université PARIS-SACLAY

CentraleSupélec

NNT : 20XXSACLXXXX

Thèse de doctorat

Thesis title. It can extend over several lines (even 4 or 5)

Thèse de doctorat de l'Université Paris-Saclay
préparée à Nom de l'établissement

École doctorale n°000 Dénomination (Sigle)
Spécialité de doctorat: voir spécialités par l'ED

Thèse présentée et soutenue à Ville de soutenance, le Date, par

FIRSTNAME LASTNAME

Composition du Jury :

Prénom Nom Statut, Établissement (Unité de recherche)	Président
Prénom Nom Statut, Établissement (Unité de recherche)	Rapporteur
Prénom Nom Statut, Établissement (Unité de recherche)	Rapporteur
Prénom Nom Statut, Établissement (Unité de recherche)	Examinateur
Prénom Nom Statut, Établissement (Unité de recherche)	Directeur de thèse
Prénom Nom Statut, Établissement (Unité de recherche)	Co-directeur de thèse
Prénom Nom Statut, Établissement (Unité de recherche)	Invité
Prénom Nom Statut, Établissement (Unité de recherche)	Invité

Dedication

Acknowledgements

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sollicitudin massa vel venenatis dictum. Aliquam erat volutpat. Phasellus accumsan eu felis at luctus. Integer neque elit, venenatis sed iaculis in, tincidunt nec augue. Aliquam erat volutpat. Nulla sodales tortor non justo tincidunt, non varius risus mollis. Aliquam est purus, cursus at nulla ac, sollicitudin placerat diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Ut at leo eget metus scelerisque venenatis. Sed quis dui nisi. Morbi sodales, leo ac scelerisque malesuada, libero sem placerat ante, sit amet ullamcorper ligula nulla vestibulum tellus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sollicitudin massa vel venenatis dictum. Aliquam erat volutpat. Phasellus accumsan eu felis at luctus. Integer neque elit, venenatis sed iaculis in, tincidunt nec augue. Aliquam erat volutpat. Nulla sodales tortor non justo tincidunt, non varius risus mollis. Aliquam est purus, cursus at nulla ac, sollicitudin placerat diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Ut at leo eget metus scelerisque venenatis. Sed quis dui nisi. Morbi sodales, leo ac scelerisque malesuada, libero sem placerat ante, sit amet ullamcorper ligula nulla vestibulum tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sollicitudin massa vel venenatis dictum. Aliquam erat volutpat. Phasellus accumsan eu felis at luctus. Integer neque elit, venenatis sed iaculis in, tincidunt nec augue. Aliquam erat volutpat. Nulla sodales tortor non justo tincidunt, non varius risus mollis. Aliquam est purus, cursus at nulla ac, sollicitudin placerat diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Ut at leo eget metus scelerisque venenatis. Sed quis dui nisi. Morbi sodales, leo ac scelerisque malesuada, libero sem placerat ante, sit amet ullamcorper ligula nulla vestibulum tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sollicitudin massa vel venenatis dictum. Aliquam erat volutpat. Phasellus accumsan eu felis at luctus. Integer neque elit, venenatis sed iaculis in, tincidunt nec augue. Aliquam erat volutpat. Nulla sodales tortor non justo tincidunt, non varius risus mollis. Aliquam est purus, cursus at nulla ac, sollicitudin placerat diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Ut at leo eget metus scelerisque venenatis. Sed quis dui nisi. Morbi sodales, leo ac scelerisque malesuada, libero sem placerat ante, sit amet ullamcorper ligula nulla vestibulum tellus. Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sollicitudin massa vel venenatis dictum. Aliquam erat volutpat. Phasellus accumsan eu felis at luctus. Integer neque elit, venenatis sed iaculis in, tincidunt nec augue. Aliquam erat volutpat.

Nulla sodales tortor non justo tincidunt, non varius risus mollis. Aliquam est purus, cursus at nulla ac, sollicitudin placerat diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Ut at leo eget metus scelerisque venenatis. Sed quis dui nisi. Morbi sodales, leo ac scelerisque malesuada, libero sem placerat ante, sit amet ullamcorper ligula nulla vestibulum tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sollicitudin massa vel venenatis dictum. Aliquam erat volutpat. Phasellus accumsan eu felis at luctus. Integer neque elit, venenatis sed iaculis in, tincidunt nec augue. Aliquam erat volutpat. Nulla sodales tortor non justo tincidunt, non varius risus mollis. Aliquam est purus, cursus at nulla ac, sollicitudin placerat diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Ut at leo eget metus scelerisque venenatis. Sed quis dui nisi. Morbi sodales, leo ac scelerisque malesuada, libero sem placerat ante, sit amet ullamcorper ligula nulla vestibulum tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sollicitudin massa vel venenatis dictum. Aliquam erat volutpat. Phasellus accumsan eu felis at luctus. Integer neque elit, venenatis sed iaculis in, tincidunt nec augue. Aliquam erat volutpat. Nulla sodales tortor non justo tincidunt, non varius risus mollis. Aliquam est purus, cursus at nulla ac, sollicitudin placerat diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Ut at leo eget metus scelerisque venenatis. Sed quis dui nisi. Morbi sodales, leo ac scelerisque malesuada, libero sem placerat ante, sit amet ullamcorper ligula nulla vestibulum tellus.

Lorem ipsum dolor sit amet, consectetur adipiscing elit. Sed sollicitudin massa vel venenatis dictum. Aliquam erat volutpat. Phasellus accumsan eu felis at luctus. Integer neque elit, venenatis sed iaculis in, tincidunt nec augue. Aliquam erat volutpat. Nulla sodales tortor non justo tincidunt, non varius risus mollis. Aliquam est purus, cursus at nulla ac, sollicitudin placerat diam. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Ut at leo eget metus scelerisque venenatis. Sed quis dui nisi. Morbi sodales, leo ac scelerisque malesuada, libero sem placerat ante, sit amet ullamcorper ligula nulla vestibulum tellus.

Contents

List of Figures

List of Tables

Chapter 1

Introduction

According to the Reuters Institute, 36% of French adults use social media as a source of news.[1] This share has declined since 2017, but this is mostly due to a decrease in the use of Facebook, while other networks are stable (like Twitter) or growing rapidly. This evolution of news consumption may reflect a growing interest for stories that are usually not covered by traditional news media, or that are covered in a different way. This raises the question of the type of news that is mostly shared on social networks: are the citizens differently informed when they use social network as a gateway to news?

In response to the transformation of news consumption, one should expect a change in the production of news by traditional media outlets. McGregor and Molyneux [2018] find that journalists using Twitter as part of their daily work consider tweets as newsworthy as headlines from the Associated Press. Does this change in the perception of journalists has an effect on the type of stories they choose to report? Does the success of a story on social media impacts the news production by traditional news media outlets?

The objective of this thesis is to investigate the role of Twitter in the evolution of both news consumption and news production in recent years. We aim at understanding what kind of news are amplified by the sphere of social networks, and, conversely, to show in which cases events born on the social networks become a subject relayed by traditional media outlets. The challenge is to quantify and analyze precisely the relationships between the two spheres, in a context of very strong influence of each sphere on the other.

Indeed, stories do not spread only in one direction, from news media to social networks or from social networks to news media. The recent Benalla case in France can be used as an illustration of the different rebounds that an event can have in both spheres: videos of Alexandre Benalla wearing a police helmet and hitting a protester where published on social media on May 1, 2018. However, he was only identified as an aide from President Macron's office on July 18, by the newspaper *Le Monde*. After this first article, traditional media outlets started to investigate on the missions of Mr. Benalla. Both journalists and Twitter users published numerous pictures

[1] http://media.digitalnewsreport.org/wp-content/uploads/2018/06/digital-news-report-2018.pdf?x89475

Notes: These preliminary results were nuanced in the full study published on August 8, 2018

Figure 1.1: Tweet from DisinfoLab, an organization working on disinformation, about the "overactivity" of some Twitter accounts during the Benalla scandal

of Alexandre Benalla in official appearances of the President, including during the period when he was allegedly suspended. The newspaper "Sud Ouest" used the term "photo hunt" to qualify the attitude of social media users and photo reporters[2]. On July 30, a Belgian organization called DisinfoLab evoked an artificial swelling of the number of messages on Twitter related to the Benalla case.[3] In the partial results published on that day, the "overactivity" of some Twitter accounts and "pro-Russian" accounts were mentioned. On August 8, DisinfoLab published the entire study,[4] showing no evidence that an organized Russian intervention has sought to amplify the Benalla case on Twitter. However, several media outlets had already relayed the (wrong) information that "Russian bots" had influenced the reaction of the public on Twitter.[5]

This is only one example of the plurality of "interaction patterns" [Ning et al., 2015] between social networks and traditional media that can occur in the news agenda. Here, social networks are first used as a source by news outlets (most videos of Alexandre Benalla used by journalists where initially published on Twitter by witnesses of the May 1 demonstration). Then, they participate to the controversy raised by traditional media outlets and amplify it. Finally, the amplitude of the echo on Twitter is discussed by media outlets. Other types of interaction

[2] https://www.sudouest.fr/2018/07/23/affaire-benalla-quand-les-reseaux-sociaux-s-amusent-5256018-10458.php
[3] https://twitter.com/DisinfoEU/status/1023903729668575242
[4] https://spark.adobe.com/page/Sa85zpU5Chi1a/
[5] https://abonnes.lemonde.fr/les-decodeurs/article/2018/08/08/l-impossible-quete-des-bots-russes-de-l-affaire-benalla_5340540_4355770.html?

patterns exist, for example “break on Twitter first” stories (like the hashtag #MakeOurPlanetGreatAgain posted on June 2017 by Emmanuel Macron that was widely commented by traditional news media) or Twitter movements that criticize traditional media (for example, Twitter users reacted to the shocking images of victims of the Nice attack with the hashtag #CSACoupezTout, which led the *France Télévisions* group to apologize[6]). We aim at developing measurement tools and analysis criteria to characterize and quantify these interaction patterns.

The originality of this project is its bi-disciplinary nature. On the one hand, it will consist of an analysis in media economics, in order to determine the factors that influence the relative impact of a story on Twitter and in traditional news media. This thesis will have to take into account several types of measures of the media impact, both absolutely (number of tweets, number of articles), but also related to the membership networks of the users emitting the information: a story spreads more widely if it is issued within a majority group [Halberstam and Knight, 2016], and information is more easily relayed if it emanates from the account of a journalist or a politician [Harder et al., 2016]. On the other hand, it will require advanced computer science research to design novel approaches to Twitter event detection and clustering, using both Natural Language Processing and Image Processing.

1.1 Choice of Twitter

Why choose Twitter over another social network? Twitter is a micro-blogging website where users can post short messages called “tweets”. Tweets are limited to 280 characters, but can also contain pictures or videos. It is difficult to find reliable statistics on the number of tweets emitted every day worldwide. In 2014, Twitter announced the figure of 500 million tweets on average per day[7], but there has been no other official statement since. Several types of interactions are possible on this social network: users can “follow” other users (that is to say, subscribe to their account in order to see all the tweets they publish), they can “retweet” a tweet (re-publish it on their own account), “reply” to it, “quote” it (re-publish the tweet with a comment of their own) or “like” it. Users can refer to other users in their tweets using “mentions” (the user’s name preceded by ”@”), and they can tag specific words as important using “hashtags” (words preceded by “#”). Tweets are publicly visible by default, which is why Twitter is used by many public personalities like politicians or journalists. This is one of the main differences between Twitter and Facebook, where posts can by default only be read by one’s “friends” (usually people that one know in real life).

Twitter is not the most used social network in France. According to the Reuters Intitute, in 2018 it is even the 4th French social network (used by 16% of respondents), behind Facebook (63%), YouTube (51%) and Facebook Messenger (31%). Nevertheless, we chose this network for our analysis of the relationships between social networks and traditional news media for several reasons.

First of all, it is predominantly used for news access. The “News Use Across Social Media Platforms 2018” study

[6] https://www.francetvinfo.fr/economie/medias/france-televisions/edition-speciale-sur-l-attentat-de-nice-france-televisions-pres
1548057.html

[7] https://blog.twitter.com/official/en_us/a/2014/the-2014-yearontwitter.html

by the Pew Research Center[8] finds that 71% of American Twitter users get news on the platform, compared to 68% for Facebook and 38% for YouTube. There is no similar study on French social networks, but the structure of Twitter makes it a privileged tool for sharing news content, independently of the country. Indeed, if favors public statements rather than private messages to family and friends, and encourages the sharing of external content (reference to other pages through URLs) because of the brevity of tweets. Kwak et al. [2010] even argue that the structure of Twitter makes it similar to a news media.

Secondly, Twitter has quickly become the preferred network of journalists, who use it both to easily contact sources and to build a connection with their audience [Swasy, 2016]. In the sample of journalists studied by McGregor and Molyneux [2018], 93% had a Twitter account. A report conducted at the request of the European Commission[9] shows a similar trend in Europe: the interviewed journalists make the distinction between Twitter, mostly used for work, and Facebook, more used in private life.

Finally, Twitter provides a larger access to its data than other social media platforms. Even if the volume of tweets that one can access through Twitter's APIs is limited, the company still provides a free access to a rather large volume of data. Conversely, despite the research effort recently launched by Facebook to protect elections,[10] it is still nearly impossible for researchers outside Facebook to get access to information on users' activity on the platform.

1.2 Choice of France

Working on French tweets and news contents is difficult due to the still little amount of available corpora for Natural Language Processing and tweet analysis compared to English. However, we can rely on the corpus provided by the French Institute for Audiovisual (INA) that contains all content published by French news media online (newspapers, radio, televisions and online pureplayers) with their precise publication date [Cagé et al., 2020]. Besides, we have access to the universe of the AFP (Agence France Presse) news agency's dispatches, which gives us a proxy for the news stories that make it to traditional media outlets – with similarly the exact time of each dispatch.

Besides, the main empirical challenge for researchers using Twitter data comes from the fact that, because of the limits of the Twitter streaming API, it is impossible for researchers to capture the universe of tweets that are posted on the platform during a given period of time. Perhaps paradoxically, the advantage of France comes here from the fact that there is less data than for example for the United States: according to our estimates, around 9.2 million tweets are published every day in French. Using the collection methods detailed in chapter **??** section 2.3.2, we are able to capture a little bit more than 4.5 million tweets a day, which means that our dataset covers nearly half of all the tweets published in French.

[8] http://www.journalism.org/2018/09/10/news-use-across-social-media-platforms-2018/
[9] http://ec.europa.eu/commfrontoffice/publicopinion/archives/quali/journsm_en.pdf
[10] https://www.facebook.com/notes/mark-zuckerberg/preparing-for-elections/10156300047606634/

1.3 Characterization of events

What is a media event? One possible definition is: a fact that is important enough to be reported in the media. In contemporary societies, news media have therefore a role to play in the definition of events. According to the historian Pierre Nora, the emergence of the mass media has transformed the nature of events: *"Press, radio, images are not only means from which events would be relatively independent, they are the very condition of their existence."*[11][Nora, 1972]. The sociologist Patrick Champagne [Champagne, 2000] shares the same view (*"The media build the events they report"*[12]) but highlights the fact that creating an event is a collective process : one media outlet alone cannot make the news if it is not picked up by others. A media event has thus to be reported by several sources to be defined as such.

With the appearance of social media, a new dimension has emerged: traditional news media cannot ignore a topic that is really bursting on Facebook or Twitter. In practice, many news events start nowadays on social media, like the #MeToo movement. Social events and media events tend to be the same in many cases, that is what we call *joint events*. However, any conversation or trend on social media cannot be considered as a *media event*. We therefore propose a distinction between *media events, social events* and *joint events* In this part, we provide definitions that formalize these intuitions.

1.3.1 Definitions

Literature review

There is no consensus on the formal definition of the problem of event detection on Twitter, mainly because of the diversity of use-cases and scenarios for which event detection may be useful. Sankaranarayanan et al. [2009] aim at providing a user interface displaying breaking news from tweets in the same way as a news wire. Aiello et al. [2013] design their tool for "an expert of some domain [who] has to monitor specific topics or events being discussed in social media". Zhang et al. [2017a] give several applications of their event detector, such as sending alarms in case of imminent disaster or helping local governments to prevent social unrest. These few examples highlight the fact that all authors working on the subject do not perceive *"event detection"* in the same way: their tools do not detect the same **type of events** (local events, conversations about specific topics, breaking news...), they do not have the same **inputs** (tweets from a range of selected sources, geo-tagged tweets, tweets containing a given keyword...) and **outputs** (all detected events, the top k events depending on some relevance criteria, a list of tweets linked to each event, a list of keywords...), and are not designed to handle the same **volume of tweets**. We propose here an overview of the existing definitions.

[11] "Presse, radio, images, n'agissent pas seulement comme des moyens dont les événements seraient relativement indépendants, mais comme la condition même de leur existence."
[12] "les médias construisent les événements dont ils rendent compte"

Even though it is not focused on tweets but rather on traditional news contents, many articles on Twitter event detection refer to the definition given in the Topic Detection and Tracking project [Allan, 2002]: an event is "something that happens at a specific time and place along with all necessary conditions and unavoidable consequences". For Aggarwal and Subbian [2012] and McMinn et al. [2013], the definition of event is similar but includes another dimension: an event has to be "of interest to the news media" [Aggarwal and Subbian, 2012]. Becker et al. [2011b] consider that events are necessarily linked to a real world fact: they define an event as "a real-world occurrence e with (1) an associated time-period T_e and (2) a time-ordered stream of Twitter messages M_e, of substantial volume, discussing the occurrence and published during time T_e." For Hasan et al. [2018], the notion of "real world" is also mentioned. An event is "an occurrence of interest in the real world which instigates a discussion on the event-associated topic by various users of social media, either soon after the occurrence or, sometimes, in anticipation of it." In these definitions, the events have to exist outside social medias: they have to be either mentioned by other traditional media outlets, or be linked to an "occurrence", a "fact" in the real world.

Other authors define a social media event depending on the actions they generate on the studied social media. Panagiotou et al. [2016] call "action" any activity on a social media, such as posting new content, interacting with another profile (sending a friend request, for example) or interacting with existing contents (like, or retweet). In their definition, an event is "something that causes a large number of actions in the OSN [Online Social Network]". This type of approaches often consider events as "breaking news" that will generate a larger activity on social media than any other topic at a given moment. This is also the approach of Guille and Favre [2015], that focus on detecting "bursty" patterns.

For our study, we need to detect on Twitter any set of tweets from several sources that discuss the same fact. These facts do not have to exist outside of the social media (in the so called "real world"). Besides, defining an event depending on its "burstiness" can lead to neglect smaller size events.

Let f be a **triggering fact**, i.e. a fact that causes an important **activity** in either the traditional media sphere or the social media sphere or the both. This fact can happen in real life or on the internet. Panagiotou et al. [2016] discuss the relevance of a distinction between *real world events* and *virtual events* such as "memes, trends or popular discussions". These distinctions are unnecessary for our applications: for example, many politicians use social media to make public statements, that are not less "real" than the ones made during interviews or press conferences. A triggering fact can thus be a tweet, a Youtube video, a speech, a sport performance, a trial, a Facebook post, etc.

In our work, we consider that a **source** s can be a media outlet or a Twitter account. A **source** can publish one or several **content objects**.

A **content object** o is a tuple (x_o, s_o, d_o, f) with x_o the object in itself (i.e. a tweet, a news article,...), s_o its source, d_o its publication date and f the related triggering fact.

Two kinds of content object will be considered in this thesis :

- **Twitter object** $t = (x_t, s_t, d_t, f)$ in which x_t is the tweet (text and /or visual contents) and s_t is a Twitter account and d_t the publication date of the tweet.
- **Media object** $m = (x_m, s_m, d_m, f)$ in which x_m is the media content (press article, video, television or radio report...), s_m is a media outlet, and d_m the publication date of the object.

Using these different concept, we can now defined an **event** as a set of content objects from several sources that are related to the same fact f.

More precisely, a **Twitter event** E_T is a set of Twitter objects $\{t_1, ..., t_n\}$ discussing the same fact f and generated by a least k different sources (i.e. Twitter accounts) in a restricted time interval $[d_{start}, d_{end}]$. For the sake of clarity, we introduce the set $X_T = \{x_t^1, ..., x_t^n\}$, $S_T = \{s_t^1, ..., s_t^n\}$ and $D_T = [d_{t,start}, d_{t,end}]$ that are respectively the set of tweets, the set of sources and the time interval of E_T. E_T is a Twitter event if $|S_T| \geq k$ and if $D_T \subseteq [d_{start}, d_{end}]$.

Similarly, a **Media event** E_M is a set of media objects $\{m_1, ..., m_n\}$ discussing the same fact f and generated by a least l different sources (i.e. media outlets) in a restricted time interval $[d_{start}, d_{end}]$. We introduce the set $X_M = \{x_m^1, ..., x_m^n\}$, $S_M = \{s_m^1, ..., s_m^n\}$ and $D_M = [d_{m,start}, d_{m,end}]$ that are respectively the set of tweets, the set of sources and the time interval of E_M. E_M is a media event if $|S_M| \geq l$ and if $D_M \subseteq [d_{start}, d_{end}]$.

1.4 Detailed Outline

Chapter 2

Building a corpus for event detection on Twitter

2.1 Introduction

Research on social media topic detection and tracking (without specification of the type of topics) lacks publicly available tweet datasets to generate reproducible results. This is particularly the case for non-English languages. Some datasets do exist, but they often have different definitions of event or topic than ours: many works centering on event detection are actually focused on burst detection (detecting topics such as natural disasters, terrorist attacks, etc., which cause an unusual volume of tweets) and do not attempt to assess the relative size of events. Therefore, we needed to build our own evaluation corpus that would fit our needs in terms of language (French), representativity of the collected tweets, quality and diversity of the annotated events. This introduction details the properties expected from such a corpus. In the rest of the chapter, we present the existing social media and in particular tweet datasets for event detection, then we detail our tweet collection method and our annotation process. Finally, we propose several ways of evaluating the built corpus.

2.1.1 Representativity and completeness of the collected data

In an ideal world, to compare news production on social media and on mainstream media, one would need the universe – during a given period of time (e.g. the year 2018) and a geographical location (e.g. France, the UK or the US) – of all documents published on the one hand on social media and on the other hand on mainstream media. Unfortunately, given the limitation of the Twitter API, it is not possible for the researcher to capture the universe of the documents (or tweets) published on Twitter. However, the researcher can work on a sample of the documents as long as this sample is “representative”, i.e. the sample matches the characteristics of the real Twitter stream

along all variables of interest. Why do we need representativity?

Let us assume that we obtain access to a subsample of the documents published on Twitter, but that this sample is not representative of the overall traffic. For example, let us assume that this sample of tweets is such that the characteristics of the tweets (perhaps because the API provides the researcher with documents tweeted by users with more followers) mean that these documents have a higher probability of making it into the mainstream media. Using this biased subsample will then lead the researcher to overestimate the probability that a news story broken on social media will appear on mainstream media.

The same issue will arise if the researcher wants to tackle the follow-up question: what are the determinants of the success of a news story initially broken on social media? Let us imagine that the researcher is using a selected sample of tweets that is not representative - for example, this sample of tweets comes mainly from journalists working for a given media, e.g. *Le Monde* - and that, at the same time, within the set of tweets posted by *Le Monde*'s journalists, only the successful ones are part of the sample. The results of the empirical analysis will be biased in favor of *Le Monde*. In other words, when the researcher studies the impact of the company for which the journalist works (independent variable) on the probability that the news story will break on Twitter and make it to the mainstream media (dependent variable), the coefficient obtained for *Le Monde* will overestimate the real causal impact of the company.

It appears to be very difficult to correct for the latter kind of biases. One can always control for a number of observable variables, but unobserved confounders will bias the estimation. Hence the necessity to have a representative sample of tweets, i.e. a sample of tweets such that the tweets included in our sample do not differ from the tweets that are not included along all the dimensions that may have a direct impact on the dependent variable of interest.

Representativity is thus critical for the study of news production on social media. However, to study the interaction between social media and mainstream media, we need more than just representativity: in a sense, we need completeness. For example, to determine whether a story emerged on Twitter or was first covered by mainstream media, representativity is not sufficient, because we may miss some tweets posted before the first news articles appeared. It is therefore of utmost importance not only to build a “representative” sample of tweets, but to build one that is as complete as possible.

2.1.2 Quality of the annotated subset

The properties of a given corpus have an impact on the implementation of the detection algorithm: for example, if all events in our corpus tend to grow at a high rate (i.e. people react very quickly to that event on social media), a simple way to increase the performance of our algorithm would be to select groups of tweets that have a high growth rate and discard others as “non-events”. However, the resulting approach would be incapable of detecting

other types of events and would thus introduce bias in our results. Therefore, the choices made during the creation of the annotated corpus are critical to ensure that our program can detect a large variety of events.

The size of events is also an important parameter that needs to be taken into account: small events that generate few tweets are difficult to detect automatically and this is precisely why they need to be represented in our test corpus. On the other hand, large events, such as the 2018 FIFA World Cup, generate such a large volume of tweets that it is impossible to annotate all of them manually. This is an inevitable flaw in any manually annotated corpus. We take this flaw into account when evaluating the event detection algorithm **(see Section?).**

It is also necessary to ensure the diversity of events in the corpus in terms of topic (sport, economy, science and technology, etc.) and regarding the origin of events (events having emerged on Twitter vs. events first covered by news media). The diversity of tweets inside each annotated event is also of importance: ideally, tweets should originate from a variety of sources, and not be written only by journalists or media accounts.

The following section details the choices made by the authors of existing datasets with regard to these quality criteria.

2.2 State of the art: event detection corpora

Twitter gives limited access to its data, but still provides some API endpoints to retrieve tweets (which is not the case with other more popular social networks, in particular Facebook), hence the large number of works based on Twitter datasets. However, few of them provide access to their evaluation corpora. We detail available event detection collections in this section.

McMinn et al. [2013] created the largest available corpus on event detection. They used several methods to generate candidate events: two automatic event detection methods on their set of 120 million tweets in English and one method based on query expansion of Wikipedia events. The automatically generated events were then assessed using Amazon Mechanical Turk, first to evaluate if the automatically generated events corresponded to their definition of event, and second to judge if the clustered tweets were all relevant to the event. They finally merged the events from the three candidate generation methods and removed duplicate events. The final corpus consists of more than 100,000 annotated tweets covering 505 events. However, this corpus dates from 2012. Because of tweets being removed and Twitter accounts being closed, a large part of this dataset can no longer be recovered.[1] In August 2018, we could only retrieve 66.5 million tweets from the original collection (55%) and 72,790 tweets from the annotated corpus (72%).

The SNOW dataset [Papadopoulos et al., 2014] can also be used as a test corpus for an event detection task. It is composed of 59 events from the headlines of the BBC RSS newsfeed and from NewsWhip UK published in

[1] In accordance with Twitter's terms of use, researchers sharing datasets of tweets do not share the content of the tweets, but only their ids. Using these identifiers, one can query the Twitter API to retrieve the full content of the tweets - but only if they are still online.

