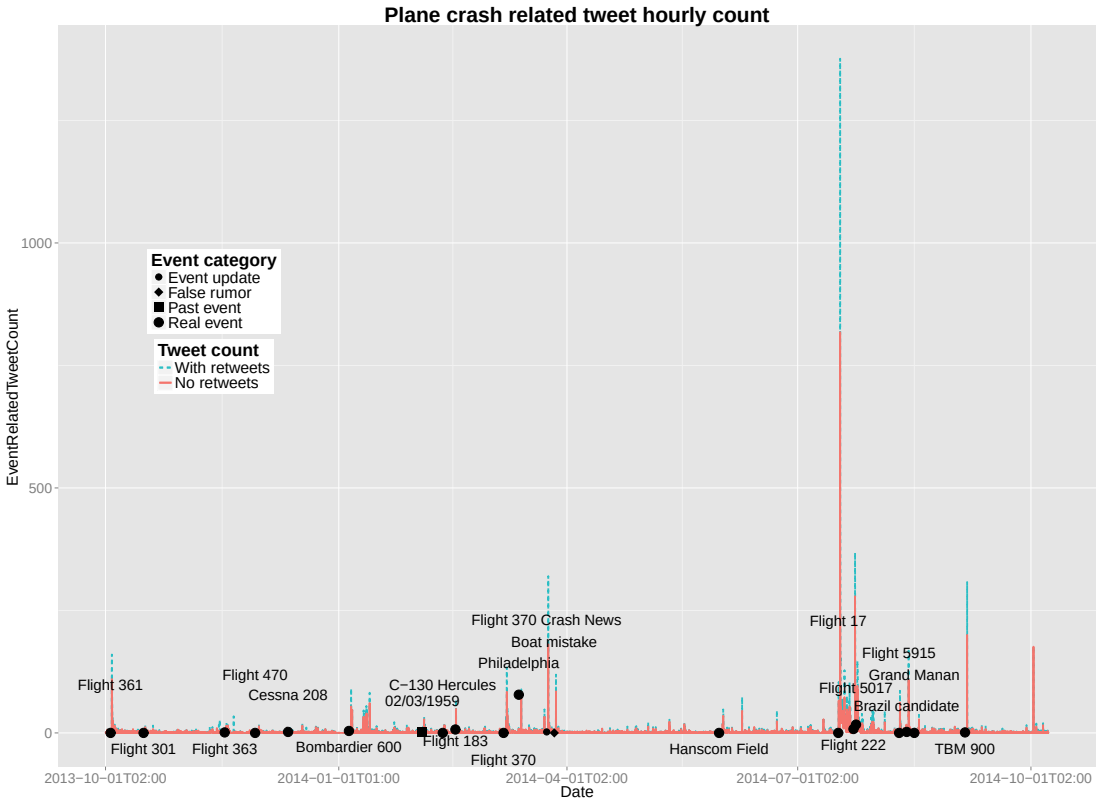
The graph shows that the classifier is working properly because most of the events are represented by a spike and the activity when no crash is occurring is low. We found very little viral jokes activity compared to legit plane crash activity in the data and as we will see in the detector results, few false positive detections meaning that the classifier works well regarding this aspect even better than for plane hijackings.

We can make similar remarks when comparing plane crash hourly and daily data than for plane hijackings data, the graph is very much compressed vertically and noisier in the low counts area. We see that many smaller scale events related spikes are very small meaning that the Twitter activity is distributed and not concentrated.

The retweet proportion is 26.58% for the daily data (hourly data has 25.72%) which is lower than for plane hijackings confirming the intuition given by the graph. We see that removing them doesn't make the spikes disappear and won't change the data general behavior so again it is a good idea to ignore the retweets during the analysis.



(a) Daily counts



(b) Hourly counts

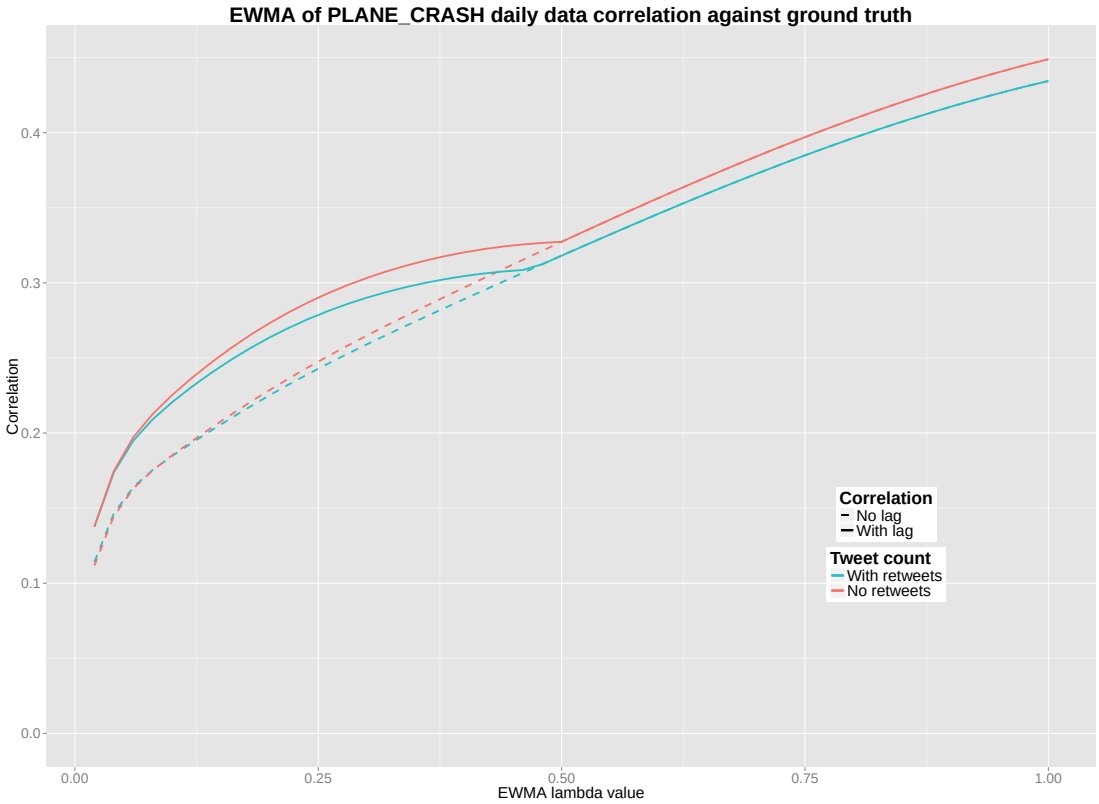
Figure 5.4: Plane crash related tweet counts

Classification Evaluation - Correlation with Ground Truth

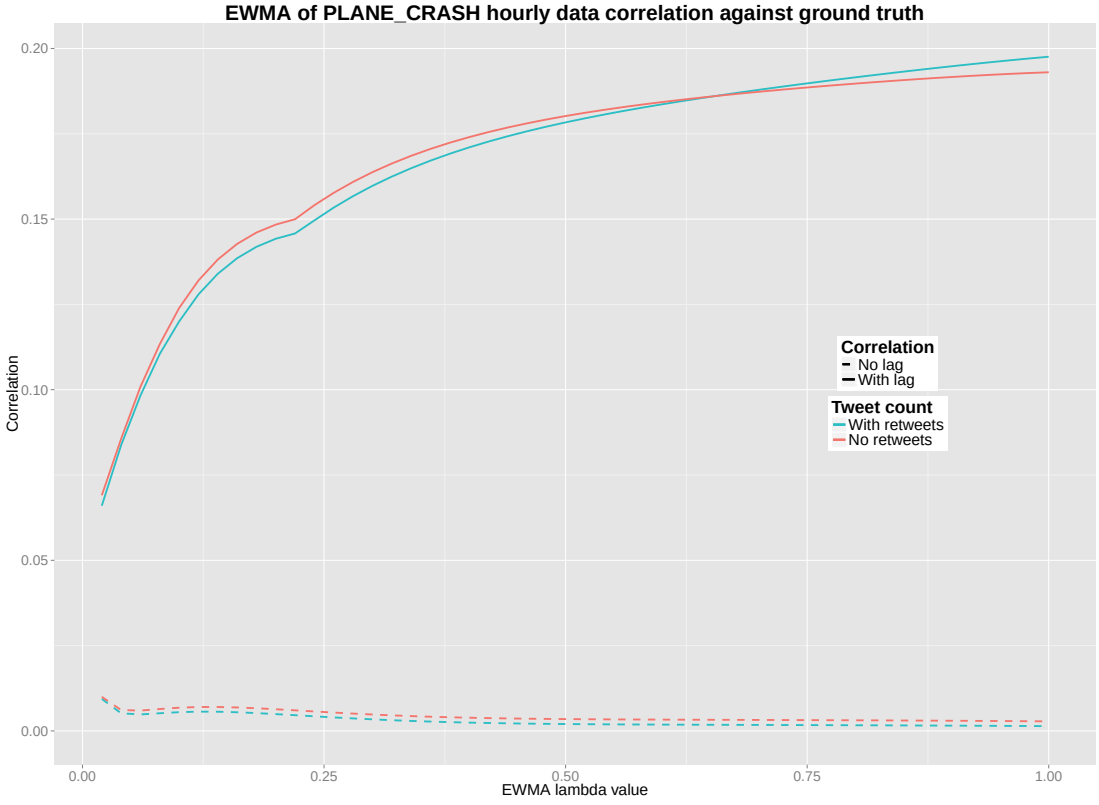
We present the results from the same analysis as in section 5.1.1. We show in figure 5.5 the graphs of the correlation between the daily/hourly plane crash data with the ground truth time series and in figure 5.6, the lag values giving the highest correlation scores.

For the same reasons as with the plane hijacking data, the daily correlation outperforms the hourly correlation.
The top correlation is again reached when we don't use EWMA filtering ($\lambda = 1$) and peaks around 0.45 which is worse than with plane hijacking. One reason is that there are much more known events occurrences which creates a noisier activity in general. It could be a possibility that the classifier simply performs worser as we use very common keywords but we also have to keep in mind that the ground truth is not complete and still misses plane crashes which are reflected in the Twitter data.

Regarding the obtained correlation, we see that removing retweets is beneficial and increases it slightly meaning that retweets add more noise to the signal. Retweets don't seem to have an effect on the lag values which means that they happen after a similar delay like the other tweets.
We remark that for daily data, when we apply a strong EWMA filter ($\lambda < 0.5$) i.e. when we strongly smoothen the peaks, the lagged correlation gives somewhat better results (with lag value 1 i.e. one days after a peak) otherwise it doesn't make a difference. This wasn't the case with plane hijackings which means that for a plane hijacking event, the activity decays faster with time than for a crash event. But the strongest correlation is when no filtering happens with no lag which implies that the Twitter plane crash data reacts also on the same day as when the events occur.
Without surprise, the lagged correlation is better for hourly data and peaks when the lag value is 18 suggesting that twitter data reacts usually 18 hours after an event occur so slower than with plane hijacking events which had a lag value of 8.

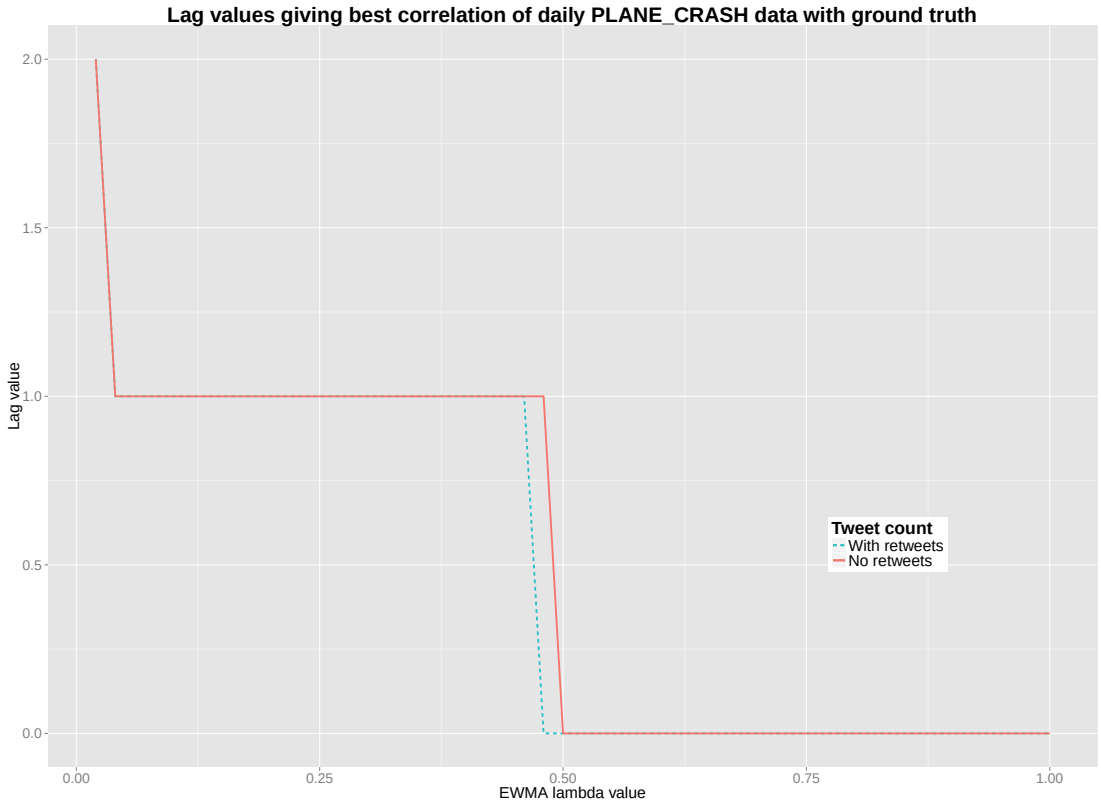


(a) Daily data correlation

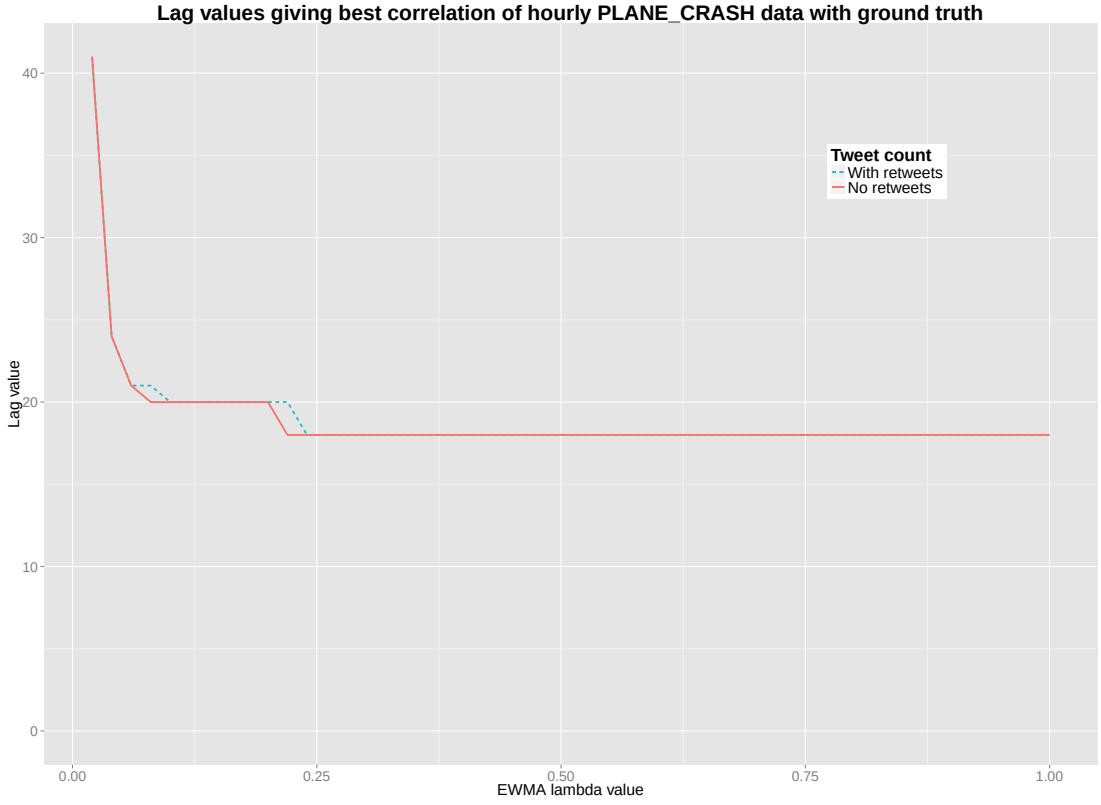


(b) Hourly data correlation

Figure 5.5: Plane crash data correlation with ground truth



(a) Daily lag values



(b) Hourly lag values

Figure 5.6: Plane crash lag values giving highest correlation with the ground truth

5.1.3 Wildfires

Data Visualization

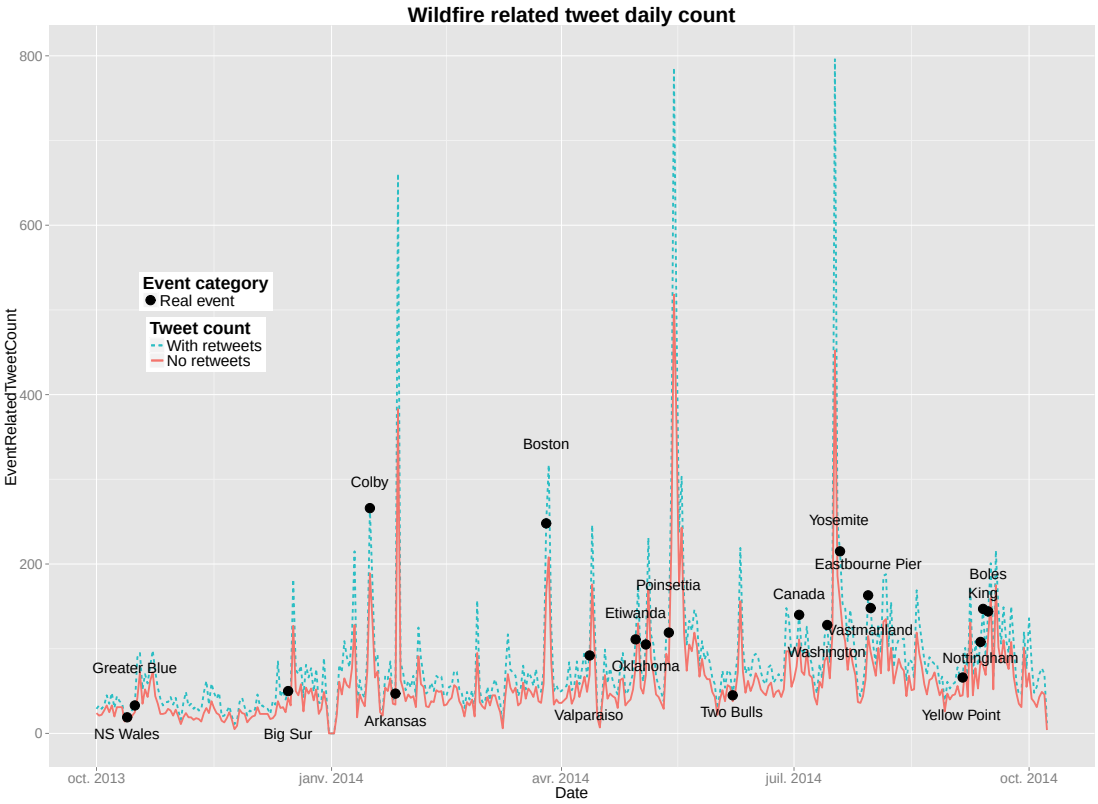
The wildfire related tweets daily and hourly counts are presented in figure 5.7.

The difference with hijackings and crashes is striking, the data is much more volatile and many spikes don't seem to correspond to an event. One part of the reason is that the classifier has more trouble separating wildfires tweets from the rest and we will see that the false positive rate is higher than with the previous event type. But the main problems lie in the wildfire event type nature as we explain in section 2.2.2 Event Types Specific Characteristics. There are many real events occurrences not present in the ground truth list (California only had 5620 wildfires in 2014) and the wildfires last from days to weeks during which the users can report new information and stay active generally. This behavior differ completely from plane crashes and hijackings which was one reason why we decided to also track wildfires. We can note than often the events impact on the Twitter data is more delayed than with hijackings and crashes. The reason for this is that the wildfire aren't always detected right at their start which is often determined exactly only later when the starting date, time and cause are deduced.

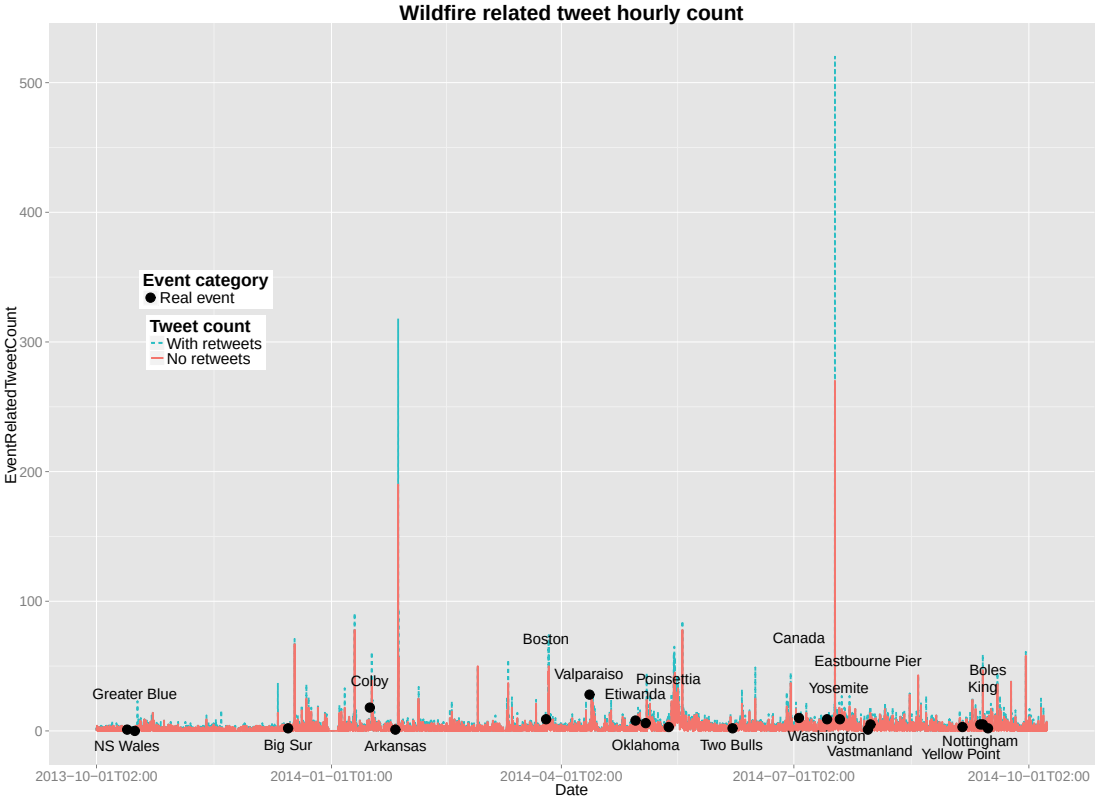
As for the previous event types, the hourly data is very volatile in the low count area. There are many activity spikes seemingly unrelated to ground truth known events like we observed in the daily graph. This continuous noise shows there are always some users talking about wildfires meaning that they are very frequent.

The retweet proportion is 28.48% for the daily data (hourly data has 27.18%). We see that the curves with and without retweets are very similar, only shifted down so removing the retweets don't change the data behavior and can be beneficial for detection.

Note: the zero counts area in January 2014 are due to the down time of our tweet recording system creating this "hole" in the data.



(a) Daily counts



(b) Hourly counts

Figure 5.7: Wildfire related tweet counts

Classification Evaluation - Correlation with Ground Truth

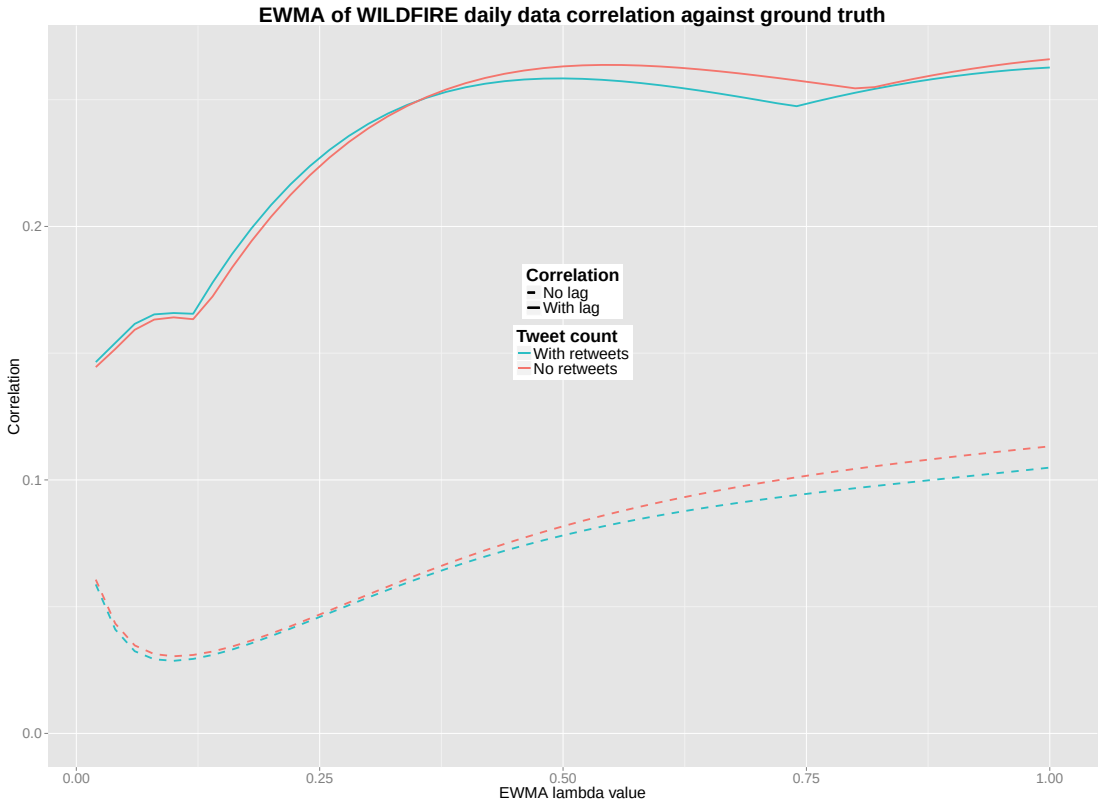
We present the results from the same analysis as in section 5.1.1. We show in figure 5.5 the graphs of the correlation between the daily/hourly wildfire data with the ground truth time series and in figure 5.6, the lag values giving the highest correlation scores.

As with the other two event types and for the same reasons, the daily correlation outperforms the hourly correlation.
For the hourly data correlation we see that the top correlation is reached when we don't use EWMA filtering ($\lambda = 1$) but it isn't as clear for the daily data. The correlation peaks two times at around 0.27 when $\lambda \approx 0.5$ and $\lambda = 1$ which shows that some filtering on the data can improve its similarity to the ground truth. The correlation is much lower than with plane hijackings and crashes but again, we should keep in mind that the ground truth is far from complete, there are many more events occurrences which makes the data volatile. The classifier also performs worser as we see in the graphs from figure 5.7 which are very noisy.

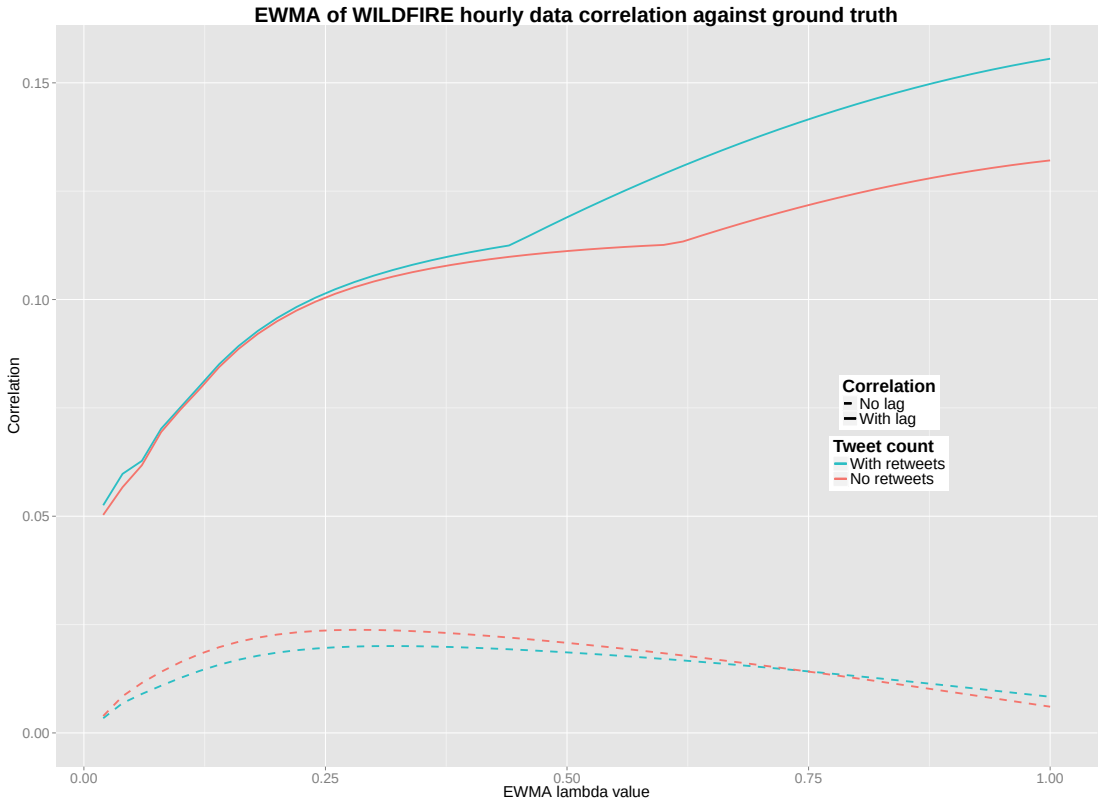
Removing retweets improves a little the daily correlation but decreases the hourly correlation meaning that retweets aren't fully noise and belong partly to the event related Twitter activity.

