Open-Source Monitoring, Search and Analytics Over Social Media

Conference Paper *in* Lecture Notes in Computer Science · November 2017

DOI: 10.1007/978-3-319-70284-1_28

5 authors, including:

Manos Schinas
Information Technologies Institute (ITI)
17 PUBLICATIONS 143 CITATIONS
SEE PROFILE

Symeon Papadopoulos
The Centre for Research and Technology, Hellas
256 PUBLICATIONS 4,719 CITATIONS
SEE PROFILE

Ioannis (Yiannis) Kompatsiaris
The Centre for Research and Technology, Hellas
1,023 PUBLICATIONS 14,027 CITATIONS
SEE PROFILE

Pericles A. Mitkas
Aristotle University of Thessaloniki
320 PUBLICATIONS 3,022 CITATIONS
SEE PROFILE

Open-source Monitoring, Search and Analytics over Social Media

Manos Schinas[1,2], Symeon Papadopoulos[1], Lazaros Apostolidis[1], Yiannis Kompatsiaris[1], and Pericles A. Mitkas[2]

[1] Information Technologies Institute,
Centre for Research and Technology Hellas, Thessaloniki, Greece
{manosetro,papadop,laaposto,ikom}@iti.gr
[2] Electrical and Computer Engineering Department,
Aristotle University of Thessaloniki, Greece
mitkas@auth.gr

Abstract. The paper describes a technical demonstration of an open-source framework for monitoring, analysis and search over multiple social media platforms. The framework is intended to be a valuable tool for media intelligence professionals, as well as a framework and testbed for scientists and developers with interest in social media research.

Keywords: social media, data collection, data mining, information retrieval, stream processing, information visualization

1 Introduction

Social media [1] is nowadays a key channel for communication, self-expression, information gathering and entertainment for billions of Internet users worldwide. The penetration of social media has consistently increased throughout the last decade and it has nowadays reached a point where the large majority of people worldwide regularly use one or more of numerous popular platforms for a wide variety of purposes. As a result, social media is often regarded as a sensor of real-world trends and events [2] and as an indispensable tool for performing social science research [3] and consumer intelligence gathering and marketing [4].

Existing social media research is typically based on one of two approaches depending on the setting where it is carried out. In *academic* settings, researchers typically use a variety of tools, scripts or libraries, and often develop their own additional custom scripts to perform the required data collection, manipulation and analysis. In *business* settings, analysts use a variety of software-as-a-service products, typically through a dashboard-like interface, offering access to statistics and analytics about a query of interest. The main disadvantage of the first approach is the increased effort that is necessary to set up the data collection and analysis pipeline. In contrast, the second approach suffers from limited flexibility, cost (most social media SaaS offerings are subscription-based) and dependence on proprietary solutions.

To address these limitations, we present an integrated open-source framework for monitoring, analyzing and retrieving social media content. The framework is easy to install locally and can then be managed as a set of services that expose data collection, indexing and retrieval capabilities via a REST API and a web-based user interface. In addition, the framework is designed in a modular way, which makes it straightforward to adapt and extend it for different user requirements, e.g. collect data from additional social media platforms, filter incoming content, compute additional metrics, etc. To our knowledge, the proposed framework is the only integrated open-source solution for social media monitoring, analytics and search, which is freely available and offers at the same time ease-of-use, rich off-the-shelf features, flexibility and extensibility.

The development of the presented framework started in SocialSensor[3] [5], where a need came up to retrieve multimedia content around trending topics that were automatically detected by the news monitoring application developed within the project [6]. The development of the tool continued in REVEAL[4], where it was combined with a focused web crawler [7] and an event summarization module [8]. Finally, the current version presented in this paper is based on extensions made to support the social media monitoring needs of the STEP[5] and hackAIR[6] research projects.