Dataset	Content	Collection method	Collection period	Events	Nb docs collected	Nb docs annotated
McMinn et al. [2013]	tweets	sample API	10/10/2012-07/11/2012	505	120 million	100000
SNOW Papadopoulos et al. [2014]	tweets	filter API: keywords and accounts	25/02/14	59	1 million	230
TREC 2015 Lin et al. [2015]	tweets	sample API	20/07/2015-29/07/2015	51	-	60000
MediaEval 2014 Petkos et al. [2014b]	images + metadata	Flickr API	-	20000	300000	300000
Signal-1M Suarez et al. [2018]	tweets + articles	search API: keywords	01/09/2015-30/09/2015	100	32 million	6000
News Event Mele and Crestani [2019]	tweets + articles	filter API: media accounts	01/03/2016-30/06/2016	57	80000	750
Event2018 Mazoyer et al. [2020]	tweets	sample API + filter API on neutral words	16/07/2018-06/08/2018	240	38 million	95000

Notes: The number of collected documents only takes tweets (or images for the MediaEval dataset) into account: news articles or RSS-feeds are not taken into account. The Event2018 dataset in the last row is not presented in this state of the art since it is our own dataset.

Table 2.1: Summary table of existing datasets for event detection on social networks

the course of one day (25th February, 2014). The tweets were collected from the accounts of 5000 "newshounds" (journalists, media outlets, politicians, etc.) and from a list of keywords linked to news events likely to generate comments on that day. However it does not provide comprehensive sets of tweets related to each event, only two to five "representative" tweets from a collection of more than 1 million tweets emitted on that date.

The *Text REtrieval Conference* (TREC) microblog datasets were released in 2014 Lin et al. [2014] and 2015 Lin et al. [2015]. The most recent dataset consists of more than 60,000 tweets posted between July 19th and July 30th, 2015, annotated in relationship with one of 51 topics. These tweets were sent by participants in response to the Microblog TREC tasks, which comprise two scenarios: (a) a "push notification" system returns tweets considered "interesting and novel" to a user having defined a topic of interest, and (b) a daily "email digest" is sent to a user with tweets containing relevant updates concerning her topic of interest. The 51 interest profiles cannot exactly be defined as "events", but rather general themes, such as "Find information about the possibility of the passage of gay marriage laws in Europe", or "Find information about bridge tournaments in the United States in July or August 2015". The tweets sent by participants to the task were then annotated by assessors as "relevant", "not relevant" or "highly relevant". This dataset is an interesting resource for our task, albeit less relevant than that of McMinn et al. [2013], because the annotated tweets have been selected for their "novelty": the dataset therefore includes many tweets announcing "breaking news", and few tweets of conversation or opinion on topics already known.

The datasets from the 2013 and 2014 MediaEval Social Event Detection tasks Reuter et al. [2013]; Petkos et al. [2014b] are also important corpora to test event detection systems, and multimodal systems in particular. The first challenge of this task consists in clustering images of social events so that each cluster matches an event. The events correspond to sport events, protest marches, BBQs, debates, exhibitions, festivals or concerts all registered on the last.fm website[2]. The authors downloaded pictures tagged with the last.fm event ids on Flickr. In total, each dataset contains 300,000 training images from approximately 20,000 events. This corpus is very interesting because of its size and the large number of different events it contains. However, the fact that it contains only "social events" (public gatherings attended by a large number of people) restricts the task to a certain type of topics. The type of images shared on Flickr is also very different from those shared on Twitter: they are mostly photos taken by users themselves, with few duplicate images. Conversely, images on Twitter are very often re-used by several

[2]last.fm is a music website, used in particular by artists to share the dates of their concerts. Its use has been extended to other types of events, not necessarily related to music.

users, sometimes in different contexts, which makes the task more complex.

Finally, let me mention two additional datasets. The Signal-1M corpora consists of a dataset of 1 million news articles from September 2015 [Corney et al., 2016], and of a dataset of tweets related to 100 randomly selected articles [Suarez et al., 2018]. The tweets dataset contains approximately 6,000 annotated tweets. The News Event Dataset [Mele and Crestani, 2019] contains 147,000 documents (including 80,000 tweets) from 9 news channels and published on Twitter, RSS portals and news websites from March to June 2016. 57 media events were annotated on a crowdsourcing platform, to label 4,300 documents, including 744 tweets. These two datasets contain too few tweets to allow a large-scale evaluation of any event detection and tracking system. In addition, events in SNOW, Signal-1M and the News Events Dataset are selected from media sources only. Only the corpus by McMinn et al. [2013] relies on a topic detection step among collected tweets before proceeding to the annotation step. This method allows for a greater variety of events, but is likely to influence the evaluation: indeed, event detection systems similar to those used by McMinn et al. [2013] for corpus creation may get better results when tested on this corpus.

We tried to avoid these biases when creating our own corpus. The methodology used to build our dataset is detailed in the two following sections.

2.3 Tweet collection

Our objective is to collect automatically and continuously as large a set of tweets as possible, and one that is representative of actual Twitter activity. In addition to being **representative**, this corpus must contain a **volume of tweets large enough for media events to be represented** in it, particularly medium and small events. Furthermore, we want to collect enough tweets so as to ensure that events that appear first on Twitter and then on mainstream media also do so in our dataset. Finally, these tweets must be **in French**. The following sections present the methods used to achieve these objectives.

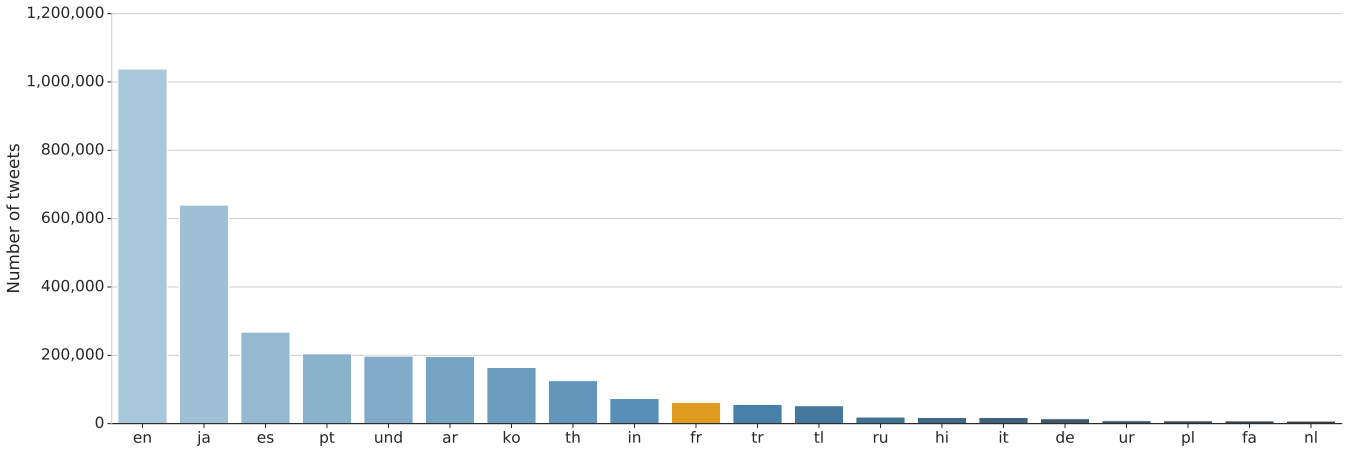
2.3.1 Constraints

There are different ways of collecting large volumes of tweets, although collecting the full volume of tweets emitted during a given period is not possible. Indeed, even though Twitter is known for providing greater access to its data than other social media platforms,[3] the Twitter streaming APIs are strictly limited in terms of volume of returned tweets. These limitations are constraints that we must integrate into our collection procedure.

Sample API: the Sample API[4] continuously provides 1% of the tweets posted around the world at a given moment in time. Once connected to the API at time t_0, the user gets 1% of all tweets emitted after t_0, and receives regular

[3] In particular, despite the research effort recently launched by Facebook, it is still nearly impossible for researchers outside Facebook to get access to information on users' activity on the platform.

[4] `https://developer.twitter.com/en/docs/tweets/sample-realtime/overview/get_statuses_sample`



Notes: This figure plots the average number of tweets collected per day using the Sample API in the 20 most frequent languages on Twitter. The language metadata is provided by Twitter. "und" stands for "undefined" language.

Figure 2.1: Average number of tweets per day in each language collected using the Sample API

batches of tweets as long as she stays connected. Twitter provides little information on how the sample is generated. However, Kergl et al. [2014] have studied it by analyzing the ids of tweets (based on a timestamp in milliseconds and on a number of series to identify tweets issued during the same millisecond). They show that all tweets provided by the API were issued between the 657th and 666th milliseconds of each second, which should guarantee that the user receives a constant stream representing around 1% of the total. Another study done on the distribution of tweets in the Sample API in comparison with another paid API, which provides full access to all emitted tweets, shows no statistically significant difference between the two samples [Morstatter et al., 2014].

This API does not meet our needs, since the proportion of tweets in French is only 1.8% of the total sample on average (Figure 2.1 illustrates the distribution of tweets in different languages). Moreover, according to Liu et al. [2016], the proportion of tweets concerning news is less than 0.2%. If we combine all these restrictions, we could only have access to 92,000 tweets in French a day, and less than 200 tweets a day concerning news if we were simply using the Sample API provided by Twitter.

Filter API: the Filter API[5] continuously provides tweets corresponding to the input parameters (keywords, account identifiers, geographical area). The language of the returned tweets can be selected. Again, the API provides only about 1% of the total flow. However, this is sometimes enough to collect all the tweets containing a relatively little-used keyword, and it is clearly enough to collect all the tweets from a given account. For a language with low representation such as French, this API could theoretically provide us with up to 55% ($\frac{1\%}{1.8\%} = 55\%$) of all the tweets emitted in French. Given this observation, we worked at identifying the keywords that maximize the number of returned French tweets.

[5] https://developer.twitter.com/en/docs/tweets/filter-realtime/api-reference/post-statuses-filter.html

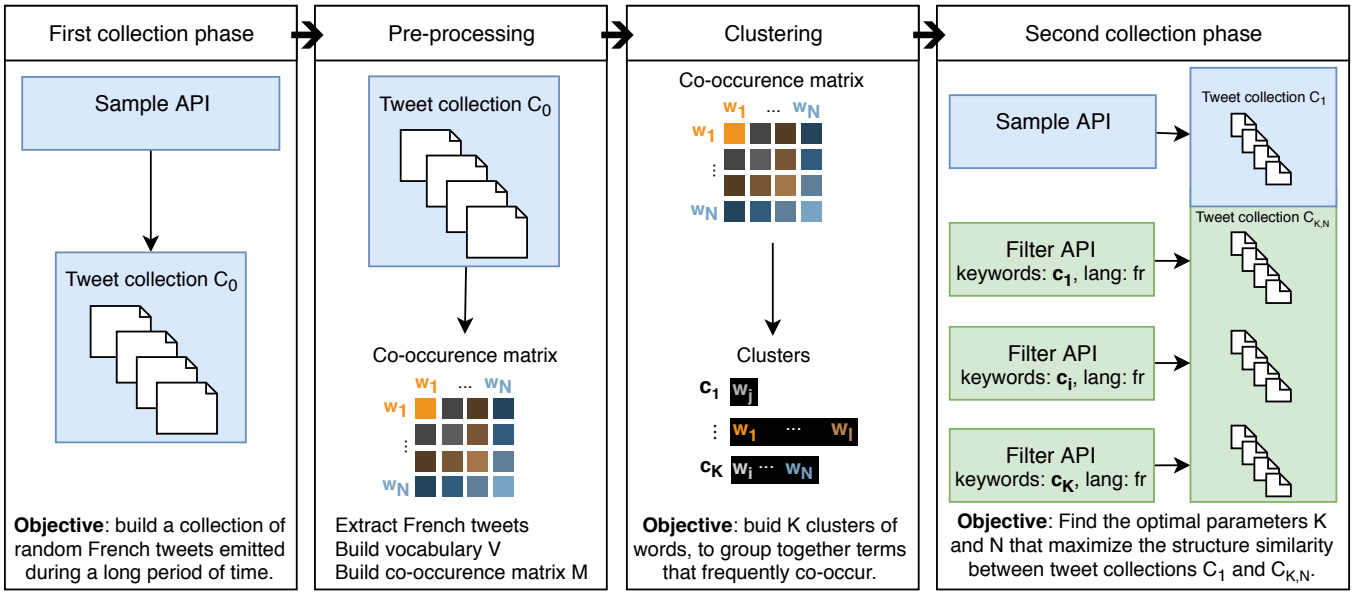


Figure 2.2: Diagram of our methodology to select the best set of keywords for each API key

Joseph et al. [2014] compare five samples collected through the Filter API with the same input keywords at the same time, using five different connection tokens[6]: they find that two connections to the Filter API at the same time with the same keywords as inputs are nearly identical". Hence, it is not useful to try to collect more tweets using a second access token with the same keywords. However, spreading different keywords over several API connections should return a higher number of tweets.

2.3.2 Proposed collection strategy

Given the constraints of the APIs, we decided to collect tweets by using the random stream proposed by the Sample API, but to increase the volume of collected data by using the Filter API as well, with "neutral" terms as keywords parameters, and "French" as the language parameter. In order to further increase the volume, we decided to use multiple tokens to connect to the Filter API.

The choice of the keywords parameters was made with a view to optimizing two metrics: the number of collected tweets, and their representativity of actual Twitter activity. The selected terms thus had to be the most frequently written words on Twitter, and we had to use different terms (and terms that do not co-occur in the same tweets) as parameters for each connection. In this way, the multiple connections would return sets of tweets with little intersection, and thus a greater total volume.

The precise strategy we used is as follows (it is schematized in Figure 2.2): given a set of tweets $C_0 = \{t_1, \ldots, t_k\}$ collected using the Sample API during a time-interval $I = [d_{t,start}, d_{t,end}]$ we select tweets in French, creating a subset $C_{0French}$ and extract from them a vocabulary $V = \{w_j, \forall j \in [1, \ldots, M]\}$ of all unique words appearing

[6]To use the Twitter API, a connection token is required. Twitter limits the access to its data by generating only one connection token per Twitter account.

in $C_{0French}$. We extract a subset of the N words of V having the highest document-frequency. We build a co-occurrence matrix $\mathcal{M} = (m_{i,j}) \in \mathbb{N}^{N \times N}$ where $(m_{i,j})$ is the number of times w_i and w_j co-occur in the same tweet of $C_{0French}$. Using spectral clustering with $\mathcal{M}$ as adjacency matrix, we extract K clusters of terms. The K obtained clusters of words are then used as parameters of K different connections to the Filter API. By doing so, we aim to separate terms that are not frequently used together and thus to collect sets of tweets with the smallest possible intersection.

Section 2.3.4 presents the method used to evaluate the similarity between $C_{K,N}$ – the set of tweets collected using N keywords spread on K Filter API connections –, and $C_1French$ – the set of French tweets collected with the Sample API during the same period as $C_{K,N}$.

2.3.3 Experimental setup

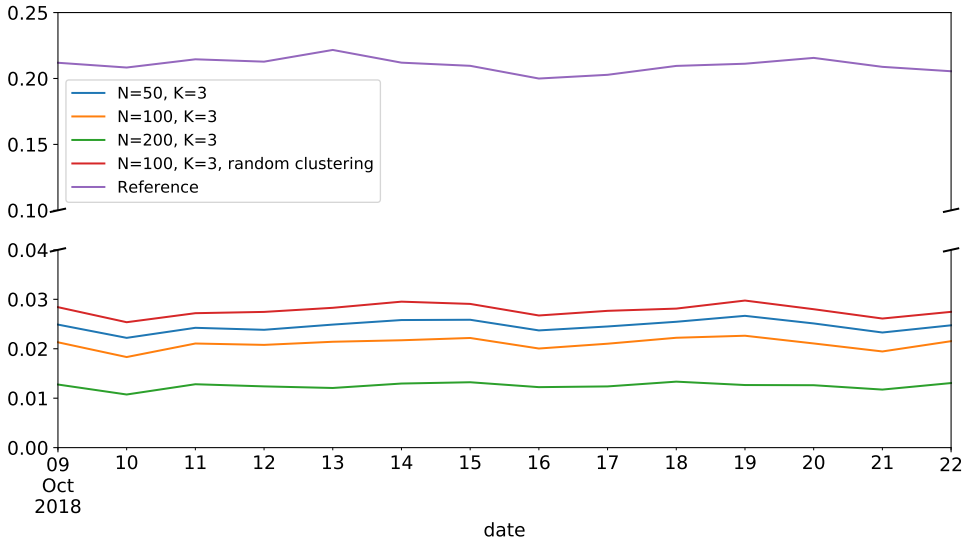
In practice, we collected our corpus C_0 of sample tweets between $d_{t,start}$ = 2018/01/15 and $d_{t,end}$ = 2018/02/15. The text of the collected tweets was tokenized on white spaces and on punctuation characters ("qu'il" was considered to be two words, "qu" and "il"). The resulting vocabulary V was lowercased and accents were removed, since we noticed that accents and capital letters are not taken into account by the Twitter API. For example, the API returns tweets containing both "à" and "a" if the parameter "a" is given as input. No other pre-processing such as stemming was applied (the vocabulary contains both "mdr" and "mdrrr"[7], for example), since the objective here is not to query the API with semantically different terms, but with the most frequently used terms.

We ran tests with $N \in \{50, 100, 200\}$ and $K \in \{2, 3\}$. This choice was motivated by our storage and CPU capacity. The clusters of terms for the different N and K values are presented in Appendix A. The resulting clusters are imbalanced in size: some contain only a few words, while others contain all remaining terms. In order to control for the effect of imbalanced clusters, we instead follow a different method and distribute terms randomly in clusters of size $\frac{N}{K}$. The samples collected using random clustering are denoted $R_{K,N}$. We ran each test for a period of two weeks, during which we also collected tweets from the Sample API and extracted tweets in French. This last set of tweets is denoted $C_{1French}$.

2.3.4 Evaluation of the collection strategy

We compared the sets $C_{K,N}$ and $R_{K,N}$, collected with each collection method, with the set $C_{1French}$, collected with the Sample API during the same time interval, in order to assess the representativity of each test dataset. This approach is comforted by the study of Morstatter et al. [2014], who had access to the entire stream of tweets and compared it with the Sample API. They find that the tweets from the Twitter Sample API are "a representative sample of the true activity on Twitter".

[7] French abbreviations similar to "lol" and "loool". "mdr" stands for "mort de rire", literally "die laughing".



Notes: This figure plots the daily evolution of the KL-divergence between the word distribution in collection $C_1 French$ and the distribution obtained using 4 collection methods. The "Reference" is obtained by splitting the collection $C_1 French$ into two sets and computing the KL-divergence between them. A KL-divergence of 0 indicates a perfect similarity between 2 distributions.

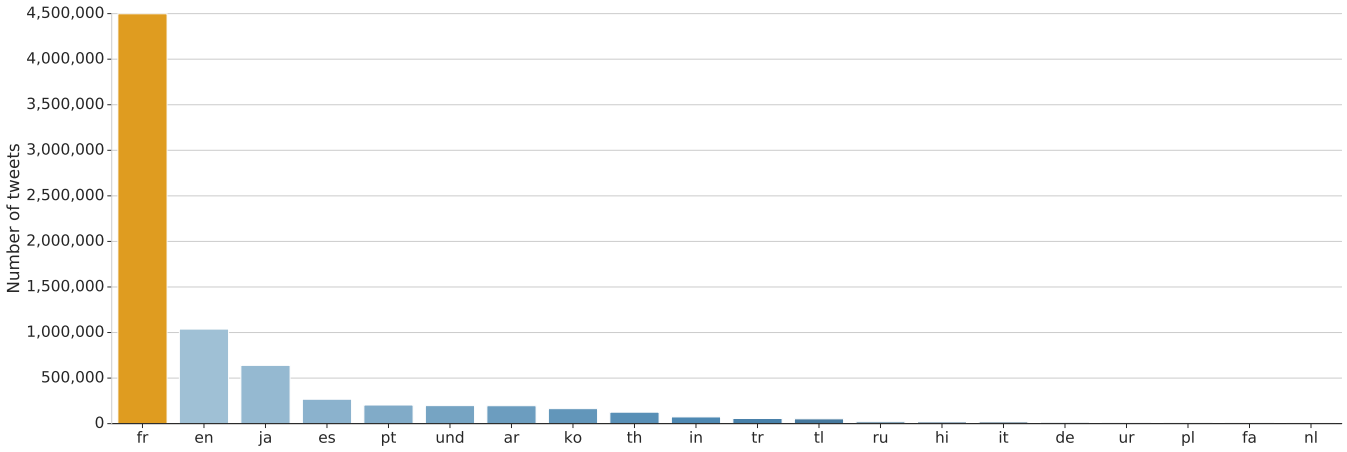
Figure 2.3: Daily evolution of the divergence between collection $C_1 French$ and the collections $C_{K,N}$ with $K = 3$

Several comparison methods can be used in order to assess the similarity between two collections of texts. A first approach consists in considering the number of times each word is used in each collection as a probability distribution, and to measure the difference between those distributions. We used Kullback-Leibler divergence [Kullback, 1997] as comparison metric. For two discrete probability distributions P and Q defined on the same probability space χ the Kullback-Leibler divergence is defined as

$$KL(Q\|P) = \sum_{x \in \chi} Q(x) \log \left(\frac{Q(x)}{P(x)}\right). \tag{2.1}$$

This measure is not symmetric, since it is meant to measure the difference between a probability distribution P to a reference distribution Q, which is precisely what we have to evaluate, Q being the distribution of words in $C_{1,French}$. For each $(N, K) \in \{50, 100, 200\} \times \{2, 3\}$, we computed the KL-divergence between the word distribution in $C_{K,N}$ and in $C_{1French}$ for the same collection period. In order to have a reference of what level of divergence can be accepted, we also split the corpus $C_{1French}$ into two sets (depending on whether the tweets had an even or odd id) and computed the KL-divergence between those sets. Figure 2.3 presents the results for $K = 3$. Overall, we found that the collection $C_{3,200}$ was the most similar to $C_{1French}$ using KL-divergence as a comparison metric.

We decided to keep $C_{3,200} \cup C_{1French}$ as our main collection method, since its similarity to the random sample was the highest. Figure 2.4 illustrates the new distribution of language we obtain using that method.



Notes: This figure plots the average number of tweets collected per day using our best collection method in the 20 most frequent languages on Twitter. The language metadata is provided by Twitter. "und" stands for "undefined" language.

Figure 2.4: Average number of tweets per day in each language collected using the Sample API combined to our best collection method

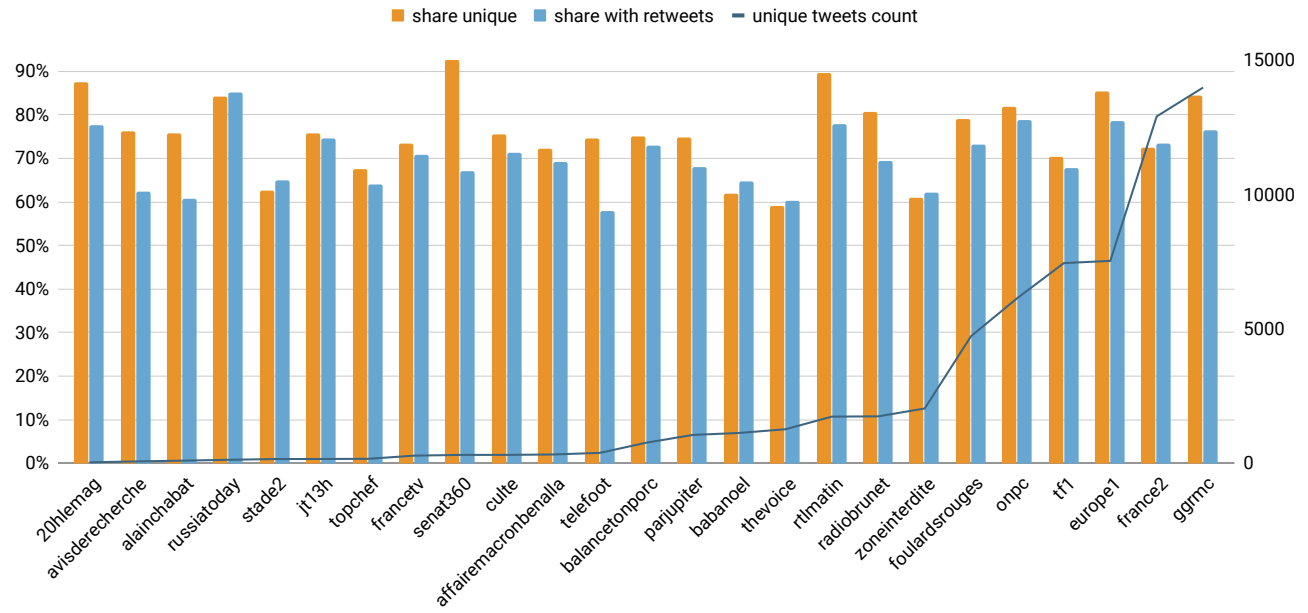
2.3.5 Evaluate the share of collected tweets

The tweet collection tool has been running from June 2018 to the present day, allowing us to store around 3 billion tweets. This long collection period allowed us to compare our corpus with other French tweet corpora collected during that time. Two teams were kind enough to give us access to part of their data: the French Internet legal deposit at the INA, and the Médialab at Sciences Po Paris. We also used metrics from Twitter (the number of tweets per user) to estimate the percentage of tweets that we collected, and conversely, the number of tweets we missed.

Corpus provided by the French Internet legal deposit (DLWeb)

The French Internet legal deposit department at the INA (DLWeb) is in charge of archiving French websites related to audiovisual media (television, radio, web TVs). As part of this mission, the team collects tweets concerning audiovisual media and therefore regularly updates a manually curated list of hashtags and Twitter accounts to be captured using the Filter API. The DLWeb team provided us with the tweets collected for 25 of these hashtags over the course of a month (248,037 tweets), and we counted how many of these tweets had been collected over the same time period using our method. The results are presented in Figure 2.5. On average, we collected 74% of the tweets from the DLWeb, and 78% if we exclude retweets. Original tweets are better captured than retweets by our collection method, because each retweet allows us to archive the original tweet to which it refers. Therefore, we only need to capture one of the retweets of a tweet to get the original tweet. Retweets, on the other hand, are not retweeted, so we lose any chance of catching them if they were not obtained at the time they were sent.

It should also be mentioned that the DLWeb does not guarantee the collection of all tweets containing a given hashtag. Capturing tweets with specific hashtags allows the DLWeb to obtain a larger share of the tweets containing these hashtags than we do, but we also captured some tweets that they did not collect. We found 11,595 tweets



Notes: This Figure plots the share of tweets from the DLWeb that we were also able to capture using our collection method. Blue columns represent the ratio for all tweets, yellow columns represent the ratio for original tweets (*i.e.* retweets excluded). The grey line shows the number of original tweets (*i.e.* retweets excluded) captured by the DLWeb for each hashtag. Tweets were collected from December 1st to December 31st, 2018.

