POLITECNICO DI MILANO
Corso di Laurea Magistrale in Ingegneria Informatica
Dipartimento di Elettronica, Informazione e Bioingegneria

POLITECNICO
MILANO 1863

A Customisable Pipeline for Continuously Harvesting Socially-Minded Twitter Users

Relatore: Prof. Carlo Piccardi

Tesi di Laurea di:
Flavio Primo, Mat. 881280

Anno Accademico 2019-2020

"Then I understood that a survivor has no right to bring a complaint.
Whoever survives has won his case, he has no right and no cause to bring charges;
he has emerged the stronger, the more cunning, the more obstinate,
from the struggle."
Sándor Márai, Embers

Summary

On social media platforms, and Twitter in particular, specific classes of users such as *influencers* have been given satisfactory operational definitions in terms of network and content metrics. Others, for instance *online activists*, are not less important but their characterisation still requires experimenting.

We make the hypothesis that such interesting users can be found within temporally and spatially localised *contexts*, i.e., small but topical fragments of the network containing interactions about social events or campaigns with a significant footprint on Twitter. To explore this hypothesis, we have designed a continuous user profile discovery pipeline that produces an ever-growing dataset of user profiles by harvesting and analysing contexts from the Twitter stream. The profiles dataset includes key network topology and content-based users metrics, enabling experimentation with user-defined score functions that characterise specific classes of online users.

The paper describes the design and implementation of the pipeline and its empirical evaluation on a case study consisting of healthcare-related campaigns in the UK, showing how it supports the operational definitions of online activism, by comparing three experimental ranking functions.

This project is part of a joint program of the Newcastle University (UK) and the Federal University of Rio Grande Do Norte (Brazil) in an effort to fight mosquito-borne diseases such as Zika, Dengue and Chikungunya.

This project has been presented and published [43] at the 2019 edition of the ICWE conference in South Korea [1].

The source code is publicly available[2].

[1] ICWE International Conference on Web Engineering - 19th edition - South Korea, Daejeon `https://icwe2019.webengineering.org`

[2] `https://github.com/flaprimo/twitter-network-analysis`

Sommario

Alcune categorie di utenti come gli *influencers*, nei social media, e in Twitter in particolare, hanno definizioni operative soddisfacenti in termini di metriche di topologia di rete e contenuti. Altre, ad esempio gli *attivisti online*, non sono meno importanti ma la loro caratterizzazione richiede un approfondimento.

Ipotizziamo che questa interessante categoria di utenti possa essere trovata in *contesti* localizzati a livello temporale e spaziale, ovvero, piccoli frammenti topici di reti contenenti interazioni relative ad eventi o campagne social con una significativa presenza su Twitter. Per esplorare questa ipotesi abbiamo progettato una procedura per la scoperta continua di profili di utenti, che produce un dataset di profili in continua evoluzione, raccogliendo e analizzando contesti da Twitter. Il dataset include importanti metriche riguardanti la topologia di rete e i contenuti pubblicati dagli utenti, permettendo la sperimentazione di funzioni personalizzate di ordinamento dei profili degli utenti online, caratterizzandoli in specifiche classi di appartenenza.

Il paper descrive la progettazione e l'implementazione della procedura e la sua valutazione empirica su un caso di studio riguardante le campagne social della sanità britannica, mostrando come supporta la definizione operativa dell'attivismo online, nella comparazione di tre funzioni di ordinamento sperimentali.

Questo progetto fa parte di un programma di lavoro congiunto tra l'Università di Newcastle (Regno Unito) e l'Università Federale del Rio Grande Do Norte (Brasile) per combattere le malattie trasmesse dalle zanzare, come Zika, Dengue e Chikunguya.

Questo progetto è stato presentato e pubblicato [43] durante l'edizione del 2019 della conferenza ICWE[3] nella Corea del Sud.

Il codice sorgente è pubblicamente disponibile[4].

[3]ICWE International Conference on Web Engineering - 19th edition - South Korea, Daejeon `https://icwe2019.webengineering.org`

[4]`https://github.com/flaprimo/twitter-network-analysis`

Acknowledgements

"It's a strange world, isn't it?" says a lost Jeffrey Beaumont in a violent and dark but yet interesting and beautiful world.

These master period is easily one of the most exciting and strangest of my life. After being emotively exhausted for personal reasons and finishing the Bachelor in Rome rushing through the thesis work, last exams and the English certification, I was enrolled at Politecnico di Milano.

It was a hard time at the beginning.

Brick by brick I got to know good friends, got out of the comfort zone, visited beautiful places, took care of myself and be independent, got a person beside me and, ultimately, got to know more about myself. From Milan to Newcastle, from Rio to Seoul, I couldn't ask for more.

I want to thank any individual that I encountered in this journey, I am the Flavio 2.0 that you got to know also because of you.

I'll list (in an importance order only known to me) the people I want to thank the most that I knew and supported me during this time:

Federica Piergiacomi, Anna Magrin, i polli (Erica Stella, Claudio Sesto, Andrei Toderean, Paolo Paterna, Salvatore Perillo, Gioele Riva), Moreno Sardella, Alessandro Verga, l'Aspromonte Dream Team (Yannick Giovanakis, Claudia Chianella, Francesco Zinnari), i punk rockers (Marta Curci, Augusto Torlonia), Isis Villarinho Caricchio, Francesco Rinaldi, Andrea Chioccarelli, La Dogana (Eloisa Zendali, Matteo Maculotti, Gaia Gambini), la Révolution (Laura Kenwell, Kaveh, Scirin Khal, Gianluca Guardo, Ana Sanchez), Junyong Effie, Vladimiro Zelaya, Søren Palmund, Annalisa Rotondaro, Noemi Popolano, Simone Mosciatti, Emanuela Furfaro, Simone Amico, Luca Cavalli, Mattia Salvini, gli abusivi (Chiara Mariani, Stefano Venegoni, Astrid Ferrari), Chantelle Louis Robinson, Chris Thornton, Jhonattan Loza, Narguess, Leonardo da Silva Sousa, Pedro Pinto da Silva, Helen Munday, Joe Garcia (we always miss you).

A special thanks goes to Paolo Missier for always believing and supporting me, Carlo Piccardi for letting me get this great opportunity and tirelessly correcting and improving the thesis, Alexander (Sascha) Romanovsky for taking care of all of us during the project.

A special thanks goes to my family, even if not physically there, we are always together.

Wish me godspeed, there's more out there.

Contents

List of Figures

Chapter 1

Introduction

In this work we present a customisable software framework for incrementally discovering and ranking individual profiles for classes of online users, through the analysis of their social activity on Twitter. Practical motivation for this work comes from an ongoing effort to support health officers in tropical countries, specifically in Brazil, in their fight against airborne virus epidemics like Dengue and Zika. Help from community activists is strongly needed to supplement the scarce public resources deployed on the ground.

The approach described in this thesis generalises past efforts (Sec. 1.1), by attempting to discover users who demonstrate an inclination *to become engaged in social issues, regardless of the specific topic*. We refer to this class of users as *activists*. The rationale for this approach is that activists who manifest themselves online on a range of social initiatives, may be more sensitive to requests for help on specific issues than the average Twitter user. In the paper we experiment with healthcare-related online campaigns in the UK. Application of the approach to our initial motivating case study is ongoing as part of a long-term collaboration, and is not specifically discussed in the thesis.

To be clear, this work is not about providing a robust definition of online activism, or to demonstrate that online activism translates into actual engagement in the "real world". Instead, we start by acknowledging that the notion of activist is not as well formalised in the literature as that of, for example, *influencers*, and we develop a generic content processing pipeline which can be customised to identify a variety of classes of users. The pipeline repeatedly searches for and ranks Twitter user profiles by collecting quantitative network- and content-based user metrics. Once targeted to a specific topic, it provides a tool for exploring operational definitions of user roles, including online activism, i.e., by combining the metrics into higher level, *engineered* user features to be used for ranking.

Although the user harvesting pipeline is generally applicable to the analysis of a variety of user profiles, our focus is on the search for a satisfactory operational definition of online activism. According to the Cambridge Dictionary, an *activist* is "A person who believes strongly in political or social change and takes part in activities such as public protests to try to make this happen". While activism is well-documented, e.g. in the social movement literature [8], and online activism is a well-

known phenomenon [31], research has been limited to the study of its broad societal impact. In contrast, we are interested in the fine-grained discovery of activists at the level of the single individual, that is, we seek people who feel passionate about a cause or topic, and who take action for it. Searching for online activists is a realistic goal, as activists presence in social media is widely acknowledged, and it is also clear that social media facilitates activists communication and organisation [42, 54]. Specific traits that characterise activists include awareness of causes and social topic and the organisation of social gatherings and activities, including in emergency situations, by helping organise support efforts and diffusion of useful information.

1.1 Past efforts

Past work in fighting airborne virus epidemics focused on identifying relevant content on Twitter that may point health authorities directly to mosquito breeding sites ([33] and [51]), as well as to users who have shown interest in those topics, i.e., by posting relevant content on Twitter [34].

The approach proposed in [33], investigates the possibility of using Twitter to gain actionable content to aide health officials to counter and prevent Dengue epidemics. It compares two different methods to get information, a supervised classification algorithm (Naive Bayes) and an unsupervised clustering algorithm (Linear Discriminant Analysis or LDA), which will be discovered to be complementary in this case. Naive Bayes is part of a family of "probabilistic classifiers" based on the Bayes theorem which makes a strong independence assumptions between the dataset features (naive) and that such features have an equal effect on the outcome. LDA, on the other hand, which works by estimating the mean and variance for each class, makes the strong assumption that the dataset has a normal distribution in the feature space. Naive Bayes is trained to classify tweets into four distinct classes of topic importance related to the Dengue epidemics, and reported an overall 84% accuracy and .83 F-measure. The problem with this approach is that a great number of manually annotated samples is necessary to train the classifier ($1,000$ tweets), and that the labelling process must take place regularly in order to stay updated with new Dengue outbreaks. LDA, while it proposed similar clusters of content type as Naive Bayes, was discovered to be far less accurate and precise than the competing algorithm, mostly due to inherent noise of the Twitter media channel. The strong point of LDA though is that it is an unsupervised approach and for this reason is better suited for discovering new outbreaks. The conclusion is that these two approaches can be combined in a semi-supervised algorithm that in an unsupervised manner exploits new topics to enhance the training set and alleviate the cost for the supervised algorithm retraining.

The method proposed in [51] builds up on the preceding work of [33] by creating a web systems called "VazaDengue" that enriches user fed information with a constant active monitoring of the Twitter stream. The VazaDengue system offers to users

three main services which are, users information reporting, social media real-time monitoring and both reports and social media content visualisation in a dynamically populated app. Users can directly submit to the system reports via web forms about the location of mosquito breeding sites, the whereabouts of a sick or a ill suspected person. The social media monitoring feature, which comprises the Twitter and the Instagram social media platforms, allows for retrieving posts from the respective services stream APIs. The Twitter monitoring feature is the more developed between the two as it features an improved supervised classification algorithm based on that in [33]. The visualisation service provides to health officials an overview of the present situation by showing a map with social posts and reports plotted with marks. Only the most recent marks are displayed due to performance issues related to the high volume and velocity of data coming from the social media platforms. Marks are also visually differentiated to understand the severity of the reports (determined by the reporting user) and the social media posts (determined by the classification algorithm).

The work in [34] distances itself from the other presented efforts. While [33] and [51] rely on social media users to produce strong, geo-located, actionable signals to indicate Dengue outbreaks on the territory, [34] explores the hypothesis of identifying selected members of the public who are likely to be sensitive to virus combat initiatives, a more akin approach with the presented work in this thesis. The work in [34] is an initial investigation into techniques to identify target users who are "social sensors" [48], which spontaneously contribute information on social media channels related to a particular topic. The tested dataset is composed of a large number of expertly annotated tweets for which the content has been reduced to bag-of-words with an N-grams ($N = 1, 2, 3$) representation. A variety of classification algorithms is tested and the best performing one has been tried to be a "Random Forest" supervised classifier with an accuracy of 84.1%. Random Forest is an ensemble learning method algorithm that operates by constructing a multitude of decision trees during training, and outputs the belonging class as an average of the individual trees. The performance were promising, but the supervised nature of the method would suffer of the same shortcomings as of [33] needing for a continuous labelling effort as the situation Dengue epidemics evolve. The presented work in this thesis proposes instead an unsupervised and generalised method to address these problems (Chapter 4).

1.2 Challenges

The definition of online activism translates into technical challenges in systematically harvesting suitable candidate users. Firstly, the potentially more subdue nature of activists, relative to influencers, makes it particularly difficult to separate their online footprint from the background noise of general conversations. Also, interesting activists are by their nature associated to specific topics and manifest their nature in local contexts, for instance as organisers or participants to local events.

Finally, we expect their personal engagement to be sustained over time and across multiple such contexts. These observations suggest that the models and algorithms developed for influencers are not immediately applicable, because they mostly operate on global networks, where less prominent users have less of a chance to emerge. Some topic-sensitive metrics and models have been proposed to measure social influence, for example, *alpha centrality* [9, 39] and the *Information Diffusion* model [40]. Algorithms based on topic models have also been proposed to account for topic specificity [55]. However, these approaches are still aimed at measuring influence, not activism, and assume a one-shot discovery process, as opposed to a continuous, incremental approach.

1.3 Approach and Contributions

To address these challenges, the approach we propose involves a two-fold strategy. Firstly, we identify suitable contexts that are topic-specific and limited both in time and, optionally, also in space, i.e., regional initiatives, events, or campaigns. We then search for users only within these contexts, following the intuition that low-key users who produce weak online signal have a better chance to be discovered when the search is localised and then repeated across multiple such contexts. By continuously discovering new contexts, we hope to incrementally build up a users' dataset where users who appear in multiple contexts are progressively more strongly characterised. Secondly, to allow experimenting with varying technical definitions of *activist*, we collect a number of network-based and content-based user profile features, mostly known from the literature, and make them available to experiment with a variety of user rankings.

The project makes the following specific contributions. Firstly, we propose a data processing pipeline for harvesting Twitter content and user profiles, based on multiple limited contexts. The pipeline includes community detection and network analysis algorithms aimed at discovering users within such limited contexts.

Secondly, we have implemented a comprehensive set of content-based metrics that results into an ever-growing database of user profile features, which can then be used for mining purposes. User profiles are updated when they are repeatedly found in multiple contexts.

Lastly, for empirical evaluation of our implementation, we demonstrate an operational definition of the activist profile, defined in terms of the features available in the database. We collected about 3,500 users across 25 contexts in the domain of healthcare awareness campaigns in the UK during 2018, and demonstrated three separate ranking functions, showing that it is possible to identify individuals as opposed to well-known organisations. The application of the approach to the specific challenge of combating tropical disease epidemics in Brazil is currently in progress and is not reported in this work.

1.4 Related Work

The closest body of research to this work is concerned with techniques for the discovery of online *influencers*. According to [26], influencers are *prominent individuals with special characteristics that enable them to affect a disproportionately large number of their peers with their actions*. A large number of metrics and techniques have been proposed to make this generic definition operational [46]. These metrics tend to favour high visibility users across global networks, regardless of their actual impact [12]. In contrast, activists are typically low-key, less prominent users who only emerge from the crowd by signalling high levels of engagement with one or more specific topics, as opposed to being thought-leaders. While such behaviour can be described using well-tested metrics [46], it should also be clear that new ways to combine those metrics are required. A method for creating Twitter user ontologies considering the content type of the tweets is proposed in [45]. This approach could be used to gain insights over a user, but fails to give a comprehensive description of the user activity as it is based only on recent user activity, also due to Twitter API limitations.