2 Design and Implementation

The open-source framework presented in this work is built upon a set of projects. The core project[7], implemented in Java, contains all the classes corresponding to a common data model shared across different social media (Figure 1). Social media abstractions[8] implements the data model for each of the supported social media platforms, e.g. Twitter, Facebook, etc. As each platform has its own way to represent data, we need to map different representations to a common model. Also, this project contains a set of wrappers that encapsulate the different APIs provided by the platforms for the collection of data. In that way, access to each supported source is made using methods with the same signature. The client project[9] contains a set of classes for data management, including wrappers around different storage solutions, such as MongoDB and Apache Solr. Finally, the Stream Manager project[10] incorporates all the orchestration and operational logic for the collection and storage of data from different social media platforms, according to the specified input.

[3] `http://socialsensor.eu/`
[4] `https://revealproject.eu/`
[5] `http://step4youth.eu/`
[6] `http://www.hackair.eu/`
[7] `https://github.com/MKLab-ITI/mklab-framework-common`
[8] `https://github.com/MKLab-ITI/mklab-socialmedia-abstractions`
[9] `https://github.com/MKLab-ITI/mklab-framework-client`
[10] `https://github.com/MKLab-ITI/mklab-stream-manager`

2.1 Data model

Figure 1 depicts the data model used to store data in the platform. The left side of the Entity-Relationship (ER) diagram contains objects collected by the platform: *items*, *media items*, *web pages* and *users*. Items correspond to the messages posted to the various social media platforms, e.g., statuses on Twitter, posts on Facebook, video posts on YouTube, etc. Media items correspond to the multimedia content that is embedded in these items, for instance, images in tweets, videos in Facebook posts, etc. Web pages correspond to the URLs that are contained in the items. Finally, users are the objects representing the user accounts publishing the items. The right side of the figure depicts the *collection* entity, which offers users of the platform a way to organize social media content. A collection can comprise multiple queries as described in section 2.2, while the same query can be included in more than one collections. The platform supports three types of query: a) keyword queries, b) account queries, and c) location queries. It is important to note that there is no explicit (stored) association between a collection and the collected content (items, media items, etc.), nor is there such association between content and queries. This is depicted in the ER diagram by the dotted relationships between the corresponding entities.

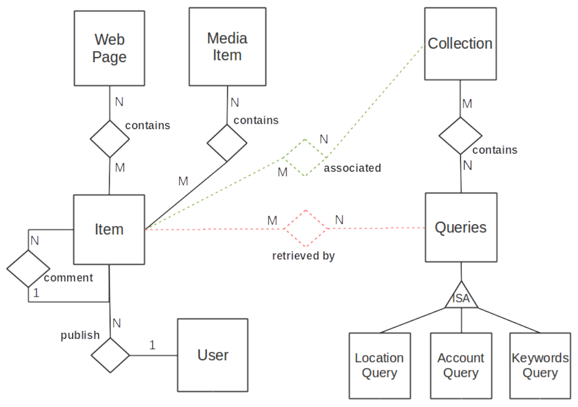


Fig. 1. Entity relationship data model of social media monitoring tool

2.2 Data collection and Processing

The collection of social media data is performed by the Stream Manager module. Currently, five different sources are supported: Twitter, Facebook, YouTube, Google+ and Flickr. The Stream Manager collects posts of interest (as defined by user queries) shared in these platforms alongside the users that published them, the embedded media items and the linked URLs. In addition to social media sources, the platform also supports monitoring RSS feeds.

Data collection by the Stream Manager is configured on the basis of user-defined collections, which, as mentioned in Section 2.1, consist of a set of three types of query (keywords, accounts, locations). Keywords can be defined on the basis of simple keywords, e.g. *"air pollution"*, or more complex logical expressions, e.g. *"election results" AND ("donald trump" OR "hilary clinton")*. Accounts refer to public sources of content for each platform, e.g., Twitter users, Facebook pages, YouTube channels, etc. Locations are represented as bounding boxes defined by pairs of latitude/longitude coordinates. For example, a user of the platform, interested in the US elections, could define a collection based on a set of keywords (e.g., names of candidates combined with state names) and a set of Twitter accounts and Facebook pages that publish content related to elections. It is worth noting that location queries are only supported on Twitter.