Figure 2.5: Share of DLWeb tweets captured using our collection method for a set of 25 hashtags

from our collection (*i.e.* 4.7% of the DLWeb collection) containing one of the 25 selected hashtags that were not present in the DLWeb collection. Overall, since Twitter does not communicate on this subject, it is difficult to evaluate precisely the reasons why the relative performance of the collection methods varies depending on the hashtag. The volume collected by the DLWeb may depend on the popularity of the hashtag, and the popularity of other collected hashtags during the same time period.

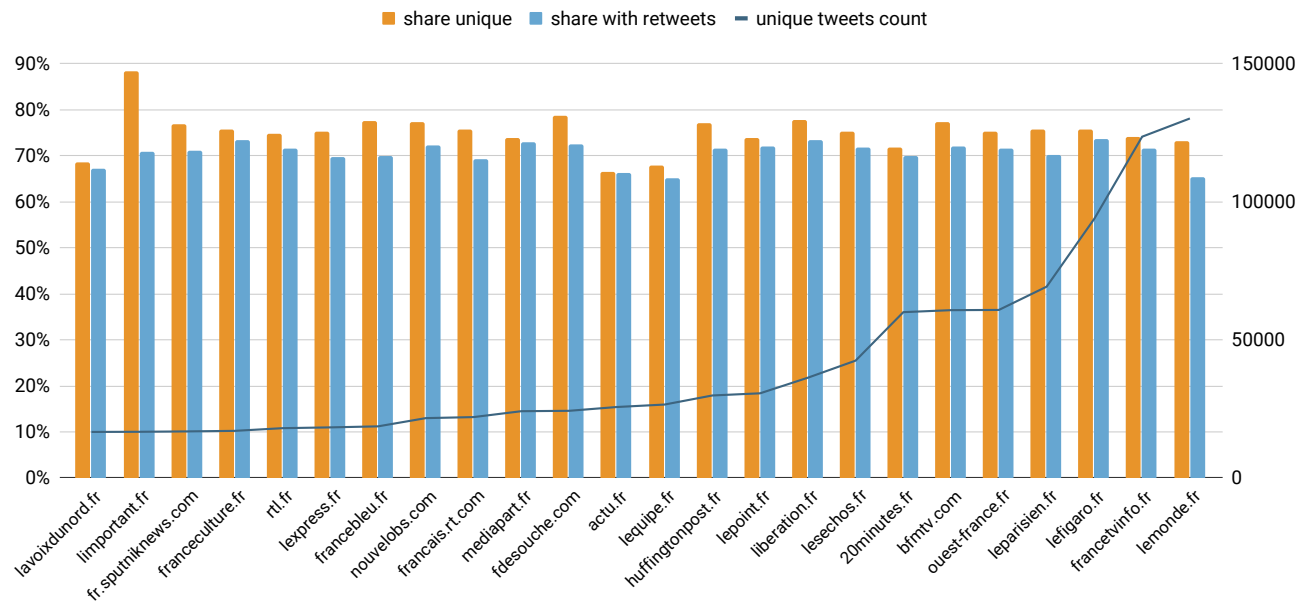
In order to validate our collection method with a more stable comparison dataset, we computed the same metrics on a second corpus, presented in the next section.

Corpus provided by the Médialab team

We compared our collection of tweets with the corpus of tweets built by Cardon et al. [2019], which consists of tweets containing URLs from a curated list of 420 French media sources. The Médialab team uses different API endpoints to search for tweets that contain one of the selected domain names, including the Filter API but also the Search API[8] that returns tweets matching a query within the last 7 days. They gave us access to a part of their dataset, namely all tweets collected between December 1st and 31st, 2018, amounting to 8.7 million tweets.

Among these tweets, we only kept those in French, i.e. 7.3 million tweets. Our sample contains 70% of these tweets in French, and 74% if we exclude retweets. Figure 2.6 shows the percentage of collected tweets for the most

[8] https://developer.twitter.com/en/docs/tweets/search/api-reference/get-search-tweets



Notes: This Figure plots the share of tweets from the Médialab that we were also able to capture using our collection method for the first 25 domain names in terms of original tweets in their dataset. Blue columns represent the ratio for all tweets, yellow columns represent the ratio for original tweets (*i.e.* retweets excluded). The grey line shows the number of original tweets (*i.e.* retweets excluded) captured by the Médialab for each domain. Tweets were collected from December 1st to December 31st, 2018.

Figure 2.6: Share of tweets from the Médialab also captured using our collection method for 25 domain names

important domain names in terms of number of original tweets (*i.e.* retweets excluded) in the Médialab dataset. The Médialab dataset appears to be more complete than the collection from the DLWeb, probably due to the fact that several collection methods are combined: we found only 223,457 tweets from our collection (i.e. 2.6% of the Médialab dataset) containing one of the selected domain names that were not present in the Médialab dataset.

Evaluate the number of tweets per user

A third method can be used to compare our sample with the real number of tweets in French emitted on Twitter. It is based on the total number of tweets sent by a user since the account creation, a metadata that is provided by Twitter for every new tweet. With this metric, we can get an estimate of the total number of tweets emitted by a given user between two of its tweets. We can then compare this value with the number of tweets actually collected for that user.

In practice, we selected the tweets of all users who had written at least 3 tweets in 3 months (from July 1st to September 30th, 2018), retweets included. In order to select users that write mostly in French (tweets written in other languages are not collected with our method), we used the OpenStreetMap API to locate users depending on what they indicate in the ”location” field. We obtained 920,000 users localized in France who emitted 241 million tweets in 3 months, according to the ”number of tweet” field. With our collection method, we captured 147 million tweets from these users, *i.e.* 61% of the real number of emitted tweet. We found the same percentage with the

sample of users who geolocate their tweets in France (27,000 users). This method gives us a high estimate of the real number of emitted tweets in French, since some of these users probably write in other languages than French, even if they are located in France.

All three comparison methods have their flaws; however, they produce close results. We can therefore conclude that we collected between 60% and 75% of all tweets in French sent on Twitter. We were therefore able to build our annotated dataset with the guarantee that our sample of tweets is representative. The next section presents the annotation procedure.

2.4 Tweet annotation

We built our event detection corpus based on tweets collected with our collection method from July 15th to August 6th, 2018. During this period, we collected 38 million original tweets (retweets excluded). We annotated these tweets depending on their relation to Twitter events and media events that took place in France at the time of the annotation.

2.4.1 Media event selection

To select media events, we decided to draw events randomly from the hundreds of events described in the French general information media every day. We drew press articles on a daily basis from July 15th to August 6th, 2018, for a total of 23 days. We did not want to use any automatic detection method to generate events from the collected tweets, since it may have biased the results of our evaluation tests (detection methods similar to the one used to generate events in the test set may be advantaged). In addition, we did not use Wikipedia to select important events (as is the case in McMinn et al. [2013] and Petrović et al. [2012]), considering that "an event detection system should also be able to detect newsworthy events at a smaller scale" [Hasan et al., 2018]. In particular, we wanted to be able to detect events on social media than never became mainstream media events.

In practice, we drew 30 events a day, two thirds from the Agence France Presse (AFP), which is the third largest news agency in the world, and one third from a pool of major French newspapers (*Le Monde*, *Le Figaro*, *Les Échos*, *Libération*, *L'Humanité*, *Médiapart*). This selection method has the advantage of giving "big" events a higher chance of being selected, since they are covered by all news outlets, while also letting relatively "small" events emerge. Duplicates, i.e. articles covering the same event, were manually removed. We considered 30 events to be the maximum number of events the annotators could process in one day. In reality they did not have time to annotate most of them, and only processed the beginning of the list each day. Since new events were drawn every day, events that continued over several days were considered as separated events at the time of the annotation. We then grouped together these daily events manually.

2.4.2 Twitter events selection

Since our final objective is to measure differences in the coverage of events by news media and Twitter users, we did not want to miss important events in the Twitter sphere that would receive little coverage by traditional news media. We therefore monitored the trending terms on Twitter by detecting unusually frequent terms every day.

We chose a metric called JLH, which is used by Elasticsearch[9] to identify "significant terms" in a subset of documents (*foreground set*) compared to the rest of the collection (*background set*). It simply compares for each term t the frequency of occurrence in the foreground set (p_{fore}) and in the background set (p_{back}). This metric is computed as:

$$f(t,d) = \begin{cases} (p_{fore}(t,d) - p_{back}(t))\frac{p_{fore}(t,d)}{p_{back}(t)} & if\ p_{fore}(t,d) - p_{back}(t) > 0 \\ 0 & elsewhere \end{cases}$$

Where $p_{fore}(t,d) = \frac{tf(t,d)}{u_d}$ (the number of different users mentioning term t on day d divided by u_d the total number of users tweeting on day d) and $p_{back}(t) = \frac{tf(t)}{U}$ (the number of distinct users mentioning term t in the total collection divided by U the total number of users, measured by the number of different authors of tweets in our collection). We did not use a standard tf-idf metric because it gave poor results when identifying bursting terms for a given day.

We computed the 20 terms with the best JLH scoring every day and went on Twitter to discover the underlying events causing a burst of these terms. We were then able to group together terms related to the same event. For example the terms "afcbom", "bournemouth", "bouom", "afcbournemouth" - all related to a soccer match between the Association Football Club Bournemouth (AFCB) and the Olympique de Marseille (OM) - were grouped together. We then excluded any events:

- that were artificially amplified using automatic tools. In particular, the Q&A website Curious Cat was used to post the same questions ("Where do you see yourself in tens years from now?", "What is your favorite movie?") to all Twitter users registered on Curious Cat. Many of them responded using the terms of the question ("My favorite movie is...") causing a burst in the frequency of those terms.

- that had already been drawn from the media events selection process.

Once the media events and the Twitter events were selected, the annotators' work could begin. In the following section, we explain the annotation procedure.

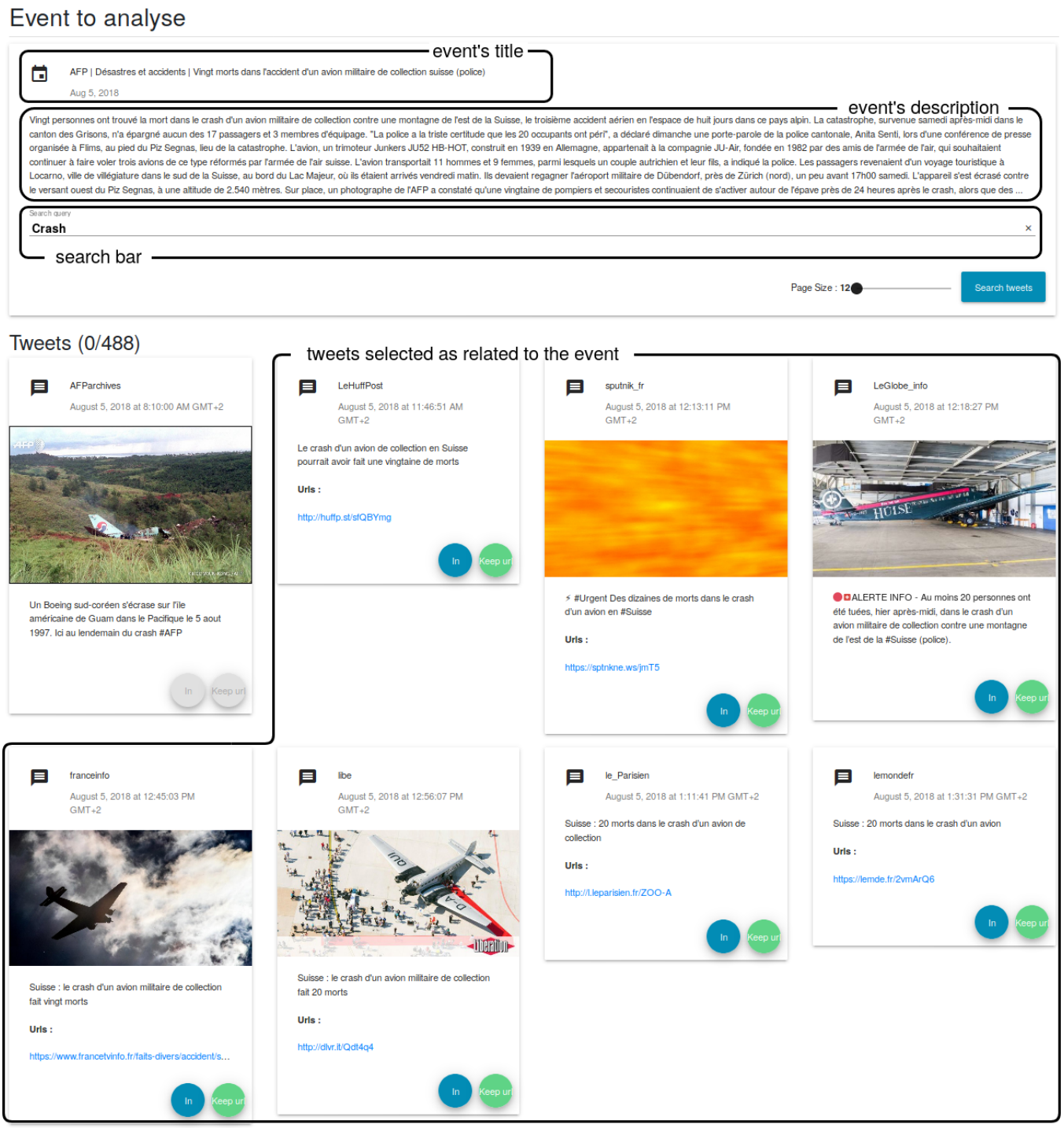


Figure 2.7: View of the annotation interface

2.4.3 Annotation procedure

User interface

We developed a user interface (see Figure 2.7) presenting each event in the form of a title and a description text. For media events we used the title and the first paragraphs of the drawn corresponding press article. For Twitter events the title constituted the bursting terms detected with the JLH scoring and the description was a tweet manually selected because it described the event clearly. Under the title and the description, a search bar was presented. The user could use that bar to enter keywords and find the collected tweets containing those exact keywords. 12 tweets per page were displayed, starting with the most retweeted one. The user could select or deselect the tweets she considered related to the event. If the tweet contained a URL, the user could click or unclick a button under the tweet to indicate whether the linked page was related to the event as well. Once the user had read all twelve tweets and selected those related to the event, the user could submit her answers and access the next 12 tweets. Displayed tweets were not pre-selected by our program depending on their content. We only excluded retweets since it would only have displayed the same tweet several times, and displayed tweets emitted on the day of the event.

The interface also made it possible to review tweets annotated by other users: once an annotator was done with an event, she could access another page displaying the same event (same title, same description) and the tweets seen by other annotators in relation to the event. The user had to go through all these tweets and annotate them, without knowing if those tweets were marked as relevant or irrelevant to the event by the other users.

Annotation task

Three political science students were hired for a month to annotate the corpus. All three of them were Twitter users and had a good knowledge of news media and the French political landscape. Every day they were presented with the new list of events. On July 16th, 2018, they began annotating events from July 15th. The first day of annotation was not included in the final dataset and served as a day of adaptation. Since the annotators did not work on Saturdays or Sundays, some days between July 15th and August 15th could not be annotated. We made the choice to annotate over a continuous period of time, from July 16th to August 6th.

For every event, they were asked to search for related tweets on the user interface, using a large variety of keywords. We insisted on the importance of named entities (persons, locations, organizations) and on the specificity of Twitter (one person could be referred to using her real name or her Twitter user name, for example). Like McMinn et al. [2013], we asked the annotators to mark tweets as related to the event if the tweet made reference to that event, even implicitly. It became clear that annotators could not treat more than 20 events a day, and often no

[9] https://www.elastic.co/guide/en/elasticsearch/reference/current/search-aggregations-bucket-significantterms-aggregation.html#_jlh_score

more than 10 events, depending on the volume of tweets generated by each event. Indeed, some major events would have required days of work in order to be fully treated. We therefore instructed them not to spend more than one hour on each event. This of course had an impact on the maximum number of tweets per event that could be annotated.

In order to ensure that the annotators worked on the same tweets, we stopped the first annotation task after four hours of work every day, and asked them to go to the second part of the user interface, where they could find tweets already seen by at least one of the other annotators. They then had to annotate those tweets without knowing the judgment made by the others. In this way, we could make sure that all tweets would be reviewed by all three students.

Maximizing annotation efficiency

The annotation interface was designed to take advantage of the annotators' intelligence and avoid repetitive tasks. Thus, it seemed unnecessary to have them annotate tweets that contained exactly the same text. For tweets containing the same url, we added a "keep url" checkbox (see the green buttons on Figure 2.7), which was checked by default. If annotators felt that the url present in the tweet did not refer to content related to the event, they had to uncheck the box. For most other tweets (for which the url did refer to an event-related article), tweets containing the same url were no longer shown to the annotators in the interface. Here are all the rules we used to avoid repetition. For a given event:

1. tweets longer than 4 words containing the same text as an already-annotated tweet were no longer displayed;

2. tweets containing the same url as an already-annotated tweet were no longer displayed, unless the "keep url" checkbox was unchecked for that tweet;

3. retweets and responses to a tweet, as well as tweets quoting a previously annotated tweet were not displayed.

These rules played an important role to improve the quality of the corpus in terms of tweet diversity within a given event. The following section details some quality evaluation metrics.

2.5 Evaluation of the created corpus

In this section, we first present the annotator agreement and discuss possible reasons for differences in agreement. We then describe the characteristics of the corpus, including the number of events, their distribution across different categories, and the number of tweets per event.

	annotated	linked to daily event	linked to macro event
tweet count	137757	95796	95796
event count	327	327	257
mean	428	296	376
std	434	449	1324
min	7	2	2
25%	150	30	22
50%	280	100	76
75%	494	344	241
max	2913	2906	18534

Note: *annotated* tweets were found by annotators using search keywords but not necessarily considered as *linked to an event. Macro events* were built manually after annotation by grouping *daily events* together.

Table 2.2: Distribution of the number of tweets per event.

2.5.1 Corpus characteristics

In total, 137,757 tweets were annotated (found by annotators using search keywords), and 95,796 tweets were considered linked to one or several events by at least 2 users. 327 daily events were selected, including 31 “Twitter events” (detected using the term frequency on Twitter) and 296 “media events” (drawn randomly in our collection of online news). Since the events were discovered day by day, we manually merged some of them to obtain 257 “macro-events”. A macro-event contains 376 tweets on average. 0.5% of the tweets were linked to several events. Additional descriptive statistics are presented in Table 2.2.

To describe the distribution of events across categories we used the classification by the French news agency AFP. AFP dispatches are labeled using the IPTC Information Interchange Model[10] Media Topics. This taxonomy is used internationally by numerous media companies to apply metadata to text, images and videos. The distribution of events across the 17 top Media Topics is detailed in Table 2.3. Among the 326 selected events, only 209 were drawn from the AFP and had a label. For the remaining 117 events (from other French press outlets or from Twitter events) we attributed a label manually.

2.5.2 Intra-event diversity

The quality of the dataset should also be measured in terms of variety of the tweets within a given event: what proportion of the tweets are simply the headline of the same linked article, for instance? Indeed, the proportion of tweets containing a url is high among the annotated tweets (71%), compared to the entire corpus of 38 million tweets (36%). However, due to the design of the annotation interface (see Section 2.4.3), very few annotated tweets share the same url. Only 4.6% of tweets linked to an event share the same url as another tweet in the same event. However, we did not anticipate during annotation that different urls can be linked to the same page. After redirection, 9.3% of tweets share the same page as another tweet in the same event, but 95% of pages are shared less than 3

[10] https://en.wikipedia.org/wiki/IPTC_Information_Interchange_Model

IPTC category	Event count
arts, culture, entertainment and media	14
conflict, war and peace	25
crime, law and justice	76
disaster, accident and emergency incident	11
economy, business and finance	53
education	4
environment	5
health	3
human interest	6
labour	3
lifestyle and leisure	2
politics	46
religion and belief	1
science and technology	2
society	17
sport	56
weather	3
Total	327

Table 2.3: Distribution of events across the 17 top IPTC Information Interchange Model Media Topics.

times in a given event.

2.5.3 Annotator agreement

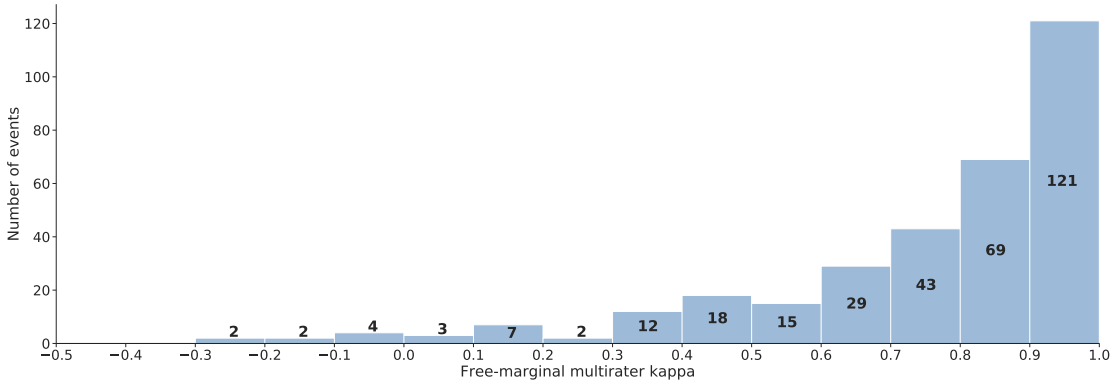
Annotator agreement is usually measured using Cohen's kappa for two annotators. Here we chose to hire three annotators in order to have an odd number of relevance judgments for each tweet. In the case of several annotators, Randolph [2005] recommends using Fleiss' kappa [Fleiss, 1971] in case of "*fixed marginal distributions*" (annotators know in advance the proportion of cases that should be distributed in each category) and free-marginal multirater kappa [Randolph, 2005] if there is no prior knowledge of the marginal distribution. Indeed, we experienced some odd results using Fleiss' kappa on our corpus, in particular for events with a strong asymmetry between categories (when a large majority of tweets were annotated as "unrelated" to the event of interest, or the opposite). We hence decided to use free-marginal multirater kappa, which is also the measure used by McMinn et al. [2013].

If one denotes P_o the proportion of overall agreement among annotators, P_e the proportion of agreement expected by chance, free-marginal multirater kappa is expressed as

$$\kappa_{free} = \frac{P_o - P_e}{1 - P_e}$$

with

$$P_o = \frac{1}{Nn(n-1)}((\sum_{i=1}^{N}\sum_{i=j}^{k} n_{ij}^2) - Nn)$$



Lecture Note: In our corpus, 121 events have a free-marginal multirater kappa higher than 0.9, 69 events have a free-marginal multirater kappa between 0.8 and 0.9, etc.

Figure 2.8: Distribution of the events depending on annotators' agreement, measured by free-marginal multirater kappa

and

$$P_e = \frac{1}{k}$$

where N is the total number of cases (here the number of tweets to annotate), n is the number of annotators and k the number of categories (two in our case: *relevant* or *not relevant* to a given event). Its values vary between -1 and 1. A value of 0 indicates a level of agreement that could have been expected by chance, and a positive value indicates a level of agreement that is better than chance. Negative values indicate worse agreement than expected by chance.

In our corpus, $\kappa_{free} = 0.79$ which indicates a strong level of agreement among annotators. We also computed the κ_{free} value for each individual event (taking into account all tweets that have been read by annotators while working on this event). Figure 2.8 describes the distribution of events depending on the κ_{free}. We observe that for some events, the agreement is very low: 8 events have a negative κ_{free} value, and 12 events have a κ_{free} value between 0 and 0.3.

We asked the annotators to come together and re-read the events on which their agreement was particularly low, in order to understand why they did not annotate tweets in the same way. The students admitted some errors in the annotation for 4 of the 17 examined events. For the other events, they explained that they had different views of the events' scope: for example, one article reported the fact that President of Nicaragua, Daniel Ortega, refused to resign in a context of severe crisis in his country. Two of the annotators included in the event those tweets related to the crisis in Nicaragua. One annotator restricted the event to the statement of Daniel Ortega. Faced with these differences in opinion we could decide to remove from the corpus the tweets where the annotators disagreed, or to remove events with a very low kappa. However we found it interesting to see how an algorithm behaves in such

borderline cases.

2.6 Conclusion

It is rare in a Machine Learning thesis to devote so much work to the constitution of a corpus: instead, the standard recommendation is to work on existing datasets recognized by the scientific community, and to evaluate performances on a common benchmark. However, there are several reasons why I took the time to set up a reliable collection method, and spent the summer of 2018 annotating tweets.

As I showed in Section 2.2, publicly accessible tweet corpora for event detection are rare, they are generally quite small and do not exist in French. I hope that I have partly remedied this lack, having published my corpus with the ids of the 38 million tweets collected during the summer of 2018. Given the number of annotated tweets and the general quality of the annotation (see Section 2.5) it may be a useful resource for researchers in the future, especially French-speaking ones.

Working on tweets in French was not only an incentive to build my own corpus, but also to set up my own method of collecting tweets, which I detailed in Section 2.3, in order to obtain a sufficient volume of tweets in French. By ruling out the sole use of the Sample API, which ensures a random distribution of collected tweets, I also had to find methods to evaluate the resulting sample, in order to guarantee its representativity. This work allows me, in the continuation of this thesis, to guarantee the reliability of my analyses.

Finally, the architecture for collecting and indexing tweets, implemented in 2018, has been running continuously until today. It is a tool for INA researchers that will serve even after my thesis is concluded. It is currently being used by Nicolas Hervé for a research project on media intensity on the subject of COVID-19 and will hopefully give rise to many other studies.

Chapter 3

Detecting Twitter events

3.1 Introduction

Twitter has been used to detect or predict a large variety of events, from flood prevention de Bruijn et al. [2017] to stock market movements Pagolu et al. [2016]. However, the specificity of social network data (short texts, use of slang, abbreviations, hashtags, images and videos, very high volume of data) makes all ”general” detection tasks (without specification of the type of topic) very difficult.

Many works on event detection are actually focused on burst detection (detecting topics such as natural disasters, attacks, etc., that cause an unusual volume of tweets), and do not attempt to assess the relative size of events. We seek to detect *all* events, both those that generate a high volume of tweets and those that are little discussed, and to group together *all* tweets related to the same event. With this definition in mind, the topic detection and tracking task is conceptually similar to clustering. Given the size of our tweet collection, the chosen clustering method has to be extremely time-efficient.

As well as deciding on the event detection algorithm, we also have to consider the type of tweet representation: should we only use the text of the tweets, or should we consider the tweet as a multimodal document, composed of text and/or images, videos, hashtags?