The algorithm proposed in [10] aims to identify influencers based on a single topic context, based on relevant social media conversations. Metrics include number of "likes", viewers per months, post frequency and number of comments per post, as well as the ratio of positive to negative posts. As some of these metrics are qualitative and difficult to acquire, however, this approach is not easy to automate. Another approach to ranking topic-specific influencers within specific events appears in [26], where network dynamics are accounted for in real-time. Once again however, the effect is to discover users who receive much attention, but do not necessarily create a real impact over users inside one topic.

Machine learning is used in [2] to analyse posted content and recognise when a user is able to influence another inside a conversation. This however requires substantial a priori ground truth, making this approach impracticable in our case. In addition, the need to create a classifier for each topic limits the scalability of the system.

A supervised regression approach is used in [35] to rank influence of Twitter users. It uses features that are not based on content, but the method performs poorly as it requires a huge training set to work effectively.

Unlike the majority of the influencer ranking algorithms, in [50] a topic-specific influencer ranking is proposed. First it harvests sequentially timed snapshots of the network of users related to a topic. Then it ranks the users based on the number of followers gained and lost in the considered snapshots.

Finally, [4] presents a model for identifying "prominent users" regarding a specific topic event in Twitter. Those are users who focus their attention and communication on the aforementioned topic event. Users are described by a feature vector, computed in real-time, which allows a separation between on-topic and off-topic users activity over Twitter. Similar to [2], problems of scalability and adaptability arise as two supervised learning methods are used, one to discriminate prominent

users from the rest and the other to rank them.

1.5 Thesis Outline

The thesis is organised as follows:

- **Chapter 2 - *Activists*** describes the figure of the activist in the literature. It explains who is an activist and what are the common behaviours that describe it. It portrays the way activists operate over social media network by raising awareness on important topics, self-organising for public demonstrations and recruiting other activists.

- **Chapter 3 - *Contexts and user metrics*** formally defines the main concepts and metrics that are used in the project to characterise social media contexts and activists.

- **Chapter 4 - *Incremental user discovery*** details the process of the profiles harvesting over Twitter and the subsequent transformation into graphs that incorporates social media interactions and relations. Then it explains how such graphs are partitioned into communities of similar users with two community detection algorithms for which the operating principles are compared. Finally it describes how users profiles are enriched with quantitative topological graph metrics and qualitative metrics that rate the tweet contents.

- **Chapter 5 - *Software architecture*** illustrates the software that supports this research in its architecture and the single modules that compose it. It explains technical decisions, how the data is acquired, handled and stored and how the software pipeline is defined.

- **Chapter 6 - *Empirical Evaluation*** validates the described approach for finding individuals with a real world use case study on UK social media health campaigns from 2018 over Twitter. Three different ranks are defined and compared for finding such individuals.

- **Chapter 7 - *Conclusion*** draws some conclusions about our work and lays out ongoing and future research directions.

Chapter 2

Activists

This chapter illustrates the figure of activist and the online activism phenomenon. It describes what an activist is and how it operates in social media, especially Twitter, and how it differs from an influencer.

2.1 Activism and activists

In modern times, ordinary people have organised to change the society with several means [21]. In the early nineteenth century workers took part to strikes and rallies to demand higher wages and women movements set up demonstrations to change family life and gender relations. Today people gather to ask for respect to the Earth's ecosystem and support the rights of the animals. Some other conservative and right-wing movements act to stop abortion clinics and to stop immigration flows. All these movements have different goals. Some claimed new rights (Fig. 2.1), others requested political and economic emancipation. Some others fought threats and violence, and others proposed different lifestyle choices. People acted together either by building formal organisations, by participating in spontaneous demonstrations and riots or by engaging in informal networks. As defined by [21], social movements are "conscious, concerted, and sustained efforts by ordinary people to change some aspect of their society by using extra-institutional means".

The origins of any social movement are fundamental to understand how it will evolve including goals, tactics and outcomes. These aspects root deeply in the cultural background of an historical moment binding together the characteristics of its participants with the environment the social movement faces. Social networks play a crucial role in defining a social movement as they are the prerequisite to join and extend it. Those who have more ties or those who are members of an existing organisation will recruit supporters and create a movement more easily. Several factors contribute to make social movements emerge, including lessened repression, social and political divisions, economic situations, demographic problems and cultural peculiarities. Large numbers of people are positively involved with slogans and catchphrases which represent shared beliefs and needs.

After the creation of initial groups of activists, social movements actively look

Figure 2.1: Black American civil rights leader Martin Luther King (1929 - 1968) addresses crowds during the March On Washington at the Lincoln Memorial, Washington DC, where he gave his "I Have A Dream" speech. (Photo by Central Press/Getty Images).

for followers and begin to think as a collective movement. Initially, before the 1960s, social movements members were thought as irrational individuals who were marginalised and alienated or, simply, loners looking for friends. On the opposite, starting from 1965, activists where seen as so rational that they would join social movement only if they thought they could get benefits by participating, thus "freeriding" the efforts of others. In the 1980s instead, the interest shifted from the "types of people" who protest to the "structural conditions" that facilitates protests such as "biographical availability" like having little work or family obligations. People with less work obligations and family ties seem more likely to join a movement; but the best predictor is whether a person already knows an activist or not, giving importance to existing social relations.

Activists churn is also very important as it directly determines a movement survival chances, even if less discussed in the literature. People reasons to remain active in a movement can be greatly different from the reasons why they joined it. Volunteers may come to dislike their peers and reject possible new directions of the organisation.

The growing of a movement depends greatly on the interaction of its activists with other decision makers such as police, media, legislators and potential allies. Common tactics applied by the activists against opponents are focused on diminishing credibility and intimidation, while they seek support from other actors and professional groups to strengthen their own group. Social movements also benefit from conducting collective actions to improve their reach [8], like staging direct actions, producing multimedia material, creating workshops and expand online presence.

2.2 Activists on social media

Activists presence in social media is widely acknowledged [42] as the latter provide powerful tools for communication and organisation. Some examples are the protests at the G-20 Summits in London (2009) and Toronto (2010), the Occupy Wall Street movement (2011) or more recently the protests for Hong Kong concerning the regional autonomy (2019).

Even if social media are greatly used by authorities for surveillance purposes, they represent an invaluable tool for activists as they facilitate the diffusion of news and opinions and the exposure of wrongdoing. An example on how social media are used by both sides comes from the Hong Kong protest of 2019 (Fig. 2.2). Both, protesters and pro-Beijing supporters, purposefully shared on the social media personally identifiable information of the opposing party in an effort of doxing and cyberbullying. Both sides also spread fake news, which heightens polarisation and reactions among the population. This includes easy manipulation tactics like selectively cutting footage news and creating false narratives. Such manipulations will be later confirmed by major social networks such as Facebook [38] and Twitter [6].

Figure 2.2: A pro-democracy activist holds a yellow umbrella in front of a police line on a street in 2014 in Hong Kong. (Photo by Chris McGrath/Getty Images)

Social media however are not neutral tools, as they can be conceptualised as "assemblages" of software processes, communication practices, political-economic relations and culture. This results in striking differences in the way the social activity shapes in social networks, for example by promoting different types of content, information, users and user relations.

Youtube plays a minor role in organising and connecting activists, while it is significant in the informative hosted content which is linked to other social networks. The platform revenue model tends to favour the content and the personalisation of the videos as opposing to their novelty [42].

Facebook's structure favours users to reconstruct offline relations on the platform as "any kind of activity takes place through a highly individualized and personalized perspective"[30]. It also encourages the creation of user groups related to specific topics where users can create and share posts with rich content like videos, photos and texts. For this reason, it is crucial in the planning phase of activism related activities, while it is less important for publishing facts and news in real-time [42].

Twitter, is a micro-blogging platform that let participants share short posts of 280 characters (as in 2017, originally the limit was only 140 characters). Twitter has developed into an important real-time news platform. To corroborate this fact, the number of published links in tweets has sensibly increased during the years (from 13% in 2007 to 25% in 2009 [24]) for information sharing purposes. Several functions of the platform encourage users in sharing fresh news and contents. Hashtags help in categorising tweets prompting users to share and search news related to a specific topic. This last feature is also enhanced by the trending topic section which highlights hashtags having a spike in volume. The non-reciprocal nature of the "follow" relations on the platform supports the existence of users who have a greater reach than others, thus creating hubs of information, in contrast with Facebook where the "friendship" relation is mutual. The information flow is an "information cascade" [31] supported by retweets which let re-share other user's tweets. For these reasons Twitter operates more as an information-sharing network than as a social network [29].

An important Twitter use case in this sense was played during the Arab Springs in Tunisia and Egypt (2011) [31] (Fig. 2.3). Both activists, planning their actions, and news sources, produced an important flow of information at the time which kept updated both locals and people from all around the world on the latest developments of the Middle-East events. In [31] it has been found that the vast majority of involved Twitter users were individuals (around 70%) opposing to organisation and mainstream media accounts. Activists, qualitatively defined as "individuals who self-identify as an activist, who work at an activist organization, or who appear to be tweeting purely about activist topics to capture the attention of others"[31] have been described as primary information sources concluding that "news on Twitter are being co-constructed by bloggers and activists alongside journalists".

Figure 2.3: Protesters jubilate in Cairo's Tahrir Square after Hosni Mubarak announced his resignation in February 2011. (Source: Al-Jazeera)

2.3 Influencers on social media

Differently from activists, the "influencers" are individuals who are "well-connected, create an impact, have active minds, and are trendsetters" [27]. One of the main acceptations is related to the marketing world, as individuals who shape the customer's purchase behaviours and eventually also the organisation behind a given product [11] [41]. Influencers in social networks are usually categorised with respect to the number of the followers [25] [13]. This is because it determines the reach (i.e., the popularity), even though it may not directly relate with the capacity of influencing people's behaviours [12] (i.e., the ability of swaying the opinion of its followers).

Online activity is fundamental for offline decision-making as it provides an important source of information and knowledge for any given topic and product for people [32]. Many influencers exploit this to change people's opinions through the use of social media. Social media allow influencers to increase the reach of their messages and establish them as opinion leaders [27]. The methods used by influencers go beyond the simple advertisement in terms of changing people's behaviours, as they seek to establish para-social relationships with the followers [14]. Influencers are persuasive in that they leverage likeability, attractiveness, but also knowledge of the field they operate into and niche expertise [52].

While some online profiles benefit from anonymity, like activists and journalists to avoid repression in dictatorships, influencers do not [11]. Influencers have to curate a transparent online presence in order to be credible, as anonymity is often perceived as something that is hidden and that undermines the trustworthiness of a source.

Influencers and activists are different online profiles with their own purpose and goals. While influencers are well-connected individuals whose goal is to maintain and orientate a following for their own benefit, activists are individuals that use social media to communicate, organise and document the world they live in and to change it.

Chapter 3

Contexts and user metrics

The aim of the pipeline is to repeatedly and efficiently discover user profiles from the Twitter post history within user-specified contexts and to use the process to grow a database of feature-rich user profiles that can be ranked according to user-defined relevance functions. The criteria used to define contexts, profile relevance functions, and associated user relevance thresholds can be configured for specific applications.

3.1 Contexts and Context Networks

A context C is a Twitter query defined by a set K of hashtags and/or keyword terms, a time interval $[t_1, t_2]$, and a geographical constraint s, which limits the tweets published on Twitter:

$$C = (K, [t_1, t_2], s) \tag{3.1}$$

Let $P(C)$ denote the query result, i.e., a limited set of posts p (Sec. 5.2.2) tweeted within the spatio-temporal constraints $([t_1, t_2], s)$, and containing at least one of the terms of K. The considered Twitter user activities are: *original tweets* and *retweets*, together with the contained user *mentions* from both. Let $u(p)$ be the user who originated a tweet $p \in P(C)$. We say that both p and $u(p)$ are *within context* C. We also define the complement $\tilde{P}(C)$ of $P(C)$ as the set of posts found using the same spatio-temporal constraints, but which do not contain any of the terms in K. More precisely, given a context $C' = (\emptyset, [t_1, t_2], s)$ with no terms constraints, we define $\tilde{P}(C) = P(C') \setminus P(C)$. We refer to these posts, and their respective users, as "out of context C".

$P(C)$ induces a user-user social network graph $G_C = (V, E)$ where V is the set of all users who have authored any $p \in P(C)$: $V = \{u(p) | p \in P(C)\}$, and a weighted directed edge $e = \langle u_1, u_2, w \rangle$ is added to E for each pair of posts p_1, p_2 such that $u(p_1) = u_1, u(p_2) = u_2$ and either (i) p_2 is a retweet of p_1, or (ii) a tweet p_1 that contains a mention of u_2. For any such edge, w is a count of such pairs of posts occurring in $P(C)$ for the same pair of users.

3.1.1 User Relevance Metrics

We support metrics that are generally accepted by the community as forming a foundation, from which many different social user roles are derived [46]. We distinguish amongst three types of features, which differ in the way they are computed from the raw Twitter feed:

Content-based metrics that rely solely on content and *not* on the user-user network. These metrics are defined relative to a topic of interest, i.e., a context;

Context-independent topological metrics that encode context-independent, long-lived relationships amongst users, i.e., follower/followee; and

Context-specific topological metrics that encode user relationships that occur specifically within a context.

All metrics are functions of a few core features that can be directly extracted from Twitter posts. Given a context C containing user u, we define:

$R1(u)$: Number of retweets by u, of tweets from other users in C;
$R2(u)$: Number of unique users in C, who have been retweeted by u;
$R3(u)$: Number of retweets of u's tweets;
$R4(u)$: Number of unique users in C who retweeted u's tweets;
$P1(u)$: Number of original posts by u within C;
$P2(u)$: Number of web links found in original posts by u within C;
$F1(u)$: Number of followers of u;
$F2(u)$: Number of followees of u

Note that, given C, we can evaluate some of the features above with respect to either $P(C)$ or $\tilde{P}(C)$ independently from each other, that is, we can consider an "on-context" and an "off-context" version of each feature, with the exception of *F1* and *F2* which are context-independent. For example, we are going to write $R1_{on}(u)$ to denote the number of context retweets and $R1_{off}(u)$ the number of out-of-context retweets by u, i.e., these are retweets that occur within C's spatio-temporal boundaries, but do not contain any of the hashtags or keywords that define C. We similarly qualify all other features. Using these core features, the framework currently supports the following metrics.

Content-based metrics:

$$\textit{Topical Focus [34]: } TF(u) = \frac{P1_{on}(u)}{P1_{off}(u) + 1} \tag{3.2}$$

$$\textit{Topical Strength [3]: } TS(u) = \frac{P2_{on}(u) \cdot \log(P2_{on}(u) + R3_{on}(u) + 1)}{P2_{off}(u) \cdot \log(P2_{off}(u) + R3_{off}(u) + 1) + 1} \tag{3.3}$$

$$\textit{Topical Attachment [5, 42]: } TA(u) = \frac{P1_{on}(u) + P2_{on}(u)}{P1_{off}(u) + P2_{off}(u) + 1} \tag{3.4}$$

All the three functions 3.2, 3.3 and 3.4 are measures of the interest of a user towards a context by exploiting its tweets content. These functions count, given a user u published tweets within the context's spatio-temporal boundaries, the number of on-topic social interactions over the off-topic ones.

Function 3.2 only considers the original posts published as a context's interest measures, while function 3.3 also takes into account the number of retweets and, together with function 3.3, the published external links.

The framework supports one **Context-independent topological metric** and one **Context-specific topological metric**, both commonly used, see e.g. [46]:

$$\textit{Follower Rank:} \quad FR(u) = \frac{F1(u)}{F1(u) + F2(u)} \tag{3.5}$$

$$\textit{In-degree centrality:} \quad IC(u) = \frac{unweighted\_indegree(u)}{N - 1} \tag{3.6}$$

where N is the number of nodes in the network induced by C. Note that the metrics we have selected are a superset of those indicated in recent studies on online activism, namely [31] and [42], and thus support our empirical evaluation, described in Chap. 6.

