FACT: a Framework for Analysis and Capture of Twitter Graphs

Daniel Thilo Schroeder
Simula Metropolitan
Center for Digital Engineering
Oslo, Norway
Technical University of Berlin
Berlin, Germany
daniels@simula.no

Konstantin Pogorelov
Simula Research Laboratory
Fornebu, Norway
konstantin@simula.no

Johannes Langguth
Simula Research Laboratory
Fornebu, Norway
langguth@simula.no

***Abstract*—In the recent years, online social networks have become an important source of news and the primary place for political debates for a growing part of the population. At the same time, the spread of *fake news* and *digital wildfires* (fast-spreading and harmful misinformation) has become a growing concern worldwide, and online social networks the problem is most prevalent. Thus, the study of social networks is an essential component in the understanding of the fake news phenomenon. Of particular interest is the network connectivity between participants, since it makes communication patterns visible. These patterns are hidden in the offline world, but they have a profound impact on the spread of ideas, opinions and news. Among the major social networks, Twitter is of special interest. Because of its public nature, Twitter offers the possibility to perform research without the risk of breaching the expectation of privacy. However, obtaining sufficient amounts of data from Twitter is a fundamental challenge for many researchers. Thus, in this paper, we present a scalable framework for gathering the graph structure of follower networks, posts and profiles. We also show how to use the collected data for high-performance social network analysis.**

***Index Terms*—Twitter, Data collection, Graph capturing, Computational social science**

I. Introduction

Computational Social Science [1] has become an important interdisciplinary field in the last decade. By evaluating data generated by online interactions, social scientists can, for the first time, observe human behaviour in real life in a way that was previously possible only in laboratory experiments of limited scale. Within the field, the study of online social networks is of particular interest because they reveal opinions, relationships, and interests of a large number of people in a quantifiable manner.

In the wake of the 2016 US presidential elections, the spread of misinformation in social networks, commonly known as *"fake news"*, has become an important area of study. In addition to political effects, fast spreading, dangerous misinformation commonly known as *digital wildfires* can have serious economic [2], personal [3], and security related [4] consequences in the real world. The decentralized nature of social media communication makes it hard for authorities to detect such problems. At the same time, misinformation arising from a single source can spread extremely fast. Therefore, the study of the formation and spreading of misinformation in online social networks has become an important area of research. The primary workflow for studies in that field is the large-scale analysis of communication connections, message contents, and the patterns in which information spreads.

Among the online social networks, Twitter has become the foremost platform for research. The reason lies in the fact that most information on Twitter is public by default and can be accessed efficiently via the Twitter API. While the restriction to public platforms excludes a large part of people's online interactions, it is very difficult to analyze non-public data without violating the user's expectation of privacy. Thus, for reasons of research ethics and availability of data, public platforms are often the only viable option.[1]

For many studies, the complexity of acquiring large amounts of Twitter data through the Twitter API in a fast and efficient manner constitutes a fundamental challenge, especially when using software that was built for more traditional data acquisition and processing methods. By the same token, the use of graph analytics is not yet widespread in the field, despite the fact that it has already been recognized as a highly promising approach [6]. Therefore, we alleviate the problem by proposing a new framework for capturing and analysing Twitter data. Several tools serving a similar purpose have been created in the past [7]–[13]. However, these tools typically have a limited functionality, which restricts the amount of data that can be captured from Twitter. In contrast, our framework makes full use of modern software design principles including parallel and distributed cloud-based computing. The components of our framework are connected within a service-oriented and scalable software architecture. Captured data is stored in an efficient, graph-representing database, which allows easy access to a variety of powerful graph analytic tools. The framework's access points and API as well as our sample experiments

This work is founded by the Norwegian SAMRISK-2 project "UMOD" (#272019).

[1]This statement is based on the Norwegian Guidelines on Research Ethics in Social Sciences [5], by which the authors of this paper are bound. Research ethics requirements for other countries may vary.

are written in Python and executed in the Jupyter Notebook environment [14], which enhances the reproducibility of the research. Thus, the key contributions of our paper are the following:

- Design and development of a novel, flexible, and extensible framework for capturing Twitter data. The framework is well-integrated with the existing open source software.
- Analysis of related work and a discussion of the pros and coins of the existing solutions.
- Initial experimental research and analysis of collected Twitter data in order to demonstrate the efficiency and flexibility of our framework.

The remainder of the paper is organized as follows. First, we review existing literature and previously developed capturing tools in Section II. Then we present our new framework along with its architecture in Section III. In Section IV, we describe examples of experimental analysis of real-world Twitter data acquired and analyzed by our framework. Finally, in Section V, we conclude our work and set goals for future research and development.