The lagged correlation improves both for hourly and daily data. The strongest correlation are obtained with lag value of 3 for the daily data and 74 for the hourly data which corresponds to a delay of around three days so users are usually talking about a wildfire three days after it started.

The obtained correlations are the lowest we obtained so far but we know that the wildfire event type has a behavior which differs a lot from plane hijackings and crashes so there is room for improving the classifier using wildfires specific characteristics.

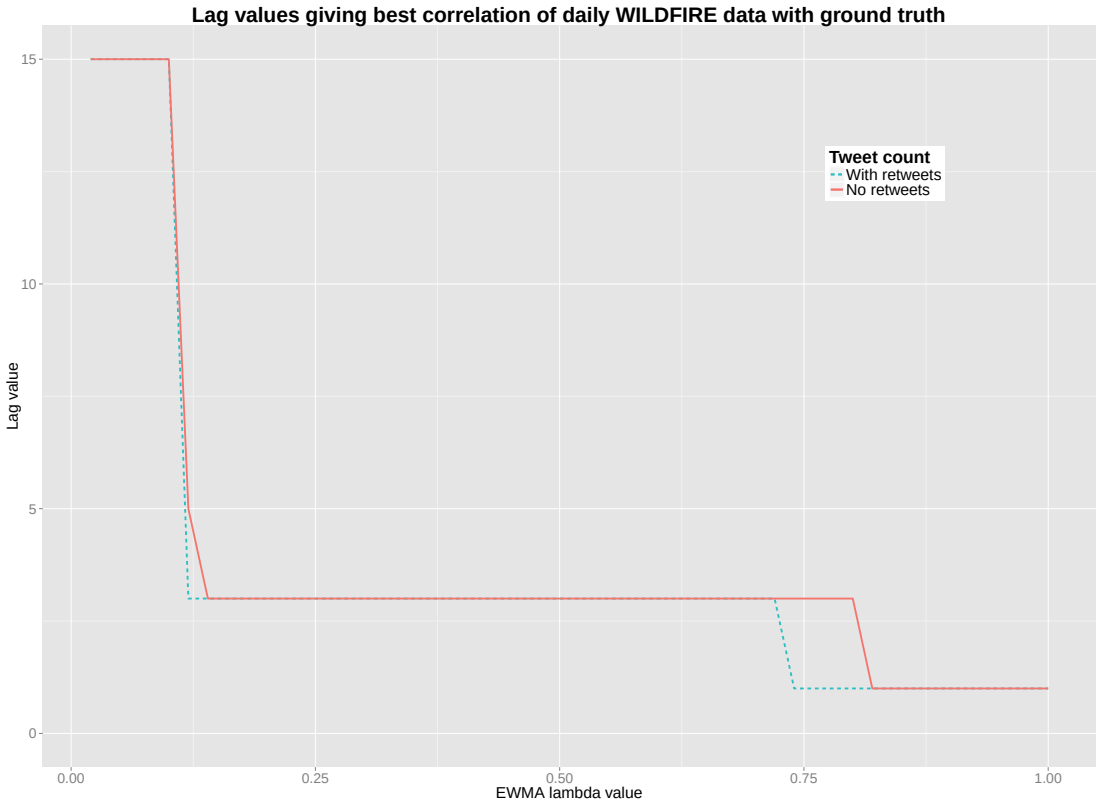


(a) Daily data correlation

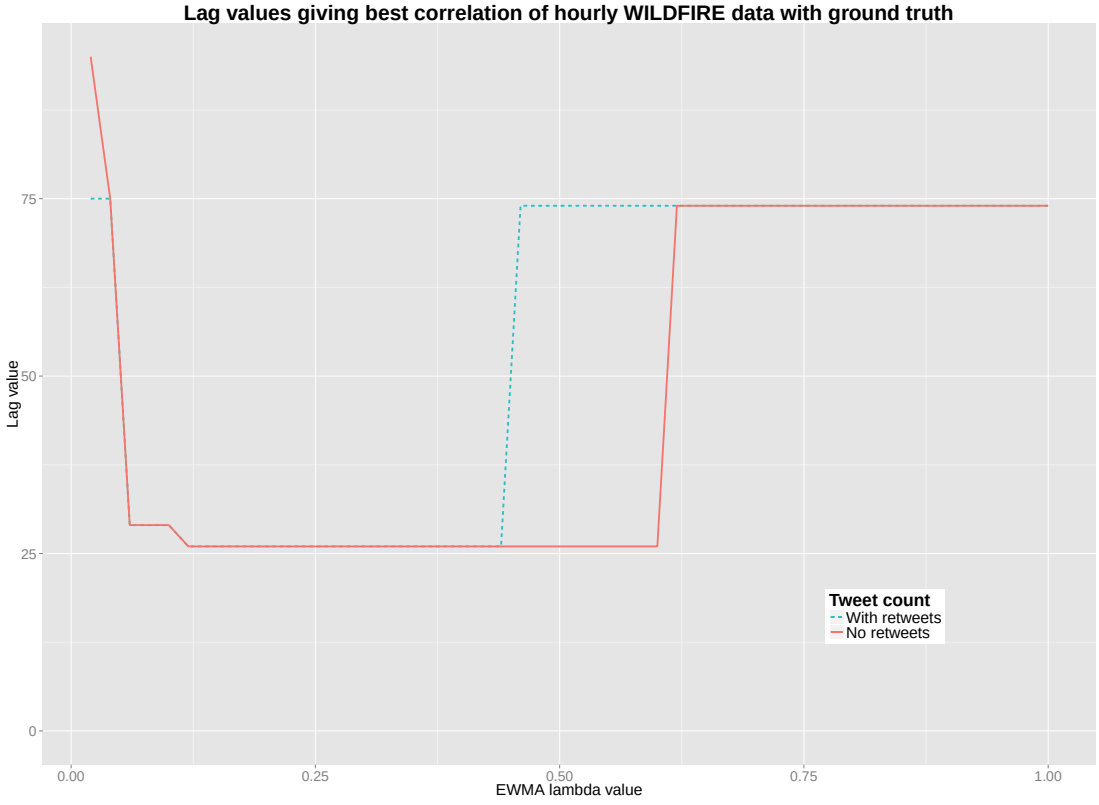


(b) Hourly data correlation

Figure 5.8: Wildfire data correlation with ground truth



(a) Daily lag values



(b) Hourly lag values

Figure 5.9: Wildfire lag values giving highest correlation with the ground truth

5.2 Ignoring Retweets

We decided to ignore retweets in our analysis of the detectors algorithms because of the following arguments:

- We have seen in section 5.1 Classification that removing retweets didn't have a strong impact on the correlation with the ground truth meaning that it shouldn't create a problem.
- We were able to do more experiments as they were running faster with less data to analyze.
- We observed that there were less duplicate detections raised by the detectors which implies less unrecognized detection thus reducing the manual work required after each experiments.
- The number of false positives was also reduced giving the algorithms a higher precision. This behavior confirmed the intuition we had after finding that event related tweets had usually a smaller retweets count than false positive tweets as shown in figure 5.10.

The graph 5.10 was obtained from the retweet count data of all the tweets related to detected viral jokes and a set of randomly chosen event related tweets from the detector's alerts. We see that both boxplot median are at 0 but the third quartile of the viral joke tweets is higher by almost 500 retweets than the third quartile of the events tweets which means that up to 25% of the viral jokes tweets have a higher retweet count than the event tweets.

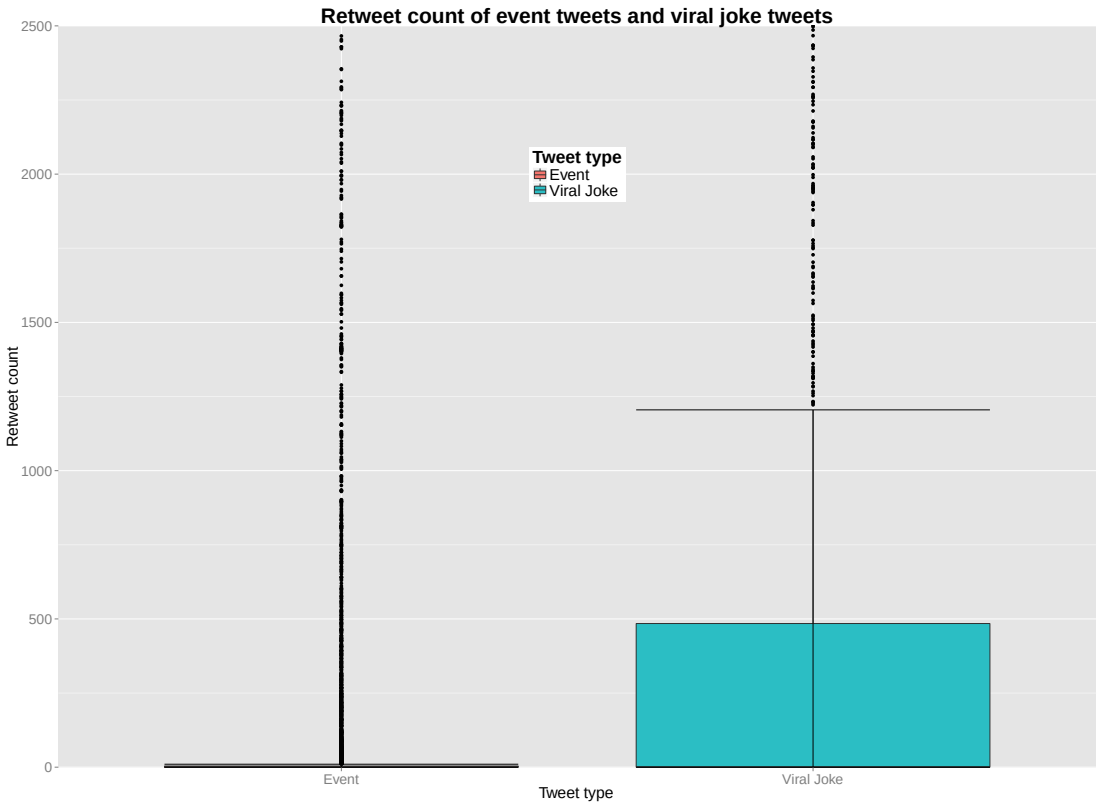


Figure 5.10: Retweets count of event tweets and viral joke tweets

5.3 Event Detectors performance evaluation

In this section we present the results of the two detector modules individually and then compare their performance with traditional medias. As we explained in section 4.5, for each detector and for each event type, we run experiments with different parameters and compute the associated performance. The false positive and detection counts used to compute the precision and recall are available in Appendix D. The delays are obtained over the same set of detected events.

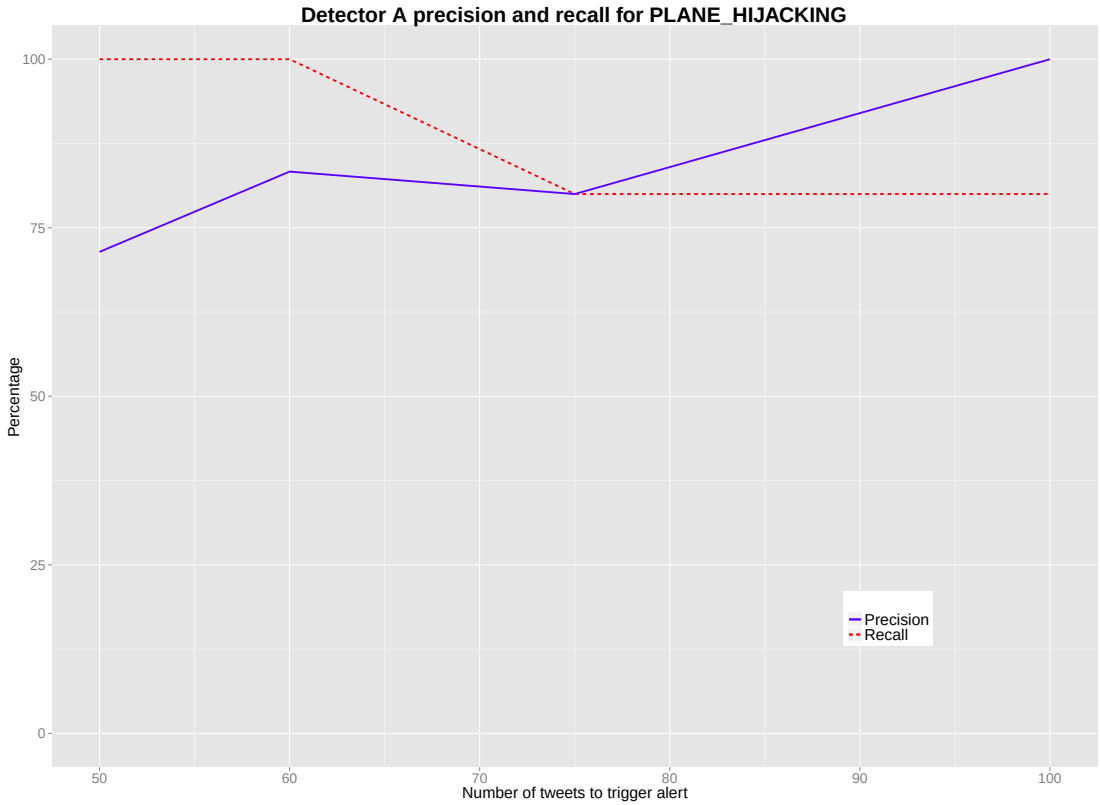
5.3.1 Frequency based Detection

The parameters we focus on are the two thresholds of the detector: the minimum number of tweets in the time window and the ratio of the first half of the time window tweet count by the second half tweet count. The sizes of the time windows are 1 hour, 5 hours and 10 hours.

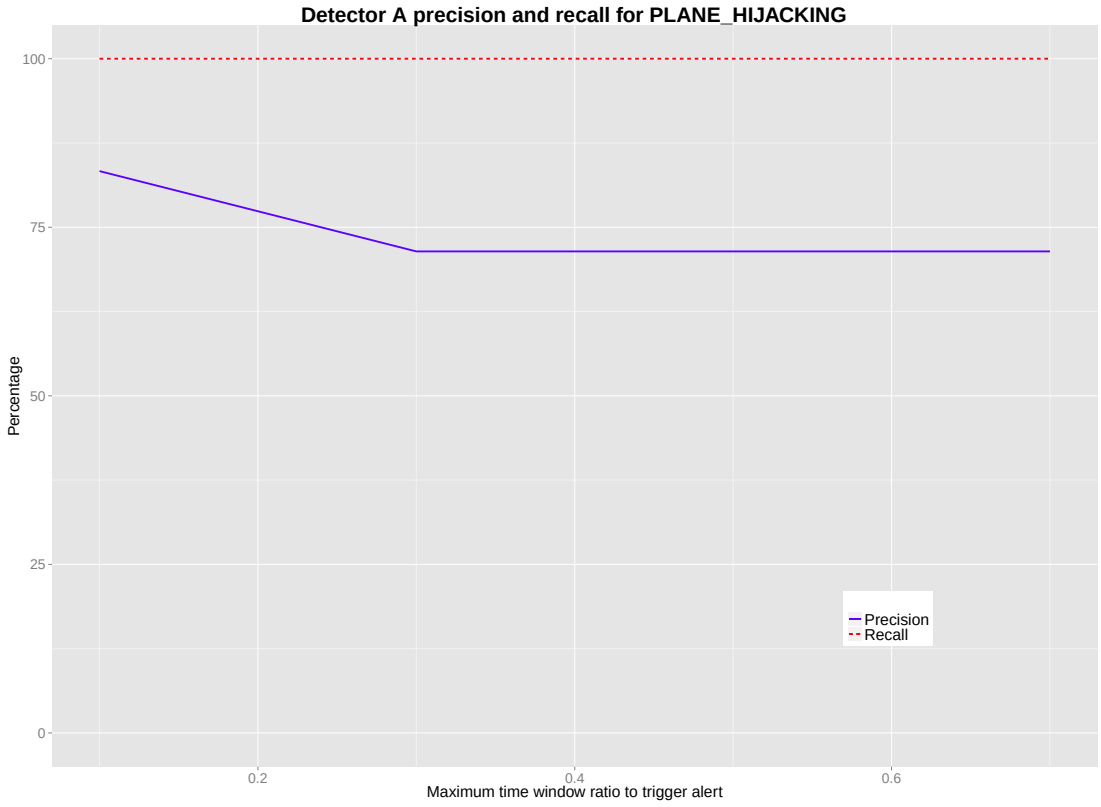
Plane Hijackings

First of all, it is important to recall that the ground truth for plane hijackings contains only 5 events. This small number makes the precision and recall very sensible, for example missing one event would lower the recall to 80%. We start by showing the precision and recall graphs in the figure 5.11.

We see from the graph 5.11a that the more tweets we require to be present in a time window before triggering a detection alert, the better precision we get. This is logical because if there are more tweets related to plane hijackings, it is more probable that a real plane hijacking event occurs. This assumption isn't true in the case of a very popular viral joke (the algorithm detects up to two viral jokes here) but as we ignore retweets their impact is diminished (cf section 5.2). We see that there is a trade-off between precision and recall as the recall diminishes when we increase the minimum tweet count threshold. This is explained by the fact that not all events may be able to simultaneously satisfy the ratio threshold and still produce enough tweets to



(a) Minimum tweet count with fixed ratio of 0.5



(b) Ratio with fixed minimum tweet count of 50

Figure 5.11: Detector A precision and recall for plane hijackings

reach the minimum count.

Regarding the ratio threshold in graph 5.11b, it is better for precision to require a small ratio (e.g. a ratio lower than 0.5 means that we require the tweet count to at least double between the first half of the window and its second half) which is easily explained by the fact that only when an event happens, will the activity be high enough to reach the ratio threshold.
There should be again a trade-off between precision and recall as not all events will generate such an activity burst and being more lenient on the ratio threshold would increase the recall. However this isn't visible for plane hijackings which means that these events are very popular and much discussed on Twitter.
Again popular viral jokes are detected (up to two viral jokes) even with a high ratio but we would detect more if we were not ignoring retweets.

The precision and recall are satisfactory, we are able to detect all plane hijackings and over the full data which has a time span of over one year we have only two false positives which is manageable for a human. Now we present the detection delays in graphs 5.12.

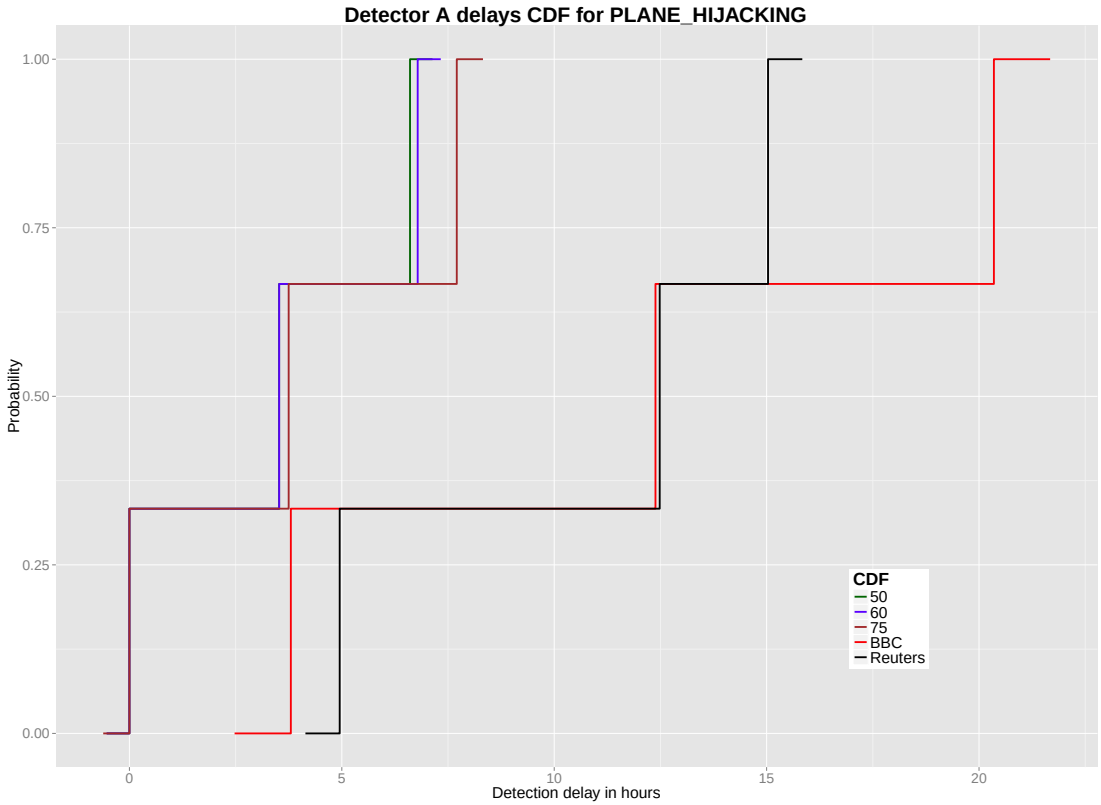
The first thing to note is that graph 5.12a delays are obtained over a common set of 4 plane hijackings (the third curve has a recall of only 80%) and that the graph 5.12b delays are obtained over the full set of 5 plane hijackings (minimum recall of 100%) which explains the difference of the traditional media delay cumulative distribution function.

The top delays are obtained with 50 tweets and 0.5 ratio. It is obvious that setting a lower threshold on the tweet count means that the detector has to wait for less tweets before launching a detection and thus gives shorter delays.
In graph 5.12b we can see only one curve because they all overlap meaning that they have really close detection delays. This behavior is explained because of the nature of the plane hijackings events but first we should understand the following. The ratio threshold is used to trigger a detection in case of a bursty activity meaning that the tweets all arrive in a relatively short amount of time. If we choose a stricter ratio this means that there will be cases where the detector won't trigger an alert because the tweets didn't arrive in a sudden enough burst and will wait for the right moment where such a situation arise. Therefore a stricter ratio should normally incur a longer delay. However we can't observe this with plane hijackings events because we see from the graph 5.1a that their occurrences are very popular and also lead to sudden bursts (the huge spikes) from the very first tweets after an event occurrence which nullifies the delay incurred by stricter ratios. Thus plane hijackings events behavior doesn't allow us to verify this argument.

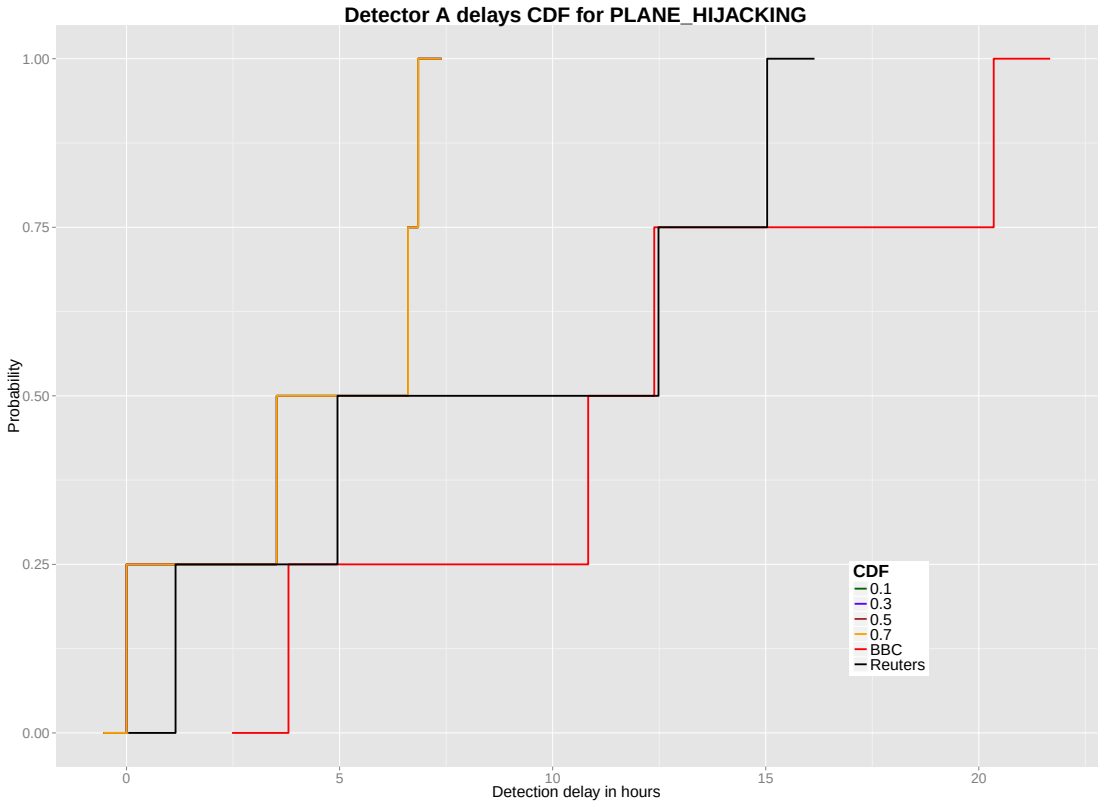
We detect all considered events in 7.5 hours which is really close to the lag value 8 obtained in section 5.1.1. The detector is faster than traditional media as shown in graph 5.13. This information has to be taken with a grain of salt as explained in section 4.4 because the traditional media data consists of the article/report latest update timestamp (the creation data aren't available through their website).

We now show the visualization of the plane hijackings detections in the graph 5.14 with the daily count (hourly data is too crowded) data without retweets using the configuration which gives the best delays and almost the best precision/recall (50 tweets and 0.5 ratio). We precise that in the graph's legend, by *automatically matched*, we mean matched to its corresponding known events from the ground truth in the way described in 4.5. Please note as well that the recognized duplicates detections aren't present in the graph.

The graphs shows that the detections correspond to the spikes and happen on the same day as the known events as their dots overlap. There are almost only one detection per event meaning that detection aggregation wouldn't be really needed in this case. The viral joke 3 remains undetected. The isolated dot corresponds to an unrecognized duplicate detection.



(a) Delays with fixed ratio of 0.5



(b) Delays with fixed minimum tweet count of 50

Figure 5.12: Detector A plane hijacking detection delays

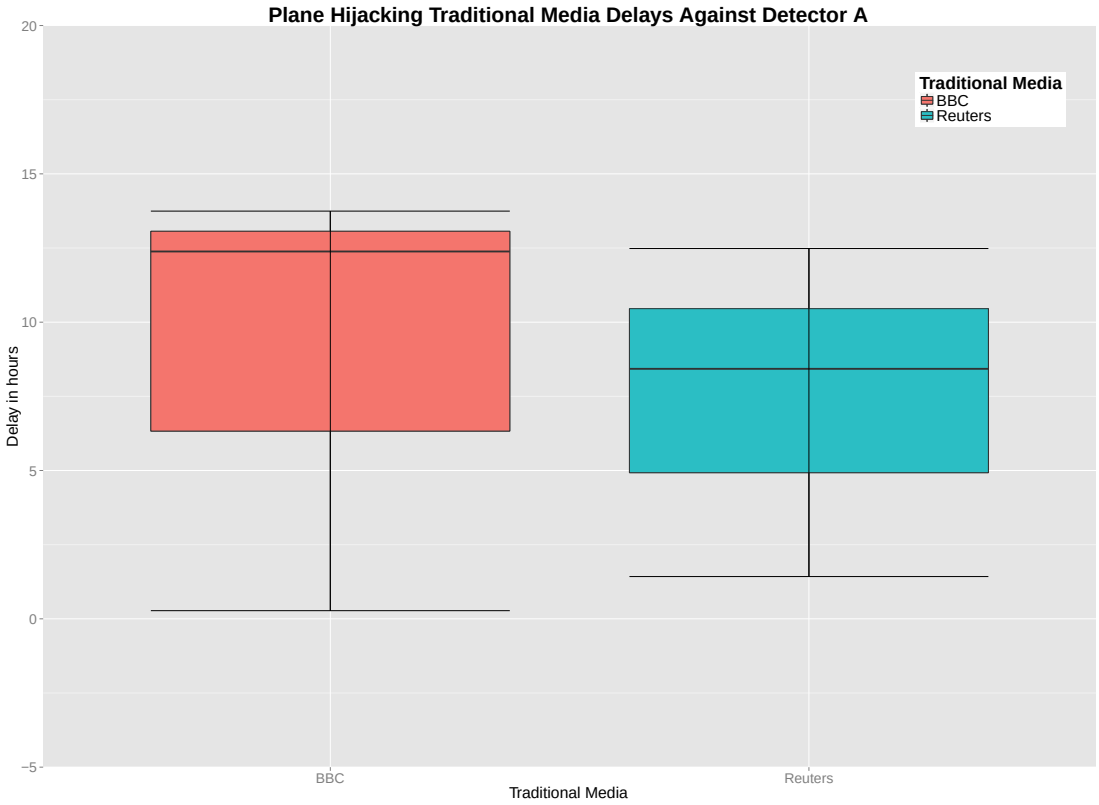


Figure 5.13: Traditional media delays for plane hijackings against detector A (a positive value e.g. 5 implies that the detector is 5 hours faster than the media and the opposite for negative values)

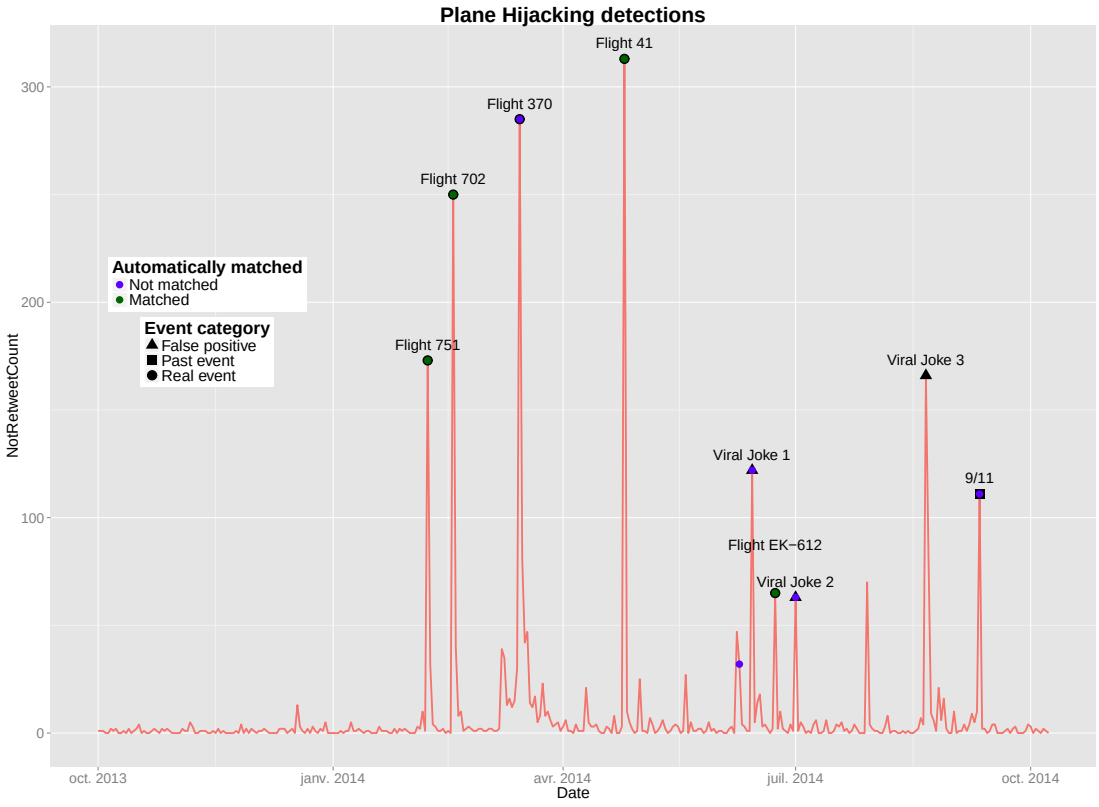


Figure 5.14: Detector A plane hijacking detections

Plane Crashes

The ground truth for plane crashes contains 18 events so the precision and recall will be less sensible than with plane hijackings. The precision and recall graphs are shown in figure 5.15.