Chapter 4

Incremental user discovery

The content processing pipeline operates iteratively on a set of contexts within a given area of interest, for instance *2018 UK health campaigns*. This set is initialised at the start of the process and then updated at the end of each iteration, in a semi-automated way. The user discovery process is therefore potentially open-ended, as long as new contexts can be defined. The new contexts are expected to be within the same topic area, but contexts that "drift" to new areas of interest are also acceptable. Each iteration takes a context C as input, and selects a subset of the users who participate in C, using the topological criteria described below, along with the set of their features and metrics. These users profiles are added to a database, where entries for repeated users are updated according to a user-defined function. The pipeline structure is described below, where the numbers are with reference to Fig. 4.1.

Given C as in (2), all Twitter posts $P(C)$ that satisfy C are retrieved, using the Twitter Search APIs. Note that this step potentially hits the API service limitations imposed by Twitter. For this reason, in our evaluation we have limited our retrieval to 200 tweets/context. This proved to be sufficient, considering that repeated users appear consistently in our evaluation (Chap. 6). Twitter API limitations can be overcome by either extending the harvesting time, or by choosing more recent contexts, as the Twitter API is more tolerant with recent tweets.

The context network G_C is then generated (3), as defined in Sec. 3.1. The size of each network is largely determined by the nature of the context, and ranges between 140 and 400 users (avg 254, see Table 6.3 in chapter 6).

4.0.1 Community detection

Next, G_C is partitioned into communities of users (4). The goal of this partitioning is to further narrow the scope when computing in-degree centrality (3.6), to enable weak-signal users to emerge relative to other more globally dominant users.

A community, in a network, can be thought as a set of individual nodes that are very similar, or close, to each other, more than to anybody else outside the community [15] [20]. In networks it is intended as a set of nodes densely connected to each other and less connected with the rest of the network.

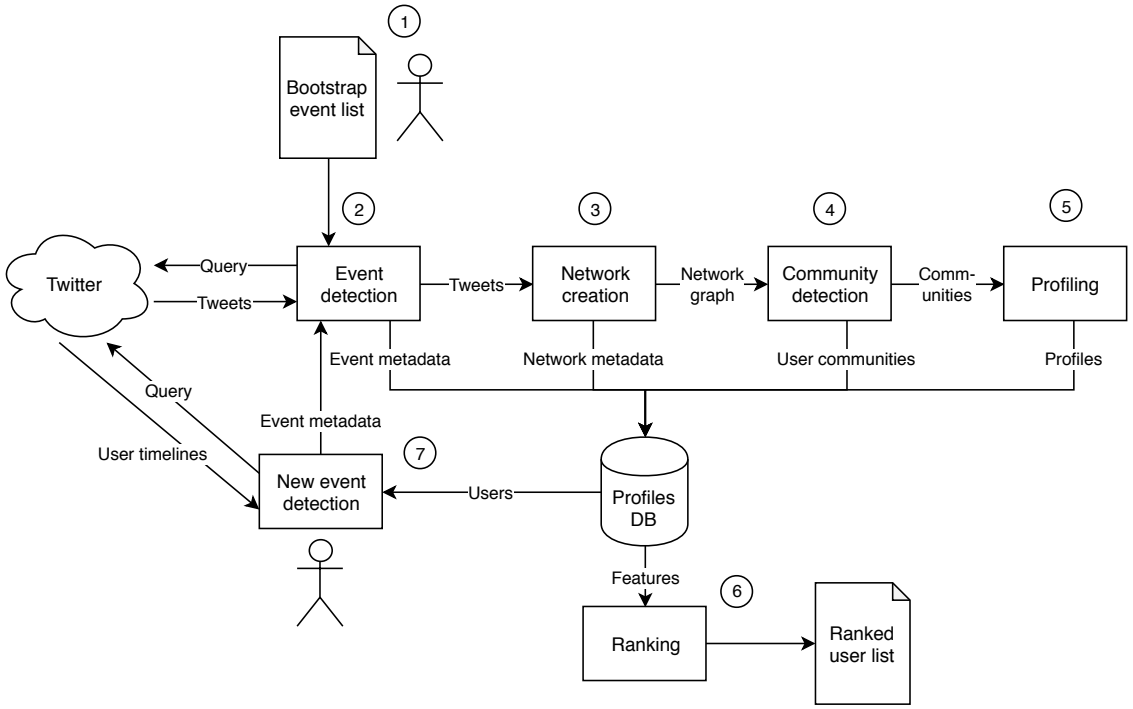


Figure 4.1: Schematic diagram of the user discovery framework. Note that an initial list C of contexts (events) is provided to initialise the event detection step (1). The outputs from each of these steps are stored into the Profiles DB.

We have experimented with two of the many algorithms for discovering virtual communities in social networks, namely DEMON [16] and Infomap [47]. Both are available in our implementation, but based on our experimental comparison (Chapter 6) we recommend the latter. For almost half of our context networks, DEMON actually fails to discover any community. In contrast, Infomap generates valid communities in all cases. As some of them are very small, our implementation discards communities with less than 4 users (see Chapter 6). Once communities are identified, using either method, we calculate in-degree centrality (3.6) for each node, either relative to their own community if available, or to the entire network otherwise.

DEMON

DEMON [16] is based on *ego networks* [1], and uses a *label propagation* algorithm [44] to assign nodes to communities.

Ego networks are networks that forms around a particular node, a social actor, that can either be a human, a corporate or a national government [17]. It has been suggested [1] that ego networks are a useful model not only to describe social relationships amongst people offline, but also the structure of their online connections.

Recognising that any individual may have different types of social relationships with different people (i.e., family, friends, colleagues, etc.), DEMON allows an individual to participate in multiple communities. This is an attractive feature when users are active in more than one community within the same context, i.e., a social

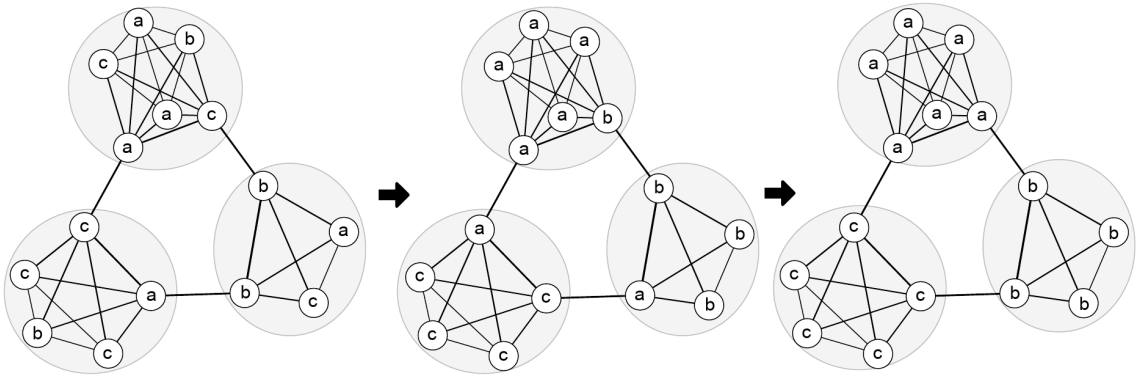


Figure 4.2: Example of how Label Propagation works for community detection (from [16]).

event or a campaign.

Specifically, DEMON operates on one node v at a time in our context network G_C. It applies a label propagation algorithm (Fig. 4.2) to each neighbour v' of v, as follows. First, a new label l, which identifies a new community, is tentatively assigned to v'. Then, with probability α, v' changes its label to that of the majority of its own neighbours. At this point, each of v's neighbours has a label, which is either new or that of the majority of its own neighbours (except v itself). v is then assigned the majority labels amongst those of its neighbours. This determines v's community. When more than one label has the same count, v is assigned to all of those communities.

As a final step, communities that overlap by more than some percentage ϵ are merged.

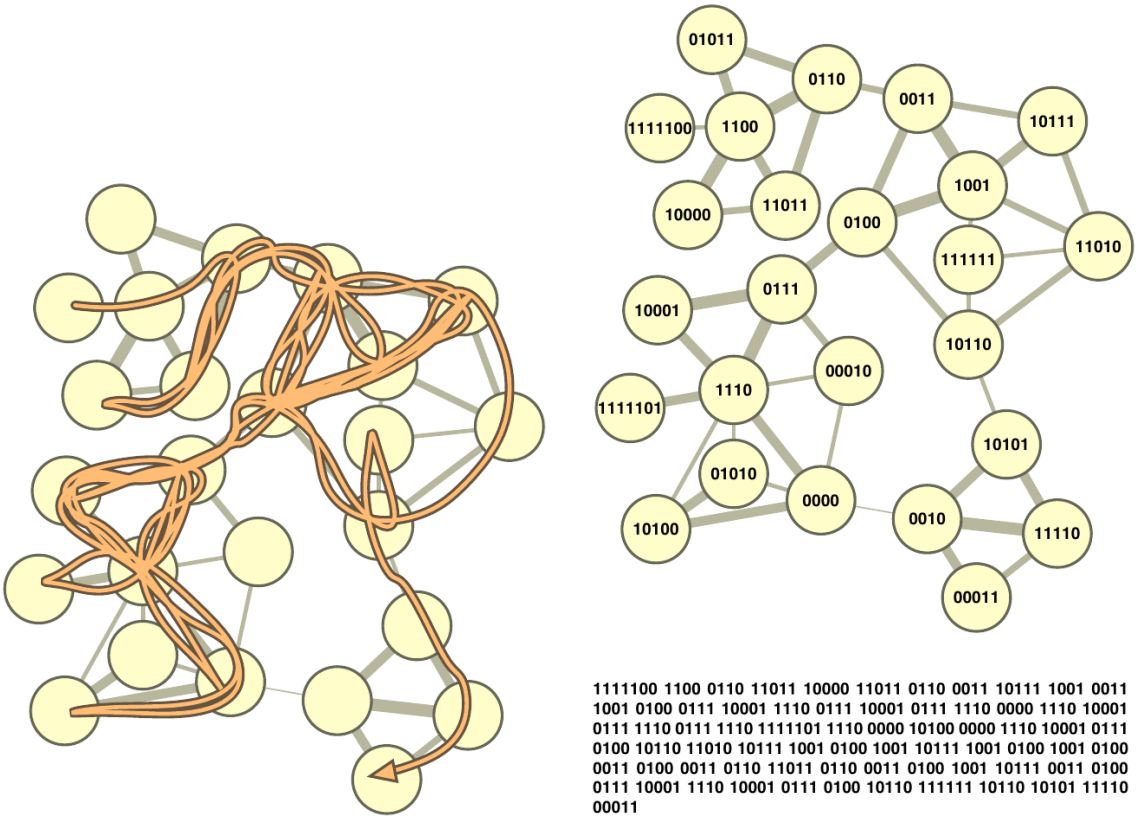
Infomap

Infomap [47] is a renowned community detection algorithm which works by exploiting the information-theoretic duality, resulting in two data representations, between the problem of compressing data flow and the problem of detecting and extracting significant patterns within those data flows.

The first representation (Fig. 4.3) consists in describing a network data flow, represented by a Random Walker (Fig. 4.3a), by orderly listing the visited nodes. Nodes are uniquely identified with a binary Huffman encoding [23]. A simple node naming method would be to randomly assign a unique identifier to each node. But an optimised and compressed way would instead be to encode the most frequent patterns with a lower number of bits (Fig. 4.3b). The most frequent pattern is determined on how many times a node is crossed by the Random Walker path. So a node that will be crossed many times will have a low bit representation whereas nodes that will be crossed a lower number of times will have an higher bit representation.

This efficient representation also gives us, in an implicit way, an hint on where the information mostly flows.

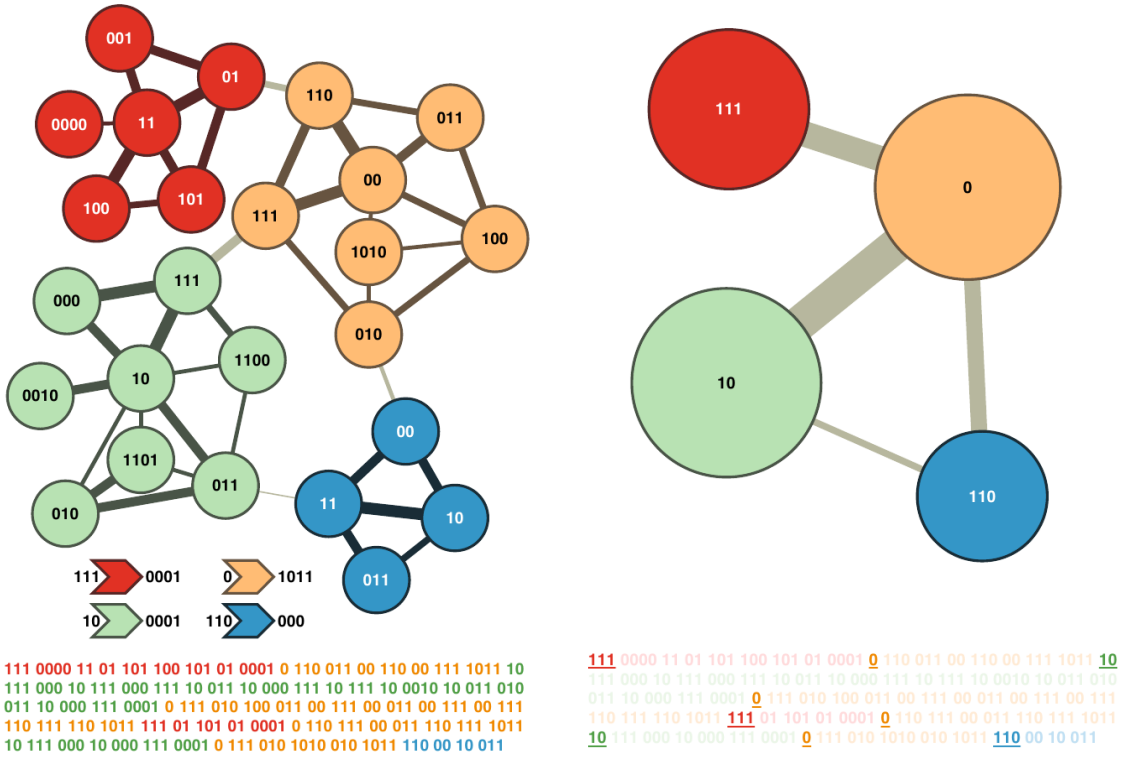
In the second representation (Fig. 4.4), to get an even smaller description representation of the data flow, we optimally partition the network. This is obtained



(a) The trajectory of Random Walker is represented by an orange arrow that travels in the network by visiting nodes.

(b) A unique and efficient Huffman encoding of the nodes and the resulting 314 bit trajectory. The trajectory starts from node 11111100 (upper left) and ends on 00011 (lower right).

Figure 4.3: Infomap first representation of a sample network, where information flow of a Random Walker is encoded in an ordered list of visited node (from [47]).



(a) Enter and exit codes are optimally placed defining network zones. Nodes are renamed locally accordingly in a more efficient way.

(b) Set of nodes identified by enter and exit codes are considered as part of the same community and thus assigned a new unique name.

Figure 4.4: Infomap second representation of a sample network, where a network is partitioned into communities to better compress the information flow of a Random. Walker (from [47])

by creating enter and exit code prefixes for each zone (community) (Fig. 4.4a), reducing the number of bits required to uniquely identifying a single node.