II. Related Work

The popularity of Twitter as a research platform has led to the development of a significant number of tools capable of capturing and analyzing Twitter data in the last decade. An overview over earlier tools is presented by Gaffney and Pushmann [7]. However, most of these systems, such as [8], [9] use the Twitter Streaming API, which provides tweet contents in real time while they are being posted. This has the significant disadvantage of requiring researchers to select their area of interest beforehand. Thus, it is essentially impossible to look at events or people which turn out to be significant in hindsight. Some larger companies use the so-called *firehose* bandwidth of the Streaming API, which allows them to retrieve and store the entire content that is posted on Twitter. However, doing so requires an immense infrastructure which is neither feasible to operate nor affordable for most academic users. Furthermore, some tools [10] are no longer compatible with Twitter's terms of service after they changed significantly in 2018 [15]. Finally, in 2018, the U.S. Library of Congress withdrew from its earlier policy of collecting the complete Twitter archive [16].

Because the extraction of information on trends and opinions has significant commercial value, there is a sizable number of commercial tools and services for doing so. Unlike research tools, these programs typically focus on ease of use for a well-established set of analytics [11]. Our framework on the other hand is focused on allowing highly flexible querying. Other research focused tools such as *IndexedHBase* [12] and *DMI-TCAT* [13] follow similar goals.

The original paper for *DMI-TCAT* [13] provides a detailed description of the underlying motivations and assumptions made in the development of that project. The authors' key observations are as follows: (a) only access to social media data by independent researchers guarantees reproducible, independent science; (b) the design choices made in such an analysis tool influence the science that is based on it, and as a consequence; (c) the tool should stay as close as possible to the raw data rather than selecting aggregates. We wholeheartedly agree with these ideas. However, we also believe that previous work has focused more on contents than on network structure. Therefore, keeping in line with Observation b, we aim to enable network structure analysis in addition to content analysis which current tools already perform reasonably well.

We are particularly interested in the comparative analysis of Twitter network structures in the European countries. As of now, most tools for contents or sentiment analysis are geared towards the English language, and multilingual analysis still suffers from significant deficiencies [17], [18]. However, analyzing the structure of these networks (graphs in the mathematical sense) allows language-independent social networks analysis, which is helpful in Europe due to the large number of actively spoken languages compared to America. Thus, we are interested in extracting the underlying networks between users.

In addition, we aim to modernize the software infrastructure. Recent advances in software technologies allow the creation of a highly flexible architecture that can easily incorporate additional tools. As a consequence, unlike *DMI-TCAT*, our framework can easily be extended to distributed computing. Because of the focus on graph analytics, our framework outputs Twitter data directly into a graph database. Also, because of its popularity, we chose *Neo4J* [19] for our current implementation although this can be changed in the source code with moderate effort.

DMI-TCAT provided data enrichment by adding *Klout* scores, a proprietary service aimed at measuring influence across multiple social networks. *Klout* has come under heavy criticism in the past [20], [21]. The service was deactivated in 2018 and is thus no longer available [22]. As similar events can happen again, we designed our framework for easy integration with new external tools. This allows us to easily swap out services that have become defunct or obsolete.

As an example for data enrichment using third party sources, we have added support for obtaining *Botometer* [23] scores of Twitter users. These scores assess the likelihood that a given account is in fact controlled by software (that is, a *social bot*) [24]. However, *Botometer* and research that is based on it [25] is not without criticism of its own [26], [27]. Our framework can be used to investigate the results, and it can easily integrate a different tool for the purpose of comparison.

III. Architecture

FACT is a framework for capturing and analysing massive amounts of graph structured Twitter data in multi-experiment environments. It is designed to support long-term operations as well as execution of simultaneous experiments. Here, an experiment is a piece of user-case specific code prescribing the data to be captured and the analysis to be run on the captured data. Because our system's architecture follows a service-oriented principle, each experiment constitutes a component provided to other components through a network. This means not only that experiments can be attached and detached from the framework at run-time, but also that the resulting system can be extended by many other components such as graph or time-series databases at low development cost. This ensures maintainability and access to state-of-the-art technology.

Since providing capture functionality for Twitter is the cornerstone of the system, Twitter's API restrictions are the limiting factor for performance and throughput. Thus, have chosen the concept of a request management service in form of a Twitter API proxy as a fundamental building principle for our framework. Besides the Twitter proxy and the experiments which are explained in more detail in Sections III-C and III-E respectively, there are two additional categories of FACT components. A storage component offers persistent data storage, while an analysis component represents an interface that interact with experiments and storage, which is described in Section III-D. All the components are connected in one framework to form an experiment setup at runtime. An experimental setup usually consists of one or more experiments connected to a storage component. This is controlled by an analysis interface. Figure 1 shows the experimental setup which was used to analyse the Norwegian newspaper Twitter graph. The results of this analysis are described in detail in Section IV-C.

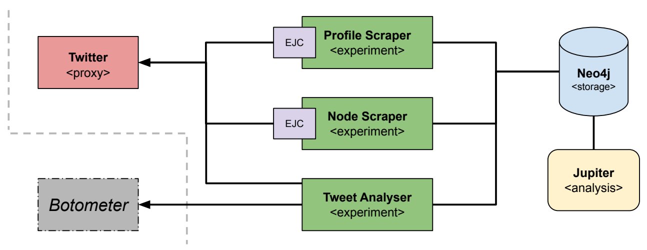

Fig. 1: The structure of the experimental setup used to analyze the Norwegian newspaper Twitter graph. There are three active experiments. One scrapes the follower graphs of the Norwegian newspapers and stores an id for each visited node in the graph database. Another requests the corresponding profile data for each node from the database. The third looks at the last 200 tweets for each profile, analyzes the language, and determines the Botometer score.