We see in graph 5.15a a perfect precision but we think for the same reasons that the behavior would be the same as with plane hijackings events i.e. with low tweet count threshold, there is more chance to get false positives than with a high threshold on the tweet count. The recall goes down fast when we increase the tweet count threshold confirming what we couldn't see with the plane hijackings, i.e. that smaller scale events aren't popular enough to breach a high tweet count threshold. This behavior confirms the trade-off between precision/recall and the tweet count threshold we supposed.

Graph 5.15b is more surprising. The recalls behaves as we expected and increases when we weaken the ratio threshold (which we couldn't confirm with plane hijackings) but the precision doesn't and decreases when strengthening the ratio from 0.3 to 0.1. Like we observed with plane hijackings the precision should have stayed at 100% but there is one false positive detection which doesn't occur when the ratio is 0.3. After careful analysis of the logs we found the following: in both cases the same detection occurs but with ratio 0.1 it contains 100% unrelated tweets making it a false positive whereas with ratio 0.3 it contains a small proportion of valid tweets related to a small undetected plane crash event thus making it a valid detection. The reason behind this difference lies in the fact that the detector chose another time window, a shorter one missing the event tweets, to raise the detection because the longer time window wouldn't breach the ratio threshold but the shorter window full of the viral joke tweets would. So this surprising behavior in the precision curve doesn't infirm our deduction but on the other hand highlights the difficulty of classifying the detections as valid or not. We chose the point of view of a human reading through such alerts which would then ignore the garbage tweets but get informed by the small valid tweets proportion of the event occurrence.

Overall the precision is really good meaning that the classifier does a proper job and also that they were few viral jokes involving plane crashes over the last year. The recall is fine as long as we choose a small tweet count threshold. We now show the detection delays in graphs 5.16.

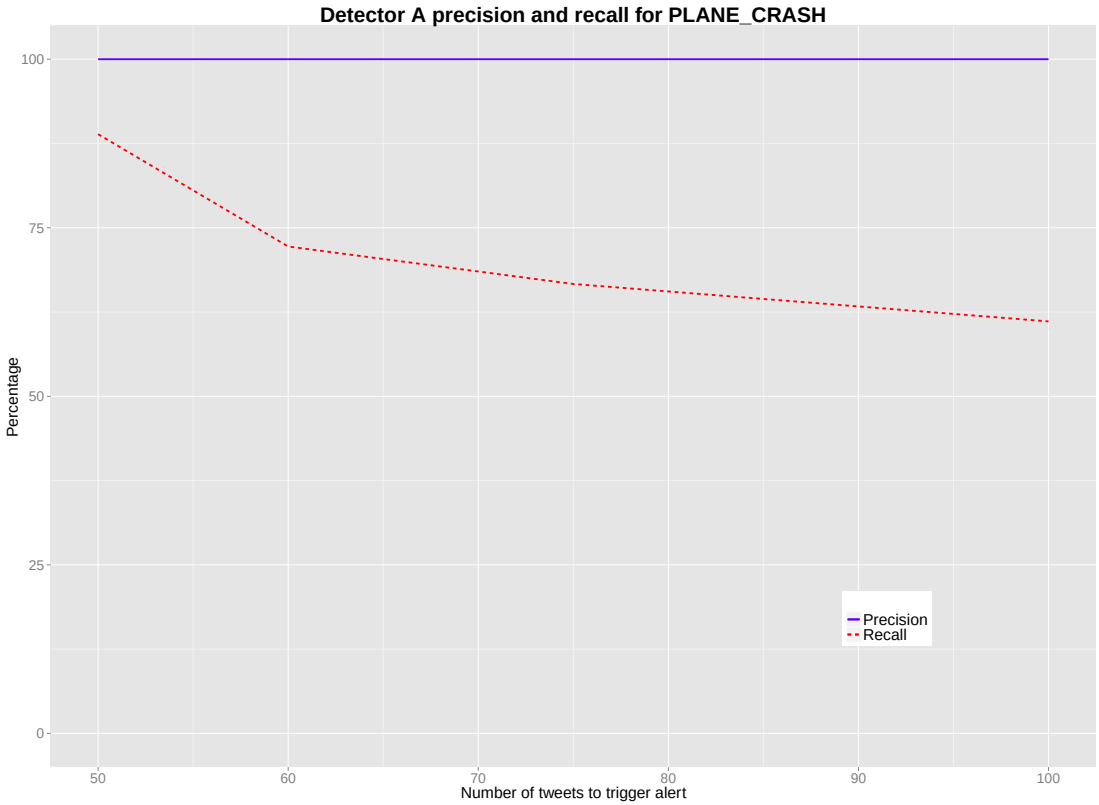
The delays are obtained over a common set of around 60% for graph 5.16a and around 70% for graph 5.16b of the ground truth.

Again and for the same reasons as with plane hijackings, the best value for the tweet count threshold is the smallest i.e. 50 tweets. For the ratio threshold, we see that the shortest delays are obtained with a ratio of 0.7. This also confirms our argument from plane hijackings a stricter ratio threshold increases the detection delays.

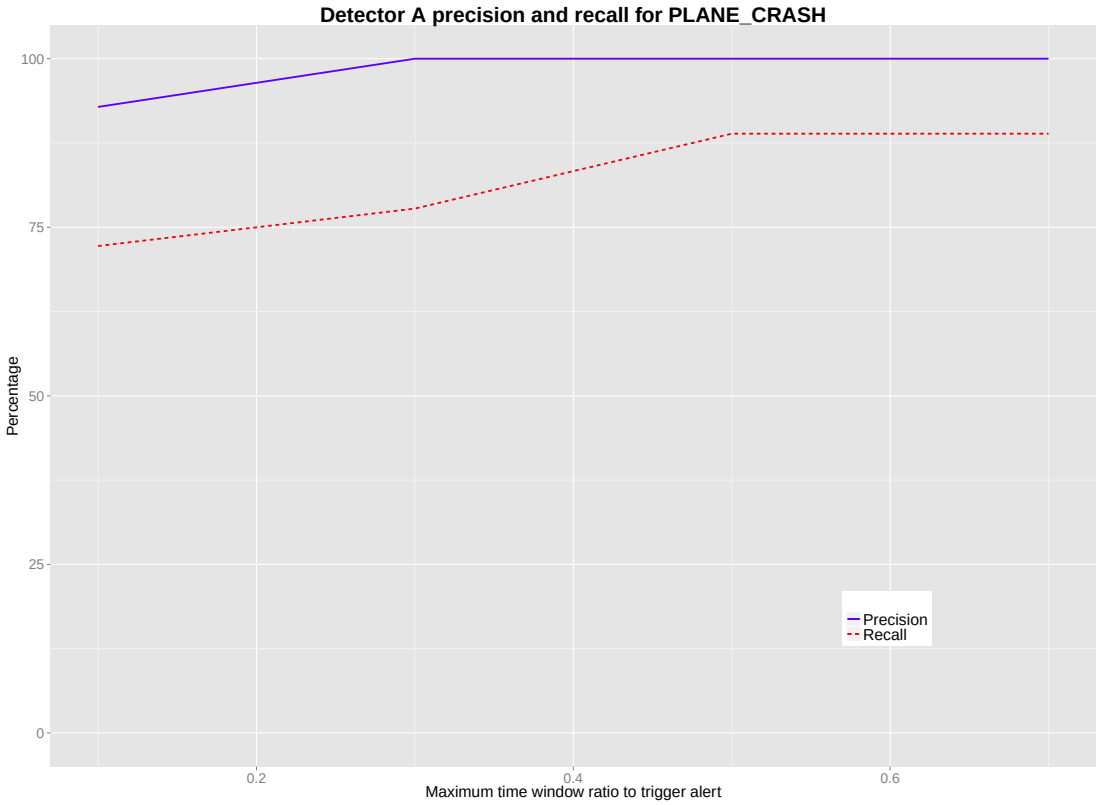
We detect all considered events in around 25 hours which is higher than the lag value 18 obtained in section 5.1.2 but we still detect around 80% in 18 hours. The detector is most of the time faster than BBC but slower than Reuters as shown by graph 5.17.

We show the visualization of the plane crashes detections in the graph 5.18 with the daily count data (hourly data is too crowded) without retweets using the configuration which gives almost the best delays and the best precision/recall (50 tweets and 0.5 ratio). Please note that the recognized duplicates detections aren't present in the graph.

Some detections don't seem to correspond to an annotated spike from the ground truth; in fact they match to smaller scale events which were not added to the ground truth. All the large spikes were recognized as events by the detector. For the events Flight 222 and Flight 5017 one of the two detections is delayed because of the time limit of 24 hours between which the detector doesn't allow any other detections. Overall the detector works well for plane crashes, it detects most of the events and is on par with the traditional media regarding the detection delays.

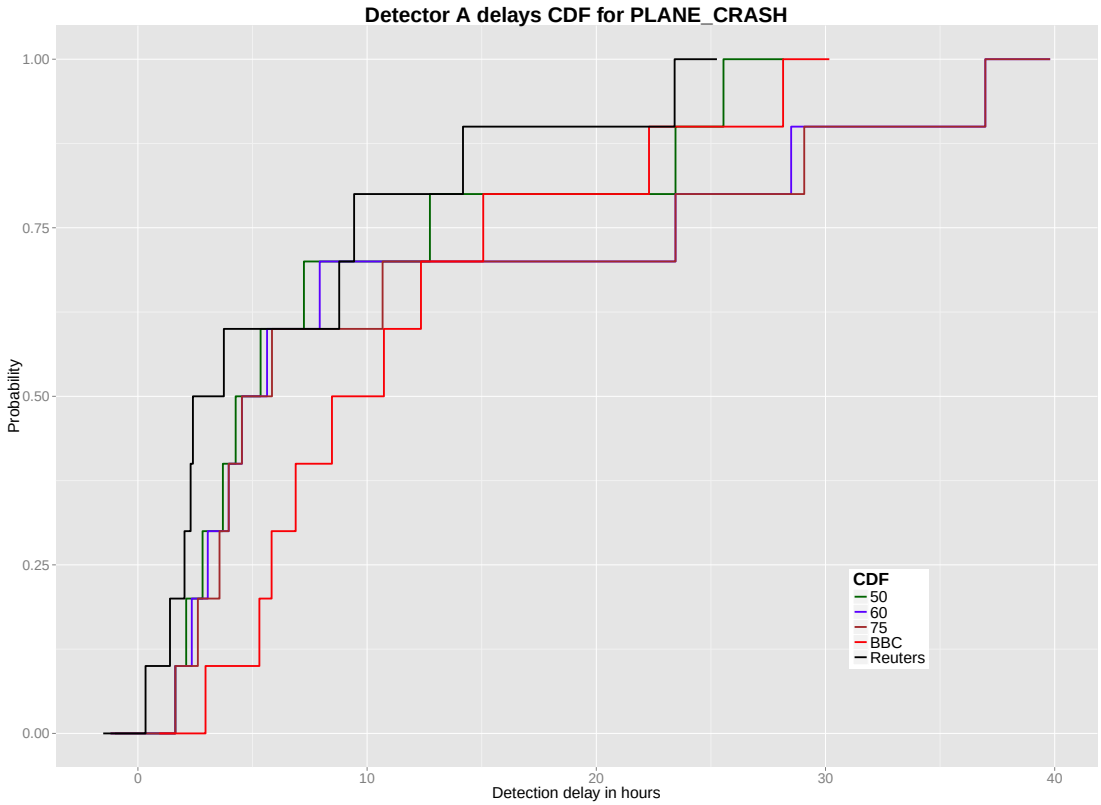


(a) Minimum tweet count with fixed ratio of 0.5

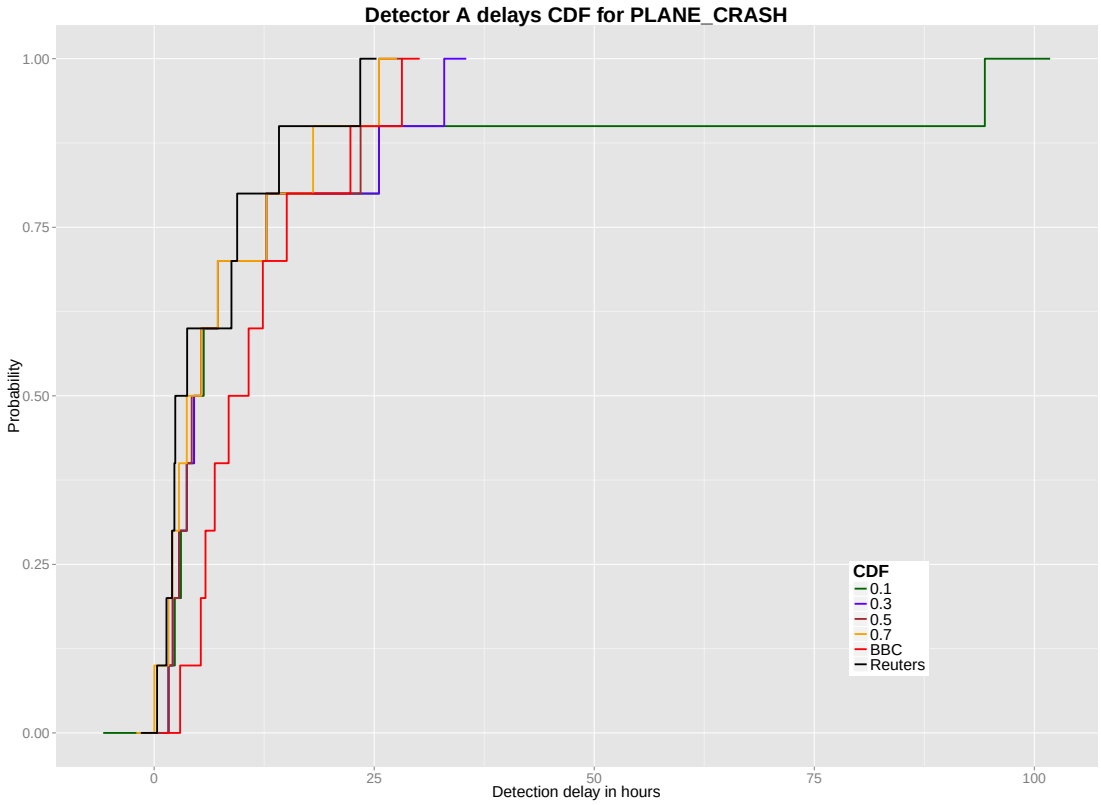


(b) Ratio with fixed minimum tweet count of 50

Figure 5.15: Detector A precision and recall for plane crashes



(a) Delays with fixed ratio of 0.5



(b) Delays with fixed minimum tweet count of 50

Figure 5.16: Detector A plane crash detection delays

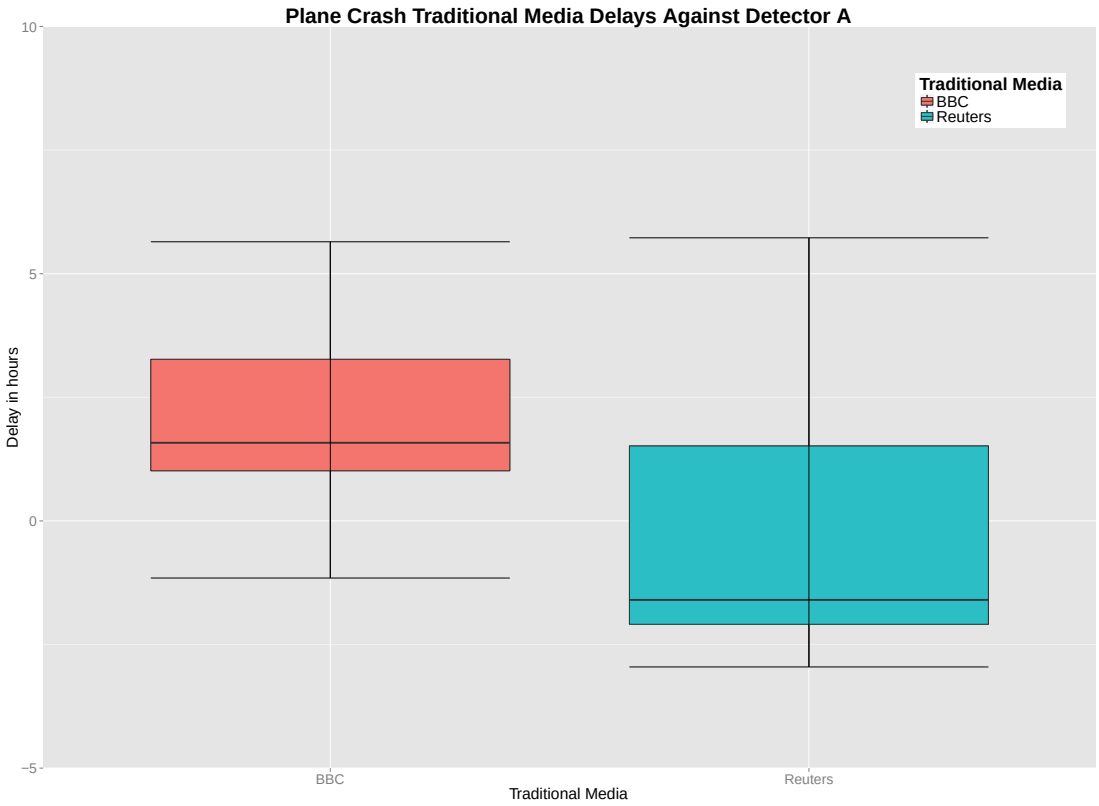


Figure 5.17: Traditional media delays for plane crashes against detector A (a positive value e.g. 5 implies that the detector is 5 hours faster than the media and the opposite for negative values)

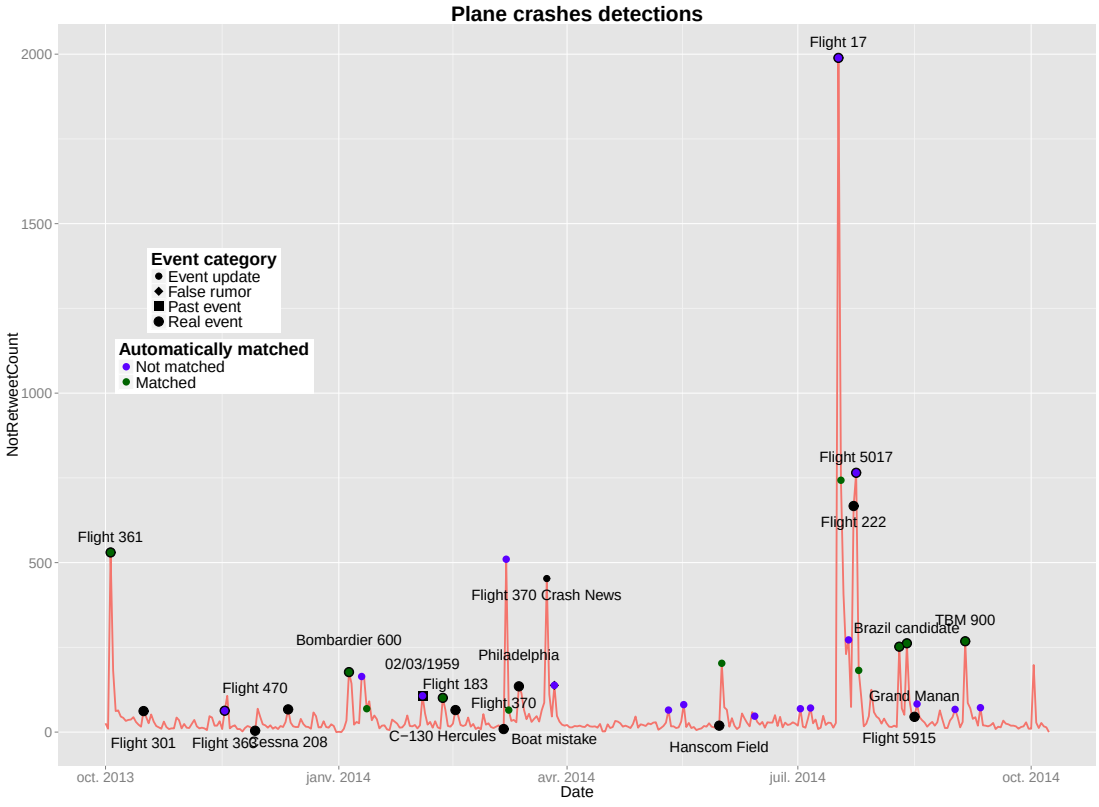


Figure 5.18: Detector A plane crash detections

Wildfires

The ground truth for wildfire contains 20 events, we use it as the basis to compute precision and recall. The precision and recall graphs are shown in figure 5.19.

The detectors performs worse than with the previous event types. This is due as we explained in section 2.2.2 to the wildfire event type characteristics. We see in graph 5.19a a confirmation of the hypothesis that recall decreases when we increase the tweet count threshold for the same reasons that we stated before. However the behavior of the precision seems to differ from what we observed with plane hijackings and plane crashes, it doesn't follow a trend and switches between decreasing and increasing. The cause for this is the decreasing recall. To understand what happens we need to check the false positive and detections count from Appendix D. We see that the number of false positive keeps decreasing and never increases which corresponds to the behavior we observed with the previous event types and expected i.e. less positive with an increasing tweet count threshold. The precision curve doesn't reflect this behavior because of the number of detections which also decreases.

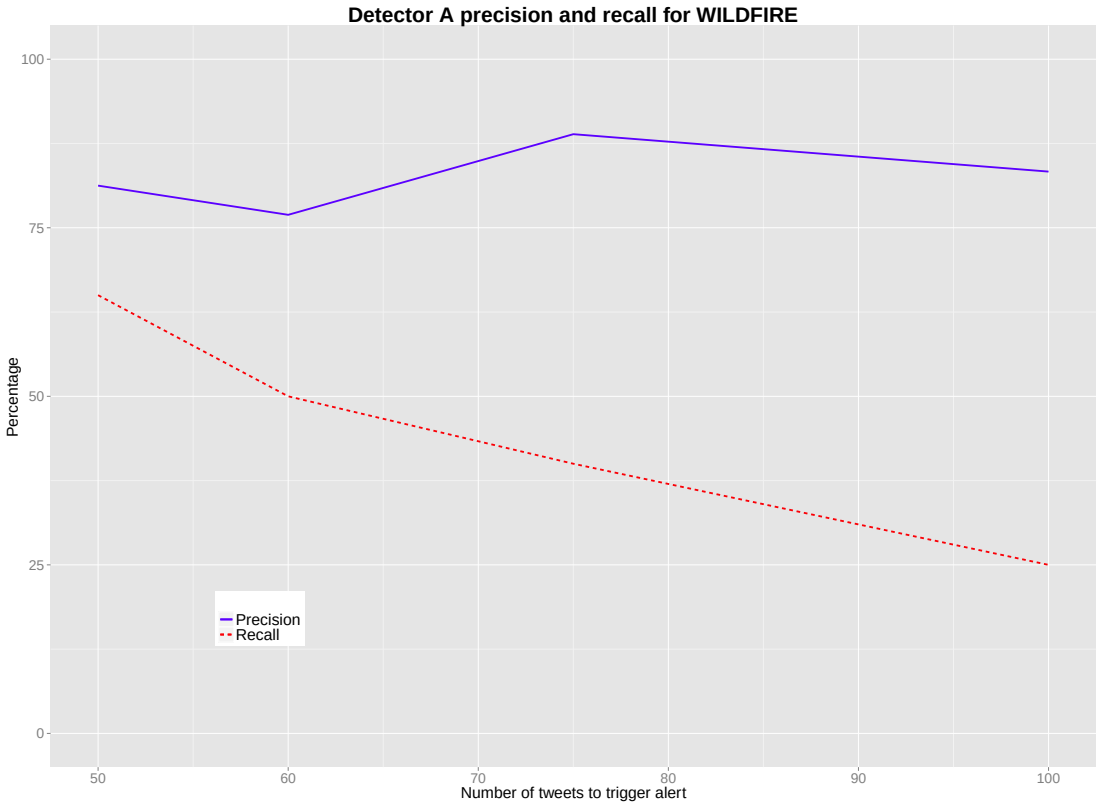
Graph 5.19b also confirms that the recall increases when we weaken the ratio threshold for the reasons we already explained. Again the precision seems to differ to what we expected but after checking Appendix D we see that the increase in precision is due only to the increase of the number of detected events and that the false positive count stays the same. So the behavior of the precision is what we expected and decreases when we weaken the ratio threshold.

Overall the precision stays at an acceptable level but the recall performance really depends on choice of the parameters. We now show the detection delays in graphs 5.20.

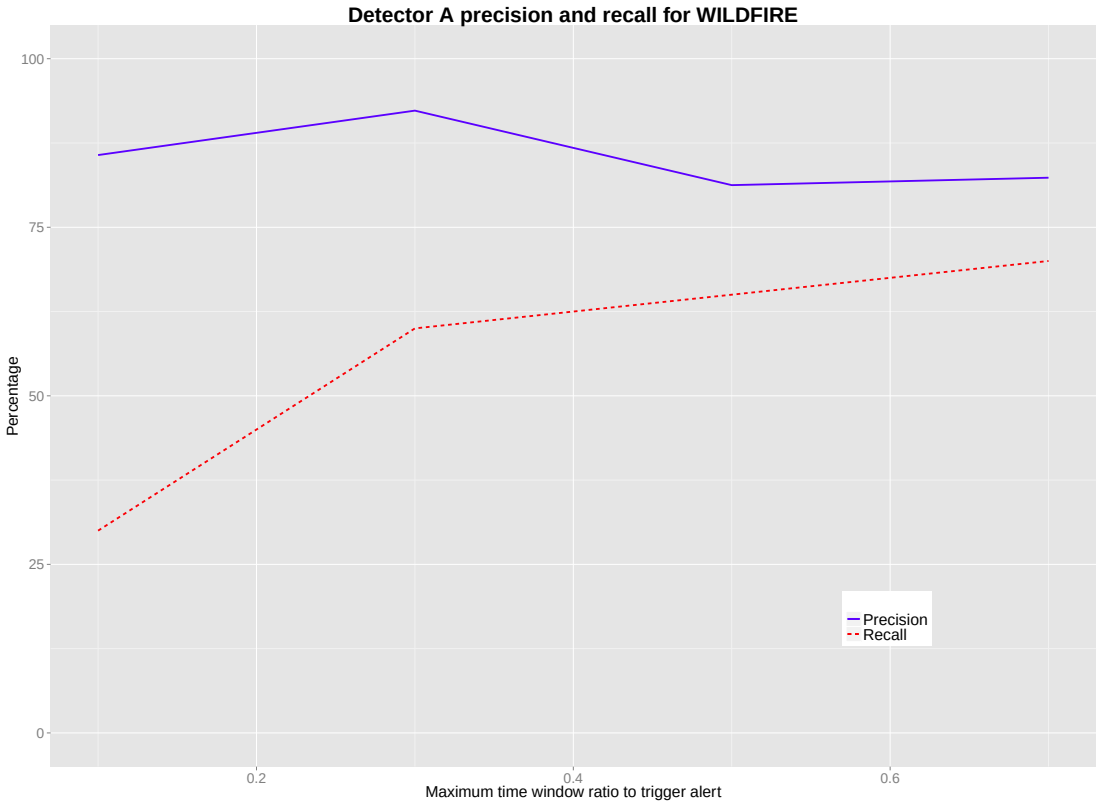
The delays are obtained over a common set of around 60% of the ground truth for both graphs 5.20a and 5.20b.
The shortest delays are obtained for the same reasons as the other event types when the tweet count threshold is 50 and the ratio 0.5 or 0.7 (no improvement between the two ratio). We detect all considered events in around 60 hours, better than the lag value of 74 obtained in section 5.1.3. The detector is almost always faster than BBC but it isn't as clear with Reuters as shown by graph 5.21.

We show the visualization of the wildfire detections in the graph 5.22 using the same settings as with the other event types and the configuration which gives almost the best delays and almost the best precision/recall (50 tweets and 0.5 ratio).

As we have seen in section 5.1.3 the spikes are often delayed compared to the events occurrences which explains why the detections are also delayed. The numerous smaller spikes get ignored most of the time showing that the threshold mechanism works.