In the field of language processing, recent works have made it possible to reach a performance level close to human capacity in several tasks, particularly when evaluating the semantic similarity between two sentences[1]. However, these advances, based on the training of neural networks on very large corpora of texts, may not always be suited to our task. Indeed, despite rapid progress in recent years in the adaptability of language processing (the GLUE benchmark Wang et al. [2018] consists of 9 different tasks, and the models are evaluated according to their average performance on all these tasks), it remains difficult to adapt these models to new tasks. Any transformation of the initial task requires fine-tuning a deep neural network on (at least) a few thousand sentences, which involves

[1] See the results on the GLUE benchmark: `https://gluebenchmark.com/leaderboard`

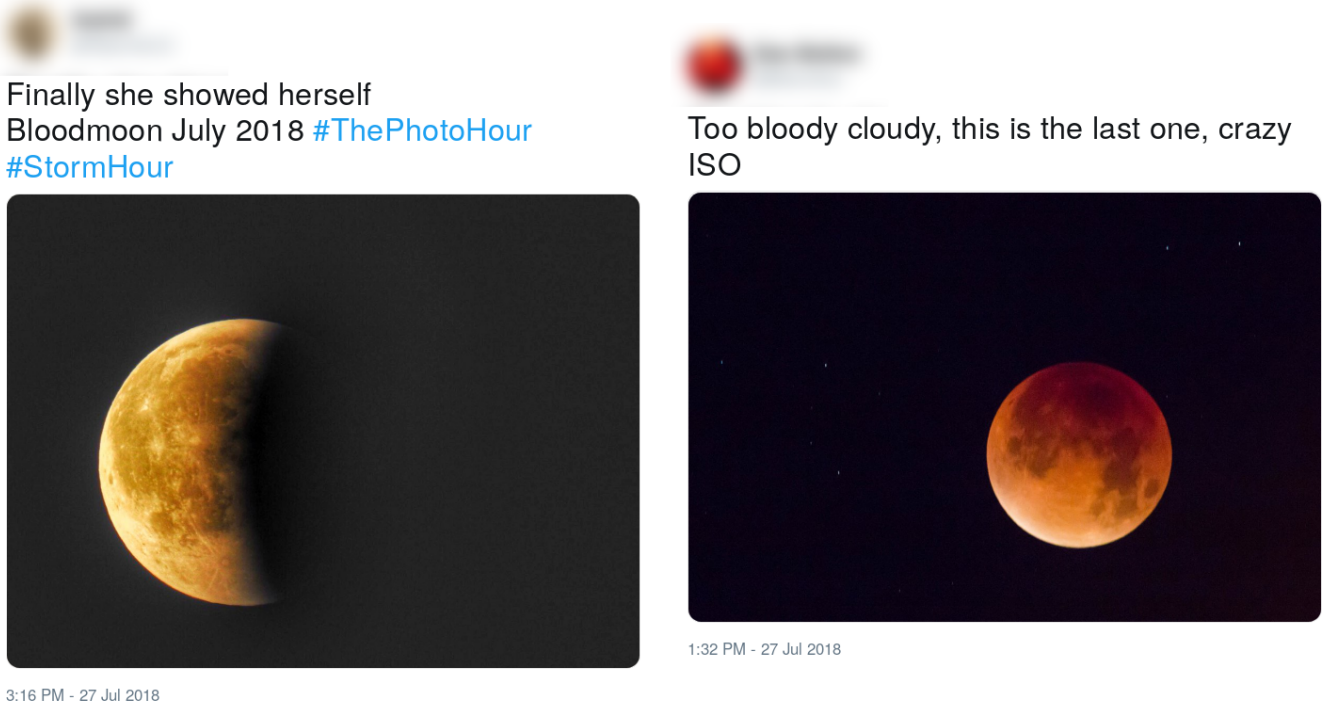


Figure 3.1: Example of a topic (the July 2018 lunar eclipse) where multimedia contents provide critical information for topic detection

hours of manual annotation to create a suitable dataset. In our work we focus on short-text topic similarity (evaluate whether two sentences / short texts address the same subject), which in some cases differs from semantic similarity (evaluate whether two sentences mean the same thing) evaluated in GLUE, and we do not have a corresponding training dataset. Besides, even on a strictly identical task, the performances announced in the literature are perfectly reproducible only on English language corpora.

Finally, most of these models are designed to be used as input to end-to-end systems. For example, to calculate a similarity score between sentences with BERT [Devlin et al., 2018], it is necessary to process each couple of sentences instead of each sentence. Using the example proposed in [Reimers and Gurevych, 2019], to find the two most similar sentences in a corpus of $n = 10,000$ sentences, the number of treatments to be performed is $n\frac{(n-1)}{2} = 49,995,000$ operations, which represents approximately 65 hours of processing with BERT on a V100 GPU. These architectures do not apply well to information retrieval systems that involve comparing hundreds of thousands of sentences. For clustering or information retrieval tasks, it is more efficient to represent each sentence in a vector space where similar sentences are close (so-called *embeddings*), and then apply conventional distance measurements (cosine, euclidean distance, etc.).

With regard to images, recent progress in the field of visual content description due to deep neural networks provides new ways of representing social media documents by including rich visual features. There are cases where image provides decisive information for event detection (see Fig. 3.1). However, dealing with image tweets

often requires a contextual knowledge external to the document. Without this knowledge, image can become a source of error for event detection algorithms compared to text alone.

In this chapter, we show that the First Story Detection clustering algorithm is the most suitable for our event detection task. We show that the best vector representation of tweets for the FSD algorithm is the tf-idf approach. We compare this algorithm with another standard algorithm (DBSCAN) and a method from the literature (DMM). We are also testing a two-step clustering method, to group tweets according to their textual content and then according to their visual content.

We conduct our demonstration by starting with a review of the state of the art, then a presentation of the approach we propose. We then detail several concurrent algorithms tested, and we review the different experiments carried out with these algorithms. Finally, we present our results in detail.

3.2 State of the art

3.2.1 Twitter event detection approaches

We divide the event detection methods into three types of approaches: term-weighting-based approaches, topic modeling, and clustering. This is also the classification used in the survey on real-time event detection by Hasan et al. [2018]. The last approach, clustering, is dealt with in more detail in this state of the art, as it is the one we chose to implement for our own event detection tool.

Term-weighting-based approaches

These approaches rely on tracking the terms likely to be linked to an event (often due to a high frequency of some terms during a given time window). They usually return a list of the top k trending events on Twitter, which does not meet our objective of detecting events in an exhaustive manner.

The event detection system TwitterMonitor [Mathioudakis and Koudas, 2010] detects bursty keywords in the Twitter stream by comparing their term frequencies in previous periods to current term frequency. Bursty keywords are then grouped together in “trends” depending on their co-occurrences in recent tweets.

EnBlogue [Alvanaki et al., 2012] measures the correlation of hashtag pairs within a given time window. Emergent topics are then detected among the pairs with the highest shift in their correlation. EnBlogue produces an overall scoring of the topics depending on the shift in correlation and on the total popularity of each topic. This score is smoothed in order to give a higher rank to new topics.

MABED [Guille and Favre, 2015] does not only use the textual content of the tweets: the frequency at which users interact with each other using “mentions” (i.e. the name of another Twitter account preceded by “@”) is also taken into account to detect events. The system models the number of tweets that contain word t and at least

one mention during a given time-window as a binomial distribution. It detects positive anomalies if the creation of mentions associated to word t is strictly greater than the expectation of the model. The magnitude of impact of an event on a time interval I is computed by integrating the anomaly function on the interval I. For each word associated with a mention, the system finds the interval I that maximizes its magnitude of impact. After other steps of event description and duplicates removal, the events with the highest impact are returned.

The Twitter Live Detection Framework (TLDF) [Gaglio et al., 2016] modifies the Soft Frequent Pattern Mining (SFPM) algorithm [Petkos et al., 2014a] to adapt to the dynamic nature of tweets. The authors use the relevance score of term t proposed for SFPM: the ratio of the likelihood of appearance of the term in the current time-window and in a reference set of tweets. This relevance score is combined with a parameter boosting the score of named entities and multiplied by the tf-idf of term t. Moreover, the size of the detection time-window is not fixed, but is controlled by a sigmoid function depending on the volume of emitted tweets at a given moment.

Topic models

Topic models are widely used techniques in the natural language processing field to discover the topical structure from a corpus of textual documents (news articles, scientific papers, tweets, etc.). Latent Dirichlet Allocation (LDA) is the most common one [Blei et al., 2001]. In this model, each document is considered a mixture of different topics drawn from a topic distribution. In the case of topic detection in a collection of tweets, LDA has several drawbacks: 1) the model does not take into account the fact that topics change over time; 2) the number of topics has to be known in advance; 3) it assumes that a document is a mixture of several topics, which is very rare in short texts such as tweets. There is, however, a vast literature working on strategies to overcome these limitations.

The first point has been addressed by Blei and Lafferty [2006]; however, they assume that the number of topics remains the same over periods, whereas in a stream of tweets, topics can emerge and disappear in each new period.

With regard to the number of topics, some methods exist to estimate the optimal parameter k [Brunet et al., 2004; Griffiths and Steyvers, 2004; Arun et al., 2010; Greene et al., 2014]. However these methods rely on re-generating topic models for each candidate $k \in [k_{min}, k_{max}]$. This can be achieved for a small number of topics ($k_{max} < 100$), but testing each k in a range $[2100, 10500]$ (between 100 and 500 events a day in our corpus of 21 days) is not an option.

Regarding the third assumption, a number of articles have specifically tackled the issue of applying topic models to short texts. The recent survey by Likhitha et al. [2019] summarizes the topic modeling techniques used to find topics within short text documents.

Based on the assumption made by Nigam et al. [2000] that each document is assumed to be generated from a single topic, i.e. the words within a document are all sampled from the same topic distribution, Yin and Wang [2014] propose the Dirichlet Multinomial Mixture (DMM) model. The generative process of DMM can be described

as follows:

1. Choose a proportion of topics $\theta \sim Dirichlet(\alpha)$.
2. For each topic $k \in \{1, \ldots, K\}$:

   Draw a distribution of words linked to this topic $\phi_k \sim Dirichlet(\beta)$.
3. For each document $d \in \{1, \ldots, D\}$:

   (a) Draw a topic $z_d \sim Multinomial(\theta)$.

   (b) For each word w:

   Draw a word $w \sim Multinomial(\phi_{z_d})$

Yin and Wang [2014] provide a collapsed Gibbs Sampling algorithm (GSDMM) to approximate the hidden variables in this generative process. The interest of this method in our case is that over iterations, some clusters (topics) become empty, while others progressively gather more documents. This means that the algorithm can approximate the number of documents in the collection, as long as the initial number of topics (K) is large enough at initialization. DMM is therefore very relevant to our study because it addresses both problem 2 (number of clusters unknown) and problem 3 (only one topic per tweet).

This simple and efficient method has been adapted and improved by several authors: Nguyen et al. [2015] integrates Word2Vec embeddings into DMM by replacing the topic-word Dirichlet multinomial component with a draw from a Bernoulli distribution to determine whether the Dirichlet multinomial component or a word embedding component will be used to draw the word w. Li et al. [2017] propose PDMM, an extension of the DMM model allowing short texts to be generated from one to three topics and drawn from a Poisson distribution, and GPU-PDMM, which exploit word-embeddings to enrich the latent topic learning with external semantic relations.

However, these advances seem to increase calculation times by several orders of magnitude. Li et al. [2017] provide a comparative table of the time efficiency of these different models, which we compare with their accuracy in Table 3.2. According to their experiments (all models are implemented in Java), the extended DMM models seem to multiply by 10 or even 100 the calculation time of plain DMM, on the BaiduQA dataset[2] for an increase in accuracy of 2 to 3 points. Therefore, we decided to use DMM as comparison model with the First Story Detection algorithm rather than more recent approaches.

Dynamic clustering

Dynamic clustering approaches do not require a fixed number of clusters. These methods are well fitted for discovering clusters of textual documents dynamically as new documents are added to the collection. For this type of

[2] a corpus of 648,514 questions from a Chinese Q&A website, containing 35 labels.

Model	*K=40*	*K=60*	*K=80*
DMM	0.355	0.566	0.843
LF-DMM	13.0	20.5	31.8
PDMM	11.6	11.7	11.8
GPU-PDMM	37.8	40.7	45.2

(a) Time cost (in seconds) per iteration of each model

Model	*K=40*	*K=60*	*K=80*
DMM	0.523	0.548	0.553
LF-DMM	0.424	0.449	0.487
PDMM	0.553	0.562	0.577
GPU-PDMM	0.545	0.569	0.583

(b) Average classification accuracy of each model

Figure 3.2: Comparison of time efficiency and classification accuracy of several DMM-adapted models with different numbers of topics K on the BaiduQA dataset. These values are provided by Li et al. [2017]

.

clustering, the algorithms that are often used take into account both the thematic similarity of the documents and their temporal proximity, in order to avoid grouping in the same cluster tweets sent at very distant times. This type of approach generally uses the state-of-the-art First Story Detection (FSD) [Allan, 2002] algorithm as a reference method. In this method (see Algorithm 1), documents are represented using tf-idf (see Section 3.2.2 for a presentation of tf-idf encoding) and their similarity is evaluated using cosine similarity. The cosine similarity of two vectors a and b is computed as follows:

$$cos(a, b) = \frac{a \cdot b}{||a||||b||} \tag{3.1}$$

We denote $\delta(a, b)$ the cosine distance between a and b, which is defined as:

$$\delta(a, b) = 1 - cos(a, b) \tag{3.2}$$

Each new document joins the cluster of its nearest neighbor in the collection. If the cosine distance to the nearest neighbor is higher than a pre-defined threshold t, a new cluster is created containing the new document. The oldest documents are dropped from the collection at regular intervals. This insures that each new document is only compared to the most recent documents.

Existing works on tweet incremental clustering have adapted this reference method initially developed for streams of news (such as RSS streams) to process much higher volumes of documents, and very short texts. Indeed, the maximum size of a tweet is 280 characters (140 characters before 2018). Changes in that baseline method are made either in the type of text representation or in the clustering algorithm itself. A step of noise filtering is also frequently added before the clustering step, to distinguish event from non-event tweets (according to Liu et al. [2016], the proportion of event tweets in well-established corpora such as that of McMinn et al. [2013] is less than 0.2%). This section explores the variations introduced in the clustering algorithm.

In TwitterStand, Sankaranarayanan et al. [2009] first perform a filtering step, which classifies tweets as either “junk” or “news” using a naive Bayes classifier. Next, their online clustering algorithm associates a vector to each cluster, which is composed from the contained tweets’ terms weighted with tf-idf. Each new tweet is represented

Algorithm 1 "First Story Detection"

Input: threshold t, window size w, corpus C of documents in chronological order
Output: thread ids for each document

```
T ← [] ; i ← 0
while document d in C do
    if T is empty then
        thread_id(d) ← i
        i ← i + 1
    else
        d_nearest ← nearest neighbor of d in T
        if δ(d, d_nearest) < t then
            thread_id(d) ← thread_id(d_nearest)
        else
            thread_id(d) ← i
            i ← i + 1
        end if
    end if
    if |T| ≥ w then
        remove first document from T
    end if
    add d to T
end while
```

using tf-idf and compared to the clusters' vectors using a modified cosine distance that accounts for the temporal dimension of clusters. The distance formula is

$$\dot{\delta}(t,c) = \delta(t,c) \times e^{\frac{-(T_t - T_c)^2}{2\sigma^2}}$$

where $\delta(t,c)$ is the cosine distance between tweet t and cluster c, T_t is tweet t's publication time and T_c is cluster c's mean publication time. The distance is only computed for clusters with a mean publication time T_c more recent than 3 days, and that have a word in common with t. If $\dot{\delta}(t,c^*) \leqslant \epsilon$ (where c^* is the nearest cluster to t), the tweet t is added to the cluster c^*. Otherwise, a new cluster is created. To speed up the search for the nearest cluster, an inverted index of the cluster's words is maintained: for each word w, the inverted index stores pointers to the clusters that contain w. The clustering algorithm in TwitterStand is adapted to the noisy nature of tweets by maintaining a list of reputable sources. A cluster is dropped from the list of active clusters if none of the first k tweets are from a reputable source. The system also deals with fragmentation (the fact that several clusters about the same topic are created) by removing duplicate clusters. If a cluster c' is identified as the duplicate of an older cluster c, c' is marked as "slave" of the cluster c, and c as "master" of c'. Any tweet that should be added to c' is now added to c.

Petrović et al. [2010] speed up the standard FSD algorithm by replacing the nearest neighbor search by an approximate nearest neighbor search using Locality Sensitive Hashing. Instead of searching for the nearest neighbor in the full set of documents, the search is done among a small set S of potential nearest neighbors. The set S is the

set of documents that have the same hash as current document x:

$$S(x) = \{y : h_{ij}(y) = h_{ij}(x), \exists i \in [1 \dots L], \forall j \in [1 \dots k]\}$$

where L is the number of hash tables, and k the number of hyperplanes in each hash table. The hash functions h_{ij} are defined as:

$$h_{ij}(x) = sgn(u_{ij}^T x)$$

the sign of the scalar product of random vector u_{ij} and x. To reduce the risk of failing to find the nearest neighbor (if the nearest neighbor y does not collide with x in the set $S(x)$), if no nearest neighbor is found, the system starts a search for an exact nearest neighbor through an inverted index containing only the most recent documents. To limit the growth of the sets S (also called buckets), the number of documents in a single bucket is limited to a constant parameter. The oldest documents in the bucket are then removed.

The system proposed by Petrović et al. [2010] was subsequently improved by the same authors [Petrović et al., 2012] using public synonym and paraphrase corpora in order to expand the words of the tweets with all their potential synonyms. This approach is a way to compensate for the scarcity of information contained in tweets and to increase the chances of finding a potential neighbor even if there is no tweet containing the same words in the collection.

Becker et al. [2011b] use the same clustering algorithm as Sankaranarayanan et al. [2009], with each new tweet being represented as a tf-idf vector and compared using cosine similarity to the centroid of each cluster (the centroid is the mean weight of all terms in all tweets contained in the cluster). The specificity of their approach lies in the step following the clustering: instead of processing to a noise/event classification at the tweet level, the classification step takes place at the cluster level. They use a wide range of features in order to run a classification algorithm able to distinguish event clusters from noise clusters. The considered features are temporal features (taking into account the frequency of emission of the tweets in the cluster), social features (percentage of messages being retweets, replies, or mentions), topical features (based on the assumption that event clusters have a smaller diversity of topics), and Twitter-Centric features (hashtag usage, presence of multi-word hashtags).

McMinn and Jose [2015] base their event-detection approach on Part Of Speech (POS) Tagging and Named Entity Recognition (NER). Their first step is to extract named entities (persons, locations and organizations) and lemmatized nouns and verbs from each tweets. They then proceed to an aggressive filtering step (95% of the tweets are removed): the system removes retweets, tweets with no named entities, and tweets containing terms associated with noise (“follow”, “watch”, etc.). Then two steps are conducted in parallel: a clustering and a burst detection step. The clustering is based on an inverted index of all named entities: for each new tweet, the tweets containing the same named entities are retrieved from the index. The nearest neighbor is searched for among these tweets using tf-idf representation, cosine similarity and a similarity threshold. If a nearest neighbor is found and does not already belong to a cluster, a new cluster is created containing the two tweets. In parallel, the burst

detection module looks for bursts in the frequency of the detected entities. Once a burst has been detected, the clusters associated with the given entity are associated to it if the average timestamp of the tweets in the cluster is after the initial burst. The clusters associated to one burst form an event. If an entity associated to another event is mentioned in more than 50% of the event's tweets, the two events are merged.

With Reuters Tracer, Liu et al. [2016] propose an event detection system that is also largely based on POS tagging and NER. This system combines a tweet-level noise filtering and a cluster-level noise filtering. We consider as "noise" all tweets and all clusters of tweets that are not linked to an event (as defined by McMinn et al. [2013]). The clustering algorithm itself is quite different from the standard FSD algorithm. New tweets are added to clusters based on three criteria: (1) Retweets of the same tweets are all added to the same cluster. (2) Tweets containing the same url are all added to the same cluster. (3) Finally, a similarity metric with the existing clusters is computed as follows:

$$S_i = aN_e + bN_n + cN_v + dN_h$$

where N_e, N_n, N_v, N_h are the numbers of matching Named Entities, nouns, verbs and hashtags between the tweet and the cluster, and a, b, c, d are learned parameters. Liu and al. explain that their system is primarily based on named entities ("An event is usually defined by *who*, *where* and *what*" according to McMinn et al. [2013]), and that there is no need for high dimensional tf-idf vectors to compute a good similarity function. The obtained clusters, called "unit clusters" are progressively merged into bigger clusters using the same steps (merging based on retweets, urls, then similarity metric) as the unit clustering step. However, the used a, b, c, d parameters are different.

Hasan et al. [2016] use the technique of First Story Detection with LSH developed by Petrović et al. [2010]. However, they observe that the k independent random vectors in a single hash table need to be updated every time the size of the input tf-idf vectors increases (which happens at a high rate in Twitter data since the vocabulary evolves faster than in traditional news media). To alleviate this problem, they propose a technique using random indexing [Sahlgren, 2005]. This aims to represent terms with fixed-size vectors. Each term t is associated with two vectors: an index vector and a context vector. The index vector is a random vector. The context vector has the same size as the index vector and is initialized with zeroes. When a co-occurrence of the term t with another term t' is observed, the context vector of t is updated by adding the index vector of t'. A tweet can then be represented as an average of the vectors of each term in the tweet. This tweet representation is then used to perform the LSH approximate neighbors search of tweet d. However, once the set S of all documents that collide with d in a hash table is computed, the exact nearest neighbor search is done using cosine similarity on tf-idf vectors, rather than random indexing vectors. Hasan et al. [2016] use this method as a first algorithm in order to detect "non-unique" tweets, which are then clustered using a second algorithm close to that of Sankaranarayanan et al. [2009]: this time, the similarity to the clusters' centroids is computed. Hasan et al. explain the use of this two-step clustering method

by the necessity to restrain the number of “one tweet clusters”. A defragmentation step is finally performed in order to merge similar clusters.

In these works, tweets are represented in the form of tf-idf vectors in the vast majority of cases [Sankaranarayanan et al., 2009; Petrović et al., 2010; Becker et al., 2011a; Hasan et al., 2016]. Repp and Ramampiaro [2018] test different types of representation of tweets (Word2Vec average, GloVe average, Doc2Vec, Word2Vec average weighted by the idf of each word). However, these representations are only tested on a classification task, and the best representation (the average of Word2Vec) is then used for clustering. We therefore wanted to update this work by testing recent embeddings, and in particular those developed for representation of sentences. The next section details recent advances in text embeddings, from word embeddings to sentence or short-text embeddings.

3.2.2 Text embeddings

The most commonly used “vectorization” method until the 2010s was the tf-idf, introduced by Sparck Jones [1972]. This is an improvement on the principle “bag of words” vectors Harris [1954], where each document is described by the number of occurrences of the words it contains (”term frequency”). The tf-idf representation uses the same vectors, but weights each of the words in inverse proportion to the number of documents in which it appears.

Word embeddings

The publication of Word2Vec [Mikolov et al., 2013] and GloVe [Pennington et al., 2014], two methods based on the prediction of the context of each word (or the prediction of each word depending on its context), made it possible to create word vectors carrying semantics other than the word frequency in the corpus. However, these representations lost the ability to describe each document by a single vector. To get around this problem, each document is often represented by an average of word vectors.

With ELMo [Peters et al., 2018] a new generation of models appears, allowing words to be represented not only depending on their usual context (the words with which they are frequently used in the training corpus), but also according to their local context: the word representation is specific to a given sentence. This constitutes an important advance in language processing, since a word no longer has a single representation that aggregates its different meanings, but several representations for each of the contexts in which it is used. ELMo is based on a bi-directional LSTM neural network trained to predict the next word in a sequence in both directions (i.e. to predict the next word in a sentence, but also, given the end of a sentence, to predict the word just before it). However, ELMo is not designed to produce sentence embeddings, but rather to be used as input to task-specific neural models. Nevertheless, the authors test the performance of word vectors directly from the first or second layer of their model (which contains three layers) for a disambiguation task by searching for the first nearest neighbor. The results obtained are close to the state of the art.

BERT Devlin et al. [2018] is even more generic than ELMo, because this model does not require a specific architecture for each type of task: it can be fine-tuned to a new dataset by simply adding an output layer. BERT is built with a Transformer-type architecture Vaswani et al. [2017], and pre-trained on two types of pretext tasks: predicting hidden words in a sentence and predicting the next sentence in a text. As in ELMo, the authors of BERT do not intend to create sentence embeddings from their model, but rather to provide an architecture that should be trained differently for each specific task. However, they demonstrate that a simple transfer learning (extraction of word vectors used at the input of a new model without fine-tuning) can match the state of the art for a named entities detection task.

Sentence embeddings

There is a large number of works that attempt to represent sentences by generic vectors that can be used in a wide variety of tasks, especially for transfer-learning. For example, Skip-Thought [Kiros et al., 2015] is based on an encoder-decoder architecture trained to generate the sentences framing a given sentence in a text.

Conneau et al. [2017] show with InferSent, a bi-directional Siamese LSTM network (Siamese means that it takes two sentences as input, but applies the same weights in both parts of the network), that supervised learning provides better results for the creation of generic sentence vectors. InferSent is trained on a classification task using the SNLI dataset Bowman et al. [2015], which contains $570,000$ pairs of English sentences manually annotated in three categories: the first sentence implies the second, the first sentence contradicts the second, or the first sentence and the second sentence are mutually neutral.

With Universal Sentence Encoder, Cer et al. [2018] apply the results of Kiros et al. [2015] and Conneau et al. [2017] by training a Transformer architecture both on unsupervised tasks, as was done for Skip-Thought, and on the SNLI dataset, like InferSent.

Sentence-BERT Reimers and Gurevych [2019] does not provide universal vectors, but a fine-tuning architecture of the BERT model specifically adapted to produce sentence embeddings adapted to certain types of tasks. This model modifies BERT into a Siamese network, with a final layer depending on the type of task on which the network is trained. The authors test their representations on the STS dataset Cer et al. [2017] ($8,628$ pairs of sentences with a similarity score between 0 and 5) by computing a simple cosine similarity score between the vectors associated with each sentence. They show that the best performances on the STS dataset are obtained by a first fine-tuning on SNLI and then a second fine-tuning on the STS training set.

All these embedding methods are potential ways of representing the text of tweets. We compare their performance in Section 3.6.1. However, we were also interested in the information carried by the image, because communication on social networks is increasingly carried by visual supports. The next section reviews the existing literature on multimodal event detection.

The two images present variations of the same meme, known as “Distracted Boyfriend”. It is a classical example of meme, with identical visual content but different captions.

Figure 3.3: Example of a meme

3.2.3 Text-Image event detection

In the last three years, some works on Twitter data have started to use multimodal embeddings to solve various tasks: Lu et al. [2018] incorporate tweet pictures in order to improve Named Entities Recognition in the text of the tweets. Other works combine text and image of tweets in order to recommend hashtags [Zhang et al., 2017b] or users’ mentions [Ma et al., 2018]. These approaches show that using visual and textual context for tweet-related tasks improves performance in several cases. However, it is unsure whether current visual content description systems can improve Twitter event detection.