A. *Twitter API*

FACT's main data capturing component provides Twitter proxy functionality that forwards requests to Twitter's application programming interface (T-API). T-API is only accessible to registered users who have been granted a Twitter developer account. Although the eligibility criteria have tightened in the recent years [15], Twitter still issues developer accounts for scientific projects, as long as they submit a comprehensive description of how the T-API will be used. Developer accounts are entitled to create Twitter developer apps. Only these provide the credentials in form of the API-key and secret, which have to be attached to the requests in order to authorize them. Moreover, Twitter users can authorize Twitter developer apps with a specific sets of permissions. By doing so, developers, in exchange, receive OAuth-tokens, which, along with the API key, authorize them to perform requests on behalf of the users.

A variety of T-API endpoints grouped to *Tweets and Users*, *Filtered Stream* and *Recent Search* groups are offered by Twitter. Within the scope of this work, we focus on the *Tweets and Users* endpoints, since they are the most relevant for graph structured data capture and analysis. In order to prevent large numbers of incoming requests from overwhelming the Twitter infrastructure, access to T-API is limited by assigning a request quota for each separate endpoint to each user within a time window of 15 minutes.

Despite the high efficiency of the provided native T-API requests, in many cases it is necessary to combine several requests and corresponding API endpoints to perform advanced search and complex data aggregation. An example of such a combined query could be a language detection task on tweet contents in order to identify the native (or most commonly used) language of the user. This is an important task for many text-based analyses, since the language setting in many profiles is set to English, no matter what language the users speak. Now, if we want to capture data from a specific language area, analyzing the dominant language of user tweets can help us identifying its boundaries. We will show an example of using this method in Section IV-C. For this analysis, a complex query has to be constructed and executed that performs language detection on the tweets for each enumerated relevant user profile. This requires a combination of multiple different endpoints for one particular investigation in one search job.

Furthermore, in practical experimental setups for multi-purpose research investigations, it is common for a team of researchers to run several experiments performing different complex queries at the same time. Hence, the challenge is to divide the contingents and corresponding application subscriptions of all the participants among each other in such a way that the experiments are completed as quickly as possible while providing a fair round-robin API request sharing policy among all the users. This means that a user who, e.g., evaluates the language of tweets, lends his quota for the analysis of user profiles to his colleagues and vice versa. Implementation of this fair-share functionality requires development of a specialized intermediate layer to wrap and dispatch T-API calls.

Twitter has set a number of general API access limitations which make building an efficient, large-scale Twitter data analysis framework difficult. The most important factor is that Twitter not only limits the number of queries per endpoint within a fixed time window, but also applies this limit per-user depending on the previous requests history. This is a strict boundary that limits the number of requests per one API-associated user account.

Additionally, Twitter performs logging and complex IP- and geo-location-based user account activity analysis, denying massive API requests from a single physical server, even if they are executed from different legal user accounts. Therefore, building an efficient, parallelizable, and robust framework for

Twitter graph capture and analysis requires not only a T-API-proxy which allows management of all available API requests globally, but also a comprehensive, crowd-based solution that allows cross-user automatic T-API collaboration, requests dispatching, and balancing. This is especially important if an experiment can take days or weeks to perform.

B. Crowd Based Collaboration

The way Twitter restricts access to its API leaves enough room for a centralised crowd-based quota management. As mentioned in Section III-A, authorising a Twitter app empowers developers to perform user requests on their behalf. On the one hand, we use this characteristic in order to give interested people an opportunity to support our research and, on the other hand, to manage internal request quotas in a less centralised and, therefore, more distributed manner.

To archive this, it is necessary to obtain a Twitter developer account. We launched a project website on which Twitter users can support our work on Fake News Detection by donating their quotas. As shown in Figure 2, a user authorizes the experiment application. After this, we receive their OAuth-token and store an encrypted version of it in our back-end database. The incoming OAuth-tokens were pulled and made accessible to the token storage of our proxy on a daily basis.

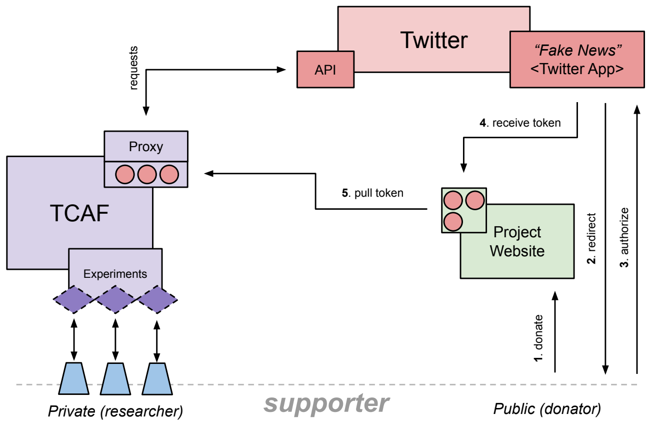

Fig. 2: The diagram of interaction within the system's environment including Twitter, the project website, and the participants. Steps from one to five describe the process of donating a token. After pressing the donate button on our project website (1), an OAuth authorization is initiated. The donor is first forwarded to Twitter (2) where they authorize our "Fake News App" with read-only authorization (3). Then our web server obtains an OAuth-token (4) which is finally collected by the proxy (5).