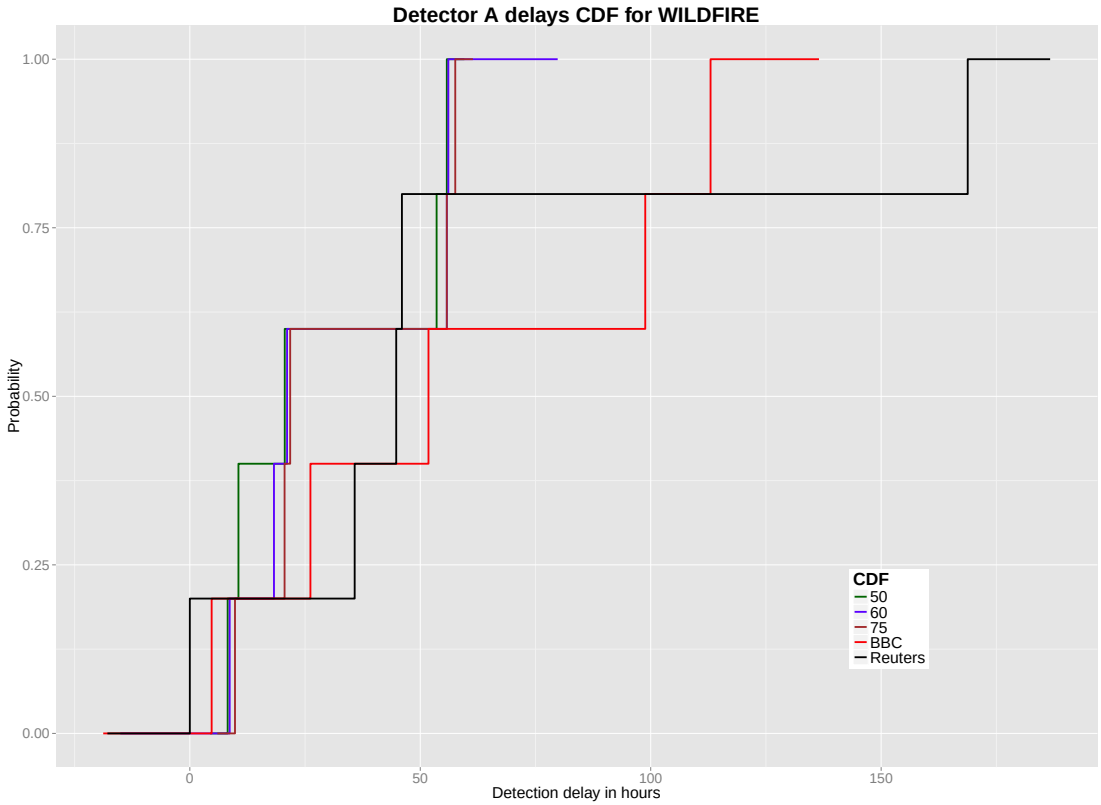


(a) Minimum tweet count with fixed ratio of 0.5

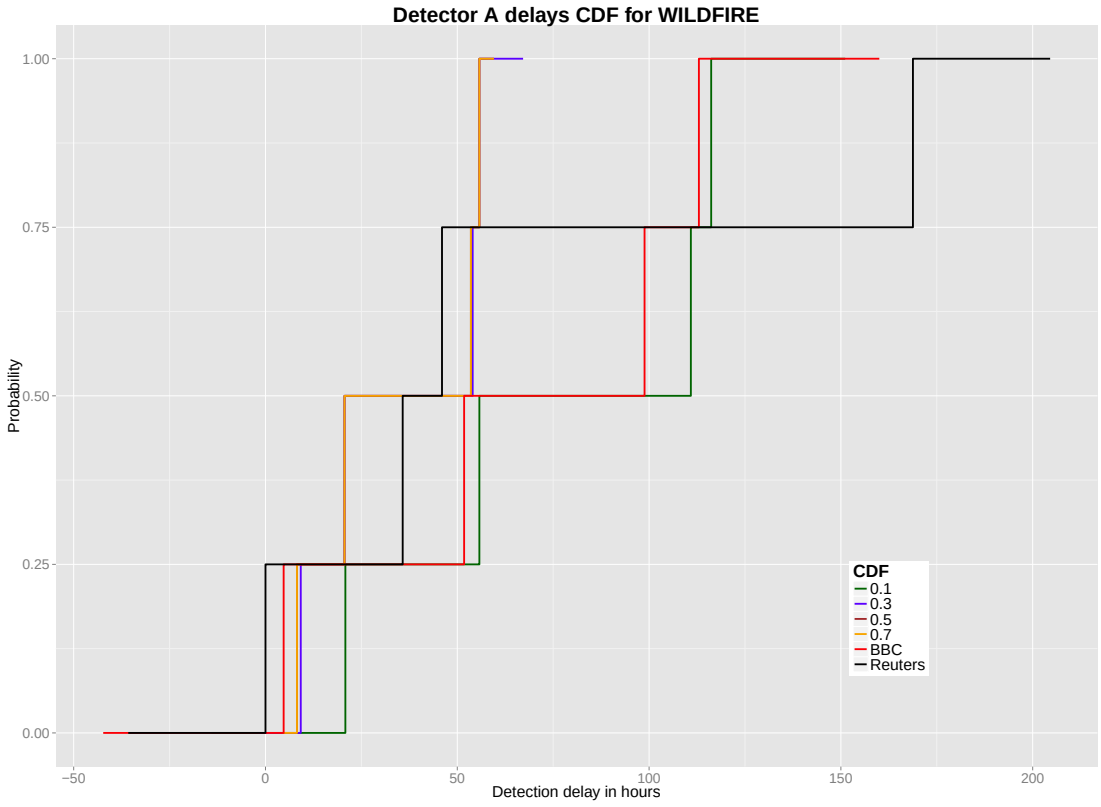


(b) Ratio with fixed minimum tweet count of 50

Figure 5.19: Detector A precision and recall for wildfires



(a) Delays with fixed ratio of 0.5



(b) Delays with fixed minimum tweet count of 50

Figure 5.20: Detector A wildfire detection delays

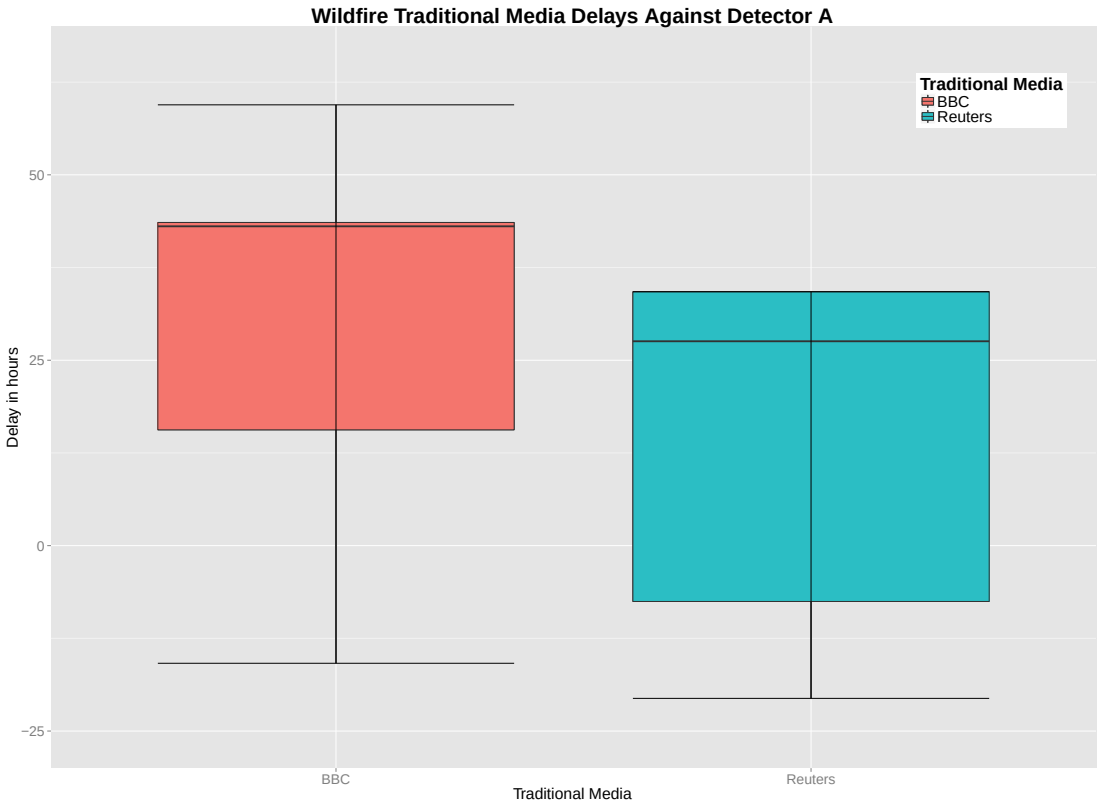


Figure 5.21: Traditional media delays for wildfires against detector A (a positive value e.g. 5 implies that the detector is 5 hours faster than the media and the opposite for negative values)

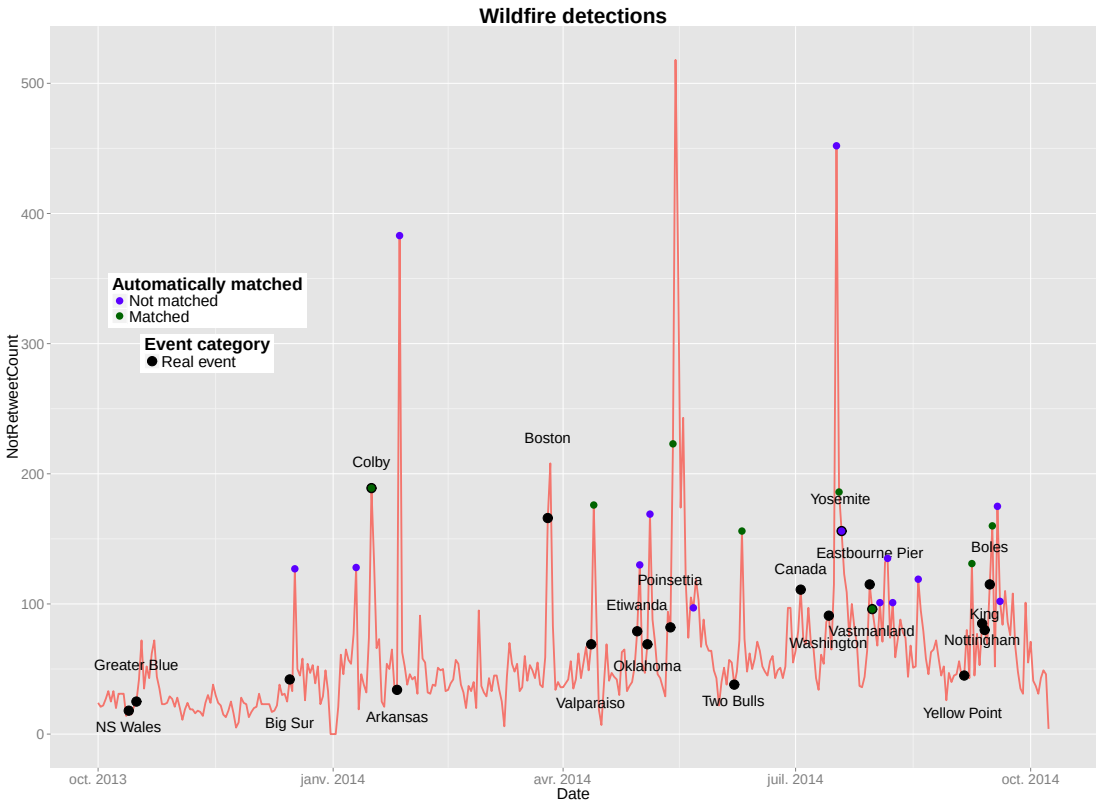


Figure 5.22: Detector A wildfire detections

5.3.2 Content based Detection

The parameters we focus on is the main threshold of the detector i.e. the minimum number of tweets in a candidate event cluster before the algorithm triggers an alert. The other parameters stay constant at 10 for the maximum hashtag count in a cluster and that a cluster's latest update isn't older than one day.

Plane Hijackings

The precision and recall graphs are shown in figure 5.23.

Similarly to what we observed with the frequency event detector we expect that if the cluster tweet count threshold is set to a high value, the precision will increase (an alert will be raised only if a large number of tweets with the same hashtags are present which is usually the sign of an event happening), and the recall will go down (smaller scale events won't generate enough tweets with the same hashtags and thus stay undetected).

The graph 5.23 confirms our expectations, the precision goes up with the tweet count threshold and the recall finally decreases. The overall precision and recall are good, we detect all the 5 events of the ground truth and have a maximum of two false positives over the full year. Let us now analyze the detection delays in graphs 5.24 obtained over the full set of the plane hijacking ground truth.

Logically the detector with the smallest cluster tweet count threshold will have the fastest delays because it needs to wait for less tweets than the detectors with higher thresholds. This expectation is confirmed by graph 5.24, the shortest delays are obtained when the cluster tweet count threshold is 5 tweets. We detect all considered events in around 6 hours, better than the frequency based detector which needs 7.5 hours. The detector is almost always faster than BBC and Reuters as shown by graph 5.25.

We present a visualization of the plane hijacking detections in the graph 5.26 using the configuration giving the best delays (cluster tweet count threshold of 5).

We see that the main events are detected but also relatively many detections are triggered because of small spikes. Two of these detections are viral jokes and the rest unrecognized event duplicates. The "main" viral jokes (the ones annotated in the graph) probably didn't have any hashtags in their tweets and thus stayed undetected.

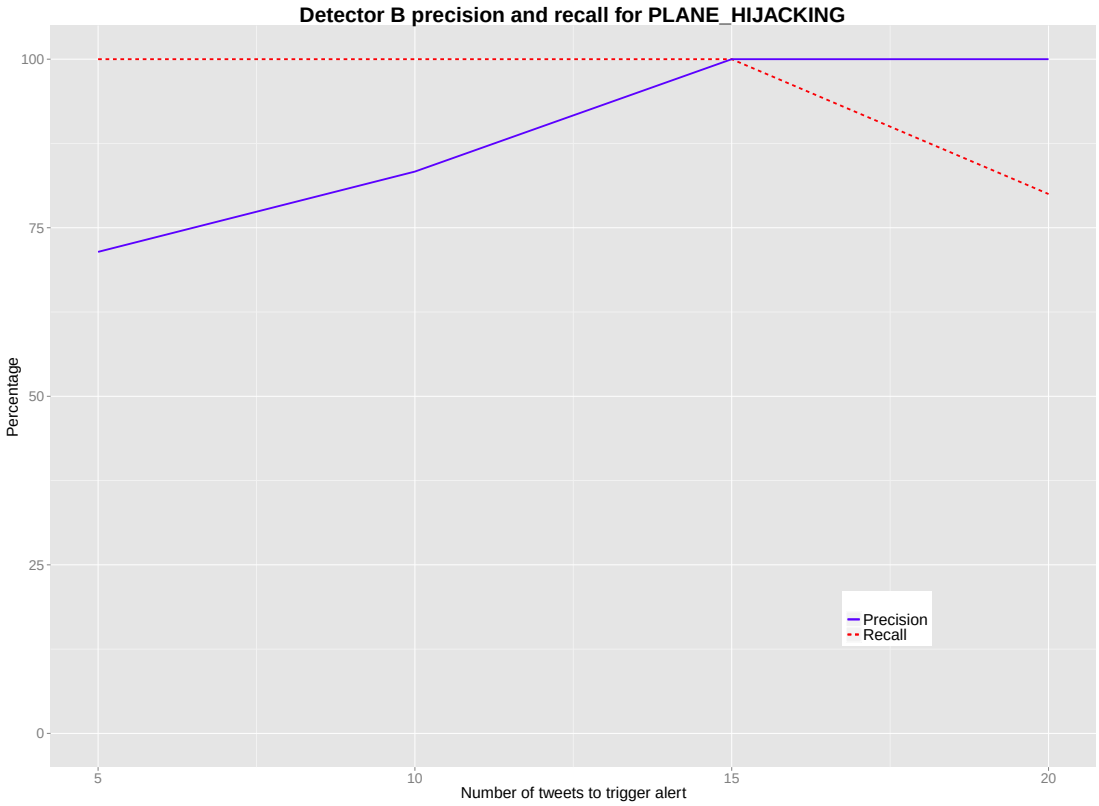


Figure 5.23: Detector B precision and recall for plane hijackings

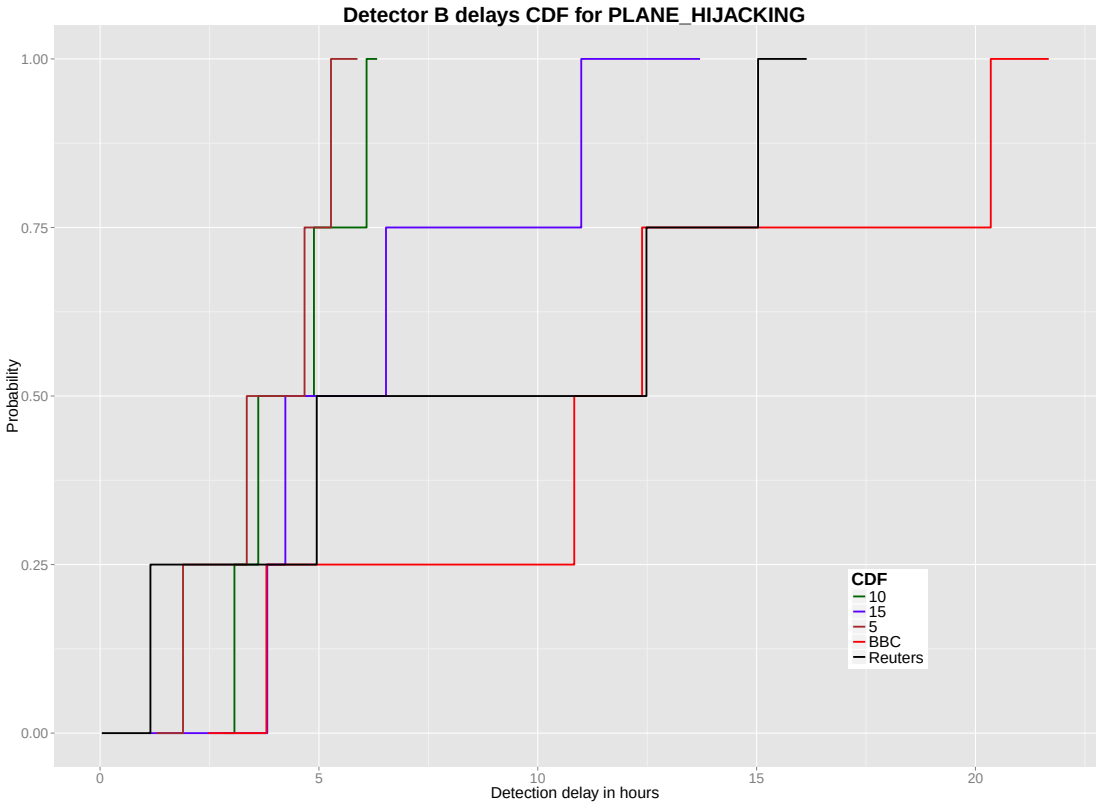


Figure 5.24: Detector B plane hijacking detection delays

Figure 5.25: Traditional media delays for plane hijackings against detector B (a positive value e.g. 5 implies that the detector is 5 hours faster than the media and the opposite for negative values)

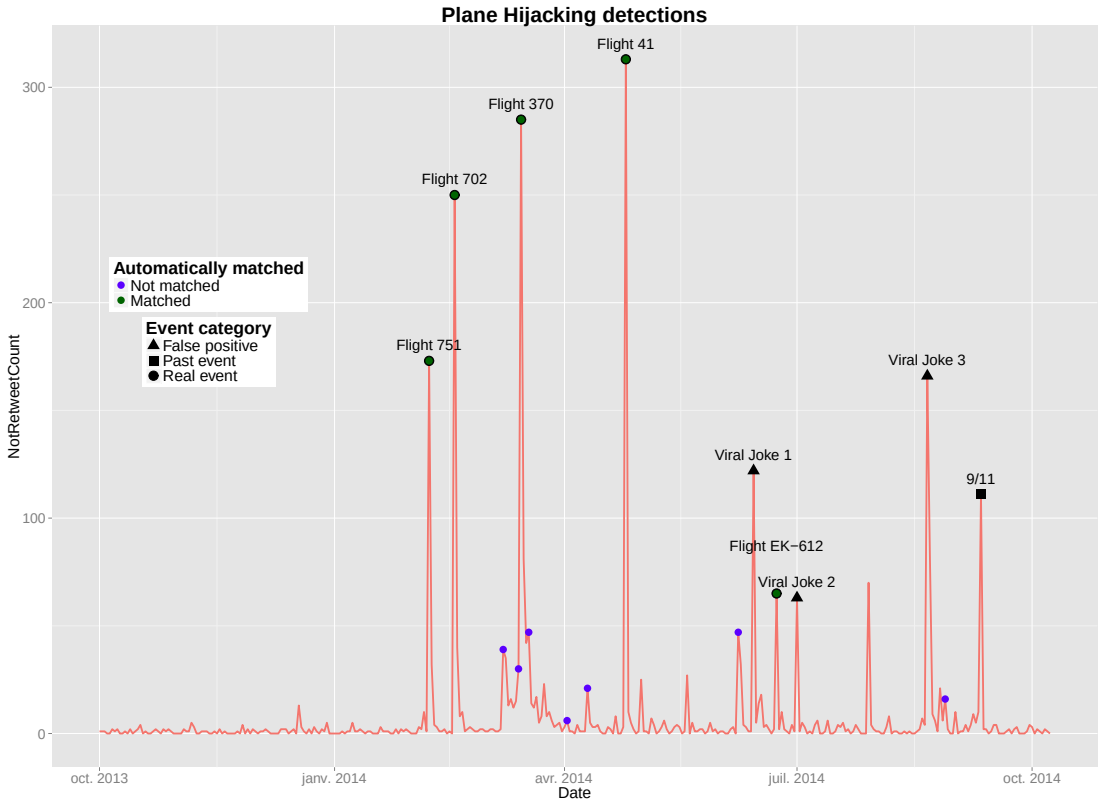


Figure 5.26: Detector B hijacking detections

Plane Crashes

The precision and recall graphs are shown in figure 5.27.

Following our intuition, the precision goes up with the tweet count threshold while the recall decreases. It is more obvious than with plane hijackings because there are more small scaled plane crashes events which are quickly affected by the higher threshold.

The overall precision and recall are good with a cluster tweet count threshold of 5 but the recall decreases quickly if we increase this threshold making 5 the only real option. Graph 5.28 present the detection delays obtained over 50% (as shown in graph 5.27 the detector didn't reach higher recall for a threshold of 15 tweets) of the plane crash ground truth.

As expected, the quickest detector is the one using a threshold value of 5 tweets. We detect all considered events under 15 hours, always faster than BBC but slower than Reuters as shown by graph 5.29.

We present a visualization of the plane crashes detections in the graph 5.30 using the configuration giving the best delays (cluster tweet count threshold of 5).

Overall there are many more detections and more smaller scale events get detected but most of the detections are unmatched. The reasons is that our algorithm is based on n-grams and has only 5 tweets per detection to work upon which makes it harder to match two detections which may be talking about the same events but about different aspects of it. With so many detections, even if it is still manageable by a human, it becomes more interesting to cluster the detections.

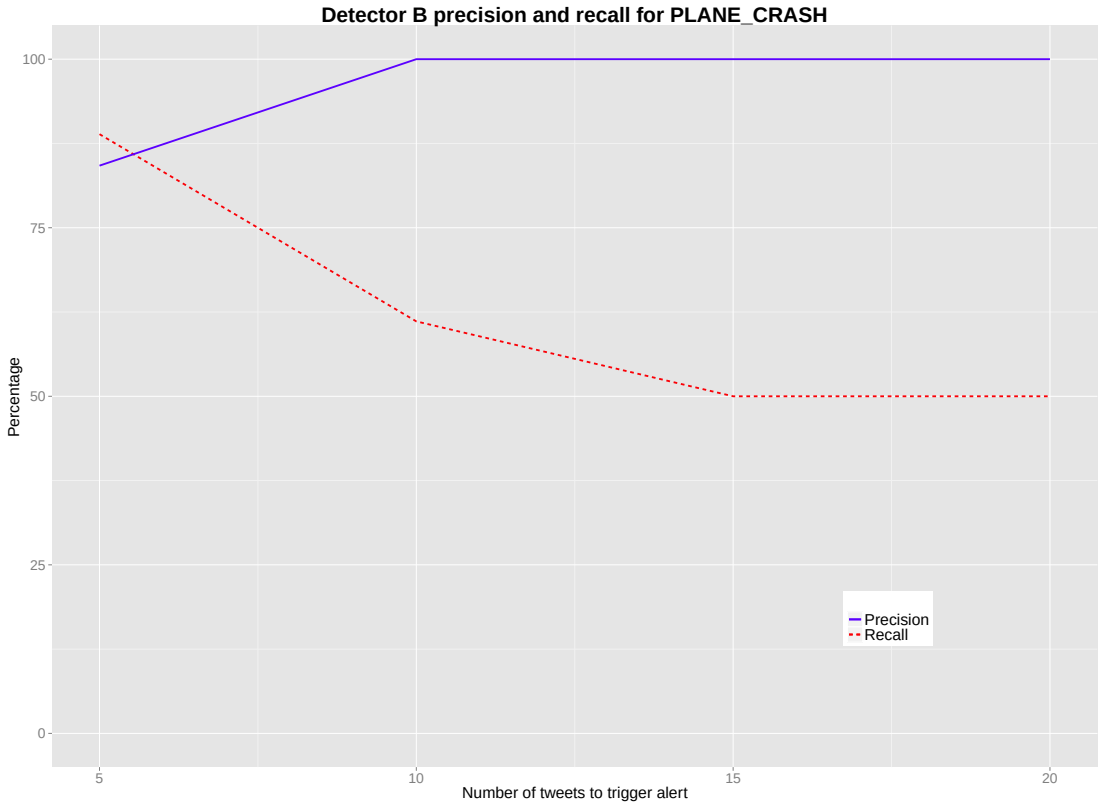


Figure 5.27: Detector B precision and recall for plane crashes

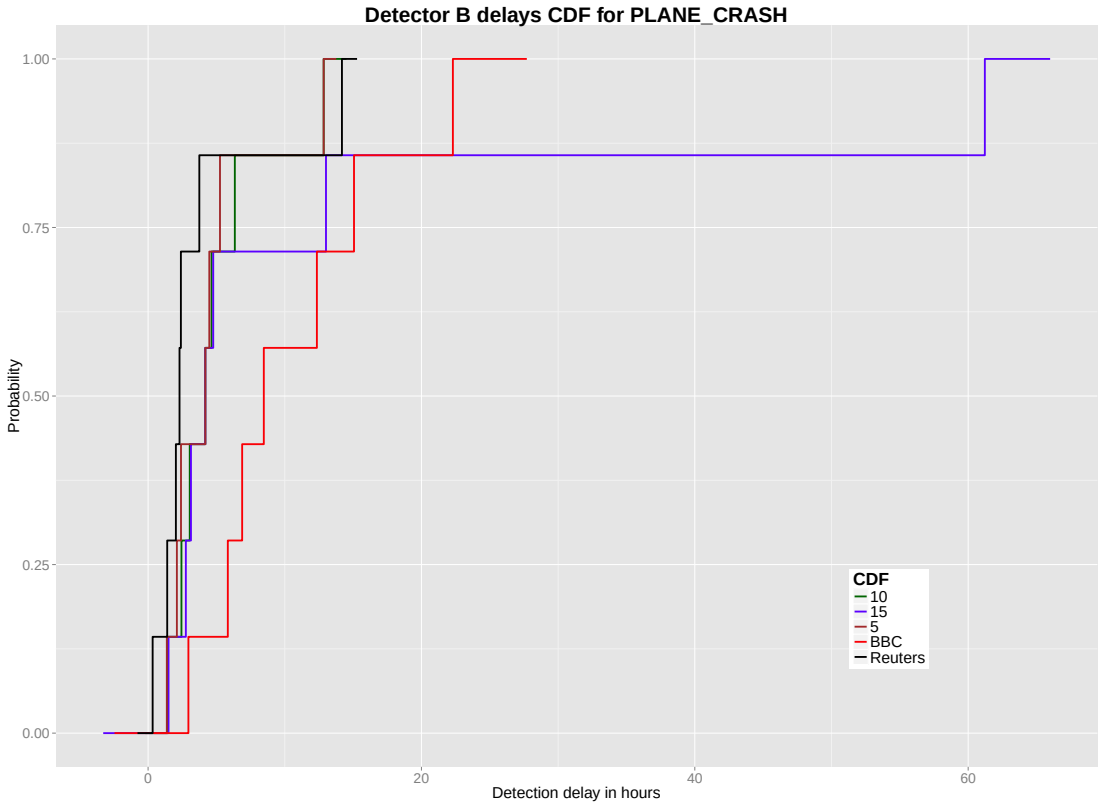


Figure 5.28: Detector B plane crash detection delays

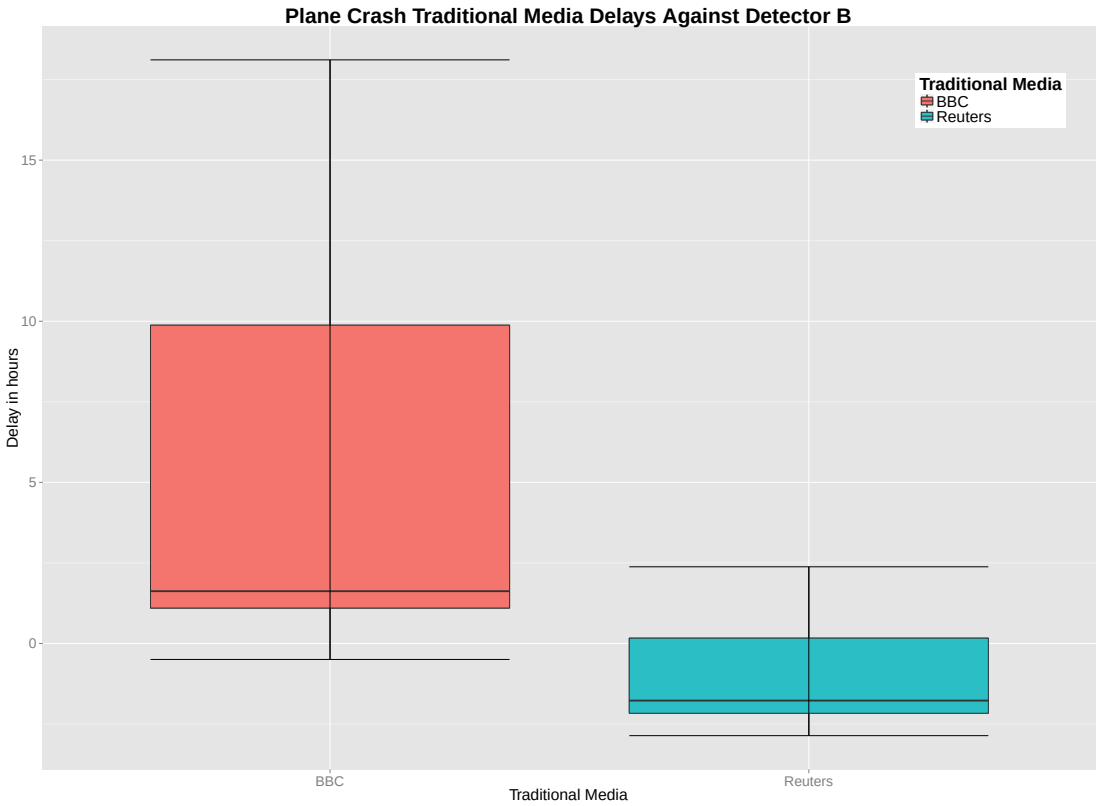


Figure 5.29: Traditional media delays for plane crashes against detector B (a positive value e.g. 5 implies that the detector is 5 hours faster than the media and the opposite for negative values)

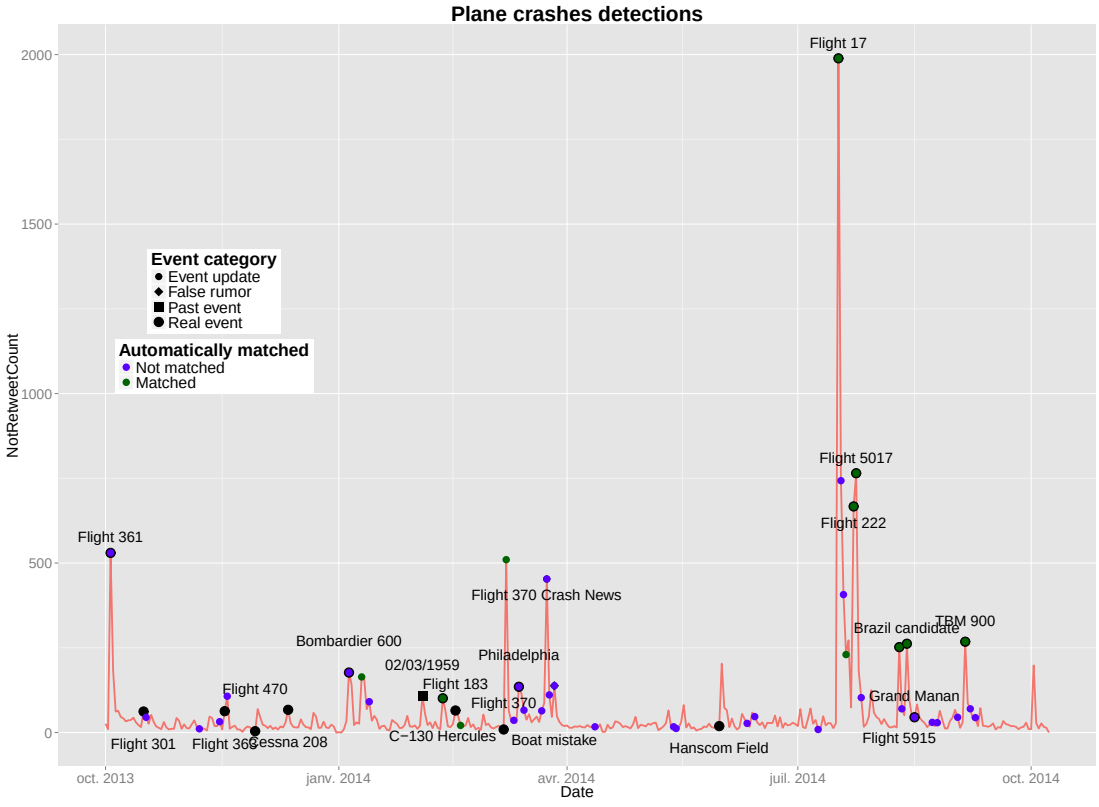


Figure 5.30: Detector B crash detections

Wildfires

The precision and recall graphs are shown in figure 5.31.