We found relatively little work on the joint use of image and text for the specific task of event detection in tweets. Alqhtani et al. [2018] use text and image to detect tweets related to earthquakes. They extract visual features by using scale-invariant feature transform (SIFT) to automatically detect keypoints from images. These keypoints are then clustered to generate a visual vocabulary. Images are then represented as “bags of visual words”. Text is represented using tf-idf vectors. After a step of features selection using PCA, they use a linear combination of two kernels to train a classifier detecting “earthquake tweets” from other tweets. This contribution shows that a kernel fusion of text and image provides better results than text alone on the specific task of detecting earthquakes. However, these results do not give any conclusion on the role of the image in the detection of other types of events.

Zhang et al. [2018] generate image captioning using an encoder-decoder neural architecture to enrich the text of tweets with a description of images. They then apply an improved-LDA algorithm to extract topics. The authors obtain better results with their method than with the tested “text-only” LDA approach. However, the evaluation corpus is rather small (only 10 subjects), and the selected topics favour the contribution of image, as they are subjects for which image captioning tools provide satisfactory results, such as “Giraffe”, “Polar Bear” or “Sunrise”.

The “MediaEval Social Event Detection” challenges Reuter et al. [2013]; Petkos et al. [2014b] have led to the publication of research articles on social event detection, unfortunately it is based on Flickr datasets. A description of the datasets is provided in Section 2.2. Strictly speaking, Flickr is an image hosting service and not a social network: text plays a very limited role, and the images posted are personal photos rather than memes[3] or images containing text.

On Twitter, conversely, not all tweets contain images (in our annotated corpus, 23% of tweets contain a visual content - image, video or animated GIF), and a very large share of the images contains text. Chen et al. [2016] find that 35% of the images in their dataset contain text, mainly in the form of memes (37%), photos of text (22%) or tweet screenshots (8%). Due to the high proportion of textual content and memes in Twitter images, the contribution of images to the event detection task may be small or even negative compared to text alone, given that tweets about very different events may contain the same image. In the remainder of this chapter, we test several types of text and image representations, as well as several types of event detection, in order to answer this question.

3.3 Our event detection approach

In this section, we detail our own methodology. We first recall the required properties of our event detection algorithm. We then detail the choices we made to match these requirements.

3.3.1 Requirements

Our algorithm should have the following properties:

- **The number of events can not be a fixed parameter**. The algorithm has to automatically discover the optimal number of topics, which could vary depending on the time of year or due to a specific situation – such as the COVID pandemic, which restricted the number of topics discussed on social networks since the virus was the public’s primary concern.

- **Scalable and time-efficient**. Ultimately, we want to process all the tweets resulting from our capture method over a whole year in a period of no more than a few days. Our algorithm must therefore be capable of changing scale, shifting from tests conducted on a few tens of thousands of tweets to an implementation on hundreds of millions of tweets.

- **Able to take into account the time of issue of tweets**. Some subjects appear in the news cyclically (soccer games, TV shows, etc.), often with a similar vocabulary. The date of the related tweets is therefore an important feature for discerning them.

[3] A meme is an amusing image or video that spreads virally on the Internet, often with slight variations. See Figure 3.3 for a classical example of meme.

- **Able to deal with noise**. Liu et al. [2017] classify the tweets in different categories: spam, advertisements, mundane/everyday conversation, general information, events, news and breaking news. In this spectrum, they classify all content from spam to general information as noise. Their experiments show a signal-to-noise ratio of less than 0.2%. We did not conduct a manual assessment to confirm or refute these results, but they tend to indicate that our algorithm should be able to handle the vast majority of non-event-related tweets.

3.3.2 Modified First Story Detection algorithm

In order to match the listed requirements, we model the event detection problem as a dynamic clustering problem, using modified versions of the FSD algorithm.

Mini-batch First Story Detection

Our first change consists in introducing "mini-batches" of b tweets. Strictly speaking, we do not parallel the search for a closer neighbor. We use the fact that computing cosine distances between two sets of documents is essentially a matrix multiplication. In the multiplication of two sparse[4] matrices, it is much more efficient to multiply a matrix $A_{(b,|V|)}$ ($|V|$ is the size of the vocabulary) by a matrix $B_{(|V|,w)}$ (w is the number of past tweets among which we search for a nearest neighbor) than to multiply b vectors of size $|V|$ by the same matrix $B_{(|V|,w)}$. Indeed, using the SMMP algorithm[5] [Bank and Douglas, 1993], the complexity of sparse matrix multiplication is $o(bZ^2+max(b,w))$ where Z is the maximum number of non-zero values in a row of A and in a column of B. If we took the documents one-by-one (no batch), computing the cosine similarities for b documents would be of complexity $o(b(Z^2+max(1,w))$.

Our version of the algorithm also deals with "empty" documents, i.e. documents that contain no words or only stop-words. This is frequently the case in datasets of tweets. Empty documents are not clustered, i.e. labeled as -1. Our version of the algorithm is described in Algorithm 2.

Re-clustering

One way of introducing multimodality into the First Story Detection algorithm is to perform a re-clustering step after the first clustering. More precisely, we proceed as follows: given a stream of tweets represented using a text-only representation, we perform a first clustering using the mini-batch FSD algorithm. We then use an image representation of the tweets to perform a re-clustering: for each cluster c, we take all the tweets in c and find their neighbors outside of c that are at a distance lower than a pre-defined threshold t_1. If a proportion p_1 of these neighbors is part of the same cluster c', c and c' are merged. This is repeated until the number of clusters converges.

[4] A sparse matrix is a matrix in which most elements are zeros. This is the case for tf-idf matrices: since each column represents a word in the vocabulary, and since each document only contains a few words, the rows (which represent documents) are therefore mostly composed of zeros. Sparse matrices require a specific format, where only nonzero elements are stored.

[5] This algorithm is implemented for sparse matrix multiplication in the python scipy module.

Algorithm 2 "Mini Batch First Story Detection"

Input: threshold t, window size w, batch size b, corpus $C = \{d_0 \dots d_n\}$ of documents in chronological order
Output: thread ids for each document

$T \leftarrow [] ; i \leftarrow 0; j \leftarrow 0$
while $i < n$ **do**
  $batch = \{d_i, \dots d_{i+b-1}\}$
  for document d in $batch$ **do**
    if d is empty **then**
      $thread\_id(d) \leftarrow -1$
    else
      if T is empty **then**
        $thread\_id(d) \leftarrow j$
        $j \leftarrow j + 1$
      else
        $d_{nearest} \leftarrow$ nearest neighbor of d in T
        if $\delta(d, d_{nearest}) < t$ **then**
          $thread\_id(d) \leftarrow thread\_id(d_{nearest})$
        else
          $thread\_id(d) \leftarrow j$
          $j \leftarrow j + 1$
        end if
      end if
      if $|T| \geq w$ **then**
        remove first document from T
      end if
    end if
    add d to T
    $i \leftarrow i + b$
  end for
end while

The distance metric used in the re-clustering part depends on the type of image representation chosen (see the next Section for the detail of tweet representations). For ResNet vectors, we use euclidean distance. For SIFT features, we compute our own distance metric based on the estimated linear transformation between each image and a combination of the matching points similarities.

Irrelevant tweets

In order to be able to process a large number of tweets in an acceptable time frame, we had to introduce some modifications to the ”Mini Batch FSD” algorithm. These changes also aim to manage the large volume of ”noise” (very short documents, whose subject is difficult to know, even for a human reader) naturally present in a dataset of non-pre-selected tweets. To this end, we introduce the notion of ”irrelevant document”:

- a document is considered irrelevant if its tf-idf vector does not contain any element with a value greater than a r threshold. In other words, the tweet contains only extremely common words;
- a tweet is also considered irrelevant if it contains less than m words.

Irrelevant documents can be clustered if a nearest neighbor is found, however, if no nearest neighbor is found within a radius t they are excluded from the collection and labelled as -2. Whether they are clustered or not, irrelevant tweets cannot be selected as nearest neighbors in the following iterations of the algorithm.

This method has two advantages: first, it prevents very vague tweets from extending the vocabulary of the cluster to which they are attached. This reinforces the stability of the vocabulary of each cluster. Second, it reduces the number of past tweets to which each tweet must be compared, thus increasing the time efficiency of the algorithm. Indeed, the complexity of the FSD algorithms shifts from $O(nw)$ (with n the number of documents in the collection and w the number of documents in the comparison window) to $O(nwp)$, with p the proportion of relevant tweets in the collection.

This approach cannot be tested on the annotated dataset, since almost all documents selected by our annotators are relevant (since they contain enough significant words to be considered as referring to an event). However, we use this method in order to detect events on the entire, very noisy corpus.

3.4 Other tested approaches

We compare the performance of our approach (mini-batch FSD algorithm with or without re-clustering and irrelevant tweets) with other algorithms existing in the literature. Some of these algorithms are not directly suited for our event detection task, since they take the number of classes or topics as an input, whereas we need an algorithm capable of automatically discovering the optimal number of clusters. This is the case for SVM classifiers, which are

rather used to evaluate the quality of different embeddings. In total, we present three methods: SVM classification, DBSCAN, and Dirichlet Multinomial Mixture model.

3.4.1 Support Vector Machines (SVM)

Evaluations of the "quality" of tweet representations can be made according to different approaches: first, test if the representation allows a good separability of the different classes (events). Second, make sure that the vectors produced are suitable for distance measurements, which are used for clustering. To evaluate the separability of classes, we initially reduced the task of unsupervised event detection to a classification task. We used SVM classifiers with two types of kernel: a triangular kernel Fleuret and Sahbi [2003] and a Radial Basis Function (RBF) kernel.

The triangular kernel function is the following:

$$k(x, y) = 1 - \|x - y\| \tag{3.3}$$

Our experiments with this type of kernel show that, in addition to being invariant to scale changes [Fleuret and Sahbi, 2003], it can be applied effectively to both dense and sparse vectors. It achieves performance similar to parametric kernels on text classification, with the advantage of not requiring a lengthy grid search process to select the right parameters.

Our first experiments were conducted only with a triangular kernel. However, as this is not a very well-known form, several reviewers have requested further tests to be conducted with a more common kernel. We therefore also conducted classification experiments with an RBF kernel:

$$k(x, y) = \exp(-\gamma \|x - y\|^2) \tag{3.4}$$

3.4.2 DBSCAN

We use the DBSCAN algorithm [Ester et al., 1996] as a baseline since it is a classical clustering algorithm that does not require a fixed number of clusters as input. We use the implementation provided by scikit-klearn.[6]

3.4.3 Dirichlet Multinomial Mixture (DMM) model

We chose DMM as our main comparative approach since it has several interesting properties for our problem:

- it can automatically infer the number of clusters
- it is fast to converge

[6] https://scikit-learn.org/stable/modules/generated/sklearn.cluster.DBSCAN.html#sklearn.cluster.DBSCAN

- it is designed to cope with the highly-dimensional problem of short texts

The generative process of DMM is presented in Section 3.2.1. We use a Python implementation[7] of the Gibbs Sampling for DMM (GSDMM) algorithm presented by Yin and Wang [2014].

3.5 Experimental setup

For each of the tested approaches, we optimize the parameters in order to obtain the best possible results for a given algorithm. In this section, we first present our choice of parameters for each model. We then detail the different types of vector representations of tweets tested with the FSD algorithm. Finally, we justify our choice of evaluation metrics.

3.5.1 Choice of parameters

FSD parameters

The parameters of the FSD algorithm are w, the number of past tweets among which we search for a nearest neighbor, and t, the distance threshold above which a tweet is considered sufficiently distant from past tweets to form a new cluster. The value of w has been set differently for each corpus: it is set to approximately one day of tweet history, based on the average number of tweets per day in each corpus. We then tested different values of t for each type of representation. Generally speaking, lower t values lead to more frequent clustering, and thus better intra-cluster homogeneity (better precision), but may lead to over-clustering (lower recall).

Re-clustering parameters

In the re-clustering step, the parameters are t_1, the distance among which candidate neighbors are selected, and p_1, the minimum proportion of candidate neighbors from the same cluster necessary to merge two clusters. When testing the second step (re-clustering), it is important that the first clusters have a good homogeneity. This is why, during our experiments, we set the t parameter to 0.6, slightly lower than the best t value for the mini-batch FSD algorithm alone. In this way, at the end of the first step we obtain more homogeneous clusters. The w parameter remains set to the average number of tweets per day in the dataset.

Relevance thresholds

Due to the size of the corpus (38 million tweets), it is not possible to test many different values of the relevance thresholds, since each test takes more than 3 days. The parameters were thus set as follows: m (minimum number

[7] `https://github.com/atefm/pDMM`

of words in a document to consider it as relevant) was set to 5, and r (minimum idf value that one of the elements of a vector must reach to consider it as relevant) to 0.21. The value of r was calculated from our annotated collection C as follows:

$$r = \min_{d \in C} \max_{w \in d} idf(w)$$

Thus, we select the minimum value of r on a dataset where all tweets are considered relevant. Any tweet from the rest of the collection that has no word of at least this significance is considered irrelevant.

SVM parameters

Both RBF and triangular classifiers are trained on a random sample of 50% of the corpus. For the RBF kernel, we perform a grid search to optimize the γ parameter.

DBSCAN parameters

DBSCAN uses 2 parameters: the distance ϵ and the minimum number of points m that must be within a radius ϵ for these points to be considered a cluster. The ϵ parameter controls the local neighborhood of each point. A too large ϵ leads to different clusters being merged, while a too small ϵ will cause most points to be labelled as "outliers". The m parameter controls the tolerance to noise. Since we run tests only on the annotated part of the datasets (only event-related tweets), the level of noise is rather low, and small values of m are sufficient. We test different values of m and ϵ for each corpus.

The vectors given as input to the DBSCAN algorithm are the same tdidf vectors as for the FSD algorithm (tfidf-all-tweets, see Section 3.5.2 for the detail of textual representations). Therefore, we also selected the cosine distance as metric.

DMM parameters

The different parameters are K the number of clusters at initialization, i the number of iterations of the GSDMM algorithm, the parameter α of the Dirichlet topic distribution, and the parameter β of the Dirichlet topic-word distribution.

Yin and Wang [2014] provide an in-depth analysis of the role of the different parameters. They show that the parameter α has a rather small influence on the performance of the algorithm when set to values between 0 and 1. The value of β has an influence on the number of non-empty clusters, and hence on the homogeneity of GSDMM: higher values of β will result in a lower homogeneity, and a higher completeness. Concerning the number of iterations, GSDMM seems to reach a stable number of clusters within 10 iterations on all 4 tested datasets. Finally,

the optimal value of the initial number of clusters K depends on the actual number of topics within the dataset: K has to be significantly larger than the ground truth number of topics. However, setting K to a high value can decrease the time efficiency of the algorithm, since the time complexity of GSDMM is $O(KD\bar{L})$ with D the total number of documents, and $\bar{L}$ the average document length.

Taking into account these different factors, we conducted the experiments by setting $\alpha = 0.1$, $i = 20$, and $\beta \in [0.08, 0.12]$, which seem to be the values for which the algorithm's performance is the best. We set K differently for each corpus: for the French corpus, with 257 events, we set K to 450. For the corpus by McMinn et al. [2013], with 505 events, we set K to 700. For each corpus, we apply the same preprocessing as for the FSD algorithm with tf-idf (see Table 3.1) : remove mentions, stop-words, urls, long numbers, punctuation, split hashtags on capital letters, lowercase and transpose to ASCII characters.

Figure 3.4: Four clusters created using the vizualisation algorithm t-SNE on the ResNet representation of images from our dataset.

3.5.2 Types of representations

In this part we present the different types of embeddings tested for the mini-batch FSD and re-clustering algorithms. We also report our fine-tuning tests aimed at improving the performance of SBERT on the French corpus. Finally, we detail the text preprocessing applied depending on each model.

Image

1. *ResNet layer.* To create image vectors, we test a CNN model trained on ImageNet as the encoder. More precisely, we experiment with the 50-layer version of the ResNet model [He et al., 2016]. We use the penultimate layer as the vector representation of images. The dimension of the image embedding is $2,048$.

   The drawback of this type of representations trained on ImageNet is that the semantic expressed by vectors is categorical: vectors allow distinctions to be made between categories such as “people”, “animals”, “buildings”, but cannot, for example, distinguish easily between two soccer games or two different buildings. Figure 3.4 illustrates this by showing some of the clusters created using the t-SNE visualization algorithm Maaten and Hinton [2008] on images from one day of our dataset represented as ResNet layers. All faces tend to produce similar vectors, whether they are the faces of politicians or “memes” such as the little girl to the right of the picture. On the other hand, the Amazon building is not in the same cluster as the press conference of Amazon executives that appears in one of the pictures above, even though a human knowing the company’s logo would probably have grouped them together.

2. *SIFT features.* Knowing this characteristic of ResNet vectors, we have also used a standard approach based on bag of local features for image descriptions. This category of image retrieval methods is known to integrate low-level visual information, less semantic and more local than that conveyed by global vectors obtained from deep-learning networks. Thus, it focuses on the existence of visually very similar details, in all or part of the images. The underlying idea of this approach is to enable the detection of places or similar objects between images, which can be useful to create links between the tweets. Figure 3.5 shows an example of the kind of similarities that can be detected using local features.

   All images are described with SIFT features Lowe [1999], which are then compressed to 128-bit binary hash codes. This is done by first computing a principal component analysis in the original feature space followed by a binary quantization. The distance between any two features can then be efficiently approximated by the Hamming distance between the hash codes. To avoid scanning the whole dataset, the hash codes are then indexed in a hash table whose keys are the t-length prefix of the hash codes. At search time, the hash code of a query feature is computed, as well as its t-length prefix. We then use a probabilistic multi-probe search algorithm inspired by that of Joly and Buisson [2008] and allowing to efficiently return the approximate K-Nearest Neighbors (K-NN) of each query feature. The raw visual matches returned by the approximate K-NN

search are finally filtered by a spatial consistency checking. We therefore estimate a linear transformation (by a RANSAC algorithm) between the query image and each matched image.

We did not use SIFT features to create image vectors, and therefore we cannot test this representation with the FSD algorithm. Moreover, the distance between images is only computed between the query image and its neighbors as returned by the approximate K-NN search algorithm. Therefore, we do not have the complete distance matrix of images and cannot compute an adapted version of the FSD algorithm with a custom distance metric. However, we can use the distance metric between an image and its close neighbours in the second step of our re-clustering method (3.3.2).

Figure 3.5: Example of near neighbors detected using SIFT features among the images from our dataset. The first image is used as query, and the following ones are matched neighbors.

Text

We chose models that have both French and English versions. This sub-section details the models used.

1. *Tf-idf.* Due to the inherent brevity of tweets, we simplified the calculation of tf-idf to a simple calculation of idf, since it is rare for a term to be used several times in the same tweet. The form used to calculate the weight of a term t in a tweet is therefore $idf(t) = 1 + log(n + 1/df(t) + 1)$, where n is the total number of documents in the corpus and $df(t)$ is the number of documents in the corpus that contain t.

We have distinguished two calculation modes for n and $df(t)$: **tfidf-dataset** denotes the method that counts only annotated tweets, and **tfidf-all-tweets** denotes the calculation method that takes into account all tweets in the corpus (38 million tweets) to obtain n and $df(t)$. For each method, we restrict the vocabulary with a list of *stop-words* and a threshold df_{min}, the minimum number of tweets that must contain t for it to be included in the vocabulary. In all our experiments, $df_{min} = 10$. We thus obtain a vocabulary of nearly 330,000 words in English and 92,000 words in French for **tfidf-all-tweets**, and 5,000 words in English and 9,000 words in French for **tfidf-dataset**.

2. *Word2Vec.* We used pre-trained models for English, and trained our own French models. For each corpus, we distinguish between **w2v-twitter**, models trained on tweets, and **w2v-news**, models trained on press articles. For English, w2v-twitter is a pre-trained model published by Godin et al. [2015][8] (400 dimensions) and w2v-news is a model trained on Google News and published by Google[9] (300 dimensions). In French, w2v-twitter was trained with the CBOW algorithm on 370 million tweets collected between 2018 and 2019, and w2v-news was similarly trained on 1.9 million AFP dispatches collected between 2011 and 2019. Both models have 300 dimensions. As Word2Vec provides word embeddings and not sentence embeddings, the representation of tweets is obtained by averaging the word vectors of each word. Two methods were used for averaging: a simple average, and an idf-weighted average using the **tfidf-all-tweets** calculation method.

3. *ELMo.* For English, we used the model published on TensorFlow Hub[10]. For French, a model trained on French published by Che et al. [2018][11]. In each case, we use the average of the three layers of the network as a representation of each word. The representation of a tweet is produced by averaging these vectors (of dimension $1,024$).

4. *BERT.* Google provides an English model and a multilingual model[12]. In order to improve the performance of the multilingual model on French tweets, we continued training for 150,000 steps on tweets collected in June 2018. We refer to the simple multilingual model as **bert** and the model trained on tweets as **bert-tweets**. In each case, we used the penultimate layer of the network (of dimension 768) as embedding, by averaging the tokens to obtain a tweet representation.

5. *Universal Sentence Encoder.* The provided models[13,14] (English and multilingual) are designed to provide sentence embeddings, so we were able to use them as is, without averaging vectors as in the previous representations. The vectors are of dimension 512.

[8] github.com/loretoparisi/word2vec-twitter
[9] code.google.com/archive/p/word2vec/
[10] tfhub.dev/google/elmo/2
[11] github.com/HIT-SCIR/ELMoForManyLangs
[12] github.com/google-research/bert models: bert-large, uncased and bert-base, multilingual cased
[13] tfhub.dev/google/universal-sentence-encoder-large/3
[14] tfhub.dev/google/universal-sentence-encoder-multilingual-large/1

6. *Sentence-BERT* The authors of SBERT provide pre-trained models for English[15]. For French, we had to perform a fine-tuning of the multilingual BERT model, which we present in Section 3.5.2. The vectors obtained are of dimension 768.

Fine-tuning Sentence-BERT for French

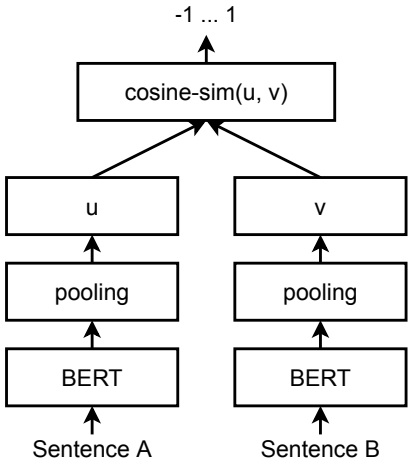


Figure 3.6: SBERT architecture at inference, to compute similarity scores. This architecture is also used during part of the training with mean-square-error as regression objective function. This diagram is provided in the Sentence-BERT paper Reimers and Gurevych [2019].

The SBERT model is specifically trained to provide cosine similarity scores: the architecture presented in Figure 3.6 is used to train the model on the STS corpus. The objective function is a mean-square-error loss between $cosine\text{-}sim(u, v)$ and the similarity score evaluated manually in the dataset. This model seems particularly suitable for a clustering algorithm based on cosine similarity, and indeed, among the sentence embeddings (*Universal Sentence Embedding* and SBERT), it is the one that provides the best clustering results in English.

However, the English pre-trained model is based on fine-tuning on supervised tasks (see 3.2.2 for details of SNLI and STS tasks), which cannot be done in French without an annotated corpus. We have therefore implemented two strategies to perform a fine-tuning of the **bert-tweets** model on French data: first we have used *Cloud Translation API*[16] within the free use limit to translate part of the STS dataset (we obtained $2,984$ pairs of sentences in French). Second, we manually annotated 500 pairs of headlines of selected newspaper articles (we selected headlines that contained common terms). The annotation was done on a scale of 0 to 5, in the same way as for STS. However, instead of indicating the degree of semantic similarity between the sentences, we sought to assess whether the two headlines described the same event. The two types of fine-tuning (translated corpus, or translated corpus + annotated corpus) are designated by **sbert-tweets-sts-short** and **sbert-tweets-sts-long**. The performances of the different representations are described in Section 3.6.

[15] `github.com/UKPLab/sentence-transformers`. Model: bert-large-nli-stsb-mean-tokens
[16] `cloud.google.com/translate/docs/reference/rest/`

model	rm mentions	unidecode	lower	rm punctuation	rm stop-words	hashtag split	rm long numbers	rm repeated chars	rm urls
tfidf_all_tweets	X	X	X	X	X	X	X	X	X
tfidf_dataset	X	X	X	X	X	X	X	X	X
w2v_afp_fr	X	X	X	X		X	X	X	X
w2v_twitter_fr	X	X	X	X		X	X	X	X
w2v_gnews_en	X			X		X	X	X	X
w2v_twitter_en				X		X	X	X	X
elmo						X	X	X	X
bert						X	X	X	X
bert_tweets						X	X	X	X
sbert_sts						X	X	X	X
sbert_nli_sts						X	X	X	X
sbert_tweets_sts_short						X	X	X	X
sbert_tweets_sts_long						X	X	X	X
use						X	X	X	X

Table 3.1: Preprocessing applied for each model

Text-image representation

We build a common vector of text and image by concatenating the ResNet vectors with each type of textual representation. We test several coefficients applied to the image vectors, in order to increase or decrease their weight in the clustering process.

Preprocessing

Each text embedding model takes different text formats as inputs: for example, models able to deal with sentences, such as BERT, Sentence-BERT, ELMo or Universal Sentence Encoder, take the full text with punctuation as input. For Word2Vec and tf-idf models, we lowercase characters and remove punctuation. Table 3.1 summarizes all preprocessing steps depending on the type of model. Each column correponds to a preprocessing step:

- remove mentions: mentions are a Twitter-specific way of referring to another Twitter user in a tweet, so that she is notified that the tweet is talking about her or is addressed to her. Entries take the following form: @name_of_the_user. For tf-idf models, removing mentions is a way to reduce the size of the vocabulary. For most Word2Vec models, mentions are not part of the vocabulary, except for w2v_twitter_en.
- unidecode: we use the Python module unidecode to convert Unicode characters to ASCII characters. In French, for example, all accents are removed: “Wikipédia” becomes “Wikipedia”.
- hashtag split: we split hashtags on capital letters. “#HappyEaster” becomes “Happy Easter”.
- lower: we set the text in lowercase letters.
- remove long numbers: we remove numbers longer than 4 digits.
- remove repeated characters: we limit the number of repeated characters inside a word to three. “looooool” becomes “loool”.

3.5.3 Evaluation metrics

Classification

The classification is evaluated by the macro average of the F-score of each class. For a given class, if we denote tp the number of true positives (documents that are correctly classified in the class),fp the number false positives (documents that are incorrectly attributed to the class) and fn the number of false negatives (documents that are incorrectly classified into another class), we can define precision, recall and F1-score as follows:

$$precision = \frac{tp}{tp + fp} \tag{3.5}$$

$$recall = \frac{tp}{tp + fn} \tag{3.6}$$

$$F1\text{-}score = 2 \cdot \frac{precison \cdot recall}{precision + recall} \tag{3.7}$$

Event detection

Event detection performance is evaluated by a measure we call “best matching F1”. It is defined by Yang et al. [1998]: we evaluate the F1 score of each (cluster, event) pair.[17] Each event is then matched to the cluster for which the F1-score is the best. Each event can be associated to only one cluster. The best matching F1 thus corresponds to the average of the F1-scores of the cluster/event pairs, once the matching is done.