C. Proxy

The proxy component is the most important actor in our system. It implements the token sharing logic in a transparent and application-friendly way. It was designed to provide the Twitter-compatible external API in a way that ensures the framework in which users will execute proxy calls using the same syntax and arguments as if they were connecting to Twitter directly. This has the additional advantage that proxied requests for already developed Twitter clients such as Twitter4j, Tweepy, etc. are possible. Our proxy hides all the complex token share logic and makes individual applications less complex, since the query logic is separated from the application, which would otherwise have to manage resources or handle API call over-utilization errors in some way. An additional advantage of the proxy is request caching. In case of repeatable user requests within a short time frame that are expected to give the same results, the proxy uses cached responses, thereby significantly reducing request quota usage and increasing performance.

Besides its task to forward the requests to the T-API, the proxy contains a mechanism for quota leasing. The quota leasing manages contingents of the participating scientists for the respective endpoints of the T-API in the form of OAuth-tokens. This encapsulation ensures that free resources are balanced across all request-consuming components. Since the resource "Requests" are managed in a central component, all other components that require access to the T-API can be designed under the assumption that there is no limit on requests. The proxy consists of sub-modules, namely the connection context, endpoint information, and context manager, which are described in more detail below.

1) Connection Context: The connection context implements a wrapper for the T-API and manages a connection to it for each available OAuth-token. The wrapper builds on top of the Twitter4J client, which contains an important data structure that manages statistics on remaining requests per endpoint and time window. We use this information in other proxy sub-modules to maintain the quota. In addition, the wrapper performs the establishment and closing of T-API connections, and it manages the endpoint info data structure for each endpoint.

2) Endpoint Information: The endpoint information structure stores the following data for each corresponding endpoint of the T-API:

- Length of time window until restock.
- Number of results per response.
- Number of results per time window.
- Number of possible requests per time window.
- Timestamp of the last request. It is used to calculate when the quota will be restocked.
- Number of used requests.

The complexity of this structure reflects the complexity of the API access limitations applied by Twitter. The information stored in this structure allows us to monitor the endpoint usage in a detailed manner and prevent possible endpoint blocking due to over-utilization beyond free subscription limits. This endpoint information is used in the context manager.

3) Context Manager: The primary task of the context manager is to receive requests and forward them to a suitable connection context. We use a straightforward greedy strategy for request forwarding and context manager selection. If a manager accepts a request, he iterates through the list of known connection managers until an alive and free connection manager is found. When searching for a suitable manager, we use the request timestamps associated with each and every endpoint. The timestamps are updated while performing a request using the endpoint. Together with the associated request counters, it allows for simple and efficient quota control implementation. If the quota is reached, endpoint usage is paused and the required waiting period is computed. When the waiting period has elapsed, the corresponding quotas are renewed, and the endpoint becomes available again. To prevent races within our connection manager system, we use a simple *one client at a time* strategy. If a particular connection manager is selected, it becomes blocked from reselection for the duration of the request that is being executed. This ensures that no connection manager can be called by more than one client simultaneously.

D. Storage

Storage is a container intended to provide data persistence across the framework which allows the integration of various types of databases and file storages. The current version our framework offers storage built on the basis of the Neo4J [19] graph database. We selected Neo4J due to its extensive functionality for graph processing using a declarative graph query language called *Cypher* [28]. All the storage components are passive. They are always connected to one or more experiments that make requests or store intermediate and output data.

E. Experiment wrapper

Experiment wrappers are active components that are connected to storage as well as to the proxy. One or more experiment wrappers form an experimental setup for both data collection and analytical tasks (see an example in Section IV). According to our framework usage template, the user-defined payload of the experiment consist of one or more pieces of user code which could perform the aggregation of intermediate results, collection of Twitter data, and communication with additional third party services, e.g., the Botometer bot detection API or Google's natural language processing API.

1) Library Extension: In order to execute the integration of user-defined analytical tasks with our framework, we implemented the special extension for the existing Twitter4J Java-language client. We added the high-level functional layer that makes the Twitter API restrictions transparent for the high-level logic developers. This is necessary because the proxy is able to hide request limitations to a certain degree, but it is no appropriate abstraction for reflecting that on a programmers level. This leads to boilerplate code and increases complexity and, therefore, development time. Thus, the developed Twitter4J extension is capable of hiding primary resource limits, making not only the execution of complex queries possible, e.g., *find all tweets of user X in language Y*, but also performing graph queries, e.g., *capture the follower network of user X within distance two*.

IV. Example Experiments

In this section we perform a series of example investigations. They are selected to be small enough to be visualized in this paper. The goal is to demonstrate how social networks can be gathered for analysis with limited technical effort. However, the framework is by no means limited to small experiments. Larger experiments do not require additional technical knowledge, but they do require more time.