We see that the recall follows our expectations and decreases with the cluster tweet count threshold but the precision also decreases after increasing once. The reason for that is similar to what we explained in section 5.3.1, it is due to the decreasing recall. If we check the false positive and detection counts from Appendix D we see that the false positive count which starts at 18 for 5 tweets decreases to 6 for 10 and 15 tweets and finally 5 for 20 tweets but at the same time the number of detected events decreases as well resulting in the precision curve of graph 5.31. So increasing the cluster tweet count threshold really improves the results by decreasing the false positive count but because it cripples the detections rate, this progress remains unnoticed.

Detector B really has trouble detecting wildfires, it seems that users reporting wildfires don't use hashtags as well as the users reporting plane crashes and hijackings. The only option for the threshold is 5 or 10 tweets but we have to choose between missing more than half the wildfires events or having more than half of the detections potential false positives.
Graph 5.32 present the detection delays obtained over 45% of the wildfire ground truth.

As expected, the quickest detector is the one using a threshold value of 5 tweets. We detect all considered events under 50 hours, always faster than both BBC and Reuters as shown by graph 5.33.

We present a visualization of the wildfires detections in the graph 5.34 using the configuration giving the best delays (cluster tweet count threshold of 5).

The obvious problem is that there are too many detections (over one hundred) which highlights the need of clustering. But not half of all of these detections are false positives as graph 5.31 hints. As shown in Appendix D, there were only 18 false positives out of the hundred detections showed in the graph, all of the others are related to smaller scale wildfires (in 2014 California had 5620 wildfires...) or unmatched duplicates. To reduce the number of detections, further filtering parameters should be introduced like a geographical location for example.

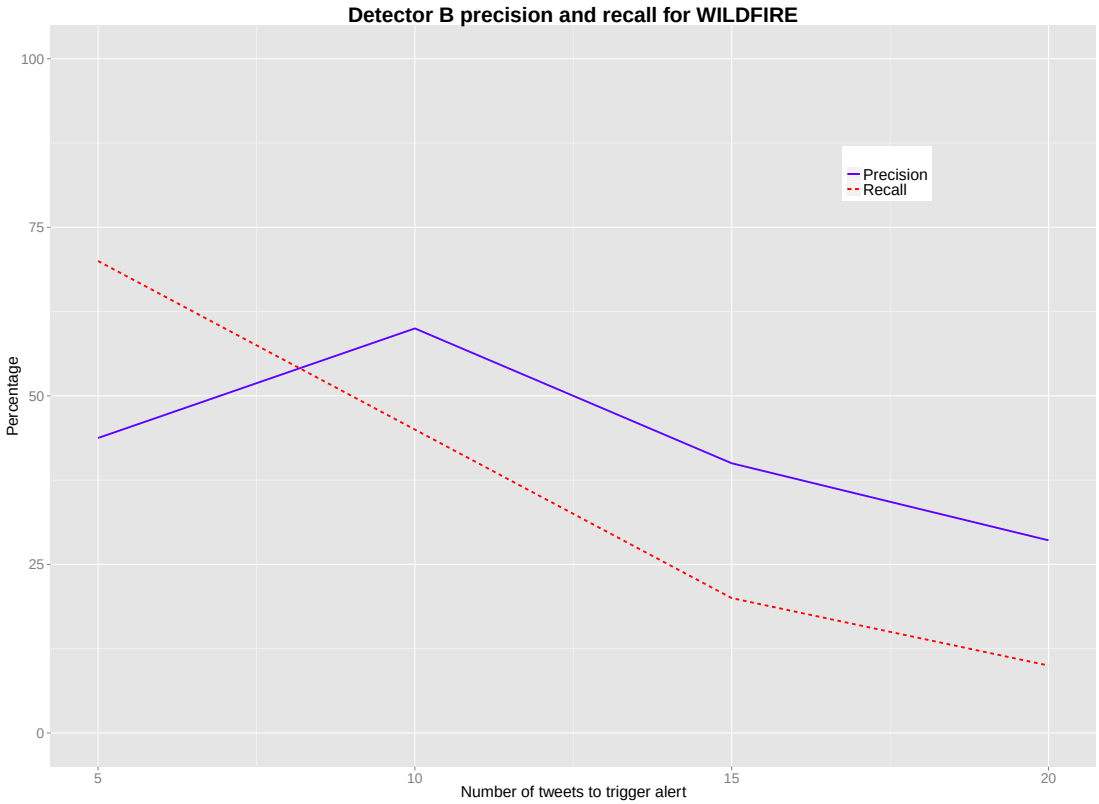


Figure 5.31: Detector B precision and recall for wildfires

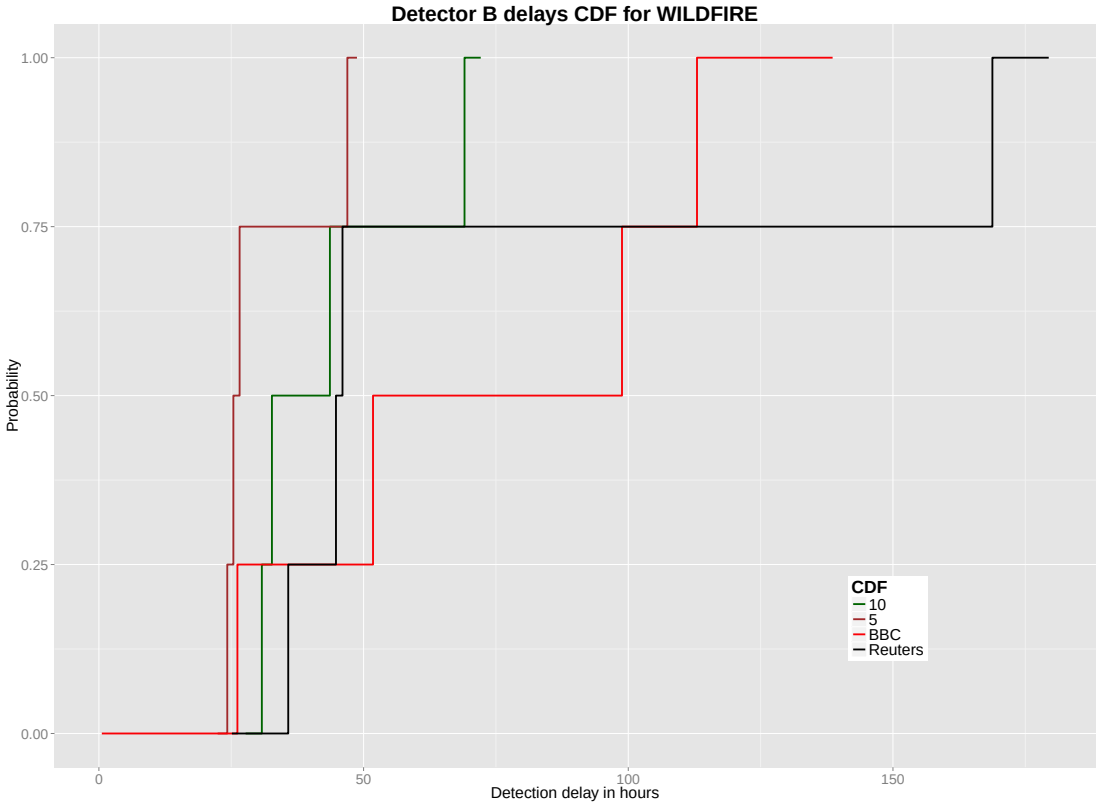


Figure 5.32: Detector B wildfire detection delays

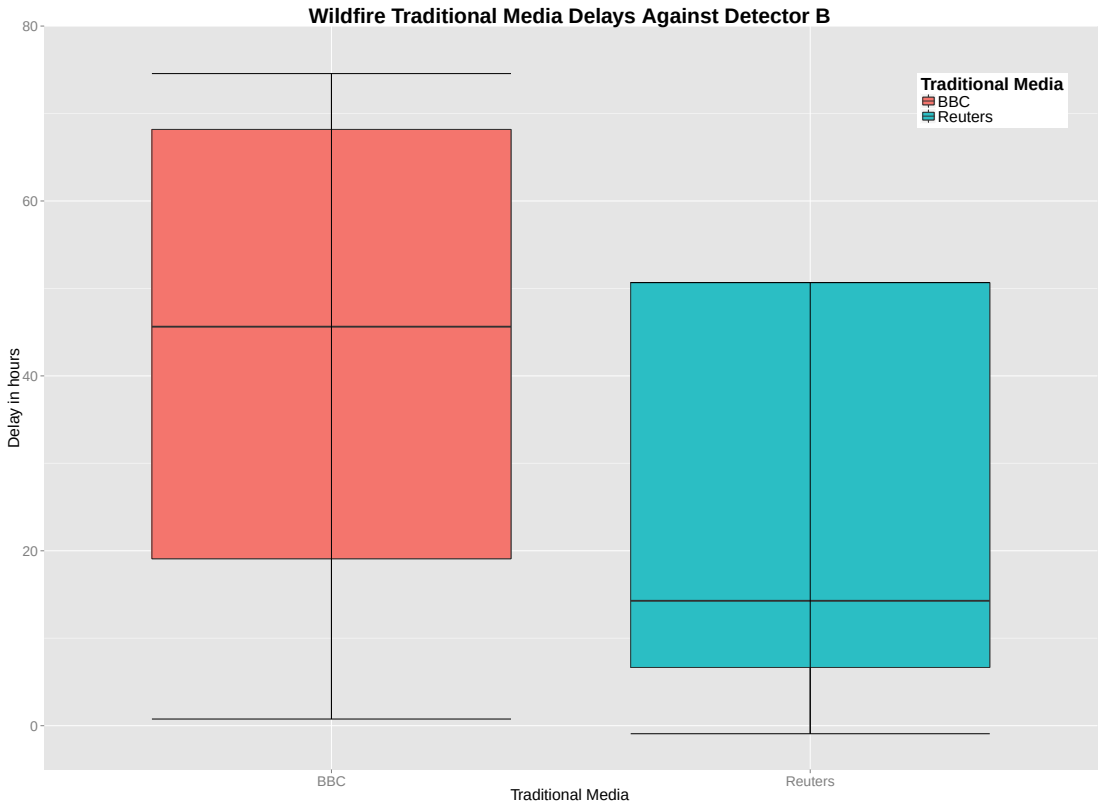


Figure 5.33: Traditional media delays for wildfires against detector B (a positive value e.g. 5 implies that the detector is 5 hours faster than the media and the opposite for negative values)

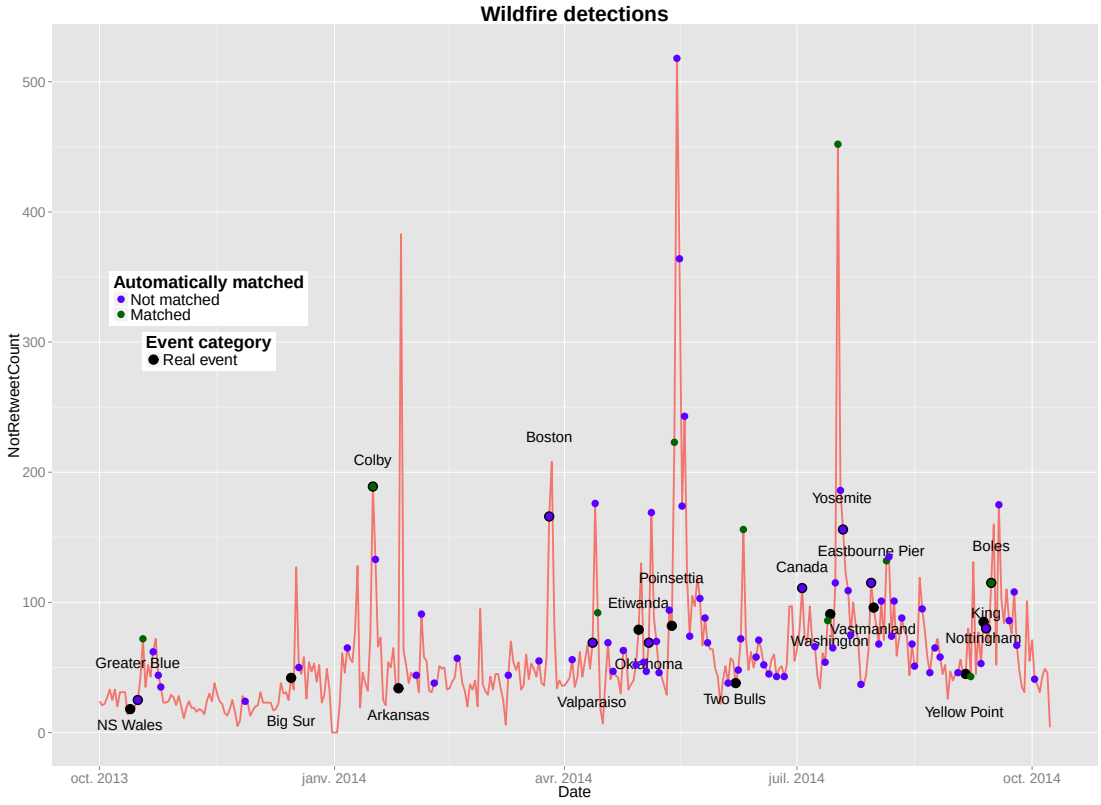


Figure 5.34: Detector B wildfire detections

5.3.3 Comparison of the Detectors Results

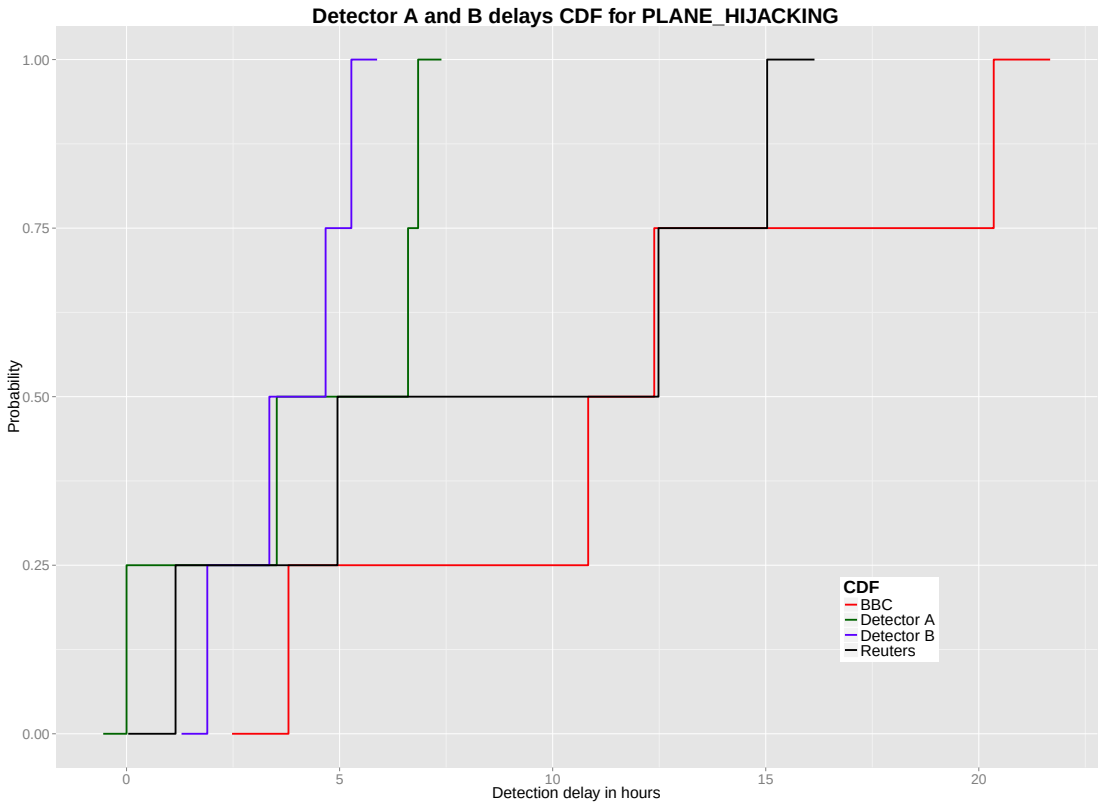
In this section we compare for each event types the results obtained by the two detectors. We call the frequency based algorithm detector A and the content based algorithm detector B.
One could expect the detector B to be always faster than detector A as it requires less tweets before triggering an event but we will see that it isn't always the case mainly because detector B has to wait for the tweets containing similar hashtags while ignoring the other tweets, thus adding a delay which can be longer than detector A's delays.
One difference to note is that detector B requires less computations than detector A; for example running an experiment over the full data without retweets using the setup described in section 4.2 takes around 15 hours with detector A and 8 hours with detector B.
We have seen that the intuition given by the correlation lag values from section 5.1 was not too far from our results especially for detector A because the lag values gives information on how lagged is the Twitter data reaction to an event occurrence and the detector A mainly uses the burst of the Twitter event related activity to decide whether or not to trigger an alert.

Plane Hijackings

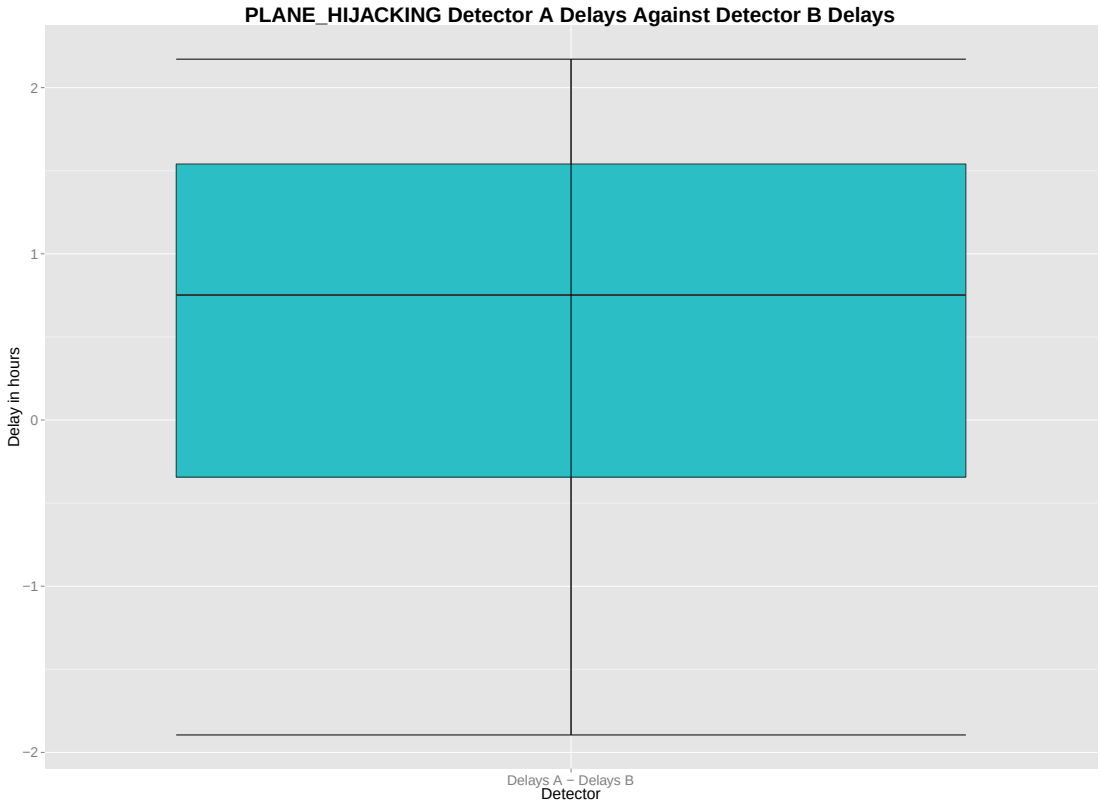
We see from the previous sections that we obtain similar precision and recall levels with both detectors. This is thanks to the popularity of the plane hijackings events which generate enough Twitter activity for detector A and tweets with relevant hashtags for detector B.
The comparison of the detection delays obtained with their respective optimal parameter choice is shown in figure 5.35.

We see that detector B is most of the time faster than detector A, with an advance of 45 minutes in 50% of the cases. We can explain this behavior with the following intuition: detector B works especially well for very popular events with popular hashtags and we have seen that plane hijackings events are very popular and well discussed in Twitter.

Regarding the number of detections (graph 5.14 and 5.26), detector B has slightly more detections than detector A but it isn't a high difference enough to make any conclusion.



(a) Delays of traditional media and detectors A and B



(b) Detector A delays for plane hijackings against detector B (a positive value e.g. 5 implies that the detector B is 5 hours faster than the detector A and the opposite for negative values)

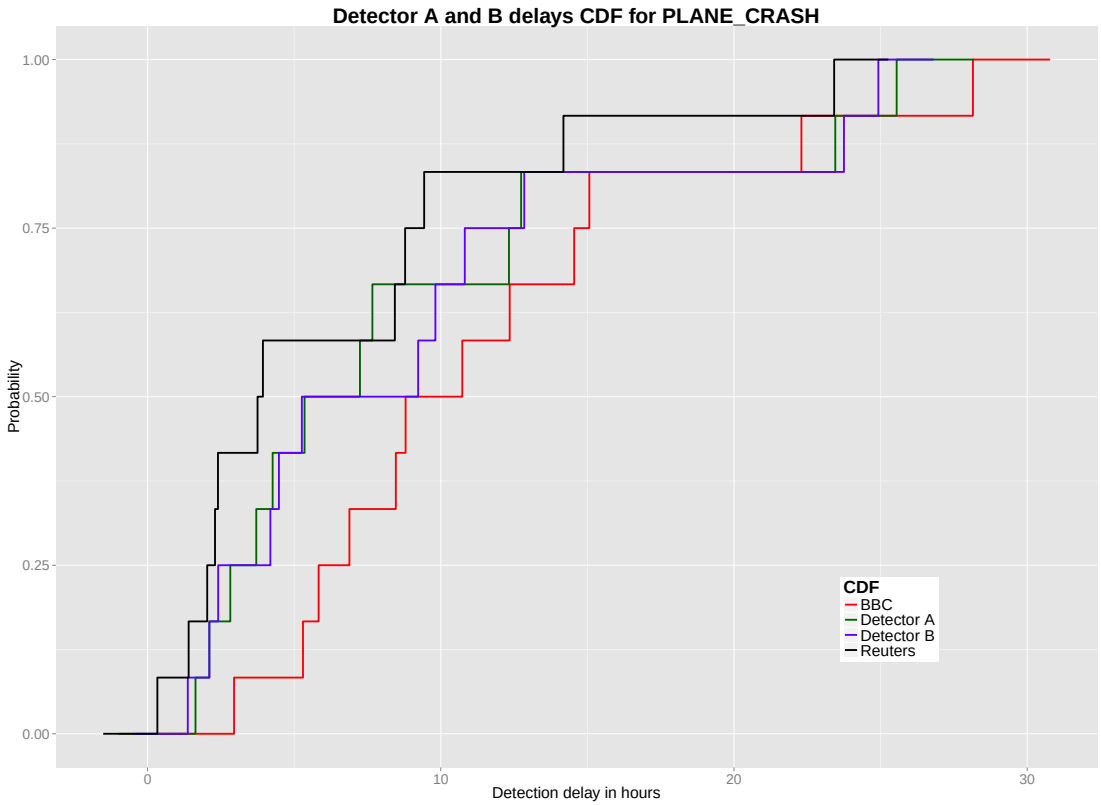
Figure 5.35: Comparison of detector A (frequency based) and B (content based) hijacking detections delays

Plane Crashes

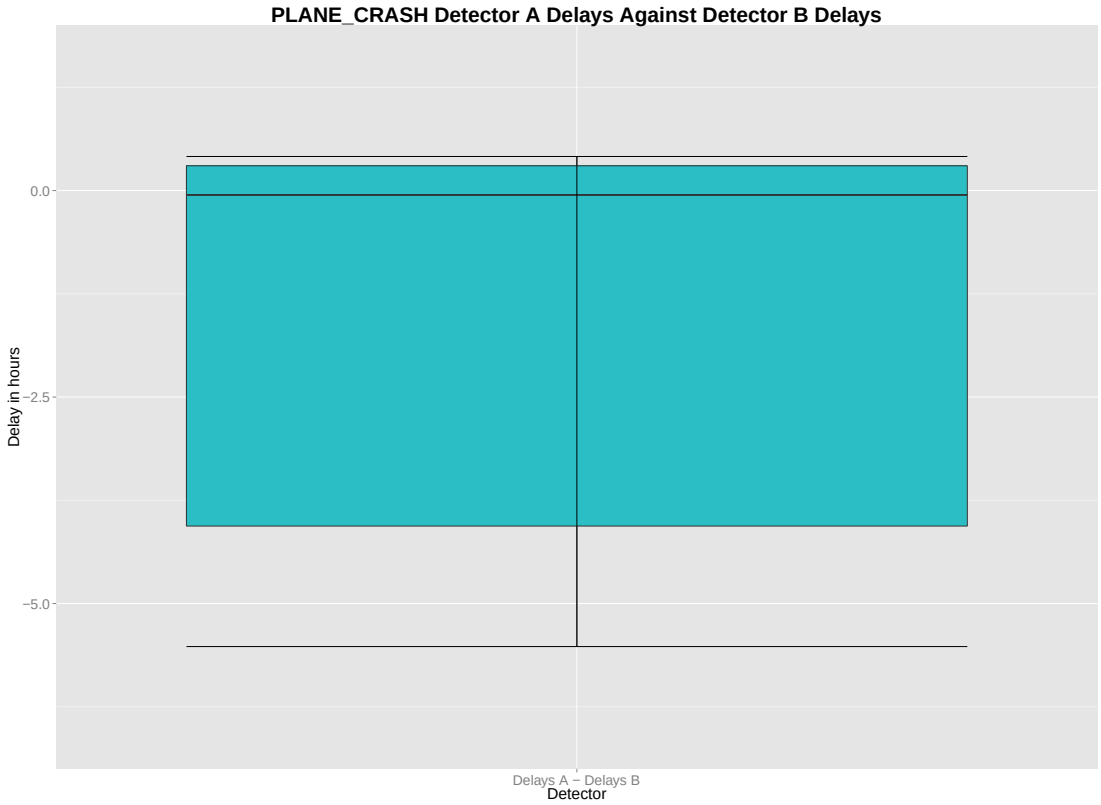
We have seen that detector B can compete with detector A only when we set its cluster tweet count threshold at 5 tweets in which case B gets the same recall but a weaker precision. This result is explained by the facts that the plane crash data is noisier than the plane hijacking data and that detector B is more sensible to noise than detector A because of its lower tweet count threshold (5 vs 50 tweets).
The comparison of the detection delays are shown in figure 5.36.