We chose to use this metric, rather than the one defined by McMinn and Jose [2015]: they consider all clusters with more than 75 tweets as candidate events. A candidate is evaluated as true positive if 5% or more than 15 tweets of the candidate’s tweets match those of an annotated event. We denote a the number of true positives. From the detail of their experimental results, we can deduce that all candidates that match with a ground truth’s event are counted as true positives, even if several clusters match the same event. Conversely, all candidates that do not match an event from the ground truth are considered false positives.

We decided not to use this metric for two reasons: first, we only have access to the portion of the McMinn et al. [2013] dataset that was not deleted over time. Therefore, we cannot keep the threshold of 75 tweets per event, and we would not be able to compare our results with those obtained by McMinn and Jose. Second, this measure favors over-clustering by not penalizing the fact that several clusters match the same event. We consider the Best Matching F1 score as a better evaluation metric for this task.

[17] Clusters are detected by the algorithm, while events are annotated in the ground truth

3.6 Results

3.6.1 Comparison of text embeddings

Whether for the classification task or for the clustering task, none of the tested models manage to outperform the tf-idf model calculated on the whole corpus (tfidf-all-tweets). However, the relative performance of the models varies by language, and by task.

Classification results

For English, SVM classification results are consistent and robust to kernel change (see Tables 3.2). They show that BERT and ELMo do not provide easily separable short-text vectors. Models intended to be used as embeddings (Word2Vec, Universal Sentence Encoder, SBERT) obtain better results. The tf-idf vectors remain the best adapted, on a par with weighted w2v-news vectors.

On the French corpus (see Table 3.3), we can make the same conclusions for BERT and ELMo, even for the BERT model further trained on French tweets (bert-tweets): these models do not provide adequate embeddings for sentences or short texts. However, the two kernels do not provide exactly the same results concerning the Universal Sentence Encoder model (use). With the RBF kernel, this model outperforms the other representations while, with the triangular kernel, tf-idf, Word2Vec and Universal Sentence Encoder perform equally well. In both cases, the difference between these latter models is rather small.

	Triangular Kernel			**RBF Kernel**			
Model	*F1*	*precision*	*recall*	γ	*F1*	*precision*	*recall*
bert	74.49 $\pm$ 0.41	85.76 $\pm$ 0.56	69.42 $\pm$ 0.38	0.01	75.31 $\pm$ 0.51	85.10 $\pm$ 0.62	70.87 $\pm$ 0.52
elmo	59.81 $\pm$ 0.41	77.17 $\pm$ 0.63	53.23 $\pm$ 0.25	0.10	57.64 $\pm$ 0.36	77.18 $\pm$ 0.29	50.55 $\pm$ 0.28
sbert-nli-sts	80.55 $\pm$ 0.33	87.85 $\pm$ 0.35	76.94 $\pm$ 0.31	0.01	80.37 $\pm$ 0.36	88.53 $\pm$ 0.37	76.26 $\pm$ 0.32
tfidf-all-tweets	**83.50** $\pm$ 0.78	90.87 $\pm$ 0.47	79.91 $\pm$ 0.78	1.00	**81.86** $\pm$ 0.77	90.98 $\pm$ 0.32	77.41 $\pm$ 0.79
tfidf-dataset	**83.46** $\pm$ 0.72	90.28 $\pm$ 0.51	80.08 $\pm$ 0.71	1.00	**82.70** $\pm$ 0.69	89.67 $\pm$ 0.51	79.10 $\pm$ 0.73
use	80.26 $\pm$ 0.38	86.17 $\pm$ 0.29	77.59 $\pm$ 0.40	1.00	79.92 $\pm$ 0.46	85.68 $\pm$ 0.45	77.40 $\pm$ 0.40
w2v-news	81.35 $\pm$ 0.53	88.94 $\pm$ 0.40	77.45 $\pm$ 0.65	1.00	80.42 $\pm$ 0.55	89.09 $\pm$ 0.41	75.89 $\pm$ 0.64
w2v-news tfidf	**82.39** $\pm$ 0.64	89.00 $\pm$ 0.35	79.02 $\pm$ 0.69	0.01	**81.57** $\pm$ 0.73	89.02 $\pm$ 0.51	77.64 $\pm$ 0.78
w2v-twitter	76.68 $\pm$ 0.53	86.82 $\pm$ 0.53	72.24 $\pm$ 0.52	10.0	77.62 $\pm$ 0.62	87.91 $\pm$ 0.39	72.71 $\pm$ 0.68
w2v-twitter tfidf	81.20 $\pm$ 0.48	88.67 $\pm$ 0.17	77.54 $\pm$ 0.54	0.10	81.07 $\pm$ 0.49	88.61 $\pm$ 0.29	77.24 $\pm$ 0.59

Note: Each cell indicates the mean and standard deviation of 5 runs (with different train/test splits), in percentages.

Table 3.2: SVM classification results on the English corpus

First Story Detection results

The tfidf-all-tweets vectors give the best results for the clustering task (see Table 3.4), even more clearly than for classification. This is due to the shape of the tf-idf vectors, which are particularly well suited for cosine similarity calculations, as well as to the event-specific characteristics of the two datasets: the same terms are obviously widely

	Triangular Kernel			RBF Kernel			
Model	*F1*	*precision*	*recall*	γ	*F1*	*precision*	*recall*
bert	78.46 ± 0.68	90.88 ± 0.78	71.26 ± 0.70	0.01	79.08 ± 0.61	90.95 ± 0.98	72.04 ± 0.56
bert-tweets	81.77 ± 0.70	91.88 ± 0.95	75.73 ± 0.75	0.01	82.68 ± 0.72	91.63 ± 1.08	77.11 ± 0.64
elmo	73.59 ± 0.64	88.53 ± 0.74	65.54 ± 0.61	0.10	74.40 ± 0.70	89.79 ± 1.07	66.21 ± 0.54
sbert-tw-sts-l	86.08 ± 0.86	93.60 ± 0.94	81.16 ± 0.81	0.10	38.47 ± 1.43	89.79 ± 0.85	27.94 ± 1.36
tfidf-all-tweets	**87.79** ± 0.58	95.24 ± 0.91	83.13 ± 0.50	1.00	86.57 ± 0.55	95.20 ± 1.17	81.08 ± 0.36
tfidf-dataset	**87.66** ± 0.69	94.78 ± 1.15	83.14 ± 0.53	1.00	86.42 ± 0.53	94.74 ± 1.06	81.08 ± 0.35
use	**87.45** ± 0.60	93.94 ± 0.56	83.44 ± 0.58	1.00	**88.40** ± 0.75	94.19 ± 0.73	84.65 ± 0.61
w2v-news	**86.59** ± 0.80	92.82 ± 1.03	82.63 ± 0.74	1.00	86.28 ± 0.88	92.84 ± 0.95	82.19 ± 0.91
w2v-news tfidf	**87.51** ± 0.71	92.94 ± 1.06	84.02 ± 0.49	0.01	86.32 ± 0.77	92.63 ± 0.87	82.27 ± 0.69
w2v-twitter	**87.01** ± 0.56	93.40 ± 0.84	83.03 ± 0.58	1.00	86.60 ± 0.60	93.47 ± 0.75	82.33 ± 0.61
w2v-twitter tfidf	**87.73** ± 0.56	93.51 ± 0.99	84.03 ± 0.38	0.01	86.71 ± 0.60	93.32 ± 0.77	82.43 ± 0.58

Note: Each cell indicates the mean and standard deviation of 5 runs (with different train/test splits), in percentages.

Table 3.3: SVM classification results on the French corpus

	English		French	
Model	*t*	*F1*	*t*	*F1*
bert	0.04	39.22	0.04	44.79
bert-tweets	-	-	0.02	50.02
elmo	0.08	22.48	0.20	46.08
sbert-nli-sts	0.39	58.24	-	-
sbert-tweets-sts-long	-	-	0.36	67.89
sbert-tweets-sts-short	-	-	0.38	65.71
tfidf-all-tweets	0.75	**70.10**	0.70	**78.05**
tfidf-dataset	0.65	68.07	0.70	74.39
use	0.22	55.71	0.46	74.57
w2v-news	0.30	53.99	0.25	66.34
w2v-news tfidf-weights	0.31	61.81	0.30	75.55
w2v-twitter	0.16	43.20	0.15	57.53
w2v-twitter tfidf-weights	0.20	53.45	0.25	71.73

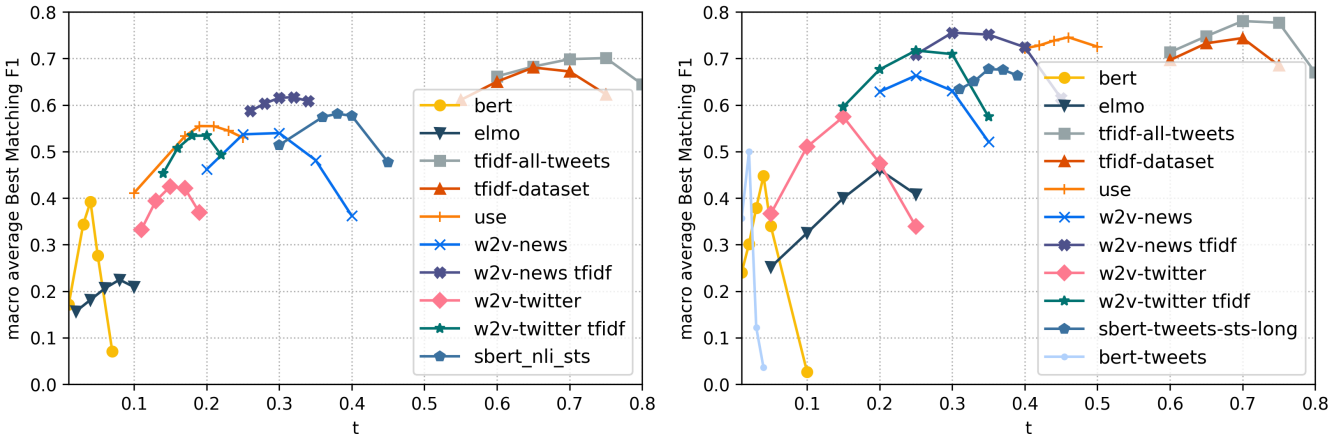
Note: Performance is assessed using the "Best Matching F1" score.
For each model, the best t threshold value was selected by successive tests.

Table 3.4: FSD clustering results for each corpus

used among tweets of the same event. These results are consistent with those of Cagé et al. [2020], who came to similar conclusions regarding event detection in press articles.

Concerning neural models adapted to sentence embeddings (SBERT, *Universal Sentence Encoder*), they do not perform better than the tf-idf weighted w2v-news models. On the English corpus, we note that the fine-tuning of *Sentence-BERT* on semantic similarity corpora (sbert-nli-sts) allows slightly better results than the generic vectors of *Universal Sentence Encoder*.

Our own fine-tuning of BERT (sbert-tweets-sts-short and sbert-tweets-sts-long) does not outperform *Universal Sentence Encoder* on the French corpus. However, we note that the thematic similarity corpus (which contains only 500 pairs of sentences) allows to increase the clustering performance by 2 points. Nevertheless, our fine-tuning does not achieve as good results as the English model, due to the fact that it could not be trained on a corpus of similar size to SNLI.



(a) English

(b) French

Figure 3.7: Best Matching F1 score for FSD clustering depending on the threshold parameter t for each corpus

When used with the FSD algorithm, the best embeddings (tf-idf, Word2Vec with tf-idf weights, Universal Sentence Encoder) also have the property of being less sensitive to variations of the threshold t, as shown in Figure 3.7. Moreover, the optimal value of t for a given embedding seems to be approximately the same for each corpus (0.7 for tf-idf). This result may indicate that the First Story Detection algorithm could be applied to other (not annotated) tweet datasets without adapting the threshold value.

It is surprising that Word2Vec models trained on tweets are no better than models trained on news. However, this can be explained by two factors: the content of the datasets, which are made up of tweets referring to news, and whose vocabulary is therefore probably closer to the corpus from the AFP or from Google News than to the average Twitter. The second factor is the great variability in the vocabulary of tweets: it is possible that the w2v_twitter_en model, trained on tweets from 2015, does not correspond to the vocabulary used in the McMinn et al. [2013] corpus, collected in 2012.

The better performance on the French corpus than on the English corpus is probably due to the fact that classification and clustering on the English corpus are more difficult tasks, for several reasons: first, the English corpus is from 2012, and therefore an substantial share of the tweets have been deleted (our last download in November 2019 allowed us to retrieve 72,484 tweets i.e. 72% of the original annotated dataset). This can have important consequences, especially for the FSD algorithm which over-segments clusters if a tweet is missing to link them. Second, it seems that many events in the English corpus could be considered “sub-events” of the same macro event: “Hurricane Sandy in the Bahamas”, “Tweets for Praying for people affected by hurricane Sandy”, “Superstorm Sandy hits the east coast of the USA”, “They all discuss about Sandy Storm” are for example four different events in the corpus by McMinn et al. [2013]. These events with very close topical similarity are probably more difficult to separate for the algorithm.

3.6.2 Comparison of text-image embeddings

Model	t	*F1*	*precision*	*recall*	*weight*
elmo	0.23	50.1	77.5	50.1	
elmo / resnet	0.28	52.5	80.2	50.3	0.05
tfidf-all-tweets	0.79	84.3	92.5	83.7	
tfidf-all-tweets / resnet	0.79	84.6	92.3	84.3	0.005
w2v-news	0.28	75.5	88.1	74.8	
w2v-news / resnet	0.35	76.9	87.2	77.9	0.01
w2v-news tfidf	0.40	81.7	89.1	83.4	
w2v-news tfidf / resnet	0.39	82.1	90.6	82.3	0.01
w2v-twitter	0.17	66.5	85.2	66.0	
w2v-twitter / resnet	0.20	69.2	87.5	66.8	0.01
w2v-twitter tfidf	0.29	79.0	90.4	78.9	
w2v-twitter tfidf / resnet	0.29	79.0	90.4	79.0	0.005

Note: Performance is assessed using the "Best Matching F1" score.

For each model, the best t threshold value was selected by successive tests.

Table 3.5: FSD clustering results on "text only" and "text-image" vectors on the tweets of the French corpus that include visual content

The results presented in this section concern only our dataset, as the English dataset does not include enough images: of all the annotated tweets that we were able to retrieve, only 570 contained a link to a video, image or animated GIF. In 2012, the use of smartphones was much less widespread than at present, which probably explains the small number of images.

Conversely, our dataset built up in 2018 contains many visual contents. Out of the 95,796 annotated tweets, we were able to download multimedia content for 22,477 (19.8% photos, 2.5% videos and 1.5% GIFs). In order to get the highest possible number of images, we processed videos and GIFs as images by using the thumbnails provided by Twitter for a static display. We thus obtained 20,481 unique images. We only considered tweets containing visual contents for clustering, which is why the clustering results that we present below are slightly different from those in the previous part, even for the text-only methods.

We present the clustering results in Table 3.5. These results show that visual features do not bring a significantly better performance to the FSD algorithm. Image may improve the results of the textual representations that perform the worst (ELMo). However, this gain is not significant enough to outperform the tf-idf representation. Moreover, the weights applied to the ResNet layer before it is concatenated with textual features are extremely small (0.005 for the tf-idf / ResNet concatenation). This in an indication of the very small role played by visual features in the nearest neighbor search. Figure 3.8 illustrates this result more clearly by showing the change in macro-F1 depending on t: ELMo and Word2Vec embeddings are distinctly improved through the concatenation with visual features, whereas the tf-idf embedding provides the same results when it is combined with the ResNet layer.

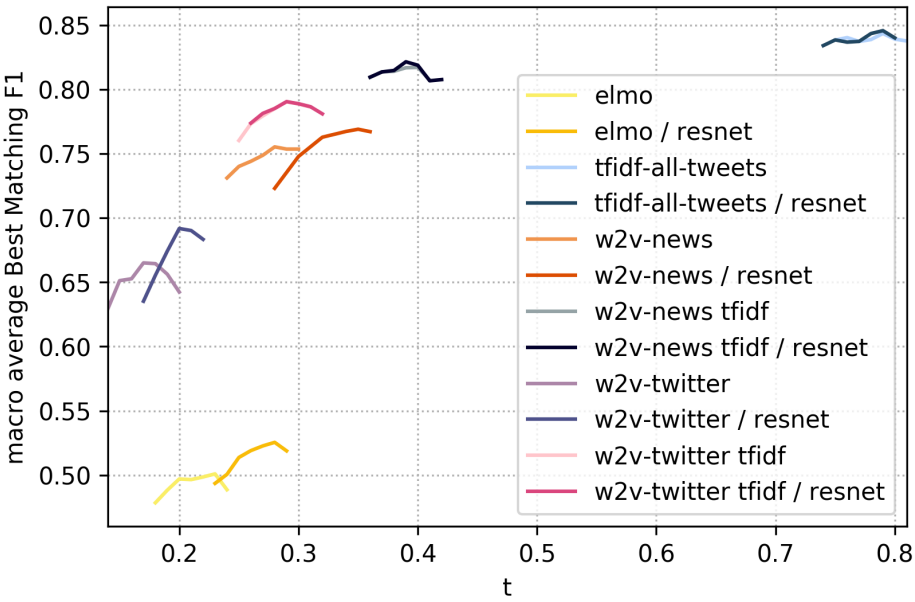


Figure 3.8: Best Matching F1 score for FSD clustering depending on the treshold parameter t for text-only and text-image vectors on the tweets of the French corpus that include visual content

3.6.3 Comparison of event detection methods

In the previous parts (3.6.1 and 3.6.2) we have determined the best embeddings to serve as input to the FSD algorithm. We now compare our best results with those obtained by other methods: re-clustering, DBSCAN and DMM.

Algorithm	t	t_1	p_1	*F1*	*precision*	*recall*
FSD with tfidf	0.79	-	-	84.3	92.5	83.7
reclustering ResNet	0.60	5	0.8	79.6	91.9	77.3
reclustering SIFT	0.60	-24	0.9	81.1	96.7	74.3

Note: Performance is assessed using the "Best Matching F1" score.

For each model, the best t, t_1 and p_1 values were selected by successive tests.

Table 3.6: Clustering performance of the FSD algorithm and re-clustering algorithms on the tweets of the French corpus that include visual content

Table 3.6 shows the results of the re-clustering approach using each type of visual representation: either a semantic representation, with ResNet vectors, or local features representation, with SIFT similarities. These results indicate that local features tend to perform best to improve the first clustering step. Overall, however, this way of combining representations degrades the performance of the FSD algorithm alone. The very high values selected for the p_1 parameter, which indicate that very few clusters are allowed to merge, seem to indicate that the only way to obtain correct results with the re-clustering algorithm is to minimize the role of the second step. The obtained results are thus close to the results of the FSD algorithm alone with a too low t parameter.

The comparative results of DBSCAN, DMM and FSD are summarized in Table **??**. The First Story Detection

algorithm outperforms the other tested methods by a clear margin on both datasets.

3.6.4 Results on the entire collection

All previous experiments were conducted on the annotated part of our dataset. However, the final objective of our study is to detect events in the entire collection of tweets (4.5 million tweets per day on average) and over a period of several months.

We tested the mini-batch FSD algorithm with and without relevance thresholds on the entire corpus (retweets excluded), i.e. 38 million tweets. The results of both methods are displayed in Table 3.7. The method with relevance thresholds seems to obtain better scores, but these results must be interpreted with caution: indeed, the thresholds ”fit” the algorithm to the characteristics of the annotated corpus, resulting in better performance on annotated tweets. This does not tell us whether performance is also better over the entire collection. However, it can legitimately be assumed that these rather low thresholds do not degrade performance, while allowing a gain in time efficiency.

Algorithm	t	m	r	*F1*	*precision*	*recall*
FSD without relevance thresholds	0.6	0	0.00	52.68	74.51	52.63
FSD with relevance thresholds	0.6	5	0.21	59.82	85.07	53.68

Note: Performance is assessed using the ”Best Matching F1” score on the annotated tweets of the collection.

Table 3.7: Clustering performance of the FSD algorithm with and without relevance thresholds

3.7 Conclusion

This chapter does not provide any brand new algorithm, but merely offers some improvements to the “First Story Detection” algorithm. However, it does allow us to verify, on two tweet corpora, some results already established on news corpora [Cagé et al., 2020]. First, we show that the FSD algorithm is extremely efficient for tweet clustering, more so than topic modeling techniques such as Dirichlet Multinomial Mixture model [Yin and Wang, 2014], or a standard algorithm such as DBSCAN [Ester et al., 1996]. The reason for this superiority probably comes from the very rapid evolution over time of the vocabulary used to talk about a given event. The two previously mentioned algorithms are not designed to take this temporal evolution into account, unlike the FSD algorithm, which allows a gradual evolution of clusters over time.

Second, we show that, out of a large panel of embedding methods, the tf-idf weighting remains the most suitable representation of documents for an algorithm such as FSD, which is based on nearest neighbor search with cosine similarity. In addition, tf-idf is not only the representation that provides the best results, but also the most stable to changes in the algorithm parameter (threshold t). These are important results for us and, I believe, for many researchers who are looking to use vector representations of short texts for Information Retrieval tasks. This does not

mean that recent neural embedding methods will not eventually outperform tf-idf for this kind of task as well. In fact, we would like in future work to re-test BERT's fine-tuning (bert-tweets) with a model specially trained on French like CamemBERT [Martin et al., 2020] and not a multimodal model. We would also like to test fine-tuning of Sentence-BERT on a forthcoming French corpus of semantic similarity [Cardon and Grabar, 2020], and further explore the idea that semantic similarity (two sentences have the same meaning) and thematic similarity (two sentences talk about the same subject) are two different things that require different approaches.

Finally, we provide partial results on the role of image for the clustering of tweets. It is true that our two approaches to include multimodality (vector concatenation and re-clustering) in clustering are quite naive and could be improved with a more appropriate text-image fusion method. However, this is a first step, which suggests that image in tweets is more often a source of additional noise than a source of additional information.

Chapter 4

Linking Media events and Twitter events

4.1 Introduction

4.2 Related Work

The task of discovering joint events from news streams including both social and mainstream media has received little attention in the recent literature. In this section, we review existing works on similar tasks: aligning social-media contents with parts or paragraphs of a longer text, matching a tweet with a relevant news article, retrieve tweets related to a news article and, finally, jointly discover events from heterogeneous streams of documents.

4.2.1 Social content alignment

A first way of linking tweets and news is to associate to each part of a text the corresponding tweets (that are considered as comments or reactions to this specific part). Hu et al. [2012] develop ET-LDA, a Bayesian model that jointly extract topics from a text and a collection of tweets, and perform text segmentation. A segment may consist of one or several paragraphs, and each segment discusses a set of topics. Tweets are ”aligned” with one segment if most of their words belong to one of the topics of this segment. Conversely, they are defined as ”general tweets” if most of their words belong to general topics. The model is tested on two texts: a speech by President Obama and the transcript of a 2011 Republican Primary debate. The tweets are collected using hashtags (“#MESpeech” and “#ReaganDebate”) that unambiguously relate to these events.

4.2.2 Match tweets-articles pairs

Another related task consists in finding, for a given tweet, the most relevant news article. This research area aims at providing more context for the reader of a tweet, or for an automatic NLP tool. Guo et al. [2013] propose a graph-

based latent variable model that extracts text-to-text correlations via hashtags and named entities in order to enrich the sense of a short text and help to identify the most related article. They also introduce a dataset of articles from CNN and the New York Times and of tweets containing a link to one of these media outlets. The tweet-news pairs are identified by matching urls, but ”trivial” pairs (when the tweet contains exactly the title of the article) are removed from the dataset.

Zhao et al. [2019] use an interactive attention deep neural network in order to learn new representations of source and target texts. The representation of the source text is enriched with the target text, considered as the neighbor information or the context of the source text, and vice-versa. Mutual differences between the original and the new representation are used to produce a similarity score. The model is tested on several applications, including the tweet-article matching task defined by Guo et al. [2013].

Compared to Guo et al. [2013], Danovitch [2020] is interested in the reverse task: given a news article, find the most relevant tweet (though their architecture seems to be symmetrical and could probably be used for both tasks). They use a deep neural attention network with a Siamese architecture[1] to jointly embed tweet/article pairs. They address the problem of the decreasing weight of tokens with article length in attention-based architectures by using a sparse activation function [Peters et al., 2019]. They also use Star-Transformer [Guo et al., 2019], a lightweight alternative to the fully-connected Transformer architecture [Vaswani et al., 2017] that reduces its complexity from quadratic to linear time. The model is trained with triplet loss. At inference time, this architecture produces embeddings for tweets and articles, allowing to perform distance computations using cosine similarity.

4.2.3 Social content retrieval

Several articles [Tsagkias et al., 2011; Tanev et al., 2012; Suarez et al., 2018] address the task of linking traditional media and social media as an Information Retrieval problem, which can be formulated as follows: given a news article, find social media posts that reference it. In this approach, there is no notion of completeness of the retrieved data: these algorithms are often evaluated without taking the “size” of the event (in terms of total number of generated documents) into account. On the other hand, the order of the results is considered important: the most relevant documents should be among the first results. Each article proposes different strategies to model the best queries, that is find the keywords that will match relevant tweets, from article title, lead and body.

Wang et al. [2015] depart from this approach as the goal of the article is not to find the best tweets for a given article, but to associate tweets to clusters of articles. To this end, they perform hierarchical clustering on news articles in order to discover events and sub-events (called “aspects” of an event). Then, a candidate pool of tweets is retrieved using the text, entities and time of each aspect. The top tweets for each aspect are selected and used as seeds to train a classifier and label more tweets for each aspect.

[1] See section 3.2.2 for a definition of a Siamese network

4.2.4 Joint event detection

Few works address the joint detection of events in tweets and news. However, taking advantage of the richer content of press articles is known to be helpful to discover events among short texts like tweets [Phan et al., 2008].

Ning et al. [2015] aim at identifying interaction patterns between tweets and news. They only use press articles to perform event detection (that they call “chaining stories”). Tweets containing the url of one of the chained articles are then downloaded and de-facto considered as linked to the event. They then extract the top 10 tweets keywords for each event, as well as named entities from news articles, and download hourly count of the occurrences of these terms from the Twitter API. They then use this hourly count to detect peaks in Twitter activity and infer interaction patterns between the activity on Twitter and the publication of a news article.