A. Single Node Follower Analysis

An example of a typical use case is the analysis of the followers of a single prominent Twitter account. Of particular interest are the accounts of politicians. A commonly held belief is the idea that the number of followers represents a heuristic measurement for a politician's influence. As a consequence, there is significant discussion about *"fake followers"*, a term that is usually intended to describe following accounts that do not correspond to real-world persons or institutions, and which are typically created with the explicit goal of inflating the perceived importance of a given account. There are specialized services for detecting such fake followers, e.g. *Twitteraudit* [29], which analyzes accounts by sampling a number of followers and assigning them a quality score based on the details and posts of each follower. For each analyzed account, it returns the number and ratio of followers it believes to be *fake*.

less than	Followers	Followers %	Tweets	Tweets %
1	236185	25.72	384601	41.89
2	366241	39.89	492779	53.67
5	561160	61.12	596053	64.92
10	668775	72.84	651634	70.97
100	827490	90.13	773068	84.2
1000	900186	98.04	857001	93.34
10000	915805	99.74	905070	98.57

Table I: The total and percentage numbers of followers of *@regsprecher* having fewer than the listed number of followers (Column 2 and 3) and tweets (Column 4 and 5)

As an example, we study the followers of the account of Steffen Seibert (*@regsprecher*), the German government's spokesperson. We chose this account because it has one of the highest numbers of followers among political actors in Germany (more than 900,000), and *Twitteraudit* reports an unusually high ratio of fake followers of 60%. Here we study the number of followers that show very little activity on Twitter. The rationale for this lies in our belief that it is very difficult to automatically discriminate fake accounts from inactive accounts because such accounts lack the data to tell those two groups apart in a meaningful way.

We use our framework to capture the follower data of *@regsprecher* and count how many other accounts the followers are followed by. We also obtain the number of times each follower has tweeted. Table I shows the results. We observe that only about 39% of the followers are followed by five or more accounts, and that less than half of the followers have posted more than once.

We also take a look at the combined counts in Table II. We observe that about 18% have no followers and no tweets, and that 60% have either at most one tweet or follower. Clearly, such accounts can be considered as inactive. Systems such as *Twitteraudit* tend to assign low trust scores to such accounts, and ultimately label them as *fake*. However, due to the lack of data it is essentially impossible to prove whether such accounts are "fake" or "real". Thus, the results of automated fake follower detection depends massively on whether a system assumes followers to be "fake" until proven "real", or the opposite.

We hypothesize that older accounts with a large number of followers tend to attract inactive followers over time, giving them a high ratio of inactive followers. For gauging influence, it might be preferable to disregard such followers. However, We cannot infer from this whether such accounts are being used to read tweets frequently. In that case, even an inactive follower would contribute to the influence of a Twitter account.

less than	OR	OR %	AND	AND %
1	455784	49.64	165002	17.97
2	567354	61.79	291666	31.77
5	670771	73.06	486442	52.98
10	725237	78.99	595172	64.82
100	839353	91.42	761205	82.91
1000	902575	98.3	854612	93.08
10000	916809	99.85	904066	98.47

Table II: The total and percentage numbers of followers of *@regsprecher* having fewer than the listed number of followers *or* tweets (Column 2 and 3) and those having fewer than the listed number of followers *and* tweets (Column 4 and 5)

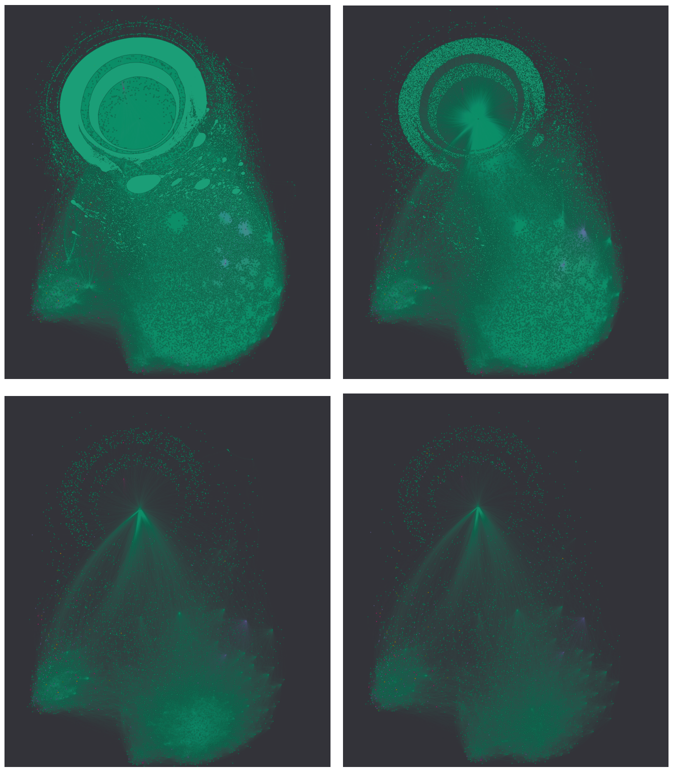
Fig. 3: Top left: the entire follower graph. Czech prime minister Andrej Babiš is in the center of the large circle near the top. Top right: sparsified graph after removing all accounts with two or fewer tweets. Bottom left: sparsified graph by removing all accounts with less than 100 followers. Bottom right: only accounts with at least 1000 tweets remain.