We see that detector A can be in 50% of the cases up to 4 hours faster than detector B but we see in graph 5.36 that the median is very close to 0 meaning also that they have similar detection performances.

Regarding the number of detections we see that detector B triggers more alerts than detector A (one reason for this is that it requires only 5 event classified tweets with same hashtags) which implies that the clustering of detections is more relevant for detector B.



(a) Delays of traditional media and detectors A and B



(b) Detector A delays for plane crashes against detector B (a positive value e.g. 5 implies that the detector B is 5 hours faster than the detector A and the opposite for negative values)

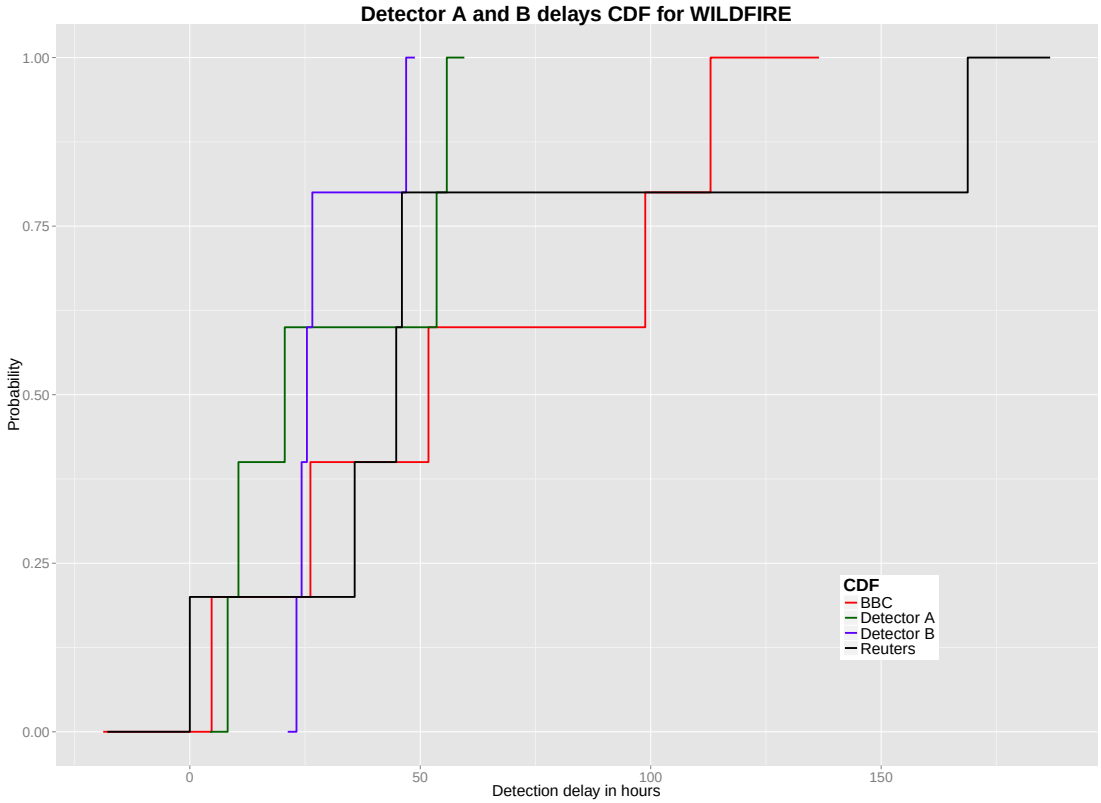
Figure 5.36: Comparison of detector A (frequency based) and B (content based) crash detections delays

Wildfires

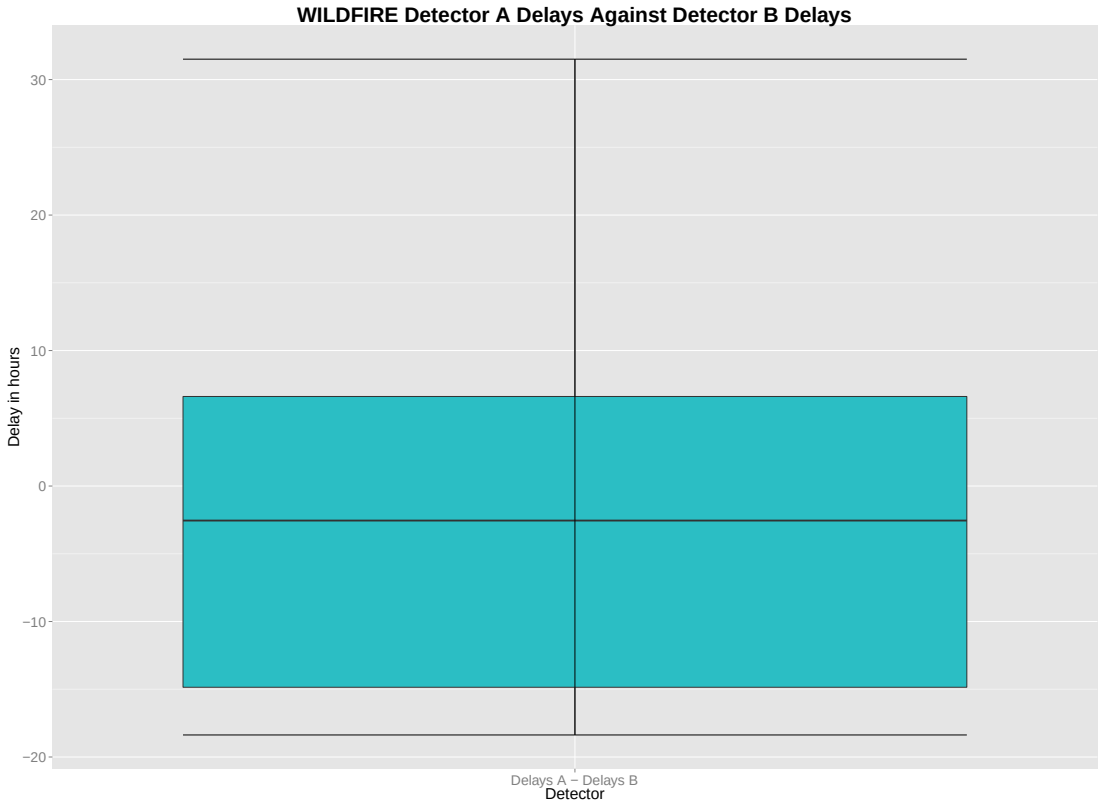
For both detectors we can only choose the smallest tweet count threshold to get an acceptable recall illustrating the fact that wildfires events individually don't generate a lot of tweets. Regarding precision, we get better results with detector A because detector B is more sensible to false positives (it requires only 5 tweets with same hashtags) and we have see that the wildfire related Twitter activity is much more noisy than plane hijacking and crash activity.
The comparison of the detection delays is shown in figure 5.37.

The delays range over a wider timespan for detector A (from 10 to 50 hours) than for detector B (from 20 to 50 hours). Similarly to plane crashes it isn't perfectly clear which detector gets better delays. As shown by the median, 50% of the events are detected faster by detector A by up to 15 hours.

The number of detections raised by detector B is much higher than detector A and we have seen that most of them aren't false positives. This is due to the fact that detector B is more sensible than detector A to smaller scale events, it can also detects concurrent events while detector A limits to one detection per 24 hours. Clustering is especially important for wildfires but we have seen that limiting the detection range with further filtering like geographical location is required because there are just too many wildfires occurrences.



(a) Delays of traditional media and detectors A and B



(b) Detector A delays for wildfires against detector B (a positive value e.g. 5 implies that the detector B is 5 hours faster than the detector A and the opposite for negative values)

Figure 5.37: Comparison of detector A (frequency based) and B (content based) wildfire detections delays

5.4 Detections Aggregation

In this section, we present the results of the aggregation module with input the detections triggered by the detectors from section 5.3. We show also the counts of false positives detections and detected events because ideally the number of aggregates should be the sum of these counts.

The general intuition on the results comes from the fact that the aggregation method used by the module relies on n-grams. This means that the fewer tweets in a detection, the more specific the extracted n-grams will be (fewer elements can be introduced in a shorter text) which can be a problem when for example two detections are related to the same event but whose tweets are talking about different aspects/informations of the event. On the other hand, a too high number of tweets in a detection may result in too general n-grams (e.g. not possible to differ between two wildfires) and the algorithm will aggregate together detections related to different events. So this is another trade-off, we have to find the number of tweets which makes the module the most robust.
This intuition describes the general trend that should happen when using this algorithm but are cases where using more tweets may actually increase the number of aggregates. This happens if the newly introduced tweets contain different elements causing the detections to differ where before they would be considered duplicates.

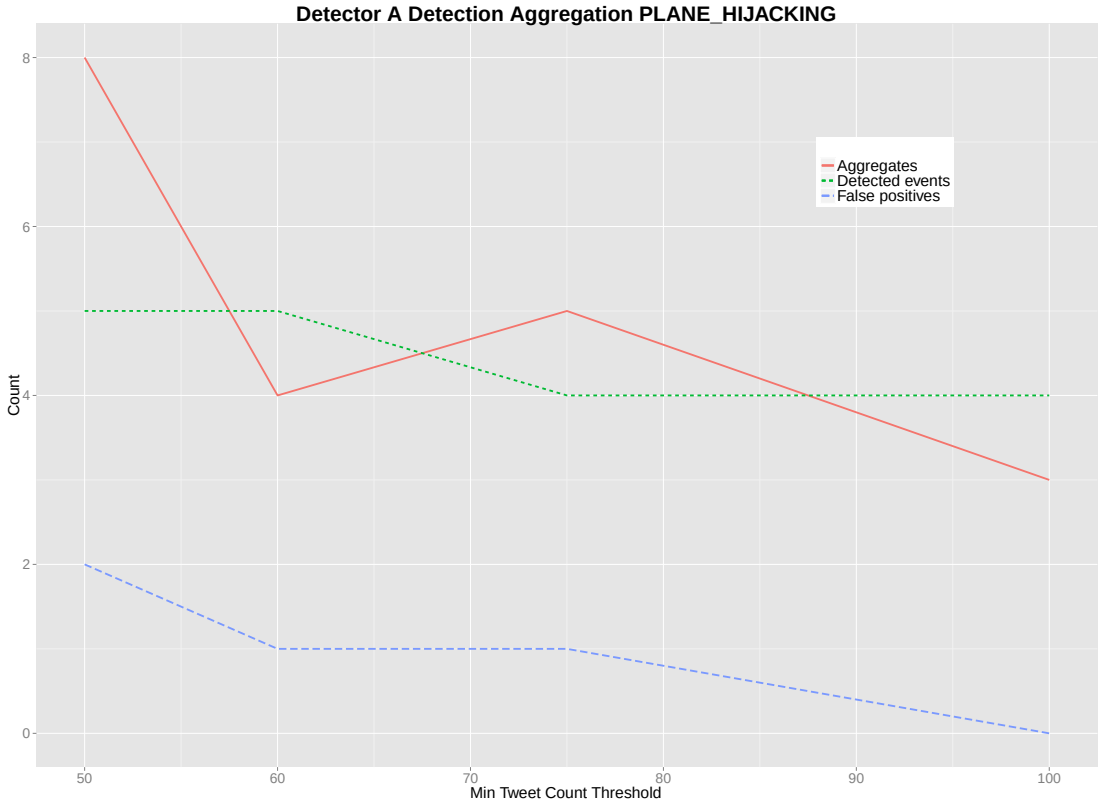
5.4.1 Plane Hijackings

The count of different aggregate representatives for detector's A detections is shown in figure 5.38 and for detector B's detections in figure 5.39.

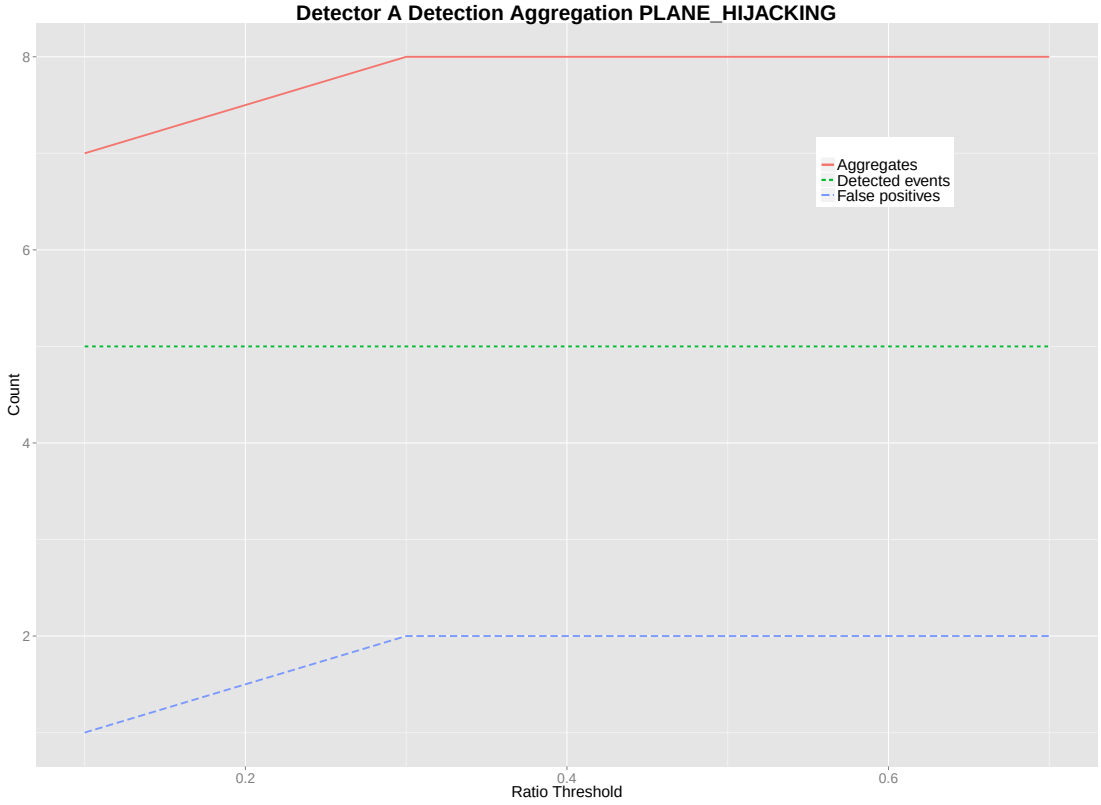
We see from graph 5.38a that following our intuition, the aggregate count curve general trend is to decrease as the tweet count threshold increases. The module works well already with 50 tweets as it detects 8 aggregates from an input of 5 events and 2 false positives which means it miss-aggregate one duplicate detection. It reaches the correct number of aggregates at 75 tweets but we see that the aggregate count increases while the detected event count decreases. This implies that previously considered duplicates are now aggregated separately because the new tweets introduced differences between the detections.

Graph 5.38b also goes in the direction of our argument; we see that the aggregate count stays constant because the number of tweets in the detections stays constant. Similarly to graph 5.38a, the modules miss-aggregates one detection leading to the count of 8 aggregates. When the ratio is 0.1, the number of false positive goes down to 1 which makes the aggregate count go down to 7 showing that the viral joke is aggregated separately as it should be.

Graph 5.39 confirms our intuition. Obviously 5 tweets isn't enough to be able to aggregate the duplicate detections properly. Like with the detections from detector A, the algorithm has trouble to correctly classify one duplicate detection, we see that there is always one aggregate more than the combined detected event and false positives counts. Aside from this mistake the algorithm seems to work properly from 10 tweets. The number of tweets to correctly aggregate detections of detector B will be lower than with the detections from detector A because detector B selects the tweets according to whether they contain certain hashtags i.e. whether the tweets core entities are the same and correspond to each other. So detector B gets rid of a lot of useless tweets which pollute the n-gram extraction from detector A's detections.



(a) Aggregation of detector A's detections when varying the tweet count threshold (ratio = 0.5)



(b) Aggregation of detector A's detections when varying the ratio threshold (tweets count = 50)

Figure 5.38: Aggregation of detector A's detections

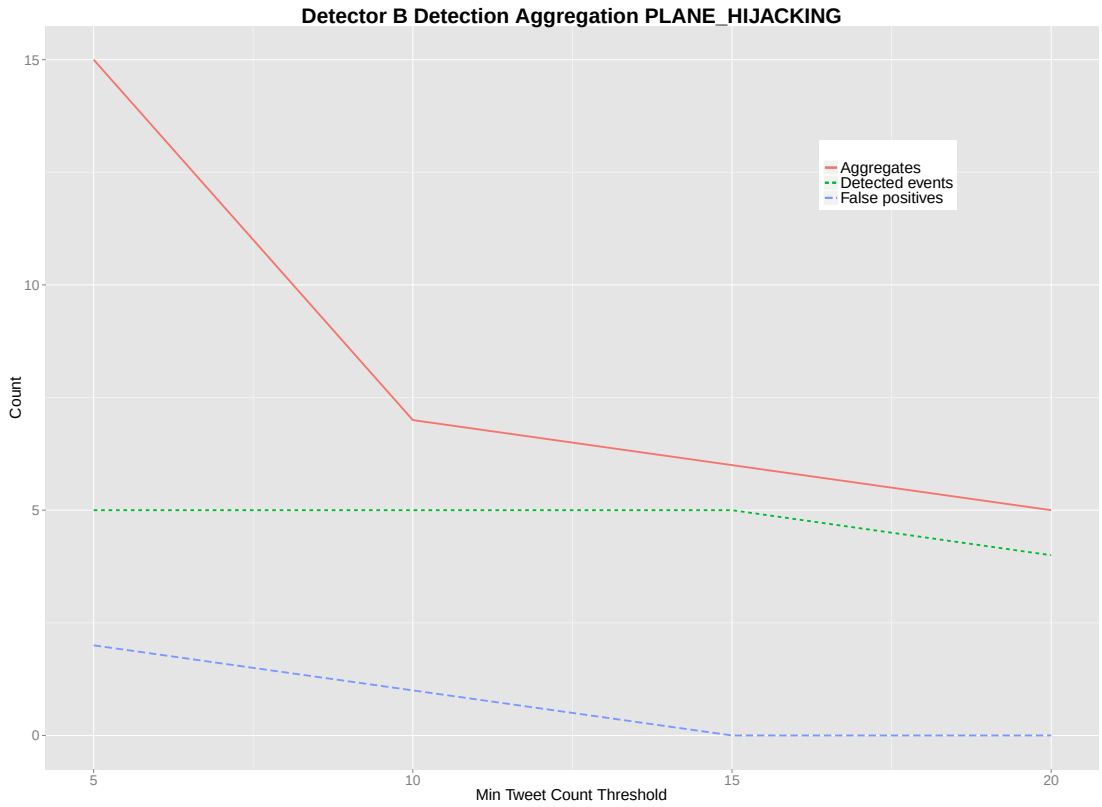


Figure 5.39: Aggregation of detector B's detections when varying the tweet count threshold

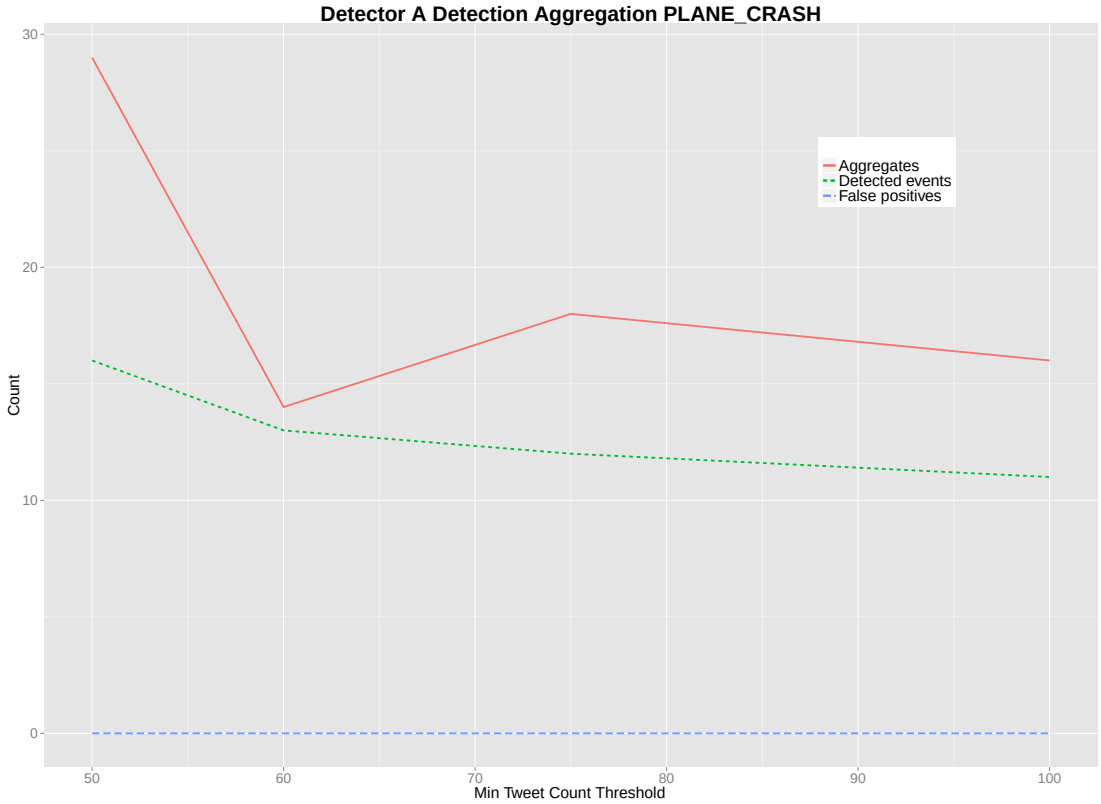
5.4.2 Plane Crashes

The count of different aggregate representatives for detector's A detections is shown in figure 5.40 and for detector B's detections in figure 5.41.

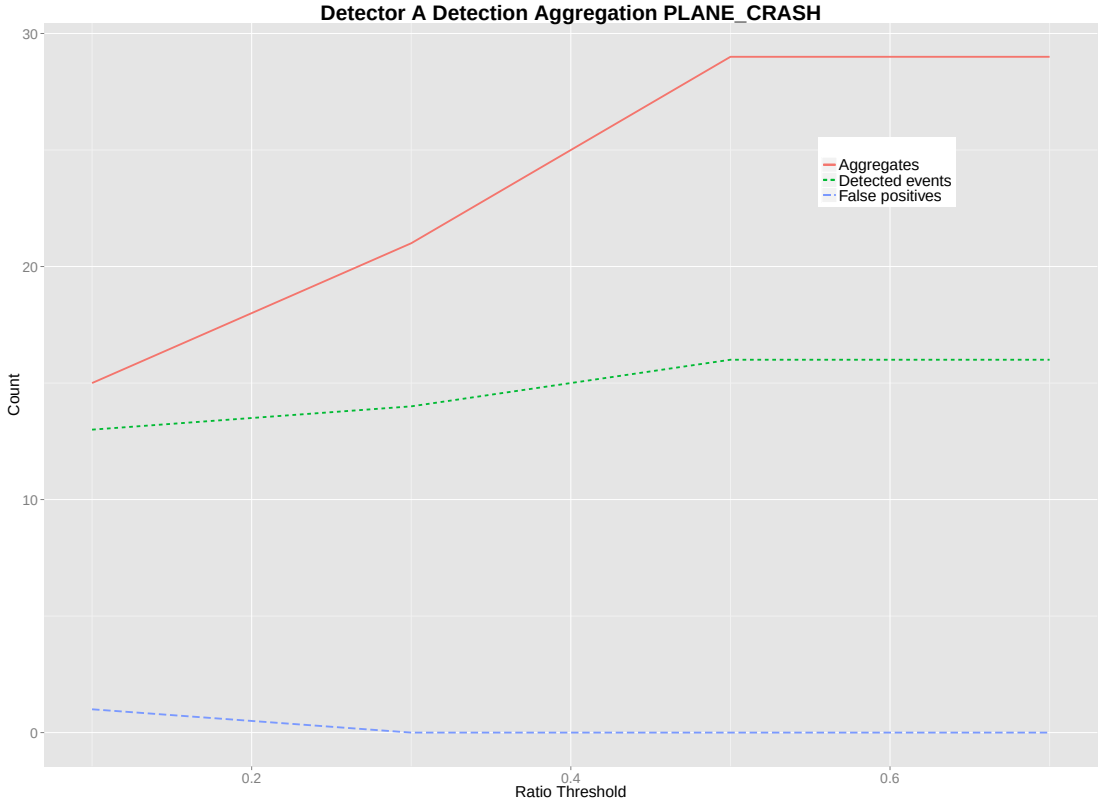
We see from graph 5.40a that for plane crashes 50 tweets isn't enough to be able to differentiate detections properly. This is a different results from plane hijackings but we have to recall that there are more plane crashes detections than plane hijacking detections which implies that the risk to make a wrong aggregation is higher, i.e. the algorithm needs a clearer differentiation between the detections. We again notice an increase in the aggregate count with 75 tweets detections. This is probably caused by the fact that these supplementary tweets may contain different informations which causes the duplicates to be miss-aggregated. The best aggregation seems to be attained with 60 tweets.

On the other hand, graph 5.40b is very different from its plane hijacking counterpart. We observe that the aggregate count increases with the number of events detection while the ratio threshold weakens. The reason behind this behavior is that each additional detected events comes with its set of potential duplicates detections. We know that a weaker ratio means an increasing recall thus an increasing number of duplicate detections which can be unrecognized and miss-aggregated. We have already seen that 50 tweets isn't enough for the algorithm to correctly aggregate plane crashes events so this larger number of detections isn't correctly matched and thus the number of aggregates increases.

Graph 5.41 confirms that 5 tweets isn't sufficient for the algorithm to properly aggregate the detections. It seems that the algorithm works best with 15 tweets. As we explained for plane hijackings, the algorithm will require less tweets to correctly aggregate detector B's detections that detector A's detections. The slight increase of aggregates with 20 tweets means that the new tweets contained different elements of information causing the duplicates detections to be aggregated differently.



(a) Aggregation of detector A's detections when varying the tweet count threshold (ratio = 0.5)



(b) Aggregation of detector A's detections when varying the ratio threshold (tweets count = 50)

Figure 5.40: Aggregation of detector A's detections

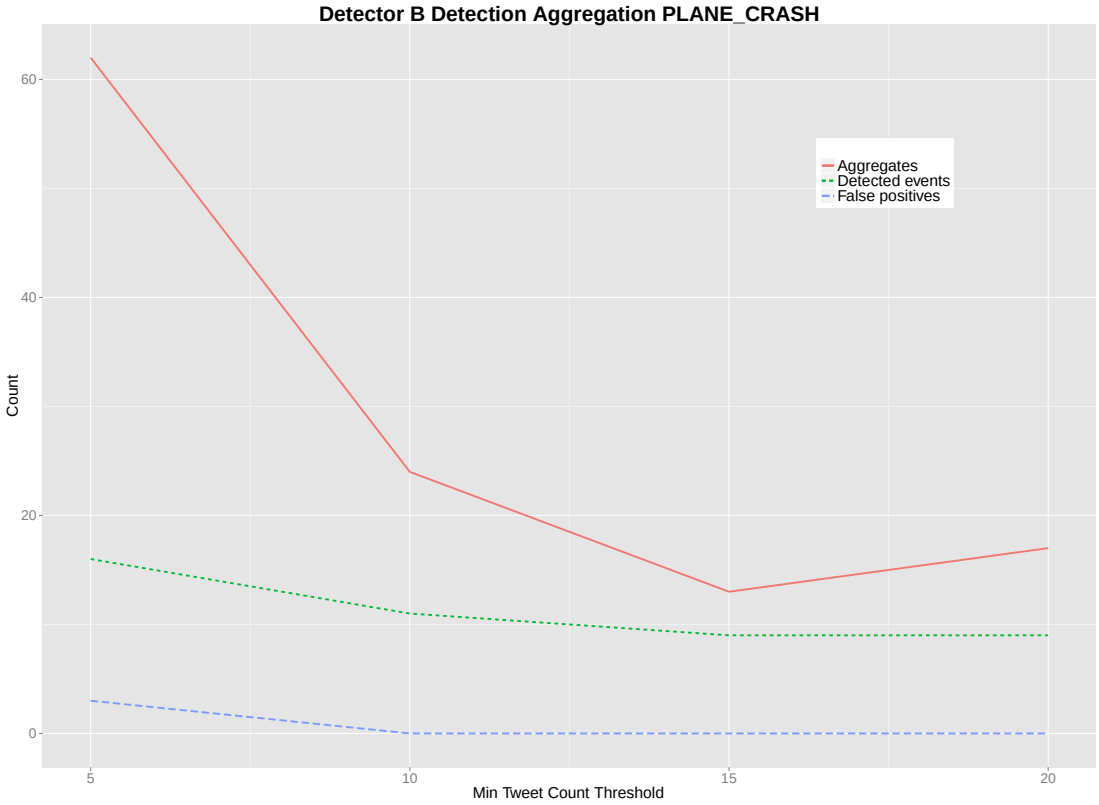


Figure 5.41: Aggregation of detector B's detections when varying the tweet count threshold

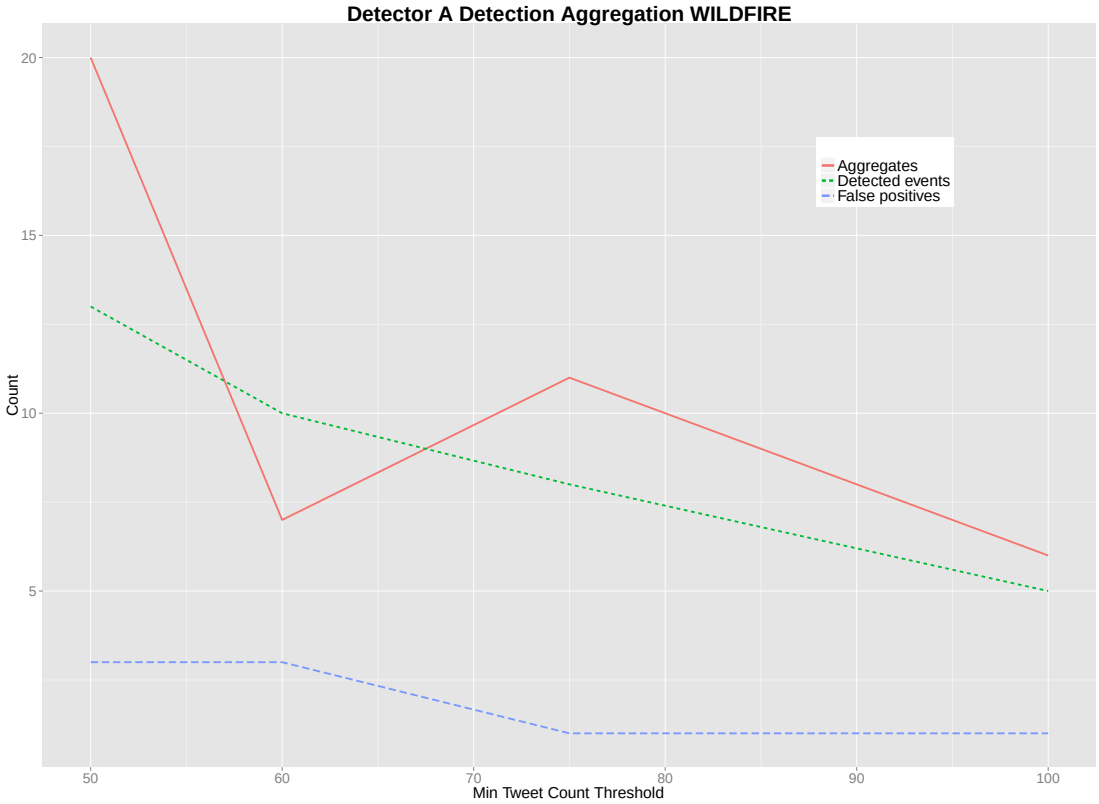
5.4.3 Wildfires

The count of different aggregate representatives for detector's A detections is shown in figure 5.42 and for detector B's detections in figure 5.43.