Hua et al. [2016] propose *News and Twitter Interaction Topic model* (NTIT), an LDA-like model to jointly discover topics from a collection of tweets and news articles. In the NTIT model, tweets are assumed to consist of words that are either sampled from news topics or from Twitter topics. This asymmetric structure (tweets are not generated like news documents) is designed to prevent noisy tweets from degrading the performance of event detection on news articles. The authors propose an algorithm inspired from Gibbs Sampling [Welling et al., 2008] for the inference and parameters estimation of this generative model. The performance of this algorithm is assessed on a dataset composed of 74 events manually selected from the top news outlets of 5 South American countries. The authors retrieve tweets considered as relevant to these events by using keywords identified with tf-idf from the title and abstract of news reports. Their relevance to the given news is then manually checked. Finally, hashtags are extracted from the most relevant tweets and used to retrieve additional tweets. The task that the authors propose to solve is close to our own objectives. Furthermore, this is the only case (to our knowledge) of joint event detection where the evaluation dataset does not only contain media tweets or tweets collected because they contain the url of an article. This seems important to ensure that their method is able to handle the language specific to Twitter users. However, the proposed algorithm does not take into account the evolution of subjects over time, which can be problematic if applied over long periods of time.

Mele et al. [2017, 2019] present a variation of Dynamic Topic Model (DTM, [Blei and Lafferty, 2006]) called *dDTM-News*, to discover events in heterogeneous and dynamic streams of news documents (news articles, RSS feeds and tweets). Similarly to DTM, dDTM-News divides the corpus into time-slices and applies LDA to each of them. However, unlike DTM, the number of topics varies with each time slice, and the topics of one slice are independent from the previous slices. The discovered topics are linked to each other using a Hidden Markov Model (HMM) [Rabiner, 1989]. The optimal number of topics and the optimal number of Markov chains are discovered using Bayesian selection models through iteration on these parameters. Topic models represent each document in the form of a distribution over topics, and each document can thus be assigned to the most represented topic in its topics distribution, in order to perform clustering on the discovered topics. The dataset used for their experiments is

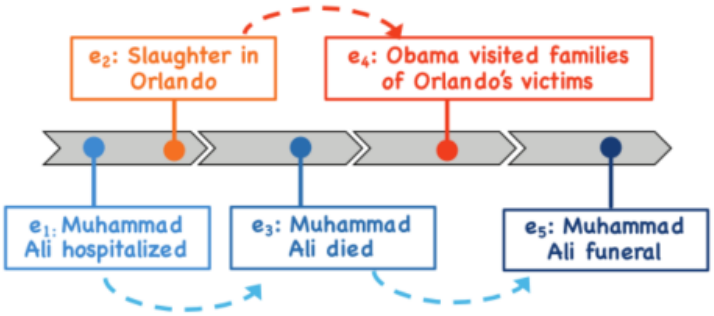


Figure 4.1: Two examples of event chains provided by Mele et al. [2019]. According to the authors, after event detection, the events are connected "based on their evolution" (see dashed arrows). However, the algorithm performing this chaining process is not detailed in the paper, and the event chains are absent from annotated data.

publicly available [Mele and Crestani, 2019], but the authors do not share their code. It contains 24,157 news articles, 43,381 RSS feeds, and 80,135 tweets issued by 9 popular news outlets (ABC News, Al Jazeera, BBC, CBC, CNN, NBC News, Reuters, United Press International, Xinhua China Agency). However, only 4,307 documents (3,681 unique documents) are annotated, and among which 744 tweets (695 unique tweets). This small number of tweets (17% of the annotated documents) poses a problem because the algorithm by Mele et al. [2017, 2019] makes no distinction in the nature of the documents, and is therefore probably less efficient on tweets, which are much shorter and therefore contain fewer words allowing them to be linked to a topic. In addition, the article does not specifies whether the algorithm is run on all the collected data, or just on the annotated data.

We downloaded the dataset and examined it: it seems that the data provided is incomplete, as the authors provide the "events" detected but not "event chains" (grouping of several events). For example, the paper [Mele et al., 2019] cites the case of Muhammad Ali's death as an example of a chain of events containing 3 sub-events: hospitalization, death and burial (see Figure 4.1). Still, in the annotated dataset, the sub-events have 3 different labels ("ali-muhammad, boxing, champion, hospitalized, respiratory", "boxing, died, louisville, muhammad-ali", "funeral, memorial, muhammad-ali, remembered"). It is therefore unclear how the authors evaluate the chaining of sub-events. Due to the incompleteness of the data provided by Mele et al. [2019], we could not test our algorithm on their dataset.

Compared to the previous works reviewed in this section, our contribution is fourfold:

1. We propose a new detection method capable of taking into account the temporal evolution of events.

2. Our approach treats articles from traditional media and Twitter separately at first, so as to preserve distinctive features of Twitter events. The identification of common events is a second step in the process.

3. We investigate the impact of URLs and hashtags (in addition to word similarity) in joint event detection.

4. We conduct our experiments on a realistic dataset, which is not only composed of tweets that contain URLs, or tweets from media accounts.

We describe the details of our joint event detection method in the following Section.

4.3 Methodology

Our approach can be broken down into three steps: first, we perform the detection of Twitter events and media events separately. Then we represent the similarity between detected events in a weighted bi-partite graph. Last, we apply a community detection algorithm in order to discover common events accross the two spheres.

4.3.1 First Story Detection

Tweets and news articles are quite different in terms of length and type of vocabulary. Detecting joint events directly from a heterogeneous collection of documents may thus lead to a poor performance (see Section 4.5.4). In order to let Twitter-specific and media-specific clusters emerge, we thus perform a first event detection step separately for each type of documents. We use the First Story Detection algorithm as described in Algorithm 2, which proved to be efficient both on tweets [Mazoyer et al., 2020] and news articles [Cagé et al., 2020]. Since the total number of news articles is much smaller than the number of tweets, we only use mini-batches for tweets, and not for news articles. Second, since words are likely to be used multiple times in each news article, we use the standard tf-idf formulation for news articles representation while we only compute idf for tweets. The formula of idf and tf-idf are given below:

$$idf(t) = 1 + log(n + 1/df(t) + 1) \tag{4.1}$$

$$tf\text{-}idf(t, d) = tf(t, d) \times idf(t) \tag{4.2}$$

where n is the number of documents in the collection, $tf(t, d)$ is the term frequency, *i.e.* the number of times a term t occurs in a document d and $df(t)$ is the document frequency, *i.e.* the number of documents in the collection that contain term t.

4.3.2 Event-similarity graph

Once events are detected separately in each sphere, we model the relationships between Twitter events and media events as a weighted bi-partite graph. In the rest of the Chapter, we denote $E_T = \{e_{T,1}, \ldots, e_{T,f}\}$ the set of all Twitter events, and $E_M = \{e_{M,1}, \ldots, e_{M,g}\}$ the set of all media events. We explore three types of links between

Twitter events and media events: word-similarity, URLs and hashtags.

Word-similarity. In order to compute a word-similarity metric between Twitter events and media events, we represent each event as the average of the tf-idf vectors of all documents it contains. The word-similarity between two events is computed as the cosine similarity between these average vectors and used to weight the edge between the two events:

$$weight_{text}(e_T, e_M) = \frac{\vec{e_T} \cdot \vec{e_M}}{\|\vec{e_T}\|\|\vec{e_M}\|} \tag{4.3}$$

Where $\vec{e}$ is the average of the tf-idf vectors of all documents in e. To facilitate the community detection step (see Section 4.3.3) , we then remove the edges with a too weak cosine similarity. We denote s the minimum similarity.

Hashtags. The graph of hashtag relationships between Twitter events and media events is built as follows: if some of the tweets of a Twitter event and some of the articles of a media event have hashtags in common, we draw an edge between the two events. The weight of hashtags is computed as follows:

$$weight_{htag}(e_T, e_M) = \frac{h(e_T, e_M)}{\max\{h(e_T, e_M) : e_T \in E_T, e_M \in E_M\}} \tag{4.4}$$

where $h(e_T, e_M)$ is the number of times the hashtags common to e_T and e_M appear in the Twitter event e_T. In order to limit the role of one individual hashtag (e.g. #BreakingNews) we remove edges where the number of different hashtags is too low.

URLs. The graph of URLs is constructed in the same way as the graph of hashtags: if tweets within a Twitter event e_T contain an URL pointing to one of the articles in media event e_M, we draw an edge between e_T and e_M. The weight of urls is computed as follows:

$$weight_{url}(e_T, e_M) = \frac{u(e_T, e_M)}{\max\{u(e_T, e_M) : e_T \in E_T, e_M \in E_M\}} \tag{4.5}$$

where $u(e_T, e_M)$ is the number of times urls linking to articles that are part of media event e_M appear in the tweets of Twitter event e_T.

Multidimensional graph. We combine these three different layers into a single multidimensional graph where the weight of the edges is computed as follows:

$$weight(e_T, e_M) = \sum_{i \in \{text, url, htag\}} \alpha_i weight_i(e_T, e_M) \tag{4.6}$$

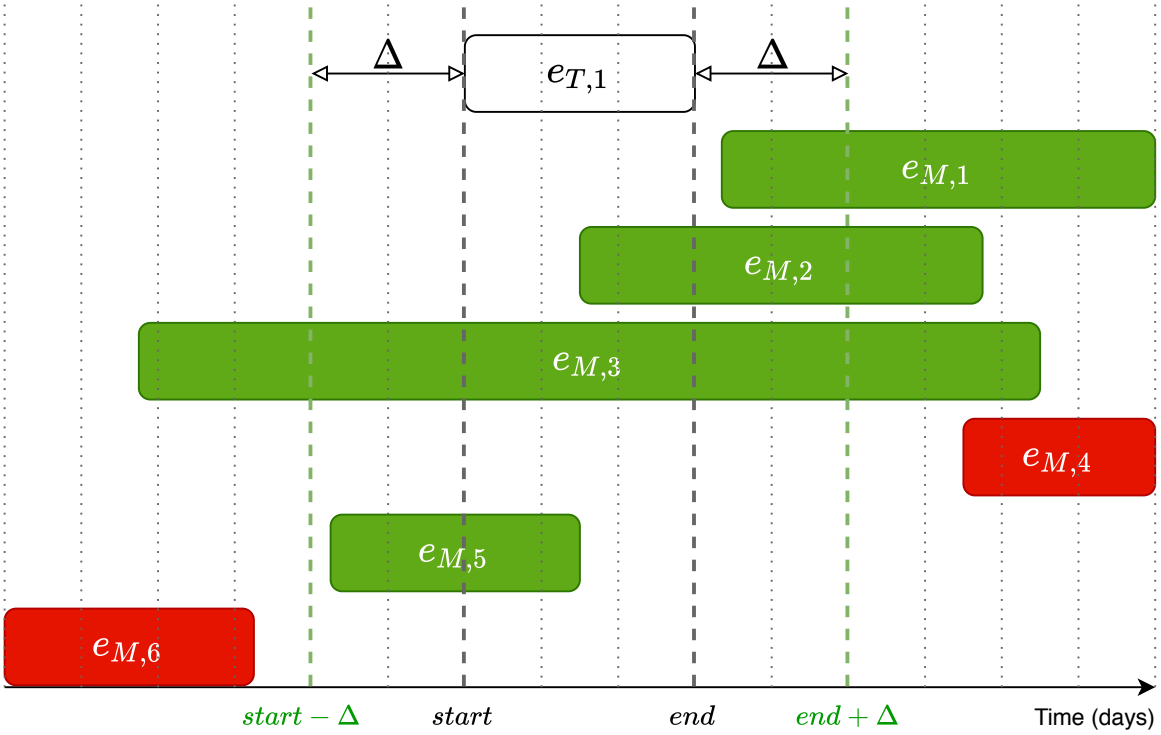


Figure 4.2: Example of different time configurations between a given Twitter event and some media events. Here the value of parameter Δ is set to 2 days. Media events ending before $start - \Delta$ or beginning after $end + \Delta$ appear in red. The edges between these events and $e_{T,1}$ are removed in the graph.

where $0 \leq \alpha_i \leq 1$ and $\sum_{i \in \{text, url, htag\}} \alpha_i = 1$.

Time of events. In addition to including word similarity, hashtags and urls in the construction of the event similarity graph, we also take into account the time dimension of the events. We therefore introduce a final parameter, Δ, which indicates the maximum time difference (in days) between a pair of events (e_T, e_M). More precisely, if we call $start$ and end the dates of the first and last document of a given event e_T, all media events containing at least one document published between $start - \Delta$ and $end + \Delta$ keep their links with e_T in the graph. Conversely, links between e_T and all other events are removed. Figure 4.2 shows examples of configurations where edges between events are kept, and others where edges are removed.

4.3.3 Community detection

Community detection within a network consists in decomposing the network into sub-groups of highly connected nodes. Researchers have proposed many strategies to solve this task, many of them based on the optimization of a given objective function.

Modularity

The most common of these objective functions is the modularity [Newman and Girvan, 2004] of the partition. The aim of this function is to isolate regions of the network where the density of links is higher than expected by chance. Modularity is defined formally as:

$$Q = \frac{1}{2m} \sum_{i,j} \left[weight(i,j) - \frac{k_i k_j}{2m} \right] \delta(c_i, c_j) \tag{4.7}$$

where

- $weight(i,j)$ represents the weight of the edge between i and j,
- $k_i = \sum_j weight(i,j)$ is the weighted degree of i,
- c_i is the community to which node i is assigned,
- $\delta(x,y)$ is 1 if $x = y$ and 0 otherwise, and
- $m = \sum_{ij} weight(i,j)$ is the weighted sum of edges.

The Louvain algorithm [Blondel et al., 2008] is, to the best of our knowledge, the fastest algorithm to find a partition of the nodes that maximizes modularity. We therefore use this algorithm to identify communities within the event-similarity graph.

Surprise

Another metric called surprise [Aldecoa and Marin, 2011] outperforms modularity on several benchmarks. It appears to be more efficient when the network contains communities of different sizes, since the modularity metric does not take into account n_c the number of nodes inside a community and n the total number of nodes, in the calculation of the density of links. Traag et al. [2015] propose an approximation of surprise in large networks, and an algorithm adapted from the Louvain algorithm, in order to maximize the surprise objective function. Their formulation of the function takes the following form:

$$Q = mD(q||\hat{q}) \tag{4.8}$$

where

- $m = \sum_{ij} weight(i,j)$ is the weighted sum of edges,
- $q = \frac{\sum_c m_c}{m}$ is the fraction of internal edges,
- $\hat{q} = \frac{\sum_c \binom{n_c}{2}}{\binom{n}{2}}$ is the expected fraction of internal edges, and

- $D(x||y) = x \log \frac{x}{y} + (1-x) \log \frac{1-x}{1-y}$ is the binary Kullback-Leibler divergence.

Our event similarity graph is expected to contain a lot of very small communities, containing only one (e_M, e_T) pair. However, in the case of long-lasting events with many sub-events, it is likely that the FSD algorithm over-clusters the events in one of the two spheres (and more likely in tweets). The community detection algorithm should then be able to detect also large communities with many nodes. We therefore also test the algorithm provided by Traag et al. [2015][2]. We detail our experiments in the next Section.

4.4 Experimental Setup

4.4.1 Dataset

We test our approach on the dataset presented in Chapter 2: it contains 95,796 tweets annotated as related to one of 327 "daily events" (events were drawn day by day and merged afterwards into 257 "macro events"). Among these daily events, 296 are actually news articles drawn randomly from a pool of French daily newspapers. The 31 remaining events were detected by monitoring unusually frequent terms on Twitter every day (see Section 2.4.2). We could manually associate 27 of them to a news article from the OTMedia collection [Hervé, 2019]. Concerning the last 4 events, we consider them as purely Twitter events that cannot be merged into a joint event, since we could not find any coverage of these events in traditional media. This makes our dataset all the more realistic: many Twitter events never lead to an article in mainstream media.

We ran the FSD algorithm on the OTMedia collection in order to automatically group the selected articles with other articles adressing the same topics. From these 323 articles (296 + 27), 167 were automatically clustered with other articles from our pool. In total, we obtained 15,544 news articles from 61 media outlets. We are aware that automatically grouping articles using the FSD algorithm may bias the dataset in favour of our approach, since part of it relies on the FSD algorithm. In order to prove the validity of our approach, we therefore systematically present two type of results: first, results computed on the entire dataset, second, results computed only on tweets (that were entirely manually annotated and are therefore not biased).

4.4.2 Evaluation metric

In an approach similar to that of Chapter 3 (see Section 3.5.3), we evaluate the performance of our algorithm using the "best matching" precision, recall and F1 score [Yang et al., 1998] for each event in the ground truth. We then compute the average of the events, to provide a macro-average result.

[2] `https://github.com/vtraag/louvain-igraph`

4.4.3 Parameter tuning

In the first step of our approach (First Story Detection applied on tweets and news articles separately), we use the same parameters as Cagé et al. [2020] for news articles (threshold t of 0.67 and time window w of one day), and the best parameters found in Mazoyer et al. [2020] (see Section 3.6.1: $t = 0.7$ and w set to one day) for tweets.

The graph construction step of our method requires many different parameters, leading to a risk of overfitting. To test the robustness of our model on different samples, we divide our dataset in 4 sub-samples of equal number of documents. The FSD algorithm requires documents to be sorted in chronological order, this is why we do not select documents randomly: we split the dataset in 4 different time periods and test each combination of parameters on each subset. We present the role of the different parameters in the next Section.

4.5 Results

The different parameters tend to have similar effects on each subset. We first detail the contribution of the different modalities of event similarity (word similarity, URLs, hashtags). We then present the best choice of Δ and s parameter.

4.5.1 Effect of word similarity, URLs and hashtags

Figures 4.3 and 4.4 show the best configuration of parameters α_{text}, α_{url}, α_{htag} and Δ for each subset. Overall, the results are quite stable to the change of parameters, event if the model performs better on all subsets when the weight of word-similarity (α_{text}) is high. URLs and hashtags seems to have a much lower effect, particularly when evaluating the model only on tweets (Figure 4.4). Increasing α_{url} or α_{htag} may slightly improve the performance on some time periods but there is no configuration that performs equally well on each subset.

We therefore chose to eliminate these modalities, in order to simplify the graph construction step and to obtain a model that is more stable to the change of dataset. We show the results with a model relying on text only in the right column of Figures 4.3 and 4.4. The low effect of URLs and hashtags can be explained by the fact that the information carried by these modalities is in most cases already present in the text of the documents.

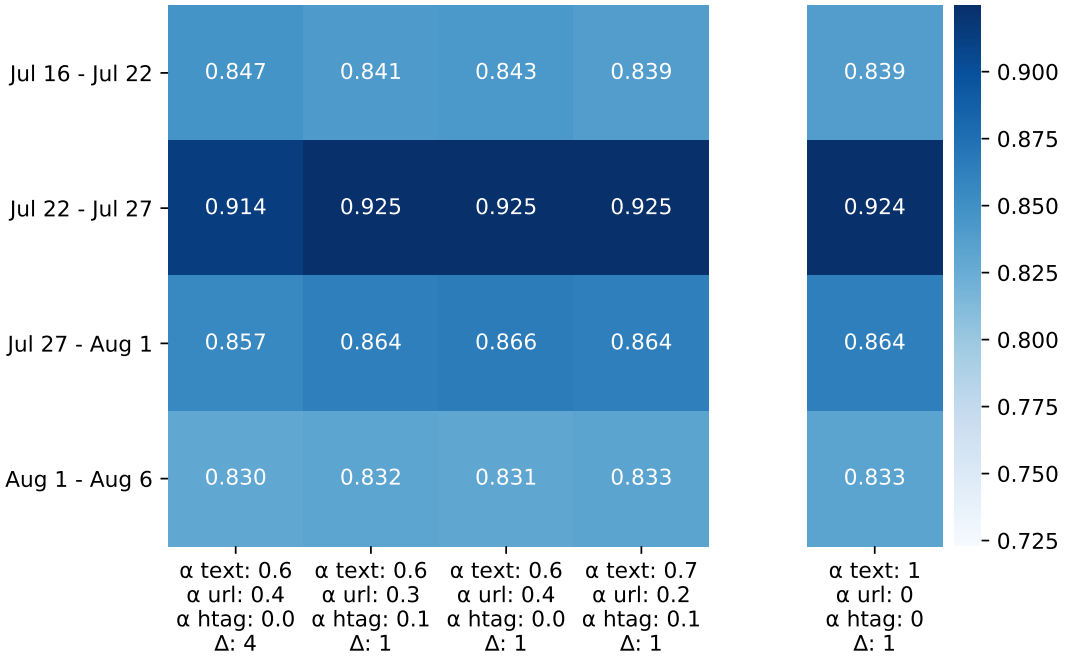


Figure 4.3: Cross-validation results on all documents. The result of community detection with the best set of parameters for a given sample is displayed on the diagonal of the square matrix. The results of that set of parameters on the other samples are displayed on the corresponding lines. The right column presents the results of community detection on the word-similarity graph (urls and hashtags are not taken into account).

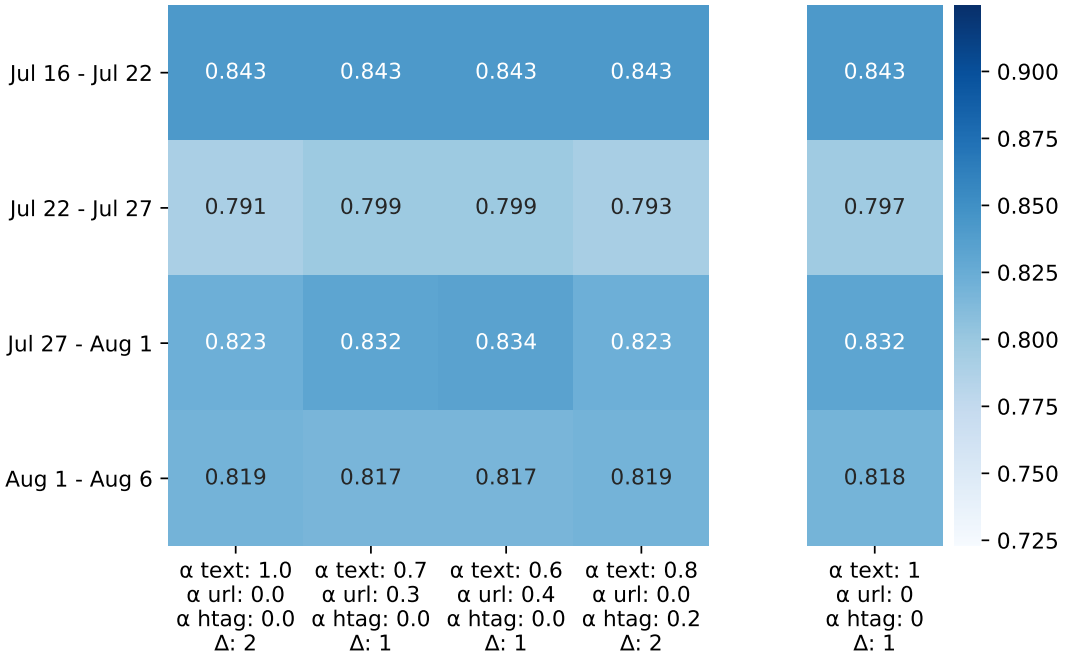


Figure 4.4: Cross-validation results on tweets only. The result of community detection with the best set of parameters for a given sample is displayed on the diagonal of the square matrix. The results of that set of parameters on the other samples are displayed on the corresponding lines. The right column presents the results of community detection on the word-similarity graph (urls and hashtags are not taken into account).

4.5.2 Effect of $\triangle$ parameter

4.5.3 Effect of s parameter

4.5.4 Results on the entire corpus

4.6 Conclusion

Chapter 5

Analysis of the spread of news on Twitter and traditional media

Chapter 6

Conclusion

Bibliography

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

L. M. Aiello, G. Petkos, C. J. Martín, D. Corney, S. Papadopoulos, R. Skraba, A. Göker, I. Kompatsiaris, and A. Jaimes. Sensing trending topics in twitter. *IEEE Trans. Multimedia*, 15(6):1268–1282, 2013.

R. Aldecoa and I. Marin. Deciphering network community structure by surprise. *PloS one*, 6, 09 2011.

J. Allan. Introduction to Topic Detection and Tracking. In J. Allan, editor, *Topic Detection and Tracking: Event-based Information Organization*, pages 1–16. Springer US, Boston, MA, 2002.

S. M. Alqhtani, S. Luo, and B. Regan. A multiple kernel learning based fusion for earthquake detection from multimedia twitter data. *Multimedia Tools and Applications*, 77(10):12519–12532, 2018.

F. Alvanaki, S. Michel, K. Ramamritham, and G. Weikum. See what's enblogue: real-time emergent topic identification in social media. In *15th International Conference on Extending Database Technology EDBT '12, Berlin, Germany, March 27-30, 2012, Proceedings*, pages 336–347, 2012.

R. Arun, V. Suresh, C. E. V. Madhavan, and M. N. N. Murthy. On Finding the Natural Number of Topics with Latent Dirichlet Allocation: Some Observations. In *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 391–402. Springer, Berlin, Heidelberg, 2010. ISBN 978-3-642-13656-6 978-3-642-13657-3. doi: 10.1007/978-3-642-13657-3_43. URL https://link.springer.com/chapter/10.1007/978-3-642-13657-3{_}43.

R. E. Bank and C. C. Douglas. Sparse matrix multiplication package (SMMP). *Adv. Comput. Math.*, 1(1):127–137, 1993.

H. Becker, F. Chen, D. Iter, M. Naaman, and L. Gravano. Automatic identification and presentation of twitter content for planned events. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*, 2011a.

H. Becker, M. Naaman, and L. Gravano. Beyond trending topics: Real-world event identification on twitter. In *Proceedings of the Fifth International Conference on Weblogs and Social Media, Barcelona, Catalonia, Spain, July 17-21, 2011*, 2011b.

D. M. Blei and J. D. Lafferty. Dynamic topic models. In *Machine Learning, Proceedings of the Twenty-Third International Conference (ICML 2006), Pittsburgh, Pennsylvania, USA, June 25-29, 2006*, volume 148 of *ACM International Conference Proceeding Series*, pages 113–120. ACM, 2006.

D. M. Blei, A. Y. Ng, and M. I. Jordan. Latent dirichlet allocation. In *Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada]*, pages 601–608. MIT Press, 2001.

V. D. Blondel, J.-L. Guillaume, R. Lambiotte, and E. Lefebvre. Fast unfolding of communities in large networks. *Journal of statistical mechanics: theory and experiment*, 2008(10), 2008.

S. R. Bowman, G. Angeli, C. Potts, and C. D. Manning. A large annotated corpus for learning natural language inference. In *Proceedings of the 2015 Conference on Empirical Methods in Natural Language Processing, EMNLP 2015, Lisbon, Portugal, September 17-21, 2015*, pages 632–642. The Association for Computational Linguistics, 2015.

J.-P. Brunet, P. Tamayo, T. R. Golub, and J. P. Mesirov. Metagenes and molecular pattern discovery using matrix factorization. *Proceedings of the National Academy of Sciences of the United States of America*, 101(12): 4164–4169, 2004. ISSN 0027-8424. doi: $10.1073/\mathrm{pnas}.0308531101$. URL `https://www.ncbi.nlm.nih.gov/pmc/articles/PMC384712/`.

J. Cagé, N. Hervé, and M.-L. Viaud. The production of information in an online world. *The Review of Economic Studies*, 2020.