Community definition at this point is trivial, because a community is defined by the joint set of all the intervals. Node sets joins are determined by the initial node in the interval. Each community is then assigned a unique name, always following the Huffman encoding (Fig. 4.4b).

We get a two-level description of the random walk, in which major communities receive unique names, but the names of nodes within communities are reused resulting in a much smaller representation. Duality reflects in the problem of finding the most efficient encoding that translates into finding the best partition for the network. In this way, by optimally solving the compression problem we found a partition of communities which describes the information flow inside the graph. Optimal partitioning is found iteratively using a modified Louvain method [7] that instead of optimising the modularity [36] optimises the Map Equation (the compression problem) (4.1).

$$L(M) = \underbrace{q_{\curvearrowright} H(Q)}_{\substack{\text{weighted entropy of movements} \\ \text{BETWEEN the communities}}} + \underbrace{\sum_{i=1}^{m} p^i_{\circlearrowright} H(P^i)}_{\substack{\text{weighted entropy of movements} \\ \text{WITHIN the communities}}} \tag{4.1}$$

Where:

- M: Partition of n nodes into m communities.
- $q_{\curvearrowright}$: probability that a random walk switches community on any given step.
- $H(Q)$: entropy of between-community movements.
- $H(P^i)$: entropy of within-community movements.
- $p^i_{\circlearrowright}$: fraction of within-community movements that occur in community i plus the probability of exiting from such community.

4.1 Computing User Features and Ranking

Next, user metrics along with the *Follower Rank* (defined in Sec. 3.1.1) are computed from the network and the user features. This is achieved through bulk retrieval of user profile information (block (5) in Fig. 4.1), namely the number of tweets, retweets, number of followers $F1(u)$ and followees $F2(u)$, along with user name, web link, and bio. Computing the other metrics: *Topical Focus* (3.2), *Topical Strength* (3.3), *Topical Attachment* (3.4) also requires the entire user post history to be retrieved for the entire time interval defined by the context. These posts are then separated into $P(C)$ (on-context) and $\tilde{P}(C)$ (off-context), depending on whether they contain a hashtag related to the context or not. Similarly, a post that contains a link is a *link on-topic* if it contains both a link and a hashtag related to the context, and a *link off-topic* otherwise. We also calculate the number of retweets for every post, i.e., $R1(u)$ and $R3(u)$, which are required to compute *Topical Strength*.

All of these features are stored in a database which is made available for ranking purposes. User-defined functions can be specified to update the Rank of pre-existing users, e.g. by combining scores assigned at different times. The DB enables user-defined scoring functions, which result in user ranking lists (block (6) in Fig. 4.1). Examples will be given later in Chapter 6. This framework approach is consistent with the experimental nature of our search for *activists*, which requires exploring a variety of ranking functions.

4.2 New Contexts Discovery

The final step of each iteration (block (7) in Fig. 4.1) aims to discover new contexts, so that the process can start again (block (2) in Fig. 4.1). Intuitively, once a score function has been applied and users have been ranked, we can hope to discover

new interesting keywords and hashtags by exploring the timeline of the top-k users. Specifically, we consider each hashtag found in their timelines, which is related to the broader topic and not yet considered in past iterations. Each stored hashtag is then enriched with the information needed to perform a new iteration of the pipeline, namely (i) the temporal and spatial information of the context, and (ii) related hashtags. Currently this step is only semi-automated, as making a judgement on the relevance of the new terms requires human expertise. While automating this step is not straightforward, this is not a very time-consuming step, and one can imagine an approach where such task is crowd-sourced.

While the process ends naturally when no new contexts are uncovered from the previous ones, the system continuously monitors the Twitter stream for recent contexts. These may typically include events that are temporally recurring, and use similar hashtags for each new edition. In this case, their relevance is assessed on the basis of their past history.

Chapter 5

Software Architecture

This chapter illustrates the architecture of the software pipeline for the user harvesting. It details language and architectural decisions, data sources from where information is pulled from, database structures that store the data, third-party libraries and the tasks that compose the pipeline.

5.1 Architectural choices

The software pipeline is implemented in Python, a multi-paradigm programming language that has become one of the most used in the domain of data science. This is due to its inherent language features such as functional style programming as well as the comprehensive software ecosystem for numeric and data manipulation such as SciPy[1].

The project presents several difficulties in the development phase which need to be addressed in order to not slow down the research process. Being a research project, thus suffering from a "trial and error" approach, the software gets constantly rewritten. For this reason it needs a modular and maintenable structure to ease the addition and removal of features without impacting the reliability and the performances. This requires extra care while designing interactions between software components and prompt code refactors if needed.

Some data harvesting tasks may take up to several hours to complete. The software must then be reliable and treat errors in such a way that the data processing continues as planned. Intermediate cached results speed up the processing.

A centralised and methodical logging component becomes crucial in identifying and addressing errors during the pipeline execution. Main errors come from exhaustion of computing resources such as the RAM, which is solved by retaining in memory only the data needed for the current task, and disk caching and errors from web resources, such as timeout errors when performing web scraping and API calls solved with error catching and retries.

Limitations in the number and the rate of API calls (even for an entire month) that external services offer require a parsimonious usage. Thus software must save

[1] `https://scipy.org`

API calls results in their entireness and then parse useful information out of it as deemed necessary in successive research iterations.

5.2 Source code

Our software is composed of two main parts: Pipelines and Datasources. While the module Pipelines is responsible for data processing by defining and managing the pipeline tasks, Datasources instead provides a centralised access to external APIs, database persistence and data files. An additional component, the Orchestrator, works as a bridge between Pipelines and Datasources.

5.2.1 Pipelines

Pipelines is loosely inspired by workflow management frameworks like Apache Airflow[2]. In Airflow dependencies among pipeline tasks are described by a DAG (Directed Acyclic Graph), which determines the execution order. Logging is centrally managed by Airflow, which keeps track of task executions and failures. Failing tasks are handled automatically by specifying the maximum number of retries and a back-off time, which determines when to make another execution attempt.

A custom and leaner solution in this case is preferred over Airflow and similar frameworks because only a restricted amount of features is actually needed. This custom framework defines pipeline task dependencies as an ordered sequential Python list of lists (Figure 5.1). While tasks in the outer list will be executed sequentially in the provided order, the ones in the inner list will be executed in parallel threads to optimise possibly intensive I/O execution. Logging is centrally controlled and failures are handled by retrying a task for a maximum specified number of times with a linear incremental back-off. Additionally, if a task output is already present, after for example stopping and resuming a pipeline execution on the same inputs, the task is automatically skipped to provide a faster execution.

Pipelines tasks are logically distributed among several "Pipeline" Python classes. Each pipeline extends a Python base class which contains the logic to initialise and execute it. Pipeline initialisation envisage output file description for reading/writing purposes (if provided) and the execution order of the contained tasks. The only public method for pipeline classes is "execute()" (contained in the pipeline base class) which executes and logs tasks in the specified order. All the tasks are instead defined as private methods as their execution flow is controlled by the pipeline class itself.

Defined pipelines for the project are listed below in execution order:

1. *ContextDetection*: harvests tweets from Tw datasource (see Sec. 5.2.2) according to the given context information.

2. *NetworkCreation*: creates a graph out of the harvested tweets in (1) together with the node list and the weighted, directed adjacency list.

[2] `https://airflow.apache.org`

```
tasks = [task1, task2, [task3, task4, task5], task6]
```

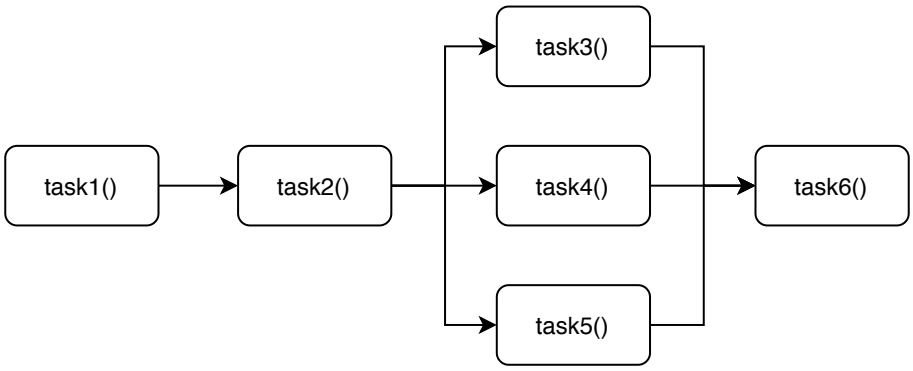


Figure 5.1: Example of the definition (above) and the resulting execution flow (bottom) of pipeline tasks.

3. *NetworkMetrics*: computes several graph metrics such as the number of nodes and edges, the average degree and the cumulative degree distribution.

4. *CommunityDetection*: applies the specified community detection algorithm on the graph created in (2). It supports both overlapping and non-overlapping community detection algorithms.

5. *CommunityDetectionMetrics*: computes the graph metrics specified in (2) on all the graph communities found in (4). Computes graph partition metrics to assess the quality of the found communities such as internal density and normalised cut (Appendix B). Finally computes node related metrics such as indegree centrality and h-index (Appendix A).

6. *ProfileMetrics*: harvests profile information from Tw datasource for each graph node (Twitter user) such as number of published tweets and bio. It computes also the FollowerRank (Chapter 6).

7. *UserContextMetrics*: harvests users timelines from Tw datasource according to context information and computes several metrics such as the Topical Attachment, Topical Focus and Topical Strength Chapter 6).

8. *Persistence*: store on the database some of the outputs of the preceding steps. It also updates user information where necessary, for example the updated Twitter bio of a user.

9. *Ranking*: Filters and ranks all the users in the database with respect to the defined ranking functions.

Pipelines 1-8 are executed once per context, while pipeline 9 is executed only once at the end, since it needs to rank users with respect to all the processed contexts.

name	start_date	end_date	location	hashtags
16-days-of-action-2018	2018-11-25	2018-12-10	United Kingdom	#16days #16daysofaction
elf-day	2018-12-03	2018-12-12	United Kingdom	#elfday #elfday2018
dry-january-2018	2018-01-01	2018-01-31	United Kingdom	#dryjanuary

Table 5.1: Example of the context format accepted by the pipeline.

5.2.2 Datasources

Datasources is a Python class that initialises multiple data sources and provides an interface to Pipelines to read and write data. Data sources from which Pipelines can exchange data are:

- *Contexts*: loads input contexts to process.
- *CommunityDetection*: loads the configuration for the community detection algorithm.
- *Database*: manages the persistence and model definitions of the entities in the database.
- *Tw*: interface to retrieve data from Twitter.
- *Files*: manages output files reads and writes for caching and storing purposes.

The following paragraphs describe each datasource and its purpose inside the pipelines.

Contexts

Contexts datasource parses and loads the list of contexts to harvest from a ".csv" file (Table 5.1) so that Pipelines can start the pipeline.

CommunityDetection

CommunityDetection datasource parses and loads from a ".json" file (Listing 5.1) the configuration for the community detection algorithm. Supported community detection algorithms are Infomap [47] and Demon [16].

```
{
  "name": "demon",
  "kwargs": {
    "epsilon": 0.25,
    "min_community_size": 3
  }
}
```

Listing 5.1: Example of the configuration format accepted by the pipeline for Demon [16] community detection algorithm.

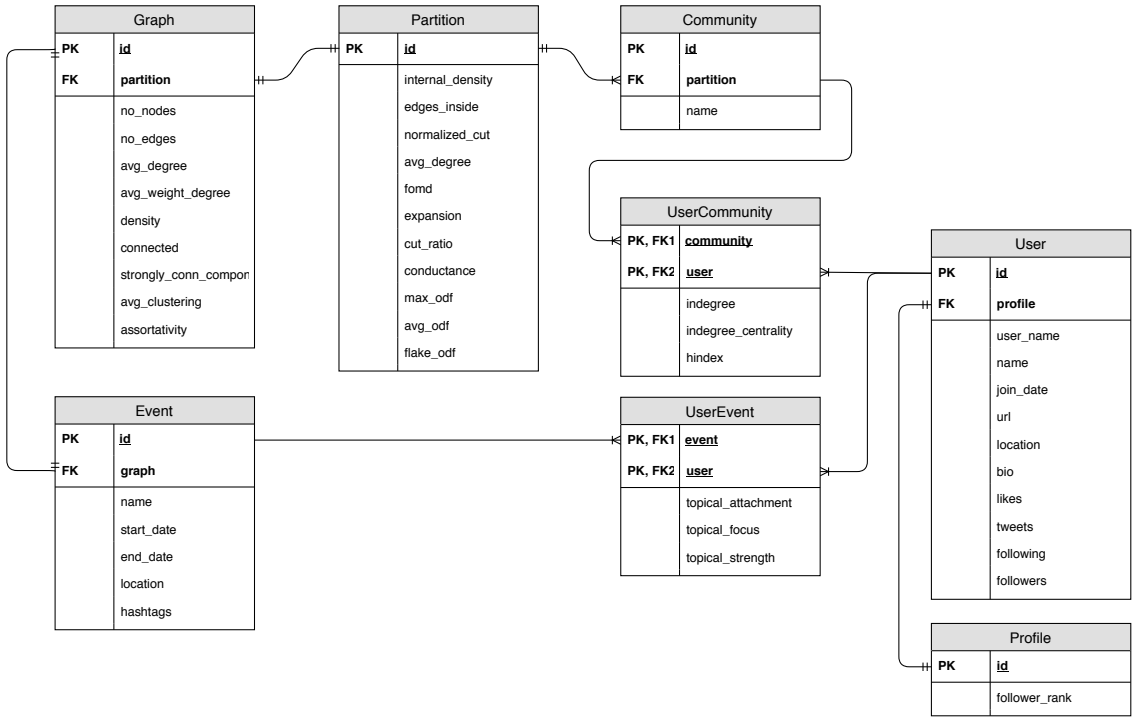


Figure 5.2: Entity Relationship diagram that describes how the database is structured.

Database

Database datasource offers relational data persistence to Pipelines (Figure 5.2). SQLAlchemy[3] is the framework chosen, through the ORM interface[4], for managing the database. It has several advantages, as it supports multiple database backends, it is natively supported by Pandas[5], it supports composite primary keys and it is reliable, being a longstanding popular Python database framework. Pandas support is particularly appreciated as the majority of the data handled by the pipeline is in the form of Pandas Dataframes which let manipulate structured data in a reliable, fast and convenient way. The project uses an embedded SQLite database[6] as SQLAlchemy backend. The choice originates from the need of an easy to manage database for prototyping. Thanks to the SQLAlchemy flexibility, it is possible to switch to a standalone database if necessary (to share it with other applications for example) with minimal code changes. Database entities are fully normalised with the exception of Profile entity. Conceptually User entity stores Twitter user profile information, while Profile entity stores metrics that are calculated over the User entity.

[3] `https://sqlalchemy.org`
[4] `https://docs.sqlalchemy.org/en/latest/orm`
[5] `https://pandas.pydata.org`
[6] `https://sqlite.org`

Tw

Tw datasource offers access to Twitter data either via Twitter Standard API[7], Twitter Premium API[8] and previously also web scraping.

Access to Twitter API is managed through TwitterAPI[9] which is a lightweight library that manages authentication, API calls and "cursoring"[10]. Cursoring is a Twitter API technique to paginate large result sets which do not fit in a single API call considering the continuous addition of new information.

The Standard API is used to retrieve user profile information[11] and user tweets timeline at the time the given context has taken place[12]. The Premium API is used to retrieve tweets in a custom filtered manner up to year 2006 with the full-archive search[13].