B. Single Node Network Analysis

As the next example, we study the network surrounding a single account and visualize it. The goal of such an analysis is to understand the position of the source account (i.e. the initial account) within its surrounding network. If the connections between other vertices (i.e. Twitter accounts) in the network are sparse, then the source account ties the network together, indicating high centrality. On the other hand, if they are dense, the source account has no special position within the network. For a given graph, there are established methods such as betweenness centrality [30] for computing similar information. However, it is not clear which network should be the basis of the computation, and even if it were, collecting all that information from Twitter and computing its betweenness centrality might be prohibitively expensive. Thus, investigating the network of a single account is much faster, and likely more meaningful w.r.t. that account.

We analyze the follower network of Andrej Babiš, the current prime minister of the Czech Republic (since December 2017), and one of the country's most followed Twitter account with more than 375,000 followers with an increase of about 30 followers per day according to *Socialbakers* [31]. We capture all vertices (i.e. accounts) in its neighborhood, and all edges (i.e. follower relationships) between these vertices. Doing so means that we have to obtain and check the entire neighborhood of each follower, which can be quite time consuming. The resulting follower network is visualized in Figure 3. Similar to Section IV-A, the large number of followers suggests that a significant fraction of them will be inactive. Such accounts say very little about the network structure. Thus, we first remove all accounts with eight or fewer tweets. This removes several dense clusters in the center of the graph, indicating that they consisted mostly of accounts with little activity. For a visual analysis, we need to reduce the the number of vertices further. Thus, we remove all accounts having 100 or less followers. We notice in Figure 3 that the ring around Babiš in the upper part of the pictures is now mostly gone. This ring contains mostly accounts that follow only the source account and no other accounts in the network. Thus, removing accounts with few followers is likely to remove most of them. Finally, we restrict ourselves to highly active users with more than 1000 tweets. Interestingly, this causes little change in the visualization. At the bottom of the image, we clearly recognize two stable clusters which would remain tightly connected even if the source vertex is removed. Thus, the centrality of the source vertex is somewhat lower than expected. A natural next step would be to find out which accounts form these clusters. However, such an analysis would be beyond the scope of this paper[2] as our primary aim is to demonstrate the capabilities of our framework.

C. Multi Source Network Analysis

Our third example demonstrates the capture and visualization of a country's newspaper Twitter sphere. The country in this example is Norway. To capture the graph, we select the Twitter accounts of the major Norwegian newspapers as source vertices and investigate their neighborhoods. The crucial question is to limit the capture to the Norwegian Twitter sphere.

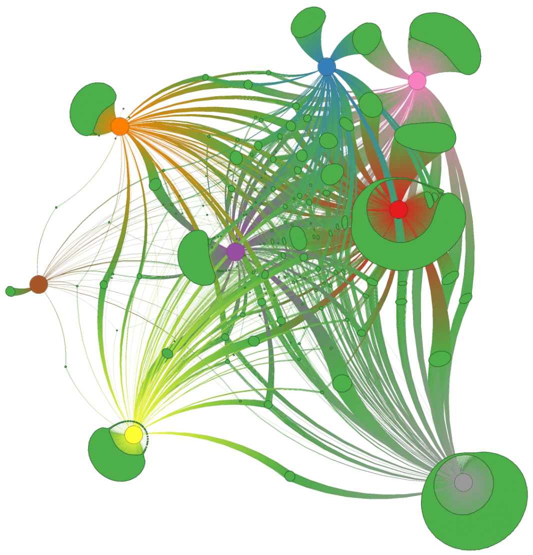
Fig. 4: Visualization of the Norwegian newspaper Twitter sphere around news websites. News outlets are color coded: red: VG, pink: Aftenposten, purple: Dagebladet, yellow: Aftenbladet, grey: Bergens Tidende, blue: Dagens Næringslivet, and orange: Adresseavisen. Brown is a right-wing fringe website (sian.no) introduced as control.

There is no way of detecting the true location of a user from Twitter data, since many users do not set their location. And even if we could detect the location, doing so would not be desirable since we have to consider some users (e.g. expats, travellers) that are physically located outside of Norway as part of the Norwegian newspaper Twitter sphere. A better criterion would be to draw boundaries by language. However, a large number of users do not change their language settings or set it to English on purpose. Thus, we instead add a module for language detection. Norwegian is not commonly spoken outside of Norway, but English is often used in Norway. Thus, we consider an account to be part of the Norwegian newspaper Twitter sphere if it has posted at least 50% of its previous 200 Tweets in Norwegian[3]. This has the side effect of filtering out inactive users, but since they do not actively take part in the conversations on Twitter, doing so is desirable. Note that this method will not be sufficient for countries that do not have a unique national language. The graph collected in this manner has almost 100,000 vertices. Since we need to check for overlapping neighborhoods, collecting this graph relatively costly. We also record the Botometer [23] scores for each account in the graph. About 33% of the accounts have a score of 4 out of 5, i.e. Botometer considers them likely to be bots. However, their distribution in the graph does not seem to follow a specific pattern. We therefore do not include the scores in the visualization.