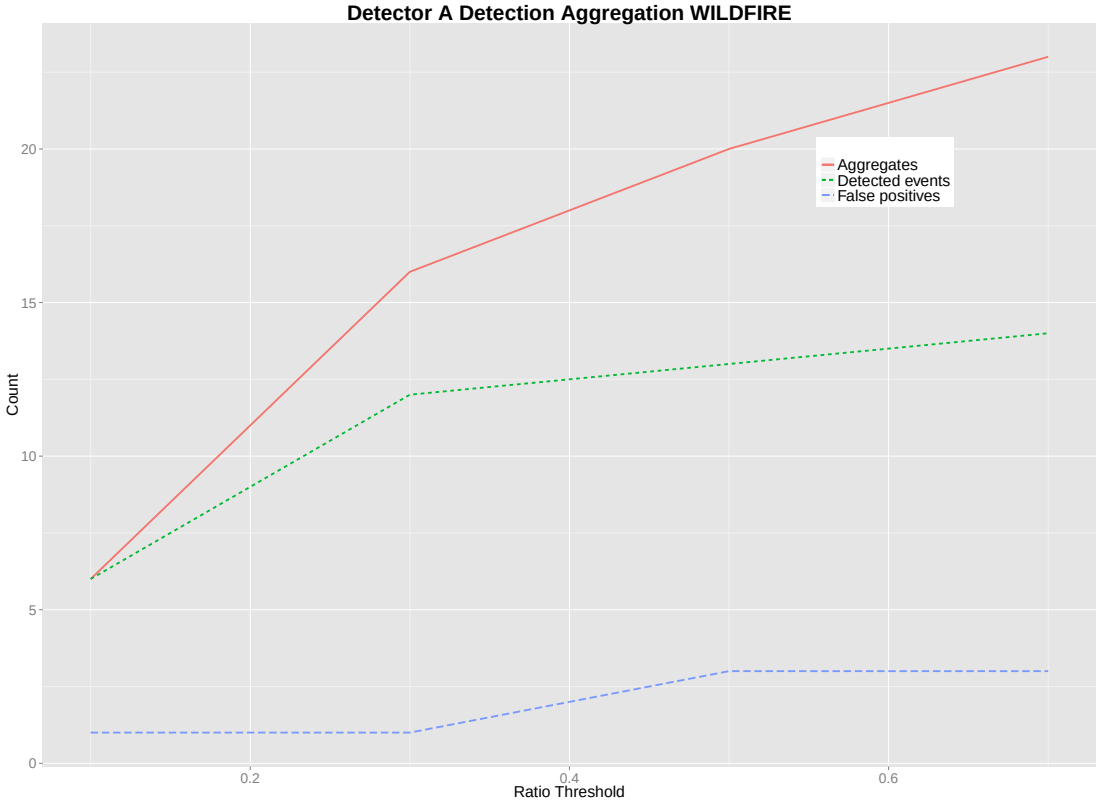
We observe from graph 5.42a a similar behavior as with plane hijackings and plane crashes. It looks like 50 tweets is also not enough to correctly aggregate the wildfire detections. The aggregate count closest to the detected event count is obtained at 100 tweets. Again we observe an increase of the aggregate count at 75 tweets meaning that the new tweets contain new information differentiating the detections from each other.

Graph 5.42b follows as well a very similar behavior to graph 5.40b for the same reasons we explained with plane crashes.

The graph 5.39 behavior follows our expectation, we understand that the algorithm works best with detections of 15 to 20 tweets.



(a) Aggregation of detector A's detections when varying the tweet count threshold (ratio = 0.5)



(b) Aggregation of detector A's detections when varying the ratio threshold (tweets count = 50)

Figure 5.42: Aggregation of detector A's detections

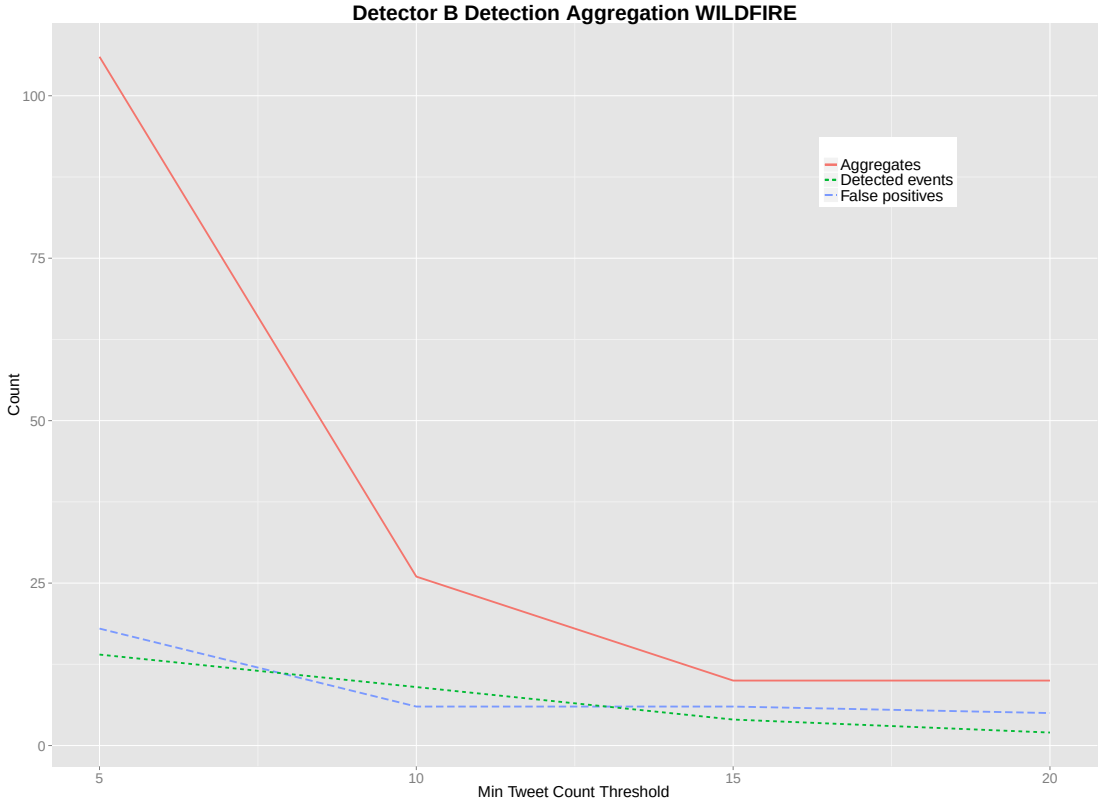


Figure 5.43: Aggregation of detector B's detections when varying the tweet count threshold

Chapter 6

Conclusion

In this master thesis we created a language independent system aiming to detect user defined events from Twitter data. The system is composed of independent modules which:

1. Read the Twitter data.
2. Classify using a keywords approach the tweets between event related and uninformative tweets.
3. Analyze the events related tweets and decide whether to trigger an event detection alert using two approaches: frequency based (focusing only on the Twitter data activity burst), and content based (focusing on the hashtags of the tweets).
4. Aggregate duplicates detections related to the same event together.

We tested our system for plane hijackings, plane crashes and wildfires events on Twitter data collected by armasuisse from October 2013 to October 2014. We chose these event types because they have different characteristics (timespan, number of real occurrences, Twitter activity) which allowed us to test our detectors on different data behavior.

Without surprise we found that the detectors were performing well mainly on popular well discussed events like plane hijackings and plane crashes for which we had high precision and recall values above 75%.
For plane hijackings, both detectors obtained shorter delays than BBC and Reuters (by up to 10 hours), the content based detector being the fastest (by up to 1.5 hour shorter than the frequency based detector). The obtained detection delays range from 1 to 10 hours.
For plane crashes, Reuters was faster than the detectors (by up to 2 hours) but BBC slower (by up to 3 hours). The frequency based detector was faster in 50% of the cases than the content based detector by up to 4 hours otherwise they had similar delays. The detection delays range from 1 to 18 hours.

On the other hand, for wildfires which have much more occurrences and noisy Twitter activity, both detectors were able to attain only 60% recall but the content based detector had poor precision 45% while the frequency based detector reached 80%. Both detectors were faster than BBC and Reuters by up to 60 hours while obtaining detection delays ranging from 10 to 70 hours. When comparing the obtained delays, the frequency based detector was faster in 50% of the cases than the content based detector by up to 15 hours otherwise they had similar delays.

Overall we achieved our goal of being able to detect real-world events faster than traditional media. Reuters still beats us for plane crashes but there are still many possible ideas to improve the detectors as we explain in chapter 7.

The obvious shortcoming of these Twitter based detectors is that they depend on the Twitter activity resulting from an event to detect it. Assuming that isn't an issue, both these detectors have the advantage of being unsupervised, language independent and to work on any type of

events once the classification keywords are chosen.

The frequency based detector is limited by the fact that it cannot detect concurrent events of the same type because it doesn't pay attention to the content of the tweets. For events which happen relatively closely in time, the minimum time limit between two events occurrences threshold can be manipulated to mitigate the problem. Ignoring tweets content is also a shortcoming because we ignore a lot of information like the geographical location, the user information, etc. This is a price we pay because we want the detector to be also able to track general events and concepts described by the set of the classification keywords. For example we can specify keywords about a country and get alert when its related Twitter activity rises suddenly which can hints that something is going to happen. With the content detector, we would probably get an alert once the event has happened if its entities are clear enough for the detector to recognize them.

The concept of the content based detector is similar to how a human would act in the sense that when it reads event related tweets, it first tries to find a few other tweets (3 to 5 tweets) talking about the same entities to confirm the information before deciding that an event is really happening. In its current implementation the detector depends on hashtags which is its main limitation. Not all relevant tweets uses "good"/any hashtags and we miss many of the first tweets before an event becomes popular. As a result, the content based detector isn't always faster than the frequency based detector. However as the content based detector requires few tweets (5 to 10 tweets) to trigger an alert, it is better suited to discover smaller scale events as long as the users mention clearly enough the related entities.

Chapter 7

Future Work

Using this work as a basis, there are many ways to improve the obtained performances; we present a few ideas in the following sections.
Once satisfying performances are obtained, with section 1.2 Motivation in mind, the next steps would be to start studying more complex concepts like crowd interests or crowd agitation and elaborate models which break down these concepts into parameters to track like a set of event types to monitor, activity indicators, sentiment analysis of tweets, etc. The methods presented in this work would then be used to obtain parts of these models' inputs.

7.1 Improving the Classification

Currently the classification relies on manually chosen keywords but this process should be automatized. An idea is to use dynamic query expansion (DQE, already used in the EMBERS[1] project for forecasting civil unrest) to automatically generate the keywords best representing the event types of interest. The algorithm would start with manually chosen keywords, extract all relevant tweets containing these keywords from the available data, compute the top most popular keywords among the obtained tweets and repeat the process until convergence. Then the obtained classification data can then be correlated like we did in section 5.1 to check if we obtain better results.

It would be interesting also to dynamically generate related keywords for each events and track their related tweets directly. Then it would be easier to separate tweets for concurrent events and aggregate correctly duplicate detections. The overall precision would be improved because the number of unrelated tweets would be much lower and this would allow to generate better summaries of the event evolution.

Further analysis should be conducted to be able to separate with more precision event related tweets from unrelated tweets (e.g. viral jokes tweets). We have seen in section 5.2 that the retweet count is an indicator, but there are other possibilities like the favorite count, evaluating the credibility of the author, the sentiment score, etc.

It is possible also to improve the precision of the classification by selecting trusted source for the tweets for example embassies, well known users, media reporters, etc.

7.2 Improving the Detectors

The frequency based detector currently doesn't use the content of the tweets. Changing this can make it possible to detect concurrent events and help further reduce the number of tweets required to trigger an alert thus reducing the obtained delays. We could use for example the geographical location of the tweet/user or the extracted entities from the text of the tweets, etc.

[1] http://www.vbi.vt.edu/ndssl/research/research-detail/EMBERS-Early-Model-Based-Event-Detection-using-Surrogates

The idea would be add some grouping mechanism on top of the time window approach i.e. combining the ideas of the two detectors.
It is important not to make the detector too specific otherwise it looses its main advantage of being able to monitor the activity of more general concepts defined by the classification keywords and raise alerts when there is a change in the activity. This track would be interesting to further develop for example by making the detector able to recognize patterns in the evolution of the Twitter activity.

We don't think that the content based detector requires conceptual changes because as we explain in chapter 6 it is very close to the how a human would recognize an event occurrence with Twitter. However the extraction process of the tweet core entities can be still greatly improved. Currently it uses the hashtags which results in many tweets being ignored even though they are classified as event related tweets. An idea would be to develop natural language processing techniques adapted for tweets to extract their entities but this question has to be further explored. With better entity extraction methods the content based detector will become more robust and faster.

7.3 Improving the Aggregation

There is a lot of room for improving the aggregation of the detections. The detections are coming in a streaming fashion but that doesn't mean that we are restricted streaming algorithm. It is possible for example to apply clustering algorithms after waiting for a sufficient number of detections. For the number of centers, we can also use assumptions like deciding on an arbitrary limit on the possible number of clusters in one day.

Similarly to the other methods, the current algorithm doesn't use any features of the tweets. For small scale events like wildfires, the geographical location of the tweets/users could help discriminate detections between each others. We could also study the authors user network and measure their similarity to come up with a probability that the two detections are related.

Appendix A

Real World Events List

A.1 Events List

Event Type	Name	Date	Time UTC
PLANE HIJACKING	Flight 751	2014-02-07	14:25
PLANE HIJACKING	Flight 702	2014-02-17	01:30
PLANE HIJACKING	Flight 370	2014-03-15	06:15
PLANE HIJACKING	Flight 41	2014-04-25	06:00
PLANE HIJACKING	Flight EK-612	2014-06-23	03:30
PLANE CRASH	Flight 361	2013-10-03	08:00
PLANE CRASH	Flight 301	2013-10-16	08:55
PLANE CRASH	Flight 363	2013-11-17	15:20
PLANE CRASH	Flight 470	2013-11-29	11:30
PLANE CRASH	Cessna 208	2013-12-12	01:15
PLANE CRASH	Bombardier 600	2014-01-05	19:20
PLANE CRASH	C-130 Hercules	2014-02-11	10:38
PLANE CRASH	Flight 183	2014-02-16	07:28
PLANE CRASH	Flight 370	2014-03-07	17:19
PLANE CRASH	Philadelphia	2014-03-13	23:05
PLANE CRASH	Hanscom Field	2014-05-31	10:30
PLANE CRASH	Flight 17	2014-07-17	13:19
PLANE CRASH	Flight 222	2014-07-23	11:06
PLANE CRASH	Flight 5017	2014-07-24	01:47
PLANE CRASH	Flight 5915	2014-08-10	05:48
PLANE CRASH	Brazil candidate	2014-08-13	13:00
PLANE CRASH	Grand Manan	2014-08-16	09:00
PLANE CRASH	TBM 900	2014-09-05	19:15
WILDFIRE	NS Wales	2013-10-13	03:30
WILDFIRE	Greater Blue	2013-10-16	03:30
WILDFIRE	Big Sur	2013-12-15	21:30
WILDFIRE	Colby	2014-01-16	22:00
WILDFIRE	Arkansas	2014-01-26	19:30
WILDFIRE	Boston	2014-03-26	19:42
WILDFIRE	Valparaiso	2014-04-12	19:40
WILDFIRE	Etiwanda	2014-04-30	21:30
WILDFIRE	Oklahoma	2014-05-04	22:30
WILDFIRE	Poinsettia	2014-05-13	18:40
WILDFIRE	Two Bulls	2014-06-07	21:30
WILDFIRE	Canada	2014-07-03	19:30
WILDFIRE	Washington	2014-07-14	18:30
WILDFIRE	Yosemite	2014-07-19	21:30
WILDFIRE	Eastbourne Pier	2014-07-30	15:00
WILDFIRE	Vastmanland	2014-07-31	12:30
WILDFIRE	Yellow Point	2014-09-05	21:30
WILDFIRE	Nottingham	2014-09-12	22:00
WILDFIRE	King	2014-09-13	21:30
WILDFIRE	Boles	2014-09-15	21:30

Table A.1: Ground truth of events occurrences

A.2 Source

Event Type	Name	Source
PLANE HIJACKING	Flight 751	http://heavy.com/news/2014/02/peguses-plane-hijacking-ukraine-sochi-turkey/
PLANE HIJACKING	Flight 702	http://en.wikipedia.org/wiki/Ethiopian_Airlines_Flight_702
PLANE HIJACKING	Flight 370	https://www.youtube.com/watch?v=qvvud1NMjUg
PLANE HIJACKING	Flight 41	http://www.ibtimes.co.uk/virgin-australia-flight-hijack-scare-boeing-737-lands-bali-drunk-passenger-mayhem-1446011
PLANE HIJACKING	Flight EK-612	http://arynews.tv/en/emirates-airlines-threaten-plane-hijacking-case-against-pml-n-govt/
PLANE CRASH	Flight 361	http://www.bellanaija.com/2013/10/03/updates-on-the-aa-plane-crash-in-lagos-agagus-casket-recovered-engine-lost-after-take-off-9-bodies-recovered-more/
PLANE CRASH	Flight 301	http://en.wikipedia.org/wiki/Lao_Airlines_Flight_301
PLANE CRASH	Flight 363	http://en.wikipedia.org/wiki/Tatarstan_Airlines_Flight_363
PLANE CRASH	Flight 470	http://en.wikipedia.org/wiki/LAM_Mozambique_Airlines_Flight_470
PLANE CRASH	Cessna 208	http://www.dailystormer.com/woman-who-confirmed-obamas-birth-certificate-dies-in-plane-crash/
PLANE CRASH	Bombardier 600	http://aviation-safety.net/database/record.php?id=20140105-1&lang=fr
PLANE CRASH	C-130 Hercules	http://aviation-safety.net/database/record.php?id=20140211-0&lang=fr
PLANE CRASH	Flight 183	http://en.wikipedia.org/wiki/Nepal_Airlines_Flight_183
PLANE CRASH	Flight 370	http://en.wikipedia.org/wiki/Malaysia_Airlines_Flight_370
PLANE CRASH	Philadelphia	http://www.dailyheraldtribune.com/2014/03/13/us-airways-flight-crash-lands-upon-take-off-at-philadelphia-international-aiport
PLANE CRASH	Flight 370 Crash News	http://www.smh.com.au/world/malaysia-airlines-mh370-hopes-dashed-as-pm-confirms-plane-crash-20140324-35e3d.html
PLANE CRASH	Hanscom Field	http://www.kathrynsreport.com/2014/05/front-range-airport-kftg-denver-colorado.html
PLANE CRASH	Flight 17	http://en.wikipedia.org/wiki/Malaysia_Airlines_Flight_17
PLANE CRASH	Flight 222	http://en.wikipedia.org/wiki/TransAsia_Airways_Flight_222
PLANE CRASH	Flight 5017	http://en.wikipedia.org/wiki/Air_Alg%C3%A9rie_Flight_5017
PLANE CRASH	Flight 5915	http://en.wikipedia.org/wiki/Sepahan_Airlines_Flight_5915
PLANE CRASH	Brazil candidate	http://en.wikipedia.org/wiki/2014_Santos_Cessna_Citation_accident
PLANE CRASH	Grand Manan	http://globalnews.ca/news/1511274/two-dead-in-new-brunswick-air-ambulance-plane-crash/
PLANE CRASH	TBM 900	http://news.nationalpost.com/2014/09/05/new-york-ceo-and-wife-killed-in-crash-off-jamaica-after-plane-lost-contact-with-air-traffic-control/
WILDFIRE	NS Wales	http://en.wikipedia.org/wiki/2013_New_South_Wales_bushfires
WILDFIRE	Greater Blue	http://en.wikipedia.org/wiki/2013_New_South_Wales_bushfires
WILDFIRE	Big Sur	http://www.ksbw.com/news/central-california/monterey/cause-of-big-sur-pfeiffer-fire-released-by-national-forest-service/30039726
WILDFIRE	Colby	http://www.nbclosangeles.com/news/local/Brush-Fire-Glendora-Angeles-National-Forest-Wildfire-240493431.html
WILDFIRE	Arkansas	http://www.arkansasmatters.com/story/200-acre-garland-county-wildfire-continues-to-burn/d/story/LOQ0AqtsVEep14xRjoFwSA
WILDFIRE	Boston	http://en.wikipedia.org/wiki/2014_Boston_Brownstone_Fire
WILDFIRE	Valparaiso	http://en.wikipedia.org/wiki/Great_Fire_of_Valpara%C3%ADso
WILDFIRE	Etiwanda	http://en.wikipedia.org/wiki/2014_California_wildfires
WILDFIRE	Oklahoma	http://cir.ca/news/logan-county-oklahoma-wildfire
WILDFIRE	Poinsettia	http://en.wikipedia.org/wiki/May_2014_San_Diego_County_wildfires
WILDFIRE	Two Bulls	http://inciweb.nwcg.gov/incident/article/3889/22021/
WILDFIRE	Canada	http://www.cbc.ca/news/canada/north/n-w-t-experiencing-one-of-its-worst-fire-seasons-ever-1.2695066
WILDFIRE	Washington	http://en.wikipedia.org/wiki/2014_Washington_state_wildfires
WILDFIRE	Yosemite	http://en.wikipedia.org/wiki/Meadow_Fire
WILDFIRE	Eastbourne Pier	http://www.bbc.com/news/uk-28572205
WILDFIRE	Vastmanland	http://en.wikipedia.org/wiki/2014_V%C3%A4stmanland_wildfire
WILDFIRE	Yellow Point	http://en.wikipedia.org/wiki/Yellow_Point_Fire
WILDFIRE	Nottingham	http://www.itv.com/news/central/update/2014-09-12/video-from-scene-of-nottingham-university-fire/
WILDFIRE	King	http://en.wikipedia.org/wiki/King_Fire
WILDFIRE	Boles	http://en.wikipedia.org/wiki/Boles_Fire

Table A.2: Source of information

Appendix B

Traditional Media Detection

B.1 First Articles/Reports Time and Date

Event Type	Name	BBC Date	BBC Time	Reuters Date	Reuters Time
PLANE HIJACKING	Flight 751	NA	NA	2014-02-07	20:57
PLANE HIJACKING	Flight 702	2014-02-17	21:51	2014-02-17	16:32
PLANE HIJACKING	Flight 370	2014-03-15	18:38	2014-03-15	18:44
PLANE HIJACKING	Flight 41	2014-04-25	09:48	2014-04-25	10:57
PLANE HIJACKING	Flight EK-612	2014-06-23	14:20	2014-06-23	04:39
PLANE CRASH	Flight 361	2013-10-03	13:18	2013-10-03	17:26
PLANE CRASH	Flight 301	2013-10-16	17:43	2013-10-16	17:21
PLANE CRASH	Flight 363	2013-11-18	05:53	2013-11-17	19:16
PLANE CRASH	Flight 470	2013-11-29	21:41	2013-11-30	08:56
PLANE CRASH	Cessna 208	NA	NA	2013-12-12	21:40
PLANE CRASH	Bombardier 600	2014-01-05	22:17	2014-01-05	19:40
PLANE CRASH	C-130 Hercules	2014-02-11	21:22	2014-02-11	19:25
PLANE CRASH	Flight 183	2014-02-17	11:54	2014-02-16	13:42
PLANE CRASH	Flight 370	2014-03-08	05:40	2014-03-08	07:30
PLANE CRASH	Philadelphia	2014-03-14	02:22	2014-03-14	10:17
PLANE CRASH	Hanscom Field	2014-06-01	14:39	2014-06-01	09:55
PLANE CRASH	Flight 17	2014-07-18	11:37	2014-07-17	15:21
PLANE CRASH	Flight 222	2014-07-23	16:56	2014-07-23	13:24
PLANE CRASH	Flight 5017	2014-07-25	13:43	2014-07-24	20:48
PLANE CRASH	Flight 5915	2014-08-10	14:16	2014-08-10	07:12
PLANE CRASH	Brazil candidate	2014-08-13	19:53	2014-08-13	15:24
PLANE CRASH	Grand Manan	NA	NA	NA	NA
PLANE CRASH	TBM 900	2014-09-06	10:19	2014-09-05	23:00
WILDFIRE	NS Wales	2013-10-18	11:26	2013-10-23	00:40
WILDFIRE	Greater Blue	2013-10-21	10:23	2013-10-17	07:02
WILDFIRE	Big Sur	2013-12-18	16:23	2013-12-17	20:49
WILDFIRE	Colby	2014-01-17	02:44	2014-01-16	22:00
WILDFIRE	Arkansas	NA	NA	NA	NA
WILDFIRE	Boston	2014-03-27	08:04	NA	NA
WILDFIRE	Valparaiso	2014-04-14	01:22	2014-04-13	19:49
WILDFIRE	Etiwanda	NA	NA	2014-04-30	22:28
WILDFIRE	Oklahoma	2014-05-06	00:40	2014-05-06	19:17
WILDFIRE	Poinsettia	2014-05-15	22:27	2014-05-15	06:26
WILDFIRE	Two Bulls	NA	NA	2014-06-10	06:30
WILDFIRE	Canada	NA	NA	2014-07-18	10:29
WILDFIRE	Washington	2014-07-18	21:19	2014-07-21	19:17
WILDFIRE	Yosemite	NA	NA	2014-08-26	17:55
WILDFIRE	Eastbourne Pier	2014-07-30	19:54	NA	NA
WILDFIRE	Vastmanland	2014-08-06	17:09	2014-08-05	09:25
WILDFIRE	Yellow Point	NA	NA	NA	NA
WILDFIRE	Nottingham	2014-09-13	02:17	NA	NA
WILDFIRE	King	2014-09-18	14:29	2014-09-15	19:31
WILDFIRE	Boles	NA	NA	2014-09-16	09:20

Table B.1: Traditional media event's latest update timestamps obtained from their websites

B.2 BBC Articles Source

Event Type	Name	BBC Article
PLANE HIJACKING	Flight 751	NA
PLANE HIJACKING	Flight 702	http://www.bbc.com/news/world-europe-26222674
PLANE HIJACKING	Flight 370	http://www.bbc.com/news/world-asia-26591056
PLANE HIJACKING	Flight 41	http://www.bbc.com/news/world-asia-27154139
PLANE HIJACKING	Flight EK-612	http://www.bbc.com/news/world-asia-27971351
PLANE CRASH	Flight 361	http://www.bbc.com/news/world-africa-24381066
PLANE CRASH	Flight 301	http://www.bbc.com/news/world-asia-24554128
PLANE CRASH	Flight 363	http://www.bbc.com/news/world-europe-24980055
PLANE CRASH	Flight 470	http://www.bbc.com/news/world-africa-25162278
PLANE CRASH	Cessna 208	NA
PLANE CRASH	Bombardier 600	http://www.bbc.com/news/world-us-canada-25617026
PLANE CRASH	C-130 Hercules	http://www.bbc.com/news/world-africa-26138101
PLANE CRASH	Flight 183	http://www.bbc.com/news/world-asia-26222528
PLANE CRASH	Flight 370	http://www.bbc.com/news/world-asia-26492454
PLANE CRASH	Philadelphia	http://www.bbc.com/news/world-us-canada-26571993
PLANE CRASH	Flight 370 Crash News	NA
PLANE CRASH	Hanscom Field	http://www.bbc.com/news/world-us-canada-27656261
PLANE CRASH	Flight 17	http://www.bbc.com/news/business-28356745
PLANE CRASH	Flight 222	http://www.bbc.com/news/world-asia-28448763
PLANE CRASH	Flight 5017	http://www.bbc.com/news/world-africa-28485503
PLANE CRASH	Flight 5915	http://www.bbc.com/news/world-middle-east-28730717
PLANE CRASH	Brazil candidate	http://www.bbc.com/news/world-latin-america-28775900
PLANE CRASH	Grand Manan	NA
PLANE CRASH	TBM 900	http://www.bbc.com/news/world-us-canada-29082666
WILDFIRE	NS Wales	http://www.bbc.com/news/world-asia-24575830
WILDFIRE	Greater Blue	http://www.bbc.com/news/world-asia-24579099
WILDFIRE	Big Sur	http://www.bbc.co.uk/weather/features/25437220
WILDFIRE	Colby	http://www.bbc.com/news/world-us-canada-25769904
WILDFIRE	Arkansas	NA
WILDFIRE	Boston	http://www.bbc.com/news/world-us-canada-26762938
WILDFIRE	Valparaiso	http://www.bbc.com/news/world-latin-america-27007884
WILDFIRE	Etiwanda	NA
WILDFIRE	Oklahoma	http://www.bbc.com/news/world-us-canada-27284939
WILDFIRE	Poinsettia	http://www.bbc.com/news/world-us-canada-27420698
WILDFIRE	Two Bulls	NA
WILDFIRE	Canada	NA
WILDFIRE	Washington	http://www.bbc.com/news/world-us-canada-28380964
WILDFIRE	Yosemite	NA
WILDFIRE	Eastbourne Pier	http://www.bbc.com/news/uk-england-sussex-28573659
WILDFIRE	Vastmanland	http://www.bbc.co.uk/weather/feeds/28678478
WILDFIRE	Yellow Point	NA
WILDFIRE	Nottingham	http://www.bbc.com/news/uk-england-nottinghamshire-29186169
WILDFIRE	King	http://www.bbc.com/news/world-us-canada-29257417
WILDFIRE	Boles	NA