To get user tweets timeline, Twitter web scraping has also been considered to speed up the retrieval and avoid the Twitter API limits. Web scraping is provided by Selenium with Python[14], a Python interface for Selenium[15]. Selenium is a library for automating the navigation of web pages through a programming interface and an actual web browser. In this project Selenium is coupled with Chromium[16] and its webdriver[17], a software component which lets Selenium interact with the Chromium web browser. Twitter web scraping supports traditional advanced search queries[18], and is used to obtain user timelines at the time the context has taken place. Performances of Chromium under Selenium are improved by: setting headless mode, enabling cache and disabling media download. Further performance improvements could be achieved if Chromium, set as headless, would either support web extensions or explicit domain blocking in order to block the download and execution of scripts related to ads and tracking by Twitter.

This method has been ultimately discarded as computationally intensive, provides less data with respect to the API and is not "future proof", as a change to the Twitter website user web interface would require a rewriting of the scraper.

Files

Files datasource offers reading and writing data files for Pipelines. It is intended for writing and reading intermediate results that are useful for successive task operations

[7] https://developer.twitter.com

[8] https://developer.twitter.com/en/premium-apis

[9] https://geduldig.github.io/TwitterAPI

[10] https://developer.twitter.com/en/docs/basics/cursoring

[11] https://developer.twitter.com/en/docs/accounts-and-users/follow-search-get-users/api-reference/get-users-lookup

[12] https://developer.twitter.com/en/docs/tweets/timelines/api-reference/get-statuses-user_timeline

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

[14] https://selenium-python.readthedocs.io

[15] https://www.seleniumhq.org

[16] https://www.chromium.org

[17] https://sites.google.com/a/chromium.org/chromedriver

[18] https://twitter.com/search-advanced

as a cache and for data analysis purposes. It provides an abstraction for Pipelines for where files actually resides on the disk. An in-memory LRU cache[19] keeps a small fixed number of files to improve access speed. Files writing and reading process is loosely inspired by a database ORM lifecycle:

1. *Registering a file (Model definition)*: a Pipelines pipeline requests Files to register a file by sending: file name, file type, pipeline name, task name and reading/writing options. It also optionally supports file prefixes and suffixes. Files then adds the file to its register and composes the path to the file to be.

2. *Writing a file (Entity persistence)*: a Pipelines pipeline requests Files to write a file by sending: file name, file type, pipeline name, task name and the file content. Files checks whether the file exists in its registry and if present writes it according to the options specified during the registration in (1).

3. *Reading a file (Entity retrieval)*: a Pipelines pipeline requests Files to read a file by sending: file name, file type, pipeline name, and task name. Files checks whether the file exists in its registry and if present reads it according to the options specified during the registration in (1).

Files are read and written by dedicated modules depending on the file type: Pandas for ".csv", Networkx[20] for ".gexf" and Json[21] for ".json".

```
file_model = {
    'pipeline_name': 'network_creation',
    'stage_name': 'create_edges',
    'file_name': 'edges',
    'file_extension': 'csv',
    'r_kwargs': {
        'dtype': {
            'source_id': 'uint32',
            'target_id': 'uint32',
            'weight': 'uint16'
        }
    },
    'w_kwargs': {
        'index': False
    }
}

datasources.files.add_file_model(**file_model)

edges = pandas.DataFrame(
```

[19] https://cachetools.readthedocs.io
[20] https://networkx.github.io
[21] https://docs.python.org/3/library/json

```
    columns=['source_id', 'target_id', 'weight'])

datasources.files.write(edges, 'network_creation',
    'create_edges', 'edges', 'csv')

datasources.files.read('network_creation',
    'create_edges', 'edges', 'csv')
```

Listing 5.2: Example of the registration, writing and reading of a file in Files. Note that by specifying the column type for Pandas Dataframes greatly reduces the RAM use and speeds up the computation.

5.2.3 Orchestrator

Orchestrator is responsible to initialise Datasources and to manage the execution of the pipelines in Pipelines in the defined order (Section 5.2.1). Orchestrator execution is timed in order to provide actionable feedback during development on possible performance regressions.

Orchestrator is initialised by specifying:

- *Project name*: the name of the project to work on. To work on several separate group of separate contexts in an easy way, Orchestrator supports "projects". Each project is composed by a set of inputs, which kicks off the pipeline.
- *Input path*: defines the base URI (Uniform Resource Identifier) for the inputs.
- *Output path*: defines the base URI for the outputs.

5.2.4 Data visualisation

After a pipeline cycle is concluded, data visualisation can be performed. Data visualisation is an important step in data analysis because it puts data in visual context in order to understand its meaning. In this way patterns, trends and correlations can be exposed and recognised by humans and exploited to make conclusions and to improve, in this project, data processing operations.

Data visualization is performed with Jupyter Notebooks[22] in conjunction with Pandas, Seaborn[23] and Matplotlib[24]. Jupyter Notebooks is a web application that produces documents that contain live code, equations, visualisations and narrative text. Pandas provides a table representation of data, while Seaborn and Matplotlib represent data through 2D plotting of informative statistical graphics.

In this paragraph we show the main data visualisation analysis that have been performed in a summarised way. While in this section we describe the analysis that have been performed, it is only in the Evaluation section (Chapter 6) that their

[22] `https://jupyter.org`
[23] `https://seaborn.pydata.org`
[24] `https://matplotlib.org`

name	no nodes	no edges	avg degree	avg weighted degree	density	connected	strongly conn components	avg clustering	assortativity
16-days-of-action-2018	396	349	1.763	1.909	0.002	False	394.0	0.01	-0.132
elf-day	365	436	2.389	2.499	0.003	False	357.0	0.1	-0.182
dry-january-2018	235	234	1.992	2.017	0.004	False	231.0	0.028	-0.283

Table 5.2: Excerpt of the contexts graph table.

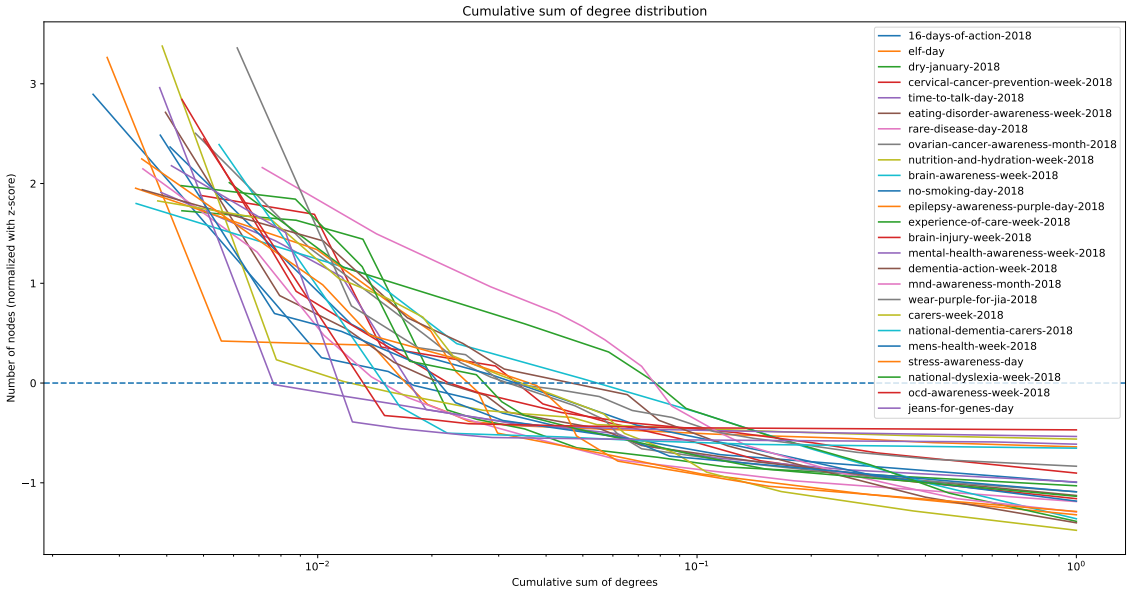


Figure 5.3: Cumulative degree distribution.

purpose is explained. Also, standard aggregated analysis (such as averages, sums, ...) of the presented tables across different contexts have been performed but left out from this document for brevity reasons.

First a context analysis (Table 5.2) is performed, which gives an idea of the size and the composition of the harvested contexts graphs. Featured metrics are computed with NetworkX and described in detail in the appendix A.

The cumulative degree distribution (Fig. 5.3) shows how the total node degree is distributed with respect to the number of nodes for each context.

$$P(k) = \frac{\#nodeswithdegree \geq k}{\#nodes} \tag{5.1}$$

Nodes with a lower degree (left on the plot) are typically way more than the ones with a high degree (right on the plot). The number of nodes are normalised with z-score $z = \frac{x_i - \bar{x}}{s}$ [28] where $\bar{x}$ is the arithmetic mean, s is the standard deviation and x_i is the actual value. Positive values are above the mean (dotted line) and conversely, negative values are below the mean. Z-score normalisation has been chosen because it preserves the range (maximum and minimum) and accounts for the dispersion of the data.

The cumulative degree distribution is more robust with respect to the simple degree distribution because it is more reliable with small numbers.

Second, results from the community detection algorithms are analysed. The same metrics of the previous table (Table 5.2) are applied to each context to get an insight of the resulting partition (Table 5.3). For each context, the ratio of

community	no nodes	no edges	avg degree	avg weighted degree	density	strongly conn components	avg clustering	assortativity
0	17	24	2.8235	2.9412	0.0882	15	0.1213	-0.7309
1	27	74	5.4815	5.4815	0.1054	27	0.302	-0.1155
2	30	38	2.5333	2.8	0.0437	27	0.2233	-0.7593
3	5	4	1.6	1.6	0.2	5	0.0	

Table 5.3: Excerpt of the partition analysis for a single context. Note that any graph to be considered as a community by a community detection algorithm is weakly connected.

name	community/no_nodes ratio
16-days-of-action-2018	0.14
elf-day	0.13
dry-january-2018	0.13

Table 5.4: Excerpt of the ratio of the number of communities over the number of nodes.

found communities over the number of nodes chosen to be part of a community is determined (Table 5.4). It helps determine how many users have meaningful interactions in sizeable groups. Each context's partition is checked with standard graph partition evaluation metrics (Table 5.5). Such metrics are computed with PQuality[25] which are detailed in appendix B.

The concepts of users appearing in more than one context ("shared" among communities) has been compared to users that are discarded as not appearing in any community to get an insight on whether repeated users are meaningful or not (Table 5.6).

- *is_present column*: describes whether a user belongs to any community or not.
- *no_participations column*: counts how many contexts a user has been in.

Users participating in more than one context are listed together with the Twitter profile information ordered by the number of participations to understand what kind of profiles are repeated the most (Table 5.7).

[25] `https://github.com/GiulioRossetti/partition_quality`

index	min	max	avg	std
internal_density	0.019	0.233	0.109	0.045
edges_inside	3.0	74.0	9.966	14.554
normalized_cut	0.0	0.438	0.082	0.119
avg_degree	1.5	5.481	2.081	0.956
fomd	0.038	0.5	0.309	0.129
expansion	0.0	1.2	0.208	0.323
cut_ratio	0.0	0.005	0.001	0.001
conductance	0.0	0.429	0.08	0.116
max_odf	0.0	31.0	1.552	5.593
avg_odf	0.0	1.2	0.245	0.356
flake_odf	0.0	0.0	0.0	0.0

Table 5.5: Excerpt of the partition quality metrics analysis for a single context.

	no_participations	is_present
dementiauk	4	True
timetochange	4	False
nhsengland	4	True

Table 5.6: Excerpt of the user description of the number of appearances across multiple contexts and whether they survived the community detection process or not.

	name	url	location	bio	lang	likes	follower rank	no participations
alzheimerssoc	Alzheimer's Society	NA	England, Wales & N.Ireland	We provide information and support, fund resea...	en	51603	0.99	4
dementiauk	Dementia UK	NA	Aldgate, London	Dementia UK provides specialist dementia suppo...	en	8086	0.98	4
mentalhealth	Mental Health Fdn	NA	UK	The UK's charity for everyone's mental health,...	en	9890	0.97	3

Table 5.7: Top shared users among multiple contexts.

Chapter 6

Empirical Evaluation

Existing methods to discover specific classes of online users are typically validated using a supervised approach, i.e., they rely on expert-generated ground truth. Such approaches, however, are prone to the subjectivity of the experts, whereby the evaluation would be measuring the fit of the model to the specific experts' own assessment of user instances' relevance. In contrast, we follow an unsupervised approach with no a priori knowledge of user relevance. We aim to demonstrate the value of our pipeline in creating a database of online profiles that are ready to be mined, along with examples of candidate user ranking functions. In this approach, human expertise only comes into play to assess and validate the top-k user lists produced by these functions. We demonstrate the pipeline in action on a significant set of 25 initial contexts, and define three alternative ranking functions aimed at capturing the empirical notion of *online activists*. To harvest such a number of contexts, the Python pipeline (Chapter 5) has been deployed on the Microsoft Azure cloud [1] to guarantee a continuous execution of the operations.

6.0.1 Deployment

Microsoft Azure is a cloud computing platform created and managed by Microsoft[2] for building, testing, deploying, and managing applications. It provides a comprehensive suite of services such as: Software as a Service (SaaS), Platform as a Service (PaaS) and Infrastructure as a Service (IaaS) and supports many different programming languages and frameworks including Python. Microsoft Azure has been chosen among other competing cloud services, such as Amazon AWS[3] or Google GCP[4], because a subscription was already available at the time and no specific service was needed from any of the other providers. The software runs on commodity hardware with no particular specification.

In Azure, all the offered services are called "Resources" (i.e. virtual machines, databases, virtual networks, ...), which are then grouped in a logical "Resource

[1] `https://azure.microsoft.com`
[2] `https://microsoft.com`
[3] `https://aws.amazon.com`
[4] `https://cloud.google.com`

Group"[5] to make them manageable (i.e. access control, pricing, dependencies, ...).

For this project a resource group has been defined, and a set of resources has been added (Fig. 6.1):

- *Virtual Machine (VM)*[6]: an IaaS product that let create a VM (either Windows or Linux).

  For this project a single Linux virtual machine "Standard B2s" has been created. It features 2 vcpus (Virtual CPUs), 4 GB of primary memory and 8 GB of temporary secondary memory. The Bs-series of general purpose virtual machines is described as an economical choice, suitable for workloads that typically run at a low to moderate baseline CPU performance.

  This resource is the computing environment where the software pipeline is executed.

- *Virtual Network*[7]: provides a virtual environment that enables virtual machines to communicate within a virtual network and to the internet.

  This resource is used by the software pipeline to exchange data to and from the internet.

- *Public IP address*[8]: provides a public IP address reachable from the internet.

  This resource is associated with the VM to make it reachable from the internet.

- *Network Security Group*[9]: filters network traffic to and from Azure resources within a virtual network with the internet.

  This resource filters out any connection to the resources (Tab. 6.1) and from the resources (Tab. 6.2) that are not related to the SSH (Secure SHell) connections and the Jupyter notebook server.

- *Network Interface*[10]: enables a VM to communicate with internet and the other resources in a virtual network.

  This resource is used by the software pipeline to connect to the internet either for the harvesting or the VM management operations.

- *Managed Disk*[11]: an IaaS product that offers persistent storage of data for Virtual Machines.

  For this project, we adopted a configuration which offers 30 GB (Standard SSD) of storage capacity.