[2]Note that finding the names of the accounts poses no technical problem.
[3]We do not differentiate between the variants of the Norwegian language here.

The result is shown in Figure 4. Each newspaper has a substantial group of accounts following only itself (within this network). The key information is contained in the clusters of accounts that follow two newspapers. Three newspapers from Oslo (*Aftenposten*,*VG*, and *Dagens Næringslivet*) have large overlap. Consequently the visualization algorithm places them close to each other. Regional newspaper such as *Aftenbladet* from Stavanger and *Adresseavisen* from Trondheim are much further away and have small overlap with the capital newspaper. *Bergens Tidende*, the largest of the regional newspapers, also has the largest cluster of exclusive followers and relatively small overlapping clusters. Thus, it is possible to infer a significant amount of plausible information about newspaper readers from this visualization.

We also observe that the website *sian.no*, while having a sizeable group of exclusive followers, has very weak connection to the major newspapers, underlining its fringe status. Disconnected communities often form so called *echo chambers*. It is widely believed that these are a breeding ground for fake news. As a result, the detection of such disconnected communities is crucial for the study of misinformation.

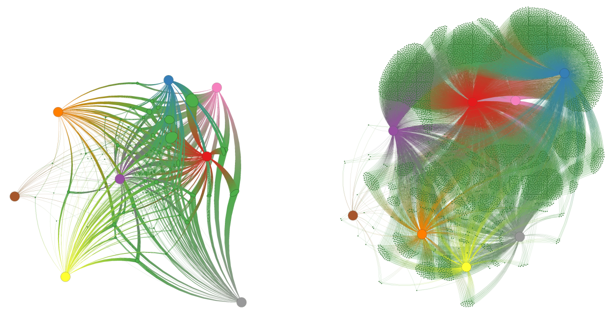
Fig. 5: Alternative visualizations of the Norwegian newspaper Twitter sphere around news websites. Left: a version of Figure 4 with the exclusive followers filtered. Right: the same graph rendered again, without the exclusive followers. (color coding as in Figure 4)

Alternative versions of this visualization are shown in Figure 5. They are created by filtering out exclusive followers (left), and by rendering a new visualization of the graph without the exclusive followers (right). In this new visualization, we observe that among the Oslo newspapers, *Dagebladet* has the strongest connection to the regional newspapers, while *Dagens Næringslivet*, a business newspaper, has the weakest.

V. Conclusions and Future Work

Research on tools to combat misinformation has lead to the development of detection software for social bots. This however has led to the a misinformation problem of its own. While researchers understand the limitations of such detection systems and the fact that there are many ways to define *social bots*, the general public does not. In Section IV-A, the statement made by Twitteraudit would most likely be understood as: "60% of the government spokesperson's followers are *fake*" which, in a political context makes it seem like so few people are interested in what the government has to say that they feel the need to inflate their perceived influence. Likewise, in a political discussion, a statement such as "50% of the comments supporting Position A are made by *social bots*" makes it seem like Position A has little backing among the population and may be the result of foreign influence. A similar case happened in Germany in late 2018 [32].

The investigation of these problems is further complicated by diverging interests which can consciously or unconsciously influence decisions. Twitter has an interest in being a valid political discussion platform. Researchers would like their tools to be relevant (especially if the tool is commercial), and politicians have strong incentives to use all results that support their positions. As a consequence, we believe that a broad discussion, especially among political and social scientists, is necessary. Coming predominantly from computer science, our contribution is to provide a software framework that allows researchers to gather the raw data, test the assessments of automatic tools, and verify results in a reproducible way.

Therefore, we have created a framework that allows formulation of complex queries to the Twitter API using a crowd-based approach in order to increase the volume of data that can be queried. Furthermore, our design consists of strongly tiered building blocks which allows to solve problems on the appropriate level of abstraction.

Our natural next step is to expand the capabilities of our framework according to the needs of social scientists. There are three primary directions for doing so. First, we will implement the required code to gather data on a specific topic or hashtag and investigate the structure of retweets. The Twitter API does not provide any information about the way that shared messages, i.e. retweets, take through the network. We aim to develop components executing stochastic methods based on follower networks analysis which allow us to compute the most likely retweet path that a tweet might have taken. We deem this functionality to be particularly beneficial in the context of identifying fake news. A second direction is to enable long-term experiments. By periodically querying for changes in a follower network, we can observe how shifts in political stances can affect social networks. Furthermore, we believe that this information can be very helpful for identifying fake followers by detecting unusual changes in the social network. The third direction is more of a technical nature. We will implement the proxy, which allows using any Twitter client to write experiments within our framework, thereby expanding its capabilities dramatically. This component is currently under development. We expect it to be freely available at the time of publication of this paper.