D. Cardon, J.-P. Cointet, B. Ooghe, and G. Plique. Unfolding the multi-layered structure of the french mediascape. 2019.

R. Cardon and N. Grabar. A french corpus for semantic similarity. In *Proceedings of the Twelfth International Conference on Language Resources and Evaluation, LREC 2020, Marseille, France, May 11-16, 2020*. European Language Resources Association (ELRA), 2020.

D. Cer, M. Diab, E. Agirre, I. Lopez-Gazpio, and L. Specia. Semeval-2017 task 1: Semantic textual similarity-multilingual and cross-lingual focused evaluation. *arXiv preprint arXiv:1708.00055*, 2017.

D. Cer, Y. Yang, S.-y. Kong, N. Hua, N. Limtiaco, R. S. John, N. Constant, M. Guajardo-Cespedes, S. Yuan, C. Tar, et al. Universal sentence encoder. *arXiv preprint arXiv:1803.11175*, 2018.

P. Champagne. L'événement comme enjeu. *Réseaux. Communication-Technologie-Société*, 18(100):403–426, 2000.

W. Che, Y. Liu, Y. Wang, B. Zheng, and T. Liu. Towards better UD parsing: Deep contextualized word embeddings, ensemble, and treebank concatenation. In *Proceedings of the CoNLL 2018 Shared Task: Multilingual Parsing from Raw Text to Universal Dependencies*, pages 55–64, Brussels, Belgium, October 2018. Association for Computational Linguistics. URL `http://www.aclweb.org/anthology/K18-2005`.

T. Chen, X. He, and M.-Y. Kan. Context-aware image tweet modelling and recommendation. In *Proceedings of the 24th ACM international conference on Multimedia*, pages 1018–1027, 2016.

A. Conneau, D. Kiela, H. Schwenk, L. Barrault, and A. Bordes. Supervised learning of universal sentence representations from natural language inference data. *arXiv preprint arXiv:1705.02364*, 2017.

D. Corney, D. Albakour, M. Martinez-Alvarez, and S. Moussa. What do a million news articles look like? In *NewsIR@ ECIR*, pages 42–47, 2016.

J. Danovitch. Linking social media posts to news with siamese transformers. *CoRR*, abs/2001.03303, 2020.

J. de Bruijn, H. de Moel, B. Jongman, and J. Aerts. Towards a global flood detection system using social media. In *EGU General Assembly Conference Abstracts*, volume 19, page 1102, 2017.

J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. *arXiv preprint arXiv:1810.04805*, 2018.

M. Ester, H.-P. Kriegel, J. Sander, X. Xu, et al. A density-based algorithm for discovering clusters in large spatial databases with noise. In *KDD*, volume 96, pages 226–231, 1996.

J. L. Fleiss. Measuring nominal scale agreement among many raters. *Psychological bulletin*, 76(5):378, 1971.

F. Fleuret and H. Sahbi. Scale-invariance of support vector machines based on the triangular kernel. In *3rd International Workshop on Statistical and Computational Theories of Vision*, pages 1–13, 2003.

S. Gaglio, G. L. Re, and M. Morana. A framework for real-time twitter data analysis. *Computer Communications*, 73:236–242, 2016.

F. Godin, B. Vandersmissen, W. De Neve, and R. Van de Walle. Multimedia lab @ acl wnut ner shared task: Named entity recognition for twitter microposts using distributed word representations. In *Proceedings of the Workshop on Noisy User-generated Text*, pages 146–153, 2015.

D. Greene, D. OCallaghan, and P. Cunningham. How Many Topics? Stability Analysis for Topic Models. In *Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 498–513. Springer,

Berlin, Heidelberg, 2014. ISBN 978-3-662-44847-2 978-3-662-44848-9. doi: 10.1007/978-3-662-44848-9_32. URL https://link.springer.com/chapter/10.1007/978-3-662-44848-9{_}32.

T. L. Griffiths and M. Steyvers. Finding scientific topics. *Proceedings of the National Academy of Sciences*, 101 (Supplement 1):5228–5235, 2004. ISSN 0027-8424, 1091-6490. doi: 10.1073/pnas.0307752101. URL http://www.pnas.org/cgi/doi/10.1073/pnas.0307752101.

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

Q. Guo, X. Qiu, P. Liu, Y. Shao, X. Xue, and Z. Zhang. Star-transformer. In J. Burstein, C. Doran, and T. Solorio, editors, *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2019, Minneapolis, MN, USA, June 2-7, 2019, Volume 1 (Long and Short Papers)*, pages 1315–1325. Association for Computational Linguistics, 2019.

W. Guo, H. Li, H. Ji, and M. T. Diab. Linking tweets to news: A framework to enrich short text data in social media. In *Proceedings of the 51st Annual Meeting of the Association for Computational Linguistics, ACL 2013, 4-9 August 2013, Sofia, Bulgaria, Volume 1: Long Papers*, pages 239–249, 2013.

Y. Halberstam and B. Knight. Homophily, group size, and the diffusion of political information in social networks: Evidence from Twitter. *Journal of Public Economics*, 143:73–88, 2016. ISSN 0047-2727. doi: http://dx.doi.org/10.1016/j.jpubeco.2016.08.011. URL http://www.sciencedirect.com/science/article/pii/S0047272716301001.

R. A. Harder, S. Paulussen, and P. Van Aelst. Making sense of twitter buzz: The cross-media construction of news stories in election time. *Digital Journalism*, 4(7):933–943, 2016.

Z. S. Harris. Distributional structure. *Word*, 10(2-3):146–162, 1954.

M. Hasan, M. A. Orgun, and R. Schwitter. TwitterNews: Real time event detection from the Twitter data stream. *PeerJ PrePrints*, 2016.

M. Hasan, M. A. Orgun, and R. Schwitter. A survey on real-time event detection from the twitter data stream. *J. Information Science*, 44(4):443–463, 2018.

K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016.

N. Hervé. OTMedia – the French transmedia news observatory. In *FIAT/IFTA Media Management Seminar.*, 2019. www.herve.name/research/nherve_ina_fiat_ stockholm_otmedia_preprint.pdf.

Y. Hu, A. John, F. Wang, and S. Kambhampati. ET-LDA: joint topic modeling for aligning events and their twitter feedback. 2012.

T. Hua, Y. Ning, F. Chen, C. Lu, and N. Ramakrishnan. Topical analysis of interactions between news and social media. In D. Schuurmans and M. P. Wellman, editors, *Proceedings of the Thirtieth AAAI Conference on Artificial Intelligence, February 12-17, 2016, Phoenix, Arizona, USA*, pages 2964–2971. AAAI Press, 2016.

A. Joly and O. Buisson. A posteriori multi-probe locality sensitive hashing. In *Proceedings of the 16th ACM international conference on Multimedia*, pages 209–218. ACM, 2008.

K. Joseph, P. M. Landwehr, and K. M. Carley. Two 1% s Don't Make a Whole: Comparing Simultaneous Samples from Twitter's Streaming API. In *SBP*, pages 75–83. Springer, 2014.

D. Kergl, R. Roedler, and S. Seeber. On the endogenesis of twitter's spritzer and gardenhose sample streams. In *2014 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2014, Beijing, China, August 17-20, 2014*, pages 357–364, 2014.

R. Kiros, Y. Zhu, R. R. Salakhutdinov, R. Zemel, R. Urtasun, A. Torralba, and S. Fidler. Skip-thought vectors. In *Advances in neural information processing systems*, pages 3294–3302, 2015.

S. Kullback. *Information theory and statistics*. Courier Corporation, 1997.

H. Kwak, C. Lee, H. Park, and S. Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th international conference on World wide web*, pages 591–600. AcM, 2010.

C. Li, Y. Duan, H. Wang, Z. Zhang, A. Sun, and Z. Ma. Enhancing topic modeling for short texts with auxiliary word embeddings. *ACM Trans. Inf. Syst.*, 36(2):11:1–11:30, 2017. doi: $10.1145/3091108$. URL `http://doi.acm.org/10.1145/3091108`.

S. Likhitha, B. Harish, and H. K. Kumar. A detailed survey on topic modeling for document and short text data. *International Journal of Computer Applications*, 975:8887, 2019.

J. Lin, M. Efron, Y. Wang, and G. Sherman. Overview of the trec-2014 microblog track. Technical report, MARYLAND UNIV COLLEGE PARK, 2014.

J. Lin, M. Efron, Y. Wang, G. Sherman, and E. Voorhees. Overview of the trec-2015 microblog track. Technical report, MARYLAND UNIV COLLEGE PARK, 2015.

X. Liu, Q. Li, A. Nourbakhsh, R. Fang, M. Thomas, K. Anderson, R. Kociuba, M. Vedder, S. Pomerville, R. Wudali, R. Martin, J. Duprey, A. Vachher, W. Keenan, and S. Shah. Reuters Tracer: A Large Scale System of Detecting & Verifying Real-Time News Events from Twitter. In *Proceedings of the 25th ACM International Conference on Information and Knowledge Management, CIKM 2016, Indianapolis, IN, USA, October 24-28, 2016*, pages 207–216, 2016.

X. Liu, A. Nourbakhsh, Q. Li, S. Shah, R. Martin, and J. Duprey. Reuters tracer: Toward automated news production using large scale social media data. In *2017 IEEE International Conference on Big Data (Big Data)*, pages 1483–1493. IEEE, 2017.

D. G. Lowe. Object recognition from local scale-invariant features. In *Computer vision, 1999. The proceedings of the seventh IEEE international conference on*, volume 2, pages 1150–1157. Ieee, 1999.

D. Lu, L. Neves, V. Carvalho, N. Zhang, and H. Ji. Visual attention model for name tagging in multimodal social media. In *Proceedings of the 56th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*, volume 1, pages 1990–1999, 2018.

R. Ma, Q. Zhang, J. Wang, L. Cui, and X. Huang. Mention recommendation for multimodal microblog with cross-attention memory network. In *SIGIR*, pages 195–204, 2018.

L. v. d. Maaten and G. Hinton. Visualizing data using t-sne. *Journal of machine learning research*, 9(Nov):2579–2605, 2008.

L. Martin, B. Muller, P. J. O. Suárez, Y. Dupont, L. Romary, É. V. de la Clergerie, D. Seddah, and B. Sagot. Camembert: a tasty french language model. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*, 2020.

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

B. Mazoyer, J. Cagé, N. Hervé, and C. Hudelot. A french corpus for event detection on twitter. In *Proceedings of the Twelfth International Conference on Language Resources and Evaluation (LREC 2020)*, pages 6220–6227, 2020.

S. C. McGregor and L. Molyneux. Twitter's influence on news judgment: An experiment among journalists. *Journalism*, page 1464884918802975, 2018.

A. J. McMinn and J. M. Jose. Real-Time Entity-Based Event Detection for Twitter. *Experimental IR Meets Multilinguality, Multimodality, and Interaction*, 9283:65–77, 2015. doi: $10.1007/978\text{-}3\text{-}319\text{-}24027\text{-}5\_6$.

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

I. Mele and F. Crestani. A multi-source collection of event-labeled news documents. In *Proceedings of the 2019 ACM SIGIR International Conference on Theory of Information Retrieval*, pages 205–208. ACM, 2019.

I. Mele, S. A. Bahrainian, and F. Crestani. Linking News across Multiple Streams for Timeliness Analysis. pages 767–776. ACM Press, 2017. ISBN 978-1-4503-4918-5. doi: 10.1145/3132847.3132988. URL http://dl.acm.org/citation.cfm?doid=3132847.3132988.

I. Mele, S. A. Bahrainian, and F. Crestani. Event mining and timeliness analysis from heterogeneous news streams. *Information Processing and Management*, 56(3):969–993, 2019.

T. Mikolov, K. Chen, G. Corrado, and J. Dean. Efficient Estimation of Word Representations in Vector Space. *arXiv:1301.3781 [cs]*, 2013. URL http://arxiv.org/abs/1301.3781.

F. Morstatter, J. Pfeffer, and H. Liu. When is it biased?: assessing the representativeness of twitter's streaming API. In *23rd International World Wide Web Conference, \WWW '14, Seoul, Republic of Korea, April 7-11, 2014, Companion Volume*, pages 555–556. ACM Press, 2014. ISBN 978-1-4503-2745-9. doi: 10.1145/2567948.2576952. URL http://dl.acm.org/citation.cfm?doid=2567948.2576952.

M. E. Newman and M. Girvan. Finding and evaluating community structure in networks. *Physical review E*, 69(2): 026113, 2004.

D. Q. Nguyen, R. Billingsley, L. Du, and M. Johnson. Improving topic models with latent feature word representations. *Transactions of the Association for Computational Linguistics*, 3:299–313, 2015.

K. Nigam, A. K. McCallum, S. Thrun, and T. Mitchell. Text classification from labeled and unlabeled documents using em. *Machine learning*, 39(2-3):103–134, 2000.

Y. Ning, S. Muthiah, R. Tandon, and N. Ramakrishnan. Uncovering news-twitter reciprocity via interaction patterns. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining, ASONAM 2015, Paris, France, August 25 - 28, 2015*, pages 1–8, 2015.

P. Nora. L'événement monstre. *Communications*, 18(1):162–172, 1972.

V. S. Pagolu, K. N. Reddy, G. Panda, and B. Majhi. Sentiment analysis of twitter data for predicting stock market movements. In *2016 international conference on signal processing, communication, power and embedded system (SCOPES)*, pages 1345–1350. IEEE, 2016.

N. Panagiotou, I. Katakis, and D. Gunopulos. Detecting Events in Online Social Networks: Definitions, Trends and Challenges. In S. Michaelis, N. Piatkowski, and M. Stolpe, editors, *Solving Large Scale Learning Tasks. Challenges and Algorithms*, volume 9580. Springer International Publishing, Cham, 2016. URL http://link.springer.com/10.1007/978-3-319-41706-6_2.

S. Papadopoulos, D. Corney, and L. M. Aiello. Snow 2014 data challenge: Assessing the performance of news topic detection methods in social media. In *Proceedings of the SNOW 2014 Data Challenge co-located with 23rd International World Wide Web Conference (WWW 2014)*, 2014.

J. Pennington, R. Socher, and C. D. Manning. Glove: Global vectors for word representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, EMNLP 2014, October 25-29, 2014, Doha, Qatar, A meeting of SIGDAT, a Special Interest Group of the ACL*, pages 1532–1543. ACL, 2014.

B. Peters, V. Niculae, and A. F. T. Martins. Sparse sequence-to-sequence models. In *Proceedings of the 57th Conference of the Association for Computational Linguistics, ACL 2019, Florence, Italy, July 28- August 2, 2019, Volume 1: Long Papers*, pages 1504–1519. Association for Computational Linguistics, 2019.

M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer. Deep contextualized word representations. In *Proceedings of the 2018 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, NAACL-HLT 2018, New Orleans, Louisiana, USA, June 1-6, 2018, Volume 1 (Long Papers)*, pages 2227–2237. Association for Computational Linguistics, 2018.

G. Petkos, S. Papadopoulos, L. M. Aiello, R. Skraba, and Y. Kompatsiaris. A soft frequent pattern mining approach for textual topic detection. In *4th International Conference on Web Intelligence, Mining and Semantics (WIMS 14), WIMS '14, Thessaloniki, Greece, June 2-4, 2014*, pages 25:1–25:10, 2014a.

G. Petkos, S. Papadopoulos, V. Mezaris, and Y. Kompatsiaris. Social event detection at mediaeval 2014: Challenges, datasets, and evaluation. In *MediaEval*. Citeseer, 2014b.

S. Petrović, M. Osborne, and V. Lavrenko. Streaming first story detection with application to Twitter. In *Human Language Technologies: The 2010 Annual Conference of the North American Chapter of the Association for Computational Linguistics*, pages 181–189. Association for Computational Linguistics, 2010.

S. Petrović, M. Osborne, and V. Lavrenko. Using paraphrases for improving first story detection in news and twitter. In *Human Language Technologies: Conference of the North American Chapter of the Association of Computational Linguistics, Proceedings, June 3-8, 2012, Montréal, Canada*, pages 338–346, 2012.

X. H. Phan, M. L. Nguyen, and S. Horiguchi. Learning to classify short and sparse text & web with hidden topics from large-scale data collections. In *Proceedings of the 17th International Conference on World Wide Web, WWW 2008, Beijing, China, April 21-25, 2008*, pages 91–100. ACM, 2008.

L. R. Rabiner. A tutorial on hidden markov models and selected applications in speech recognition. *Proceedings of the IEEE*, 77(2):257–286, 1989.

J. J. Randolph. Free-marginal multirater kappa (multirater k [free]): An alternative to fleiss' fixed-marginal multirater kappa. *Online submission*, 2005.

N. Reimers and I. Gurevych. Sentence-bert: Sentence embeddings using siamese bert-networks. *arXiv preprint arXiv:1908.10084*, 2019.

Ø. Repp and H. Ramampiaro. Extracting news events from microblogs. *Journal of Statistics and Management Systems*, 21(4):695–723, 2018.

T. Reuter, S. Papadopoulos, G. Petkos, V. Mezaris, Y. Kompatsiaris, P. Cimiano, C. de Vries, and S. Geva. Social event detection at mediaeval 2013: Challenges, datasets, and evaluation. In *Proceedings of the MediaEval 2013 Multimedia Benchmark Workshop Barcelona, Spain, October 18-19, 2013*, 2013.

M. Sahlgren. An introduction to random indexing. In *In Methods and Applications of Semantic Indexing Workshop at the 7th International Conference on Terminology and Knowledge Engineering, TKE 2005*, 2005.

J. Sankaranarayanan, H. Samet, B. E. Teitler, M. D. Lieberman, and J. Sperling. Twitterstand: news in tweets. In *17th ACM SIGSPATIAL International Symposium on Advances in Geographic Information Systems, ACM-GIS 2009, November 4-6, 2009, Seattle, Washington, USA, Proceedings*, pages 42–51, 2009.

K. Sparck Jones. A statistical interpretation of term specificity and its application in retrieval. *Journal of Documentation*, 28(1):11–21, 1972.

A. Suarez, D. Albakour, D. Corney, M. Martinez, and J. Esquivel. A data collection for evaluating the retrieval of related tweets to news articles. In *European Conference on Information Retrieval*, pages 780–786, 2018.

A. Swasy. A little birdie told me: Factors that influence the diffusion of twitter in newsrooms. *Journal of Broadcasting & Electronic Media*, 60(4):643–656, 2016.

H. Tanev, M. Ehrmann, J. Piskorski, and V. Zavarella. Enhancing Event Descriptions through Twitter Mining. In *Proceedings of the Sixth International Conference on Weblogs and Social Media, ICWSM 2012, Dublin, Ireland, June 4-7, 2012*, 2012.

V. A. Traag, R. Aldecoa, and J.-C. Delvenne. Detecting communities using asymptotical surprise. *Physical Review E*, 92(2):022816, 2015.

M. Tsagkias, M. de Rijke, and W. Weerkamp. Linking online news and social media. In *Proceedings of the Forth International Conference on Web Search and Web Data Mining, WSDM 2011, Hong Kong, China, February 9-12, 2011*, pages 565–574, 2011.

A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Ł. Kaiser, and I. Polosukhin. Attention is all you need. In *Advances in neural information processing systems*, pages 5998–6008, 2017.

A. Wang, A. Singh, J. Michael, F. Hill, O. Levy, and S. R. Bowman. Glue: A multi-task benchmark and analysis platform for natural language understanding. *arXiv preprint arXiv:1804.07461*, 2018.

J. Wang, W. Tong, H. Yu, M. Li, X. Ma, H. Cai, T. Hanratty, and J. Han. Mining multi-aspect reflection of news events in twitter: Discovery, linking and presentation. In *2015 IEEE International Conference on Data Mining, ICDM 2015, Atlantic City, NJ, USA, November 14-17, 2015*, pages 429–438, 2015.

M. Welling, Y. W. Teh, and B. Kappen. Hybrid variational/gibbs collapsed inference in topic models. In *UAI 2008, Proceedings of the 24th Conference in Uncertainty in Artificial Intelligence, Helsinki, Finland, July 9-12, 2008*, pages 587–594. AUAI Press, 2008.

Y. Yang, T. Pierce, and J. G. Carbonell. A study of retrospective and on-line event detection. In *Proc. of ACM-SIGIR*, pages 28–36, 1998.

J. Yin and J. Wang. A dirichlet multinomial mixture model-based approach for short text clustering. In *The 20th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, KDD '14, New York, NY, USA*, pages 233–242, 2014. doi: $10.1145/2623330.2623715$. URL `http://doi.acm.org/10.1145/2623330.2623715`.

C. Zhang, L. Liu, D. Lei, Q. Yuan, H. Zhuang, T. Hanratty, and J. Han. Triovecevent: Embedding-based online local event detection in geo-tagged tweet streams. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Halifax, NS, Canada, August 13 - 17, 2017*, pages 595–604, 2017a.

C. Zhang, S. Lu, C. Zhang, X. Xiao, Q. Wang, and G. Chen. A novel hot topic detection framework with integration of image and short text information from twitter. *IEEE Access*, 7:9225–9231, 2018.

Q. Zhang, J. Wang, H. Huang, X. Huang, and Y. Gong. Hashtag recommendation for multimodal microblog using co-attention network. In *IJCAI*, pages 3420–3426, 2017b.

S. Zhao, Y. Huang, C. Su, Y. Li, and F. Wang. Interactive attention for semantic text matching. *CoRR*, abs/1911.06146, 2019. URL `http://arxiv.org/abs/1911.06146`.

Appendix A

Clusters of terms used as tweet collection parameters

These clusters where obtained using spectral clustering with K clusters on the $(N \times N)$ words co-occurrence matrices.

$K = 2$

- $N = 50$
  - a, au, avec, bien, c, ca, ce, cest, dans, de, des, du, elle, en, est, et, faire, fait, il, jai, la, le, les, ma, mais, meme, moi, mon, ne, on, plus, pour, quand, qui, se, si, sur, toi, tout, trop, tu, un, une, va, vous
  - je, me, pas, que, suis
- $N = 100$
  - ah, alors, au, aussi, avec, bah, bien, bon, bonne, c, ca, ce, cette, comme, dans, deja, dire, dit, donc, du, elle, encore, etre, faire, fais, fait, faut, ils, jai, jamais, je, jsuis, la, ma, mais, mdr, me, meme, merci, mes, moi, mon, ne, non, nous, on, oui, par, pas, pour, que, qui, quoi, rien, sa, sais, se, si, son, sont, suis, sur, ta, te, tellement, tes, toi, ton, tous, tout, tres, trop, tu, une, va, vais, veux, vie, voir, vous, vraiment, y, ya, ça
  - a, cest, de, des, en, est, et, il, le, les, lui, ou, plus, quand, quil, un
- $N = 200$
  - 1, 2, 3, ah, ai, aller, alors, ans, apres, as, au, aussi, aux, avant, avec, avoir, bah, beau, beaucoup, belle, bien, bon, bonjour, bonne, c, ca, ce, ces, cetait, cette, chez, comme, comment, compte, contre, coup,

cours, crois, d, dans, deja, demain, depuis, deux, dire, dis, dit, donc, du, elle, encore, envie, es, etait, ete, etre, eu, faire, fais, fait, faut, fois, g, genre, gens, grave, gros, il, jai, jaime, jamais, javais, je, jen, jespere, jme, jour, journee, jsuis, juste, jvais, la, lui, ma, mais, mal, mdr, mdrr, mdrrr, me, mec, meme, merci, merde, mere, mes, mieux, moi, moins, moment, mon, monde, ne, nest, non, nous, oh, ok, on, ou, ouais, oui, par, parce, parle, pas, pense, personne, petit, peu, peut, peux, plus, pour, pr, ptdr, ptn, putain, quand, quel, quelle, quil, quoi, quon, rien, rt, sa, sais, sans, se, ses, si, soir, son, sont, suis, super, sur, t, ta, tas, te, tellement, temps, tes, tete, toi, ton, toujours, tous, tout, toute, tres, trop, trouve, tu, une, va, vais, veut, veux, vie, viens, voir, vois, votre, vous, vrai, vraiment, vu, y, ya, ça

  - a, cest, de, des, en, est, et, ils, le, les, leur, ont, passe, pourquoi, que, qui, un

$K = 3$

- $N = 50$
  - a, au, avec, c, ce, dans, de, des, du, elle, en, est, et, faire, fait, il, jai, la, le, les, ma, me, moi, mon, plus, pour, qui, se, sur, tout, trop, un, une, vous
  - bien, ca, cest, mais, meme, ne, on, pas, quand, que, si, toi, tu, va
  - je, suis
- $N = 100$
  - ah, alors, au, aussi, avec, bah, bien, bon, bonne, c, ca, ce, cette, comme, dans, deja, dire, dit, donc, du, elle, encore, etre, faire, fais, fait, faut, ils, jai, jamais, jsuis, la, ma, mais, mdr, me, meme, merci, mes, moi, mon, ne, non, nous, on, oui, par, pas, pour, que, qui, quoi, rien, sa, sais, se, si, son, sont, suis, sur, ta, te, tellement, tes, toi, ton, tous, tout, tres, trop, tu, une, va, vais, veux, vie, voir, vous, vraiment, y, ya, ça
  - a, cest, de, des, en, est, et, il, le, les, lui, ou, plus, quand, quil, un
  - je
- $N = 200$
  - a, cest, de, des, en, est, et, ils, le, les, leur, ont, pourquoi, qui, un
  - jen, passe, que, tu
  - 1, 2, 3, ah, ai, aller, alors, ans, apres, as, au, aussi, aux, avant, avec, avoir, bah, beau, beaucoup, belle, bien, bon, bonjour, bonne, c, ca, ce, ces, cetait, cette, chez, comme, comment, compte, contre, coup, cours, crois, d, dans, deja, demain, depuis, deux, dire, dis, dit, donc, du, elle, encore, envie, es, etait, ete, etre, eu, faire, fais, fait, faut, fois, g, genre, gens, grave, gros, il, jai, jaime, jamais, javais, je, jespere, jme,

jour, journee, jsuis, juste, jvais, la, lui, ma, mais, mal, mdr, mdrr, mdrrr, me, mec, meme, merci, merde, mere, mes, mieux, moi, moins, moment, mon, monde, ne, nest, non, nous, oh, ok, on, ou, ouais, oui, par, parce, parle, pas, pense, personne, petit, peu, peut, peux, plus, pour, pr, ptdr, ptn, putain, quand, quel, quelle, quil, quoi, quon, rien, rt, sa, sais, sans, se, ses, si, soir, son, sont, suis, super, sur, t, ta, tas, te, tellement, temps, tes, tete, toi, ton, toujours, tous, tout, toute, tres, trop, trouve, une, va, vais, veut, veux, vie, viens, voir, vois, votre, vous, vrai, vraiment, vu, y, ya, ça

Appendix B

Second Appendix

Titre: titre (en français)

Mots clés: 3 à 6 mots clés

Résumé: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.
Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.
Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Title: titre (en anglais)

Keywords: 3 à 6 mots clés

Abstract: Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.
Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.
Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Université Paris-Saclay
Espace Technologique / Immeuble Discovery
Route de l'Orme aux Merisiers RD 128 / 91190 Saint-Aubin, France