[5] https://docs.microsoft.com/en-us/azure/azure-resource-manager
[6] https://docs.microsoft.com/en-us/azure/virtual-machines/linux
[7] https://docs.microsoft.com/azure/virtual-network
[8] https://docs.microsoft.com/azure/virtual-network/virtual-network-ip-addresses-overview-arm
[9] https://docs.microsoft.com/azure/virtual-network/security-overview
[10] https://docs.microsoft.com/azure/virtual-network/virtual-network-network-interface
[11] https://docs.microsoft.com/azure/virtual-machines/linux/managed-disks-overview

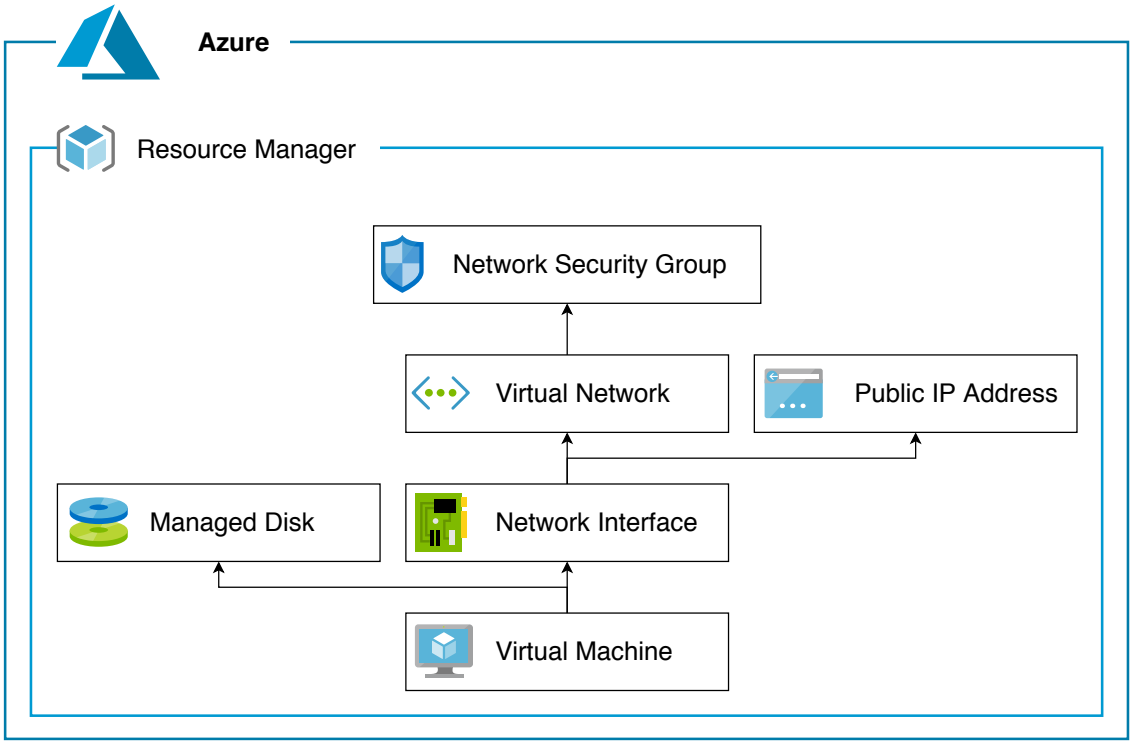


Figure 6.1: Azure schema definition for the resource group and the relations between all the defined resources for the project deployment.

Priority	Name	Port	Protocol	Source	Destination	Action
100	SSH	22	Any	Any	Any	Allow
110	Jupyter Notebook	8888	Any	Any	Any	Allow
65000	AllowVnetInBound	Any	Any	Any	VirtualNetwork	Allow
65001	AllowAzureLoadBalancerIn	Any	Any	Any	AzureLoadBalancer	Allow
65500	DenyAllInBound	Any	Any	Any	Any	Deny

Table 6.1: Inbound security rules for the resources in the virtual network. This configuration allows any connection from the internet to the ports 22 for the SSH connection and 8888 for the web Jupyter Notebook application connection. Note that in a production environment better and hardened security settings can be applied for example by using Azure Bastion[12].

This resource is used to store the software pipeline's code, the cache data and the embedded database.

All the resources are localised in the Northern Europe data-centre[13] to decrease latency access to the resources.

With this configuration, an average of 2 hours is needed for each context to be harvested and analysed. A more powerful configuration of computational resources would not provide an additional benefit, as the operations are mainly bound to the Twitter API limitations (Sec. 5.2.2).

[12] `https://docs.microsoft.com/azure/bastion`
[13] `https://azure.microsoft.com/global-infrastructure/regions`

Priority	Name	Port	Protocol	Source	Destination	Action
65000	AllowVnetOutBound	Any	Any	VirtualNetwork	VirtualNetwork	Allow
65001	AllowInternetOutBound	Any	Any	Any	Internet	Allow
65500	DenyAllOutBound	Any	Any	Any	Any	Deny

Table 6.2: Outbound security rules for the resources in the virtual network.

Context name	Period (2018)	Nodes	Edges	Density	Avg degree	Assortativity
16 days of action	11-25 / 12-10	396	349	0.002	1.8	-0.1
Elf day	12-03 / 12-12	365	436	0.003	2.4	-0.2
Dry january	01-01 / 01-31	235	234	0.004	2.0	-0.3
Cervical cancer prevention week	01-21 / 01-27	209	192	0.004	1.8	-0.1
Time to talk day	02-06 / 02-07	268	231	0.003	1.7	-0.2
Eating disorder awareness week	02-25 / 03-03	256	241	0.004	1.9	-0.2
Rare disease day	02-28 / 03-01	294	206	0.002	1.4	-0.2
Ovarian cancer awareness month	03-01 / 03-31	215	202	0.004	1.9	-0.4
Nutrition and hydration week	03-11 / 03-17	273	326	0.004	2.4	-0.3
Brain awareness week	03-11 / 03-17	307	281	0.003	1.8	-0.1
No smoking day	03-13 / 03-14	254	219	0.003	1.7	-0.3
Epilepsy awareness purple day	03-26 / 03-27	306	252	0.003	1.6	-0.2
Experience of care week	04-23 / 04-27	176	196	0.006	2.2	-0.1
Brain injury week	05-01 / 05-31	238	306	0.005	2.6	-0.1
Mental health awareness week	05-14 / 05-20	268	245	0.003	1.8	-0.5
Dementia action week	05-21 / 05-31	300	300	0.003	2.0	-0.0
Mnd awareness month	06-01 / 06-30	141	234	0.012	3.3	-0.3
Wear purple for jia	06-01 / 06-30	165	245	0.009	3.0	-0.5
Carers week	06-11 / 06-17	270	277	0.004	2.1	0.0
National dementia carers	09-09 / 09-10	184	177	0.005	1.9	-0.2
Mens health week	06-11 / 06-17	264	214	0.003	1.6	-0.2
Stress awareness day	11-07 / 11-08	293	209	0.002	1.4	-0.2
National dyslexia week	10-01 / 10-07	229	235	0.004	2.1	-0.2
Ocd awareness week	10-07 / 10-13	202	193	0.005	1.9	-0.6
Jeans for genes day	09-21 / 09-22	246	325	0.005	2.6	-0.2

Table 6.3: List of contexts used in the experiments along with network metrics.

6.0.2 Contexts and Networks

We have manually selected 25 contexts within the scope of health awareness campaigns in the UK, all occurring in 2018 and well-characterised using predefined hashtags (Fig. 6.2). Due to limitations imposed by Twitter on the number of posts that can be retrieved within a time interval (Sec. 5.2.2), only 200 tweets were retrieved from each context. Table 6.3 lists the events along with key metrics for their corresponding user-user networks. To recall, *assortativity* measures how frequently nodes with a high degree are likely to connect with other nodes with a high degree (> 0) or with a low degree (< 0). Negative figures (mean: -0.22, std dev: 0.17) are in line with what is observed on the broader Twitter network [19]. The very small figures for density, defined as $\frac{\#edges}{\#nodes \cdot (\#nodes - 1)}$ (mean: 0.004, std dev: 0.002), suggest very few connections exist amongst users within a context. This makes it difficult to detect meaningful communities, as described below, thus for some contexts the topological metrics are measured on the entire network as opposed to within each community. This view is also supported by the small average node degree (mean: 2.04, std dev: 0.46) and the ratio of strongly connected components to the number of nodes (mean: 0.98, std. dev. 0.02).

6.0.3 Communities

DEMON and Infomap produce significantly different communities in each network. DEMON identifies communities in only 48% of the networks, with an average of only

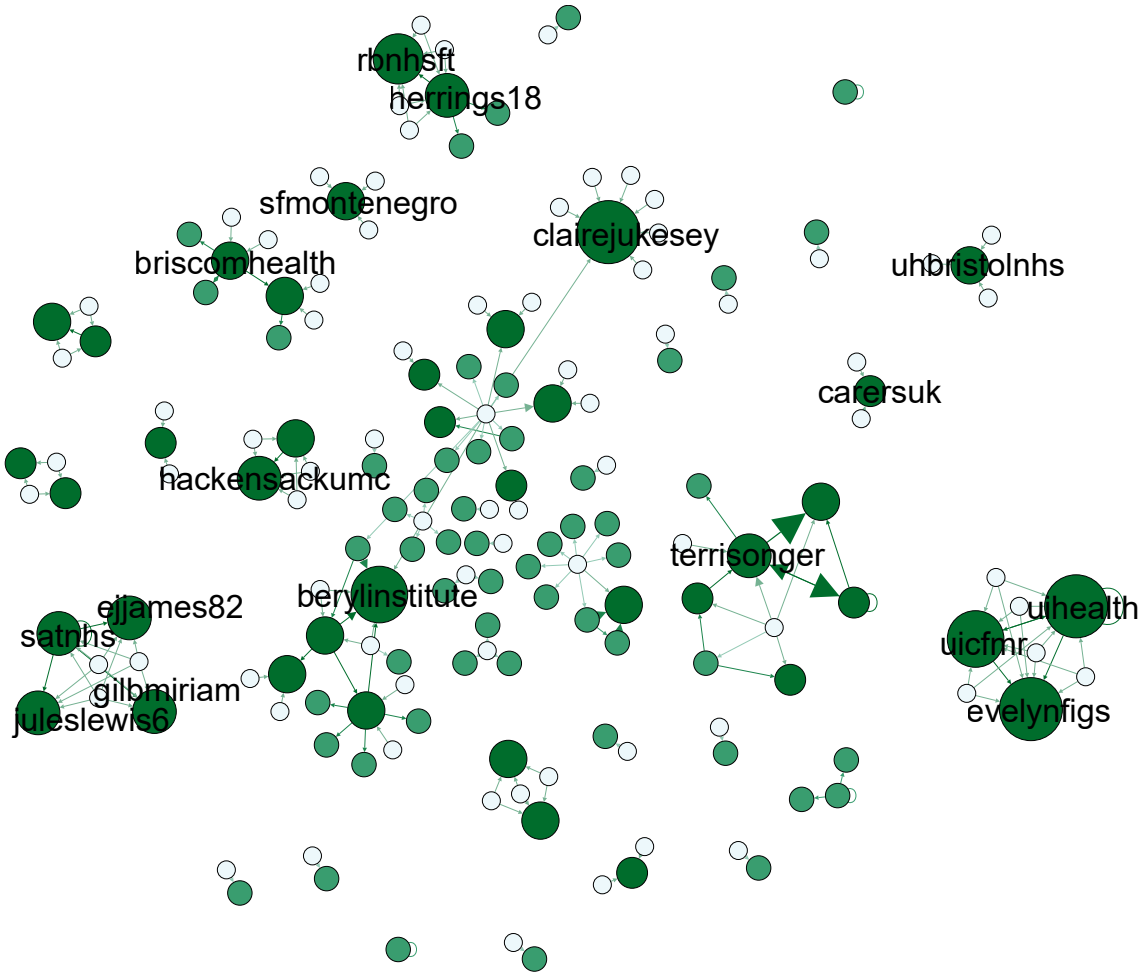


Figure 6.2: Example of the network of the context "Experience awareness week" from the UK healthcare case. Nodes, represented by circles, depict Twitter users and are proportional in size to the in-degree. Nodes with a higher in-degree also highlight the related user name. In this network, highly connected nodes are often carers, for example "clairejukesey" is a patient services manager and "sfmontenegro" self describes her as a "healthcare leader". Edges, represented by lines, depict the directed relations between Twitter users and are proportional in size to the weight.

Metric	DEMON	Infomap
Fraction of networks with no communities	0.52	0.0
Number of communities per context (avg)	1.92	18.88
Fraction of network users added to the DB (avg)	0.06	0.59
Fraction of repeat users added to the DB across networks	0.28	0.37

Table 6.4: Comparing DEMON to Infomap for community detection.

1.92 communities per network and a slightly negative (-0.28) average assortativity per community, in line with the average for their parent networks. Only the users who belong to one of those communities, about 6%, are added to the database. For the remaining 52% of networks where no communities are detected, users' in-degrees are calculated using the entire network, and all users are added to the database, for a total of 3,570 users being added to the database in our experiments using DEMON.

In contrast, Infomap provides meaningful communities for all networks (Fig. 6.3). Those with less than 3 users are discarded, leaving 18.88 communities per network on average, with 8.5 users per community on average. When using Infomap, 3,567 users were added to the database (on average 253 users per network). The average assortativity across all communities is again slightly negative (-0.43). Table 6.4 compares the two approaches on the key metrics just discussed. On the basis of this comparison, we recommend using Infomap, which we have used for our evaluation.

6.0.4 Users Discovery

Repeat users who appear in multiple contexts are particularly interesting as they provide a stronger signal. Out of the total 3,567 users, 160 of them appear at least in two of the 25 contexts. After community detection, only 61 of these users are still seen as repeat users, while the remaining 99 are either removed altogether, or they only appear once. Of the 61, 57 appear twice, 2 appear three times, and 2 appear four times. Thus, only 1.6% of users appear more than once when communities with more than 3 users are considered, compared to the overall 4.5% of overall repeat users. Table 6.5 reports the top-10 repeat users along with their *Follower Rank*, and Fig. 6.4 shows the number of repeat users per context. As the table is sorted by number of occurrences then by *Follower Rank* (Chap. 3), an indication of popularity, it is not surprising to find that top users include well-known names such as Mr. Hunt, who at the time of the events was Secretary of State for Health and Social Care in the UK, with $FR = 1$, and a number of associations and foundations active in the public healthcare space. More interesting are perhaps non-repeat users who emerge when ad-hoc ranking is applied to the database, as we illustrate next.

6.0.5 Users Ranking

To demonstrate the potential value of the database, albeit on a small scale, we have tested three user ranking functions based on the indicators introduced in Chap. 3.