Table B.2: Source of information

B.3 Reuters Reports Source

Event Type	Name	Reuters Report
PLANE HIJACKING	Flight 751	http://www.reuters.com/article/2014/02/07/us-turkey-plane-idUSBREA161E720140207
PLANE HIJACKING	Flight 702	http://www.reuters.com/article/2014/02/17/us-ethiopian-hijacking-idUSBREA1G07F20140217
PLANE HIJACKING	Flight 370	http://www.reuters.com/article/2014/03/15/malaysia-airlines-wrapup-10-pictures-tv-idUSL6N0MC0G920140315
PLANE HIJACKING	Flight 41	http://www.reuters.com/article/2014/04/25/australia-airplane-idUSL3N0NH2UO20140425
PLANE HIJACKING	Flight EK-612	http://www.reuters.com/article/2014/06/23/pakistan-cleric-diverted-idUSL4N0P412D20140623
PLANE CRASH	Flight 361	http://www.reuters.com/article/2013/10/03/us-nigeria-plane-idUSBRE99209T20131003
PLANE CRASH	Flight 301	http://www.reuters.com/article/2013/10/16/laos-crash-idUSL3N0I632L20131016
PLANE CRASH	Flight 363	http://www.reuters.com/article/2013/11/17/russia-crash-idUSL5N0J20MT20131117
PLANE CRASH	Flight 470	http://www.reuters.com/article/2013/11/30/mozambique-flight-crash-idUSL5N0JF05420131130
PLANE CRASH	Cessna 208	http://www.reuters.com/article/2013/12/12/usa-hawaii-crash-idUSL1N0JR1P320131212
PLANE CRASH	Bombardier 600	http://www.reuters.com/article/2014/01/05/usa-aspen-crash-idUSL2N0KF0F620140105
PLANE CRASH	C-130 Hercules	http://www.reuters.com/article/2014/02/11/algeria-crash-idUSL5N0LG40C20140211
PLANE CRASH	Flight 183	http://www.reuters.com/article/2014/02/16/nepal-plane-idUSL3N0LL09F20140216
PLANE CRASH	Flight 370	http://www.reuters.com/article/2014/03/08/malaysia-airlines-profile-idUSL3N0M506V20140308
PLANE CRASH	Philadelphia	http://www.reuters.com/article/2014/03/14/press-digest-canada-idUSL3N0MB1ZO20140314
PLANE CRASH	Flight 370 Crash News	NA
PLANE CRASH	Hanscom Field	http://www.reuters.com/article/2014/06/01/us-usa-massachusetts-crash-idUSKBN0EC1BF20140601
PLANE CRASH	Flight 17	http://www.reuters.com/article/idUSL4N0PS4IK20140717
PLANE CRASH	Flight 222	http://www.reuters.com/article/2014/07/23/taiwan-crash-idUSL4N0PY3ZD20140723
PLANE CRASH	Flight 5017	http://www.reuters.com/article/2014/07/24/algeria-flight-idUSL6N0PZ3AL20140724
PLANE CRASH	Flight 5915	http://www.reuters.com/article/2014/08/10/us-iran-airplane-crash-idUSKBN0GA04520140810
PLANE CRASH	Brazil candidate	http://www.reuters.com/article/2014/08/13/brazil-crash-idUSE6N0QJ00020140813
PLANE CRASH	Grand Manan	NA
PLANE CRASH	TBM 900	http://www.reuters.com/article/2014/09/05/usa-airplane-cuba-idUSL1N0R61OY20140905
WILDFIRE	NS Wales	http://www.reuters.com/article/2013/10/23/australia-fires-idUSL3N0IC4WA20131023
WILDFIRE	Greater Blue	http://www.reuters.com/article/2013/10/17/us-australia-fires-idUSBRE99G08E20131017
WILDFIRE	Big Sur	http://www.reuters.com/article/2013/12/17/us-usa-california-fire-idUSBRE9BG04H20131217
WILDFIRE	Colby	http://www.reuters.com/article/2014/01/16/us-usa-fire-california-idUSBREA0F1EX20140116
WILDFIRE	Arkansas	NA
WILDFIRE	Boston	NA
WILDFIRE	Valparaiso	http://www.reuters.com/article/idUSL2N0N504W20140413
WILDFIRE	Etiwanda	http://www.reuters.com/article/2014/04/30/us-usa-wildfire-california-idUSBREA3T14D20140430
WILDFIRE	Oklahoma	http://www.reuters.com/article/idUSBREA450SB20140506
WILDFIRE	Poinsettia	http://www.reuters.com/article/2014/05/15/usa-wildfires-california-idUSL1N0O01MW20140515
WILDFIRE	Two Bulls	http://www.reuters.com/article/2014/06/10/us-usa-oregon-wildfires-idUSKBN0EL0EH20140610
WILDFIRE	Canada	http://www.reuters.com/article/2014/07/18/press-digest-canada-idUSL4N0PT32Q20140718
WILDFIRE	Washington	http://www.reuters.com/article/2014/07/21/us-usa-wildfires-idUSKBN0FQ1XK20140721
WILDFIRE	Yosemite	http://www.reuters.com/article/2013/08/26/idUSL2N0GR0SP20130826
WILDFIRE	Eastbourne Pier	NA
WILDFIRE	Vastmanland	http://www.reuters.com/article/2014/08/05/us-sweden-fire-idUSKBN0G50U420140805
WILDFIRE	Yellow Point	NA
WILDFIRE	Nottingham	NA
WILDFIRE	King	http://www.reuters.com/article/2014/09/15/us-usa-california-wildfires-idUSKBN0HA20820140915
WILDFIRE	Boles	http://www.reuters.com/article/2014/09/16/us-usa-california-wildfires-idUSKBN0HA20820140916

Table B.3: Source of information

Appendix C

Avro Schema

```
{
  "type" : "record",
  "name" : "TwitterStatusUpdate",
  "namespace" :
"ch.armasuisse.twitterdemo.avro.v2",
  "fields" : [ {
    "name" : "schemaVersion",
    "type" : [ "string", "null" ],
    "default" : "2.0"
  }, {
    "name" : "targetId",
    "type" : [ "string", "null" ],
    "default" : "ALL"
  }, {
    "name" : "isRetweetOriginal",
    "type" : "boolean",
    "default" : false
  }, {
    "name" : "contributors",
    "type" : {
      "type" : "array",
      "items" : "long"
    }
  }, {
    "name" : "createdAt",
    "type" : [ "string", "null" ]
  }, {
    "name" : "createdAtAsLong",
    "type" : "long",
    "default" : 0
  }, {
    "name" : "currentUserRetweetId",
    "type" : "long"
  }, {
    "name" : "id",
    "type" : "long",
    "default" : -1
  }, {
    "name" : "inReplyToScreenName",
    "type" : [ "string", "null" ]
  }, {
    "name" : "inReplyToStatusId",
    "type" : "long"
  }, {
    "name" : "inReplyToUserId",
    "type" : "long"
  }, {
    "name" : "lang",
    "type" : [ "string", "null" ],
    "default" : "unknown"
  }, {
    "name" : "retweetCount",
    "type" : "long"
  }, {
    "name" : "source",
    "type" : [ "string", "null" ]
  }, {
    "name" : "text",
    "type" : [ "string", "null" ]
  }, {
    "name" : "favoriteCount",
    "type" : "int",
    "default" : -1
  }, {
    "name" : "isFavorited",
    "type" : "boolean"
  }, {
    "name" : "filterLevel",
    "type" : [ "string", "null" ],
    "default" : "unknown"
  }, {
    "name" : "isPossiblySensitive",
    "type" : "boolean"
  }, {
    "name" : "isRetweet",
    "type" : "boolean"
  }, {
    "name" : "isRetweetedByMe",
    "type" : "boolean"
  }, {
    "name" : "retweetedStatus",
    "type" : [ "null", {
      "type" : "record",
      "name" : "TwitterRetweetedStatus",
      "fields" : [ {
        "name" : "createdAt",
        "type" : [ "string", "null" ]
      }, {
        "name" : "id",
        "type" : "long"
      }, {
        "name" : "text",
        "type" : [ "string", "null" ]
      } ]
    } ],
    "default" : null
  }, {
    "name" : "isTruncated",
    "type" : "boolean"
  }, {
    "name" : "accessLevel",
    "type" : "int",
    "default" : -1
  }, {
    "name" : "coordinatesLatitude",
    "type" : "double"
  }, {
    "name" : "coordinatesLongitude",
    "type" : "double"
  }, {
    "name" : "hashtagEntities",
    "type" : {
      "type" : "array",
      "items" : {
        "type" : "record",
        "name" : "TwitterHashtagEntity",
        "fields" : [ {
          "name" : "end",
          "type" : "int"
        }, {
          "name" : "start",
          "type" : "int"
        }, {
          "name" : "text",
          "type" : [ "string", "null" ]
        } ],
        "description" : "A simpler
representation of a Twitter4j
HashtagEntity object"
      }
```

1

```
    }
  }, {
    "name" : "mediaEntities",
    "type" : {
      "type" : "array",
      "items" : {
        "type" : "record",
        "name" : "TwitterMediaEntity",
        "fields" : [ {
          "name" : "id",
          "type" : "long"
        }, {
          "name" : "mediaURL",
          "type" : [ "string", "null" ]
        }, {
          "name" : "mediaURLHttps",
          "type" : [ "string", "null" ]
        }, {
          "name" : "URL",
          "type" : [ "string", "null" ],
          "default" : "unknown"
        }, {
          "name" : "displayURL",
          "type" : [ "string", "null" ],
          "default" : "unknown"
        }, {
          "name" : "expandedURL",
          "type" : [ "string", "null" ],
          "default" : "unknown"
        }, {
          "name" : "type",
          "type" : [ "string", "null" ]
        }, {
          "name" : "end",
          "type" : "int",
          "default" : -1
        }, {
          "name" : "start",
          "type" : "int",
          "default" : -1
        } ],
        "description" : "A simpler
representation of a Twitter4j
MediaEntity object"
      }
    }
  }, {
    "name" : "urlEntities",
    "type" : {
      "type" : "array",
      "items" : {
        "type" : "record",
        "name" : "TwitterURLEntity",
        "fields" : [ {
          "name" : "displayURL",
          "type" : [ "string", "null" ]
        }, {
          "name" : "end",
          "type" : "int"
        }, {
          "name" : "start",
          "type" : "int"
        }, {
          "name" : "URL",
          "type" : [ "string", "null" ]
        }, {
          "name" : "expandedURL",
          "type" : [ "string", "null" ]
        } ],
        "description" : "A simpler
representation of a Twitter4j URLEntity
object"
      }
    }
  }, {
    "name" : "userMentionEntities",
    "type" : {
      "type" : "array",
      "items" : {
        "type" : "record",
        "name" :
"TwitterUserMentionEntity",
        "fields" : [ {
          "name" : "end",
          "type" : "int"
        }, {
          "name" : "start",
          "type" : "int"
        }, {
          "name" : "name",
          "type" : [ "string", "null" ]
        }, {
          "name" : "screenName",
          "type" : [ "string", "null" ]
        }, {
          "name" : "id",
          "type" : "long"
        } ],
        "description" : "A simpler
representation of a Twitter4j
UserMentionEntity object"
      }
    }
  }, {
    "name" : "symbolEntities",
    "type" : [ "null", {
      "type" : "array",
      "items" : {
        "type" : "record",
        "name" : "TwitterSymbolEntity",
        "fields" : [ {
          "name" : "end",
          "type" : "int"
        }, {
          "name" : "start",
          "type" : "int"
        }, {
          "name" : "text",
          "type" : [ "string", "null" ]
        } ],
        "description" : "A simpler
representation of a Twitter4j
SymbolEntity object"
      }
    } ],
    "default" : null
  }, {
    "name" : "Place",
    "type" : {
      "type" : "record",
```

```
      "name" : "TwitterPlace",
      "fields" : [ {
        "name" : "URL",
        "type" : [ "string", "null" ]
      }, {
        "name" : "streetAddress",
        "type" : [ "string", "null" ]
      }, {
        "name" : "placeType",
        "type" : [ "string", "null" ]
      }, {
        "name" : "name",
        "type" : [ "string", "null" ]
      }, {
        "name" : "fullName",
        "type" : [ "string", "null" ]
      }, {
        "name" : "id",
        "type" : [ "string", "null" ]
      }, {
        "name" : "country",
        "type" : [ "string", "null" ]
      }, {
        "name" : "countryCode",
        "type" : [ "string", "null" ]
      }, {
        "name" : "boundingBoxType",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "boundingBox",
        "type" : {
          "type" : "array",
          "items" : {
            "type" : "record",
            "name" :
"TwitterGeoLocation",
            "fields" : [ {
              "name" : "latitude",
              "type" : "double"
            }, {
              "name" : "longitude",
              "type" : "double"
            } ],
            "description" : "A simpler
representation of a Twitter4j
GeoLocation object"
          }
        }
      } ]
    },
    "description" : "A simpler
representation of a Twitter4j Place
object"
  }, {
    "name" : "User",
    "type" : {
      "type" : "record",
      "name" : "TwitterUser",
      "fields" : [ {
        "name" : "id",
        "type" : "long",
        "default" : -1
      }, {
        "name" : "description",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "userURLEntity",
        "type" : [ "null", {
          "type" : "record",
          "name" : "UserURLEntity",
          "fields" : [ {
            "name" : "end",
            "type" : "int"
          }, {
            "name" : "start",
            "type" : "int"
          }, {
            "name" : "URL",
            "type" : [ "string", "null"
]
          }, {
            "name" : "displayURL",
            "type" : [ "string", "null"
]
          }, {
            "name" : "expandedURL",
            "type" : [ "string", "null"
]
          }, {
            "name" : "text",
            "type" : [ "string", "null"
]
          } ],
          "description" : "A simpler
representation of a Twitter4j UrlEntity
object"
        } ],
        "default" : null
      }, {
        "name" : "favoriteCount",
        "type" : "int",
        "default" : -1
      }, {
        "name" : "isFollowRequestSent",
        "type" : "boolean",
        "default" : false
      }, {
        "name" : "screenName",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "createdAt",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "followersCount",
        "type" : "int",
        "default" : -1
      }, {
        "name" : "friendsCount",
        "type" : "int",
        "default" : -1
      }, {
        "name" : "isGeoEnabled",
        "type" : "boolean",
        "default" : false
      }, {
        "name" : "listedCount",
```

```
        "type" : "int",
        "default" : -1
      }, {
        "name" : "statusesCount",
        "type" : "int",
        "default" : -1
      }, {
        "name" : "isVerified",
        "type" : "boolean",
        "default" : false
      }, {
        "name" : "isTranslator",
        "type" : "boolean",
        "default" : false
      }, {
        "name" : "lang",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "location",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "name",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" :
"profileBackgroundColor",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" :
"profileBackgroundImageURL",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" :
"profileBackgroundImageURLHttps",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" :
"isProfileBackgroundTiled",
        "type" : [ "boolean", "null" ],
        "default" : false
      }, {
        "name" : "profileBannerURL",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "profileImageURL",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "profileImageURLHttps",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "profileLinkColor",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" :
"profileSidebarBorderColor",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" :
"profileSidebarFillColor",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "profileTextColor",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" :
"isProfileUseBackgroundImage",
        "type" : [ "boolean", "null" ],
        "default" : false
      }, {
        "name" : "isProtected",
        "type" : "boolean",
        "default" : false
      }, {
        "name" : "isShowAllInlineMedia",
        "type" : "boolean",
        "default" : false
      }, {
        "name" : "timeZone",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "URL",
        "type" : [ "string", "null" ],
        "default" : "unknown"
      }, {
        "name" : "utcOffset",
        "type" : "int",
        "default" : -1
      } ]
    },
    "description" : "A simpler
representation of a Twitter4j User
object"
  } ],
  "description" : "A simpler
representation of a Twitter4j Status
object"
}
```

Appendix D

Detectors Precision and Recall Data

D.1 Frequency Based Detector

D.1.1 Min Tweet Count Threshold

Event Type	Min Tweet Count	Precision	Recall	False Positives	Detected Events
PLANE HIJACKING	50	71.42	100	2	5
PLANE HIJACKING	60	83.33	100	1	5
PLANE HIJACKING	75	80	80	1	4
PLANE HIJACKING	100	100	80	0	4
PLANE CRASH	50	100	88.88	0	16
PLANE CRASH	60	100	72.22	0	13
PLANE CRASH	75	100	66.66	0	12
PLANE CRASH	100	100	61.11	0	11
WILDFIRE	50	81.25	65	3	13
WILDFIRE	60	76.92	50	3	10
WILDFIRE	75	88.88	40	1	8
WILDFIRE	100	83.33	25	1	5

Table D.1: Detector A results obtained when varying the minimum tweet count threshold

D.1.2 Ratio Threshold

Event Type	Ratio	Precision	Recall	False Positives	Detected Events
PLANE HIJACKING	0.1	83.33	100	1	5
PLANE HIJACKING	0.3	71.42	100	2	5
PLANE HIJACKING	0.5	71.42	100	2	5
PLANE HIJACKING	0.7	71.42	100	2	5
PLANE CRASH	0.1	92.85	72.22	1	13
PLANE CRASH	0.3	100	77.77	0	14
PLANE CRASH	0.5	100	88.88	0	16
PLANE CRASH	0.7	100	88.88	0	16
WILDFIRE	0.1	85.71	30	1	6
WILDFIRE	0.3	92.3	60	1	12
WILDFIRE	0.5	81.25	65	3	13
WILDFIRE	0.7	82.35	70	3	14

Table D.2: Detector A results obtained when varying the ratio threshold

D.2 Content Based Detector

D.2.1 Cluster Min Tweet Count Threshold

Event Type	Min Tweet Count	Precision	Recall	False Positives	Detected Events
PLANE HIJACKING	5	71.42	100	2	5
PLANE HIJACKING	10	83.33	100	1	5
PLANE HIJACKING	15	100	100	0	5
PLANE HIJACKING	20	100	80	0	4
PLANE CRASH	5	84.21	88.88	3	16
PLANE CRASH	10	100	61.11	0	11
PLANE CRASH	15	100	50	0	9
PLANE CRASH	20	100	50	0	9
WILDFIRE	5	43.75	70	18	14
WILDFIRE	10	60	45	6	9
WILDFIRE	15	40	20	6	4
WILDFIRE	20	28.57	10	5	2

Table D.3: Detector B results obtained when varying the cluster minimum tweet count threshold

Appendix E

Master Thesis Schedule

E.1 Original Schedule Plan

Early Detection of Real-world Events with Twitter Schedule		
Month	**Week**	**Tasks**
October	06.10.2014	. Discover Twitter, literature
	13.10.2014	. Write schedule . Choose real-world events types . Read literature
	20.10.2014	. Read literature . Write related work section . Choose first method to implement
	27.10.2014	
November	03.11.2014	. Prepare avro-impala read/query engine . Implement tweets classifier
	10.11.2014	. Test/improve classifier . Write classifier description
	17.11.2014	. Implement event detector A
	24.11.2014	. Test A performance for each event type
December	01.12.2014	. Improve A separately for each event type . Write A description
	08.12.2014	. Implement event detector B
	15.12.2014	. Test B performance for each event type
	22.12.2014	----
	29.12.2014	
January	05.01.2015	. Improve B separately for each event type . Write B description
	12.01.2015	. Implement event detector C
	19.01.2015	. Test C performance for each event type
	26.01.2015	. Improve C separately for each event type . Write C description
February	02.02.2015	. Write master thesis body (abstract, introduction, data collection, references, etc)
	09.02.2015	
	16.02.2015	. Add general event detector, new ideas, …
	23.02.2015	
March	02.03.2015	. New ideas, delays …
	09.03.2015	
	16.03.2015	
	23.03.2015	. Complete master thesis
	30.03.2015	

E.2 Actual Outcome

Early Detection of Real-world Events with Twitter Schedule		
Month	**Week**	**Tasks**
October	06.10.2014	. Discover Twitter, literature
	13.10.2014	. Write schedule . Choose real-world events types . Read literature
	20.10.2014	. Read literature . Prepare related work section . Choose first method to implement
	27.10.2014	
November	03.11.2014	. Prepare avro reader (data feed module)
	10.11.2014	. Implement tweets classifier . Test/Improve classifier
	17.11.2014	
	24.11.2014	
December	01.12.2014	. Implement detector A
	08.12.2014	
	15.12.2014	
	22.12.2014	----
	29.12.2014	
January	05.01.2015	. Prepare experiments (design, ground truth) . Code refactoring
	12.01.2015	
	19.01.2015	. Test, interpret results and improve detector A for each event types
	26.01.2015	
February	02.02.2015	. Implement detector B (first attempt with n-grams, entity recognition, finally hashtags)
	09.02.2015	
	16.02.2015	. Test, interpret results and improve detector B for each event types
	23.02.2015	
March	02.03.2015	. Compare delays with traditional medias
	09.03.2015	. Prepare results for report . Write Master Thesis
	16.03.2015	
	23.03.2015	
	30.03.2015	

Appendix F

Original Task Description

Schweizerische Eidgenossenschaft
Confédération suisse
Confederazione Svizzera
Confederaziun svizra

Eidgenössisches Departement für Verteidigung, Bevölkerungsschutz und Sport VBS

armasuisse
Wissenschaft und Technologie W+T

Master Thesis Task Assignment for
Amruthalingam Ludovic (D-INFK)

Early Detection of Real-world Events with Twitter

Main advisor:	Dr. Vincent Lenders (armasuisse)
Advisor ETH	David Gugelmann (ETH Zürich)
Supervisor	Prof. Dr. B. Plattner (ETH Zürich)
Start Date:	October 6th, 2014
End Date:	April 5th, 2015

1 Background

Open source intelligence (OSINT) is currently experiencing a big paradigm shift. The classical OSINT sources were mainly newspapers, magazines, TV, radio etc,. With these sources, the set of editors and publishers was closed and limited. Today, new sources have appeared in the social media sector and in the Web that allow anybody to publish and contribute content. This user-generated content has changed OSINT quite significantly. E.g., Wikipedia, the open encyclopedia maintained by volunteers has established itself as the prime source of knowledge in the Internet. Social networks like Twitter, Facebook or Youtube have become so popular that many persons and organizations are using these sources as one of their primary source for news.

2 Thesis Goal

The goal of this thesis is to evaluate the feasibility to use Twitter as an early warning system for real-world events. Events could for example be natural disasters, accidents, revolutions, hijacking, conflicts and wars, cyber incidents and attacks, etc. Various studies have shown that Twitter was often the first source of information to mention the occurrence of particular events. The reports on Twitter were up to hours or even days before the traditional media reported about it. This makes Twitter an attractive source of information for early detection. The problem however is that the source is very noisy with a lot of Tweets about non-event related topics.

To assess whether Twitter could be turned into an early warning system, the student shall therefore conduct a systematic study using a one-year long historical data set of Tweets. The goal is to understand how signals from Twitter could be exploited to devise a real-time event detection system. Ideally, we want to alerted about new events before the classical media sources report about it by exploiting the crowd of active users.

3 Tasks

The goal of this thesis is to explore how well Twitter data can be used for early detection of real-world events.

The detailed tasks of the thesis are:

1. Study the content publishing, retweeting and following concepts in Twitter.

2. Analyze the type of data and meta-data that can be obtained openly from Twitter through the streaming and REST APIs.
3. Review the open literature on event detection with Twitter.
4. Select together with your advisors a set of events from different categories (e.g. using [1] as categorization). These events will be used to evaluate early detection later on.
5. Develop a classifier using machine learning techniques to automatically classify Tweets to the selected categories of events. Your classifier shall integrate various types of features including temporal, content, geo-spacial and social aspects of the Tweets and/or combinations thereof.
6. Evaluate the performance of your classifier using sample Tweets for each event category. Analyze the importance of individual features and how these features are correlated.
7. Develop and implement different event detection models based on your classifier. Your models could be of statistical or probabilistic nature. Note that your models should be developed with the aim of early detection in the context of real-time warning. That means that we are interested in finding models that best reflect the beginning or pre-phase of events and not the actual event presence or post-phase.
8. Evaluate the false positive and false negative rates of the implemented models for the different event categories. Study the differences in detection rates for the different event types and explore which model works best for which category of events.
9. Evaluate the delay between the event occurrences and detection times using your models.
10. Compare these delays with the times between the event occurrences and the first reports about these events in the traditional media (e.g. CNN, BBC, etc)

4 Deliverables

- At the end of the second week, a detailed time schedule of the thesis must be given and discussed with the main advisors.
- At the end of the second month, a short discussion of 15 minutes with the supervisor and the advisors will take place. The student has to talk about the major aspects of the ongoing work using slides.
- At the end of month four, another meeting with the supervisor will take place. At this point, the student should already have a preliminary version of the written report or at least a table of content to hand in to the supervisor. This preliminary version should be brought along to the short discussion.
- At the end of the thesis, a presentation of 15 minutes must be given at armasuisse and at ETH (in English). The presentations should give an overview as well as the most important details of the work. If possible, a demonstrator should be presented (offline after the talk).
- The final report should be written in English. It must contain a summary, the assignment and the time schedule. Its structure should include an introduction, an analysis of related work, and a complete documentation of all used hardware/software tools. Two written copies of the final report must be delivered to the main advisor along with CD that includes developments undergone during the thesis.

5 References

[1] http://en.wikipedia.org/wiki/Lists_of_disasters

armasuisse
Science and Technology
C4I Networks

Dr. Vincent Lenders
Thun, 19. September 2014

Bibliography

[1] T. F. A. Java, X. Song and B. Tseng. Why we twitter: Understanding microblogging usage and communities. In *Proc. Joint 9th WEBKDD and 1st SNA-KDD Workshop 2007*, 2007.

[2] P. S. I. W. A. Tumasjan, T. Sprenger. Predicting elections with twitter: What 140 characters reveal about political sentiment. In *Proceedings of the Fourth International AAAI Conference on Weblogs and Social Media*, 2010.

[3] F. Atefeh and W. Khreich. A survey of techniques for event detection in twitter. *Computational Intelligence. doi: 10.1111/coin.12017*, 2013.

[4] J. K. B. Sharifi, MA. Hutton. Summarizing microblogs automatically. In *HLT '10 Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics Pages 685-688*, 2010.

[5] F. Chen and D. B. Neill. Non-parametric scan statistics for event detection and forecasting in heterogeneous social media graphs. In *Proc of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM*, 2014.

[6] T. Cheng and T. Wicks. Event detection using twitter: A spatio-temporal approach. *PloS one 9.6*, 2014.

[7] C. M. Corney, David and A. Gker. Spot the ball: Detecting sports events on twitter. *Advances in Information Retrieval. Springer International Publishing, 449-454,*, 2014.

[8] H. L. K. M. C. Fred Morstatter, Jurgen Pfeffer. Is the sample good enough? comparing data from twitter's streaming api with twitter's firehose. *CoRR abs/1306.5204*, 2013.

[9] C. S. Hamed Abdelhaq and M. Gertz. Eventweet: Online localized event detection from twitter. In *Proc. of the VLDB Endowment, Vol. 6, No. 12, 2013*, 2013.

[10] B. S. Ifrim, Georgiana and I. Brigadir. Event detection in twitter using aggressive filtering and hierarchical tweet clustering. *SNOWDC@ WWW*, 2014.

[11] M. H. A. W. M. Krstaji, C. Rohrdantz. Getting there first: Real-time detection of real-world incidents on twitter. *2nd IEEE Workshop on Interactive Visual Text Analytics VisWeek 2012*, 2012.

[12] J. B. M. Naaman and C.-H. Lai. Is it really about me?: message content in social awareness streams. In *ACM Computer Supported Cooperative Work (CSCW), Savannah, Georgia, USA, 2010, pp. 189-192.*, 2010.

[13] O. I. O. M. P. S. McCreadie R., Macdonald C. Scalable distributed event detection for twitter. In *Proc. of IEEE International Conference on Big Data, 2013*, 2013.

[14] M. B. P. Andre and K. Luther. Who gives a tweet?: evaluating microblog content value. In *ACM conference on Computer Supported Cooperative Work (CSCW), Seattle, Washington, USA, 2012, pp. 471-474*, 2012.

[15] K.-C. Rui Li ; Kin Hou Lei ; Khadiwala, R. ; Chang. Tedas: A twitter-based event detection and analysis system. In *Data Engineering (ICDE), 2012 IEEE 28th International Conference on. IEEE*, 2012.

[16] K. S. L. P. S .Vieweg, A. L. Hughes. Microblogging during two natural hazards events: What twitter may contribute to situational awareness. *CHI 2010: Crisis Informatics*, 2010.

[17] M. W. Schubert, Erich and H.-P. Kriegel. Signitrend: scalable detection of emerging topics in textual streams by hashed significance thresholds. In *Proc. of the 20th ACM SIGKDD international conference on Knowledge discovery and data mining. ACM*, 2014.

[18] M. Walther and M. Kaisser. Geo-spatial event detection in the twitter stream. *Advances in Information Retrieval. Springer Berlin Heidelberg, 356-367*, 2013.

[19] J. C. E. C. J. H. Y. Z. Y You, G Huang. Geam: A general and event-related aspects model for twitter event detection. *Web Information Systems EngineeringWISE 2013. Springer Berlin Heidelberg 319-332*, 2013.

[20] H. W. Z. Chu, S. Gianvecchio and S. Jajodia. Who is tweeting on twitter: human, bot, or cyborg? In *ACM Annual Computer Security Applications Conference (ACSAC), Austin, Texas, 2010, pp. 21-30*, 2010.

[21] S. Zhao. Detecting events from twitter in real-time. Master's thesis, Rice University, 2013.