References

[1] D. Lazer, A. Pentland, L. Adamic, S. Aral, A.-L. Barabási, D. Brewer, N. Christakis, N. Contractor, J. Fowler, M. Gutmann *et al.*, "Computational social science," *Science*, vol. 323, no. 5915, pp. 721–723, 2009.

[2] ZeroHedge, http://bit.ly/31YQlz7, 2012.

[3] T. Guardian, http://bit.ly/2JaL9zq, 2012.

[4] T. W. S. Journal, https://on.wsj.com/2KNkJXR, 2012.

[5] T. N. N. R. E. Committees, "Guidelines for research ethics in the social sciences, humanities, law and theology," http://bit.ly/2LzqWG6, 2016.

[6] W. M. Campbell, C. K. Dagli, and C. J. Weinstein, "Social network analysis with content and graphs," *Lincoln Laboratory Journal*, vol. 20, no. 1, pp. 61–81, 2013.

[7] K. Weller, A. Bruns, J. Burgess, M. Mahrt, and C. Puschmann, *Twitter and society*. Peter Lang, 2014, vol. 89.

[8] S. Gilbert, https://github.com/WebEcologyProject/140kit, 2010.

[9] P. Barbera, https://cran.r-project.org/web/packages/streamR/, 2018.

[10] https://github.com/540co/yourTwapperKeeper, 2011.

[11] http://twitonomy.com/, 2019.

[12] X. Gao and J. Qiu, "Supporting queries and analyses of large-scale social media data with customizable and scalable indexing techniques over nosql databases," in *2014 14th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing*, May 2014, pp. 587–590.

[13] E. Borra and B. Rieder, "Programmed method: developing a toolset for capturing and analyzing tweets," *Aslib Journal of Information Management*, vol. 66, no. 3, pp. 262–278, 2014.

[14] T. Kluyver, B. Ragan-Kelley, F. Pérez, B. Granger, M. Bussonnier, J. Frederic, K. Kelley, J. Hamrick, J. Grout, S. Corlay, P. Ivanov, D. Avila, S. Abdalla, C. Willing, and J. development team, "Jupyter notebooks ? a publishing format for reproducible computational workflows," in *Positioning and Power in Academic Publishing: Players, Agents and Agendas*, F. Loizides and B. Scmidt, Eds. IOS Press, 2016, pp. 87–90.

[15] Y. Roth and R. Johnson, http://bit.ly/2XD5Z3Q, 2018.

[16] L. of Congress, http://bit.ly/2RNvgCX, 2017.

[17] S. L. Lo, E. Cambria, R. Chiong, and D. Cornforth, "Multilingual sentiment analysis: from formal to informal and scarce resource languages," *Artificial Intelligence Review*, vol. 48, no. 4, pp. 499–527, Dec 2017.

[18] K. Dashtipour, S. Poria, A. Hussain, E. Cambria, A. Y. Hawalah, A. Gelbukh, and Q. Zhou, "Multilingual sentiment analysis: state of the art and independent comparison of techniques," *Cognitive computation*, vol. 8, no. 4, pp. 757–771, 2016.

[19] J. Webber, "A programmatic introduction to neo4j," in *Proceedings of the 3rd annual conference on Systems, programming, and applications: software for humanity*. ACM, 2012, pp. 217–218.

[20] J. Scalzi, https://cnnmon.ie/3006TF4, 2011.

[21] C. Stross, http://bit.ly/2XDQvMS, 2011.

[22] L. Technologies, https://www.lithium.com/products/klout, 2018.

[23] K.-C. Yang, O. Varol, C. A. Davis, E. Ferrara, A. Flammini, and F. Menczer, "Arming the public with artificial intelligence to counter social bots," *Human Behavior and Emerging Technologies*, vol. 0, no. 0, p. e115, 2019. [Online]. Available: http://bit.ly/2Xg6Xyu

[24] Z. Chu, S. Gianvecchio, H. Wang, and S. Jajodia, "Who is tweeting on twitter: human, bot, or cyborg?" in *Proceedings of the 26th annual computer security applications conference*. ACM, 2010, pp. 21–30.

[25] T. R. Keller and U. Klinger, "Social bots in election campaigns: Theoretical, empirical, and methodological implications," *Political Communication*, vol. 36, no. 1, pp. 171–189, 2019.

[26] M. Kreil, http://bit.ly/2XDQvMS, 2018.

[27] F. Gallwitz and M. Kreil, http://bit.ly/2RItkM2, 2019.

[28] N. Francis, A. Green, P. Guagliardo, L. Libkin, T. Lindaaker, V. Marsault, S. Plantikow, M. Rydberg, P. Selmer, and A. Taylor, "Cypher: An evolving query language for property graphs," in *Proceedings of the 2018 International Conference on Management of Data*. ACM, 2018, pp. 1433–1445.

[29] D. Caplan and D. Gross, https://www.twitteraudit.com/, 2012.

[30] M. O. Jackson, *Social and economic networks*. Princeton university press, 2010.

[31] www.socialbakers.com, 2019.

[32] J. Hermann, http://bit.ly/2RL8wmZ, 2018.