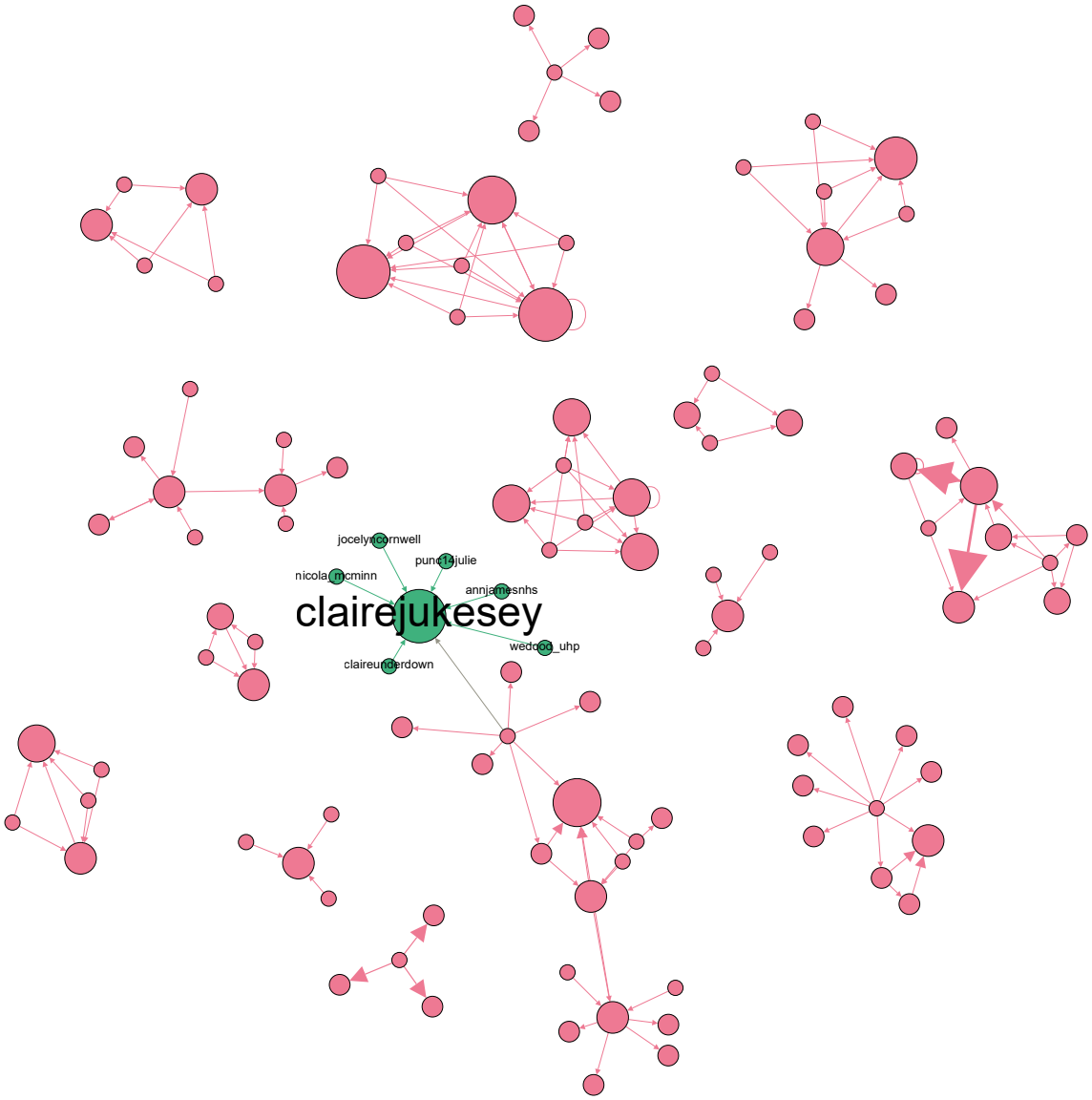


Figure 6.3: Example of the network after applying Infomap, from the context "Experience awareness week" in the UK healthcare case. In green are highlighted the nodes and the edges of a single example community. Nodes, represented by circles, depict Twitter users and are proportional in size to the in-degree. Edges, represented by lines, depict the directed relations between Twitter users and are proportional in size to the weight.

Username	Name	Follower rank	Participations
alzheimerssoc	Alzheimer's Society	0.99	4
dementiauk	Dementia UK	0.98	4
mentalhealth	Mental Health Fdn	0.97	3
colesmillerllp	Coles Miller LLP	0.65	3
jeremy_hunt	Jeremy Hunt	1.0	2
nhsengland	NHS England	0.99	2
carersuk	Carers UK	0.95	2
rdash_nhs	RDaSH NHS FT	0.88	2
alzsocseengland	Alzheimer's Society - South ...	0.64	2
mndassoc	MND Association	0.64	2

Table 6.5: Top-10 repeat users, amongst those who belong to a community.

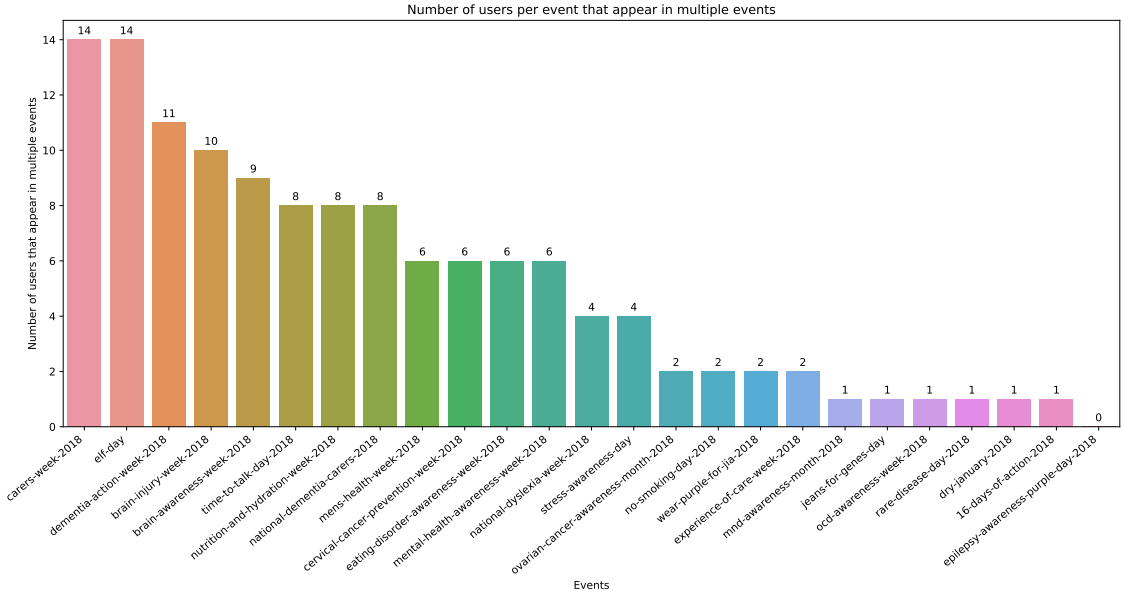


Figure 6.4: Number of repeat users for each context

As mentioned, the aim of this exercise is to provide an objective grounding for engaging with experts on finding suitable operational definitions for specific user profiles. We consider good functions those that privilege individuals over organisations or business.

$$\textit{Ranking 1: } R1(u) = \frac{1}{\sum_{u \in C} IC(u) + 1} \cdot \sum_{u \in C} TF(u) \tag{6.1}$$

$$\textit{Ranking 2: } R2(u) = |FR(u) - 1| \cdot \left(\sum_{u \in C} TA(u) + \sum_{u \in C} IC(u) \right) \tag{6.2}$$

$$\textit{Ranking 3: } R3(u) = |FR(u) - 1| \cdot \left(\sum_{u \in C} TA(u) + \frac{1}{\sum_{u \in C} IC(u) + 1} \right) \tag{6.3}$$

All the three functions 6.1, 6.2 and 6.3 consider every community C where the given user u have appeared ($u \in C$) across all the contexts.

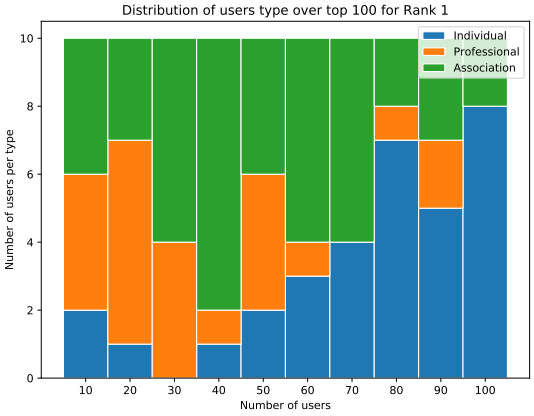
Function (6.1) is designed to promote users who are at the "fringe" of their

	Ranking 1			Ranking 2			Ranking 3		
#	**User**	**On-topic**	**Individual**	**User**	**On-topic**	**Individual**	**User**	**On-topic**	**Individual**
1	homesnutrition	X		johnneustadt	X		johnneustadt	X	
2	ficajones	X	X	jo_millar27	X	X	solutions777	X	X
3	helenvweaver	X	X	hatchbrenner			kingste29344921	X	X
4	spriggsnutri	X		nchawkes	X	X	daisylu1964		X
5	critcarelthtr	X		moz0373runner	X	X	zakariamarsli	X	X
6	danielleroisin_	X	X	aimsonhealth	X	X	meowaaaaaa		X
7	mynameisandyj	X	X	wordsharkv5		X	vecta67		X
8	fionaliu92	X	X	fullcircle_play	X		cosfordfamily1	X	X
9	ldpartnership	X		qsprivatehealth	X		hayleycorriganx		X
10	milaestevam1		X	socialissp			jhbrasfie		X

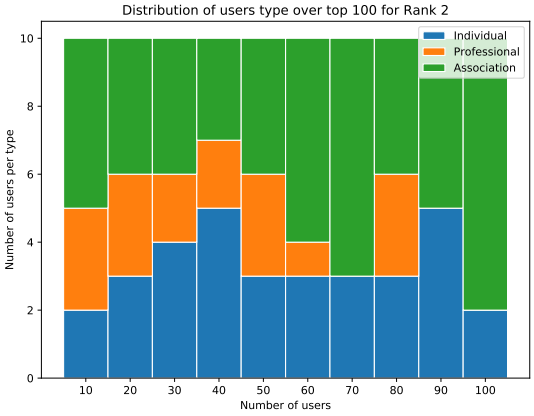
Table 6.6: Top-10 ranked users for ranking functions (6.1) and (6.2) and (6.3), with indication of whether the user is on-topic/off-topic and individual vs association/professional. Such categories are useful to evaluate the ranking functions.

community, while giving credit to generic on-topic activities during the contexts. To achieve this, *Topical Focus TF* is used as a positive contribution, while a large in-degree *IC* reduces the score. In contrast, function (6.2) penalises user popularity, i.e., by using the complement of *Follower Rank FR*, while rewarding prominence inside communities (in-degree *IC*) and information spreading by also considering shared links (*Topical Attachment TA*). Function (6.3) combines ideas from both (6.2) and (6.1).

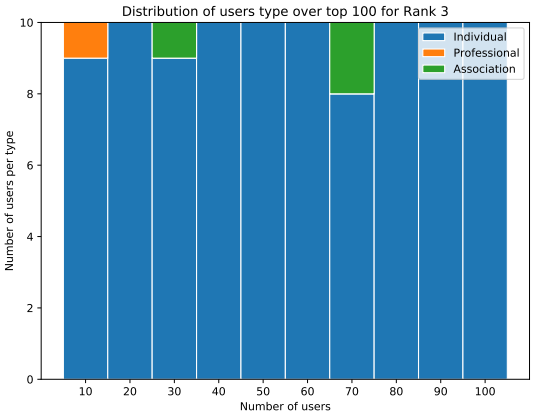
The top-10 users for each ranking are reported in Tab. 6.6. To appreciate the effects of these functions, we have manually labelled the top-100 user profiles for each of the rankings, using a broad type classification as *individuals* as opposed to *institutional players* (associations, public bodies), or *professionals*. The fractions of on-topic users are 86%, 83%, and 38% for (6.1), (6.2), and (6.3) respectively. Importantly, (6.3) identifies more individuals than institutions and professionals (96%) than (6.2) and (6.1), both at 33%. Also, repeat users are rewarded in both rankings. Users with $FR(u) = 0$ and $\frac{t(u)-min(t(u))}{max(t(u))-min(t(u))} < 0.005$, where $t(u) = |Tweets(u)|$, are considered not active and have been assigned lowest score. Fig. 6.5 shows the distribution of user types within the top-100 users for each of the three rankings, broken down into 10 users bins. We can see that individuals dominate in (6.3), and are fewer but emerge earlier in the ranks when (6.2) is used. We plan to conduct user studies to establish useful analytics to be incorporated into our framework.



(a) Rank 1: 33 are individuals, 23 are professionals, 44 are associations.



(b) Rank 2: 33 are individuals, 17 are professionals, 50 are associations.



(c) Rank 3: 96 are individuals, 1 are professionals, 3 are associations.

Figure 6.5: Distribution of user types for top-100 users and for each ranking function.

Chapter 7

Conclusion

Motivated by the need to find an operational definition of "online activists" that is grounded in well-established network and user-activity metrics, we have designed a Twitter content processing pipeline for progressively harvesting Twitter users based on their engagement with online socially-minded events, or campaigns, which we have called *contexts*. The pipeline yields a growing database of user profiles along with their associated metrics, which can then be analysed to experiment with user-defined user ranking criteria. The pipeline is designed to select promising candidate profiles, but the approach is unsupervised, i.e., no manual classification of example users is provided. We have empirically evaluated the pipeline on a case study, along with experimental scoring functions to show the viability of the approach.

The design of the pipeline show that useful harvesting of interesting users can be accomplished within the limitations imposed by Twitter on its APIs. The next challenge is to automate the discovery of new contexts so that the pipeline may continuously add new and update users in the database. Only at this point will it be possible to validate the entire approach, hopefully with help from third party users, on a variety of new context topics.

Future research directions include a (1) method to semi-autonomously harvest new contexts, (2) a dashboard that helps users to select new and promising contexts, (3) validation of the described approach on multiple topics.

Currently (1) is in an advanced validation stage, results are planned to be published and the code is currently available as an incremental software update to this project's software repository[1]. The work features an unsupervised method which combines community detection and signal analysis over a multiplex network of hashtags and users. Its goal is to discover new contexts starting from the user ranked list (Chapter 6).

A dashboard (2) would be of help for non-technical users to interact with the framework. It would allow the selection of the bootstrapping contexts as well as connecting to the top ranked users, the discovered activists, directly for a call for action.

Altough this approach has been tested on the presented use case (Sec. 6), the

[1] `https://github.com/flaprimo/twitter-network-analysis`

framework would benefit from additional testing over other topics and use cases (3).

Bibliography

[1] V Arnaboldi, M Conti, A Passarella, and F Pezzoni. Ego networks in Twitter: An experimental analysis. In *2013 Proceedings IEEE INFOCOM*, pages 3459–3464, 2013.

[2] Or Biran, Sara Rosenthal, Jacob Andreas, Kathleen McKeown, and Owen Rambow. Detecting influencers in written online conversations. In *Proceedings of the Second Workshop on Language in Social Media*, LSM '12, pages 37–45, Stroudsburg, PA, USA, 2012. Association for Computational Linguistics.

[3] Imen Bizid, Nibal Nayef, Patrice Boursier, and Antoine Doucet. Detecting prominent microblog users over crisis events phases. *Information Systems*, 78:173–188, nov 2018.

[4] Imen Bizid, Nibal Nayef, Patrice Boursier, Sami Faiz, and Jacques Morcos. Prominent Users Detection During Specific Events by Learning On- and Off-topic Features of User Activities. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, ASONAM '15, pages 500–503, New York, NY, USA, 2015. ACM.

[5] Imen Bizid, Nibal Nayef, Patrice Boursier, Sami Faiz, and Jacques Morcos. Prominent Users Detection During Specific Events by Learning On- and Off-topic Features of User Activities. In *Proceedings of the 2015 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining 2015*, ASONAM '15, pages 500–503, New York, NY, USA, 2015. ACM.

[6] Twitter Safety Blog. Information operations directed at hong kong, aug 2019.

[7] Vincent D Blondel, Jean-Loup Guillaume, Renaud Lambiotte, and Etienne Lefebvre. Fast unfolding of communities in large networks. *Journal of Statistical Mechanics: Theory and Experiment*, 2008(10):P10008, oct 2008.

[8] Chris Bobel. "i'm not an activist, though i've done a lot of it": Doing activism, being activist and the "perfect standard" in a contemporary movement. *Social Movement Studies*, 6(2):147–159, 2007.

[9] Phillip Bonacich and Paulette Lloyd. Eigenvector-like measures of centrality for asymmetric relations. *Social Networks*, 23(3):191–201, jul 2001.

[10] Norman Booth and Julie Ann Matic. Mapping and leveraging influencers in social media to shape corporate brand perceptions. corporate communications 16, 184-191. *Corporate Communications: An International Journal*, 16:184–191, 08 2011.

[11] Duncan Brown. *Influencer marketing: who really influences your customers?* Routledge, Taylor & Francis Group, 2016.

[12] Meeyoung Cha, Hamed Haddadi, Fabrício Benevenuto, and Krishna P. Gummadi. Measuring user influence in twitter: The million follower fallacy. In *ICWSM*, 2010.

[13] CMSWire.com. Social media influencers: Mega, macro, micro or nano, dec 2010.

[14] Jonas Colliander and Micael Dahlén. Following the fashionable friend: The power of social media. *Journal of Advertising Research*, 51(1):313–320, 2011.

[15] Michele Coscia, Fosca Giannotti, and Dino Pedreschi. A classification for community discovery methods in complex networks. *Statistical Analysis and Data Mining: The ASA Data Science Journal*, 4(5):512–546, 2011.

[16] Michele Coscia, Giulio Rossetti, Fosca Giannotti, and Dino Pedreschi. Demon: A local-first discovery method for overlapping communities. In *Proceedings of the 18th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, KDD '12, pages 615–623, New York, NY, USA, 2012. ACM.

[17] Nick Crossley, Elisa Bellotti, Gemma Edwards, Martin G. Everett, Johan Koskinen, and Mark Tranmer. Social network analysis for ego-nets. In *Social Network Analysis for Ego-Nets*, 2015.

[18] Giorgio Fagiolo. Clustering in complex directed networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 76:026107, 09 2007.

[19] David N. Fisher, Matthew J. Silk, and Daniel W. Franks. *The Perceived Assortativity of Social Networks: Methodological Problems and Solutions*, pages 1–19. Springer International Publishing, Cham, 2017.

[20] Santo Fortunato. Community detection in graphs. *Physics Reports*, 486(3-5):75–174, feb 2010.

[21] Jeff Goodwin and James M. Jasper. *The Social Movements Reader: Cases and Concepts*. John Wiley
Sons, 2009.

[22] J. E. Hirsch. An index to quantify an individual's scientific research output. *Proceedings of the National Academy of Sciences*, 102(46):16569–16572, 2005.

[23] D. A. Huffman. A method for the construction of minimum-redundancy codes. *Proceedings of the IRE*, 40(9):1098–1101, sep 1952.

[24] Amanda Hughes and Leysia Palen. Twitter adoption and use in mass convergence and emergency events. *International Journal of Emergency Management*, 6:248–260, 02 2009.

[25] Business Insider. Why brands are turning away from big instagram influencers to work with people who have small followings instead, apr 2019.

[26] Magdalini Kardara, George Papadakis, Athanasios Papaoikonomou, Konstantinos Tserpes, and Theodora Varvarigou. Large-scale evaluation framework for local influence theories in Twitter. *Information Processing & Management*, 51(1):226–252, 2015.

[27] Ed Keller and Jon. Berry. *The Influentials*. Free Press, 2003.

[28] S. Kokoska and D. Zwillinger. *CRC Standard Probability and Statistics Tables and Formulae*. CRC Press, 2000.

[29] Haewoon Kwak, Changhyun Lee, Hosung Park, and Sue Moon. What is twitter, a social network or a news media? In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, page 591–600, New York, NY, USA, 2010. Association for Computing Machinery.

[30] Ganaele Langlois, Greg Elmer, Fenwick McKelvey, and Zachary Devereaux. Networked publics: The double articulation of code and politics on facebook. *Canadian Journal of Communication*, 34(3), 2009.

[31] Gilad Lotan, Erhardt Graeff, Mike Ananny, Devin Gaffney, Ian Pearce, and Danah Boyd. The arab spring— the revolutions were tweeted: Information flows during the 2011 tunisian and egyptian revolutions. *International Journal of Communication*, 5(0), 2011.

[32] Mckinseyquarterly.com. The consumer decision journey, mar 2013.

[33] Alexanderand Miu Tudorand Pal Atinderand Daniilakis Michaeland Garcia Alessandroand Cedrim Diegoand da Silva Sousa Leonardo Missier, Paoloand Romanovsky. Tracking dengue epidemics using twitter content classification and topic modelling. In Peterand Pautasso Cesare Casteleyn, Svenand Dolog, editor, *Current Trends in Web Engineering*, pages 80–92, Cham, 2016. Springer International Publishing.

[34] Paolo Missier, Callum McClean, Jonathan Carlton, Diego Cedrim, Leonardo Silva, Alessandro Garcia, Alexandre Plastino, and Alexander Romanovsky. Recruiting from the Network: Discovering Twitter Users Who Can Help Combat Zika Epidemics. In *Web Engineering: 17th International Conference, ICWE 2017, Rome, Italy, June 5-8, 2017, Proceedings*, pages 437–445, Roma, Italy, 2017. Springer International Publishing.

[35] A. Nargundkar and Y. S. Rao. Influencerank: A machine learning approach to measure influence of twitter users. In *2016 International Conference on Recent Trends in Information Technology (ICRTIT)*, pages 1–6, April 2016.

[36] M. E. J. Newman. Modularity and community structure in networks. *Proceedings of the National Academy of Sciences of the United States of America*, 103(23):8577–8582, jun 2006. 16723398[pmid].

[37] M. E. J. Newman. *Networks: an introduction*. Oxford University Press, Oxford; New York, 2010.

[38] Facebook Newsroom. Removing coordinated inauthentic behavior from china, aug 2019.

[39] Lucas A Overbey, Benjamin Greco, Christopher Paribello, and Terresa Jackson. Structure and prominence in Twitter networks centered on contentious politics. *Social Network Analysis and Mining*, 3(4):1351–1378, dec 2013.

[40] Aditya Pal and Scott Counts. Identifying topical authorities in microblogs. In *Proceedings of the fourth ACM international conference on Web search and data mining - WSDM '11*, 2011.

[41] Helen Peck. *Relationship marketing: strategy and implication*. Butterworth-Heinemann, 1999.

[42] Thomas Poell. Social media and the transformation of activist communication: exploring the social media ecology of the 2010 Toronto G20 protests. *Information, Communication & Society*, 17(6):716–731, jul 2014.

[43] Flavio Primo, Paolo Missier, Alexander Romanovsky, Mickael Figueredo, and Nelio Cacho. A customisable pipeline for continuously harvesting socially-minded twitter users. In Maxim Bakaev, Flavius Frasincar, and In-Young Ko, editors, *Web Engineering*, pages 91–106, Cham, 2019. Springer International Publishing.

[44] Nandini Raghavan, Réka Albert, and Soundar Kumara. Near linear time algorithm to detect community structures in large-scale networks. *Physical review. E, Statistical, nonlinear, and soft matter physics*, 76:036106, 10 2007.

[45] G. Razis and I. Anagnostopoulos. Semantifying twitter: The influence tracker ontology. In *2014 9th International Workshop on Semantic and Social Media Adaptation and Personalization*, pages 98–103, Nov 2014.

[46] Fabian Riquelme and Pablo Gonzalez-Cantergiani. Measuring user influence on twitter: A survey. *Information Processing and Management*, 52(5):949–975, 2016.

[47] Martin Rosvall and Carl T. Bergstrom. Maps of random walks on complex networks reveal community structure. *Proceedings of the National Academy of Sciences of the United States of America*, 105:1118–23, 02 2008.

[48] Takeshi Sakaki, Makoto Okazaki, and Yutaka Matsuo. Earthquake shakes twitter users: Real-time event detection by social sensors. In *Proceedings of the 19th International Conference on World Wide Web*, WWW '10, page 851–860, New York, NY, USA, 2010. Association for Computing Machinery.

[49] Jari Saramäki, Mikko Kivelä, Jukka-Pekka Onnela, Kimmo Kaski, and János Kertész. Generalizations of the clustering coefficient to weighted complex networks. *Physical Review E*, 75:027105, 03 2007.

[50] C B Schenk and D C Sicker. Finding Event-Specific Influencers in Dynamic Social Networks. In *2011 IEEE Third International Conference on Privacy, Security, Risk and Trust and 2011 IEEE Third International Conference on Social Computing*, pages 501–504, 2011.

[51] Leonardo Sousa, Rafael de Mello, Diego Cedrim, Alessandro Garcia, Paolo Missier, Anderson Uchoa, Anderson Oliveira, and Alexander Romanovsky. Vazadengue: An information system for preventing and combating mosquito-borne diseases with social networks. *Information Systems*, 75:26 – 42, 2018.

[52] Gerrit Sundermann and Thorsten Raabe. Strategic communication through social media influencers: Current state of research and desiderata. *International Journal of Strategic Communication*, 13(4):278–300, 2019.

[53] Jaewon Yang and Jure Leskovec. Defining and evaluating network communities based on ground-truth. *Knowledge and Information Systems*, 42, 05 2012.

[54] William Lafi Youmans and Jillian C. York. Social media and the activist toolkit: User agreements, corporate interests, and the information infrastructure of modern social movements. *Journal of Communication*, 62(2):315–329, 2012.

[55] Wayne Xin Zhao, Jing Jiang, Jianshu Weng, Jing He, Ee-Peng Lim, Hongfei Yan, and Xiaoming Li. Comparing twitter and traditional media using topic models. *33rd European Conference on IR Research, ECIR 2011*, pages 338–349, 2011.

Appendix A

Context graph metrics

Several graph topological metrics are stored for each context's graph, these metrics are needed for subsequent analysis and comparisons between contexts.

Given a directed weighted graph G belonging to a given context, we call $A = [a_{ij}]$ the unweighted adjacency matrix [37] which represents the existence of directed edges between nodes of G where:

$$[A]_{ij} = a_{ij} = \begin{cases} 1, & \text{if there is an edge from node } i \text{ to node } j \\ 0, & \text{otherwise} \end{cases} \tag{A.1}$$

while the directed weighted version $W = [w_{ij}]$ also considers the edge's weights:

$$[W]_{ij} = \begin{cases} w_{ij} > 0, & \text{the weight of the edge if there is an edge from node } i \text{ to node } j \\ 0, & \text{otherwise} \end{cases} \tag{A.2}$$

In the following, we list the computed metrics for each context graph:

- *# nodes*: number of users (nodes) N.
- *# edges*: number of relations (edges) between users M.
- *avg degree*: average total number of edges per node $deg^{tot}_{avg} = \frac{\sum^N k^{tot}_i}{N}$ with each node i having total degree $k^{tot}_i = \sum_j a_{ij} + \sum_j a_{ji}$.
- *avg in-degree*: average total number of edges per node $deg^{in}_{avg} = \frac{\sum^N k^{in}_i}{N}$ with each node i having total degree $k^{in}_i = \sum_j a_{ji}$.
- *avg weighted degree*: average total weight sum per node $w\_deg^{tot}_{avg} = \frac{\sum^N w\_k^{tot}_i}{N}$ with each node i having total weighted degree $w\_k^{tot}_i = \sum_j w_{ij} + \sum_j w_{ji}$.
- *density*: expresses how sparse is the adjacency matrix (Equation A.1) $d = \frac{M}{N \cdot (N-1)}$. The limit values are 0 for a graph without edges and 1 for a complete graph (every node connected to every other node).
- *weakly connected*: the graph is weakly connected if for each pair (i, j) of nodes there exists a path that connects them, when the edge direction is disregarded.

- *number of strongly connected components*: it is the number of strongly connected components in the graph. A strongly connected component is a maximal subset of nodes such that there is a directed path in both directions between every pair in the subset [37].

- *avg clustering*: the average of the clustering coefficient for a graph is $C = \frac{1}{N}\sum_N c_i$ [37] where C_i is the clustering coefficient for node i, which, in a directed weighted graph, is intended as the geometric average of the subgraph edge weights [18] [49]:

$$c_i = \frac{1}{deg^{tot}(i)(deg^{tot}(i)-1) - 2deg^{\leftrightarrow}(i)} T(i) \tag{A.3}$$

  where $T(i) = \sum_{jk}(w_{ij} \cdot w_{ik} \cdot w_{jk})^{1/3}$ only considering directed triangles through node i, $deg^{tot}(i)$ is the sum of in degree and out degree of i and $deg^{\leftrightarrow}(i) = \sum_{j \neq k} w_{ij} \cdot w_{ik}$ is the reciprocal degree of i.

- *assortativity*: the standard Pearson correlation coefficient for the nodes degrees [37]:

$$r = \frac{\sum_{ij}(a_{ij} - k_i^{out}k_j^{in}/2M)k_i^{out}k_j^{in}}{\sum_{ij}(k_i\delta_{ij} - k_i^{out}k_j^{in}/2M)k_i^{out}k_j^{in}} \tag{A.4}$$

  where $k_i^{out} = \sum_j w_{ij}$ and $k_j^{in} = \sum_i w_{ij}$ are respectively the "out" and "in" weighted degrees for a node. The correlation coefficient varies between a maximum of 1 (perfectly assortative network) and a minimum of -1 (perfectly disassortative network).

- *h-index*: the h-index [22] of a node i is defined as the maximum value of h such that i has h inbound edges that have at least a weight h.

Appendix B

Context partition metrics

To understand the effectiveness of the community detection algorithm, several partition quality metrics have been computed [53].

Given a directed graph $G(V, E)$, where V is its set of nodes and E is its set of edges, we call S a set of its nodes and $f(S)$ a function that scores how likely the set of nodes S is a significant community. $n_S = |S|$ is the number of nodes in S, $m_S = |\{(u, v) \in E : u \in S, v \in S\}|$ is the number of edges in S, $c_S = |\{(u, v) \in E : u \in S, v \notin S\}|$ is the outer number of edges on the boundary of S and k_u^{tot} is the total degree of node u.

Featured metrics are:

- *Internal density*: Number of edges m_S in a community S divided by the total number of possible edges between all the nodes $f(S) = \frac{m_S}{n_S \cdot (n_S - 1)}$.
- *FOMD (Fraction over median degree)*: Determines the number of nodes that have an internal degree greater than the median degree of nodes in subset S, $f(S) = \frac{|\{u : u \in S, |\{(u,v) : v \in S\}| > d_m\}|}{n_S}$ where d_m is the median value of k_u^{tot} in V.
- *Expansion*: A measure of "separability" which averages the number of external connections c_S per node n_S in S, $f(S) = \frac{c_S}{n_S}$.
- *Cut Ratio*: A measure of "separability", which is the fraction of external edges c_S of S out of the total number of possible edges connecting S with the rest of the graph, $f(S) = \frac{c_S}{n_S(n - n_S)}$.
- *Conductance*: Ratio of edges inside the community to the number of edges leaving the community (captures surface area to volume) $f(S) = \frac{m_S}{c_S}$.
- *Normalised Cut*: Represents how well subset S is separated from the rest of G. Combines Conductance with the fraction of external edges over all non-community edges $f(S) = \frac{m_S}{c_S} + \frac{c_S}{(m - m_S) + c_S}$.
- *Maximum ODF (Out Degree Fraction)*: maximal fraction of external connections to internal connections for the nodes in S $f(S) = max_{u \in S} \frac{|\{(u,v) \in E : v \notin S\}|}{d(u)}$.
- *Average ODF*: Same as Maximum ODF but takes the average $f(S) = \frac{1}{n_S} \sum_{u \in S} \frac{|\{(u,v) \in E : v \notin S\}|}{d(u)}$.

- *Flake ODF*: Fraction of the nodes in S that have less internal connections than external connections $f(S) = \frac{|\{u:u\in S,|\{(u,v)\in E:v\in S\}\}|}{n_S}$.

High values for Internal density, Average degree, FOMD, Conductance, Normalised Cut and Flake ODF metrics determine an high community quality, resulting in more cohesive communities, while for Expansion, Cut Ratio, Maximum and average ODF lower values are recommended instead.