Given a collection, the associated queries are translated to the appropriate API calls. For example if a Twitter account such as *@BBC* is specified as one of the queries in the collection, that account is mapped to a call to the Twitter API that is used for the collection of Tweets posted by the specific user. In the same way, given a logical expression of keywords, multiple API calls are generated, one for each of the supported sources. However, given that the tool may be used by several end users, the defined collections might contain identical queries, e.g. the same YouTube channel or the same keywords. To make the query process more efficient, the Stream Manager first de-duplicates all input queries into unique non-redundant query elements. To support continuous monitoring of the user-defined collections, the Stream Manager periodically polls each these unique query elements, and keeps track of the number of submitted requests to each platform in order to respect the limits imposed by it. It is worth noting that the set of Stream Manager queries may change dynamically (e.g., when a user creates or updates a collection), and such changes are efficiently communicated to the Stream Manager through a Redis[11] message broker instance.

The fetched items can then be processed by a sequence of filters and processing modules, executed within the Stream Manager, before being stored and indexed. In terms of filtering, a set of heuristic rules is applied on the input items to keep only items of high quality. For example, items with limited text content, or with too many hashtags and URLs are treated as spam messages and therefore discarded. In terms of processing modules, three indicative processors are provided off-the-shelf by the Stream Manager: language detection (in case no language information is provided from the corresponding source), named entity extraction (e.g. persons and organizations), and MinHash signature extraction based on the text of the item. Finally, the collected content i.e. items, media items, users and web pages are stored in a MongoDB[12] instance.

2.3 Indexing and retrieval

As previously mentioned, there is no explicitly stored association between collections and fetched content. The queries generated for each collection are used

[11] `https://redis.io/`

[12] `https://www.mongodb.com/`

for fetching data from each social media platform, and the collected items (after filtering within the Stream Manager) are stored in the database without any association to the corresponding queries or the collections.

Data indexing is based on Apache Solr[13]. For each item, a subset of its fields are indexed. This includes all the textual fields, e.g. title, and other relevant fields, e.g., publication time, user id, number of views, etc., which can be used for filtering and faceting. To retrieve content related to a specific collection a Solr query is generated based on the queries that are associated with the collection, and the ids of the relevant items are retrieved. Then, the corresponding item metadata and all associated entities are retrieved from MongoDB by id.

In addition to content retrieval, the platform provides analytics per collection (or collection subsets). This includes simple metrics such as the number of items or unique users, to more complex ones such as *reach* (the estimated cumulative audience for a set of items) and *endorsement* (the estimated sum of "likes" for a set of items). Top users, locations, tags and named entities (based on frequency of appearance) are also provided. Finally, the platform supports the generation and visualization of timelines with different time granularities.

To support the aforementioned analytics operations, the same Solr query that is used for the retrieval of content is now used in conjunction with two Solr components: Faceting[14] and Stats[15]. For example, for the generation of the top active users in the collection (in terms of number of posted items), we make a facet request in Solr, using the user id as the field to be treated as a facet. The corresponding response by Solr contains the ton N users along with the number of items per user. In a similar manner, using the Stats component we can calculate field-oriented statistics, e.g. the average number of views or sum of shares per collection. These statistics can be used for the calculation of metrics such as reach and endorsement.

Finally, the analytics framework leverages two more Solr components to make easier the exploration of collections: Result Clustering[16] and Result Collapse[17]. Clustering is used to automatically discover groups of similar search hits, i.e. groups of items related to a specific topic. These identified topics can be used to narrow down the set of items associated to a collection. Collapse, on the other hand, is used to group search results based on the value of a field. In our case, we use collapse based on the MinHash signature of items to support de-duplication of content, i.e. items with the same signature collapse into a single item.

Figure 2 depicts the key software components of the framework along with the function that each performs. Further details regarding their deployment are provided in section 3.3.

[13] `http://lucene.apache.org/solr/`
[14] `https://cwiki.apache.org/confluence/display/solr/Faceting`
[15] `https://cwiki.apache.org/confluence/display/solr/The+Stats+Component`
[16] `https://cwiki.apache.org/confluence/display/solr/Result+Clustering`
[17] `https://cwiki.apache.org/confluence/display/solr/Collapse+and+Expand+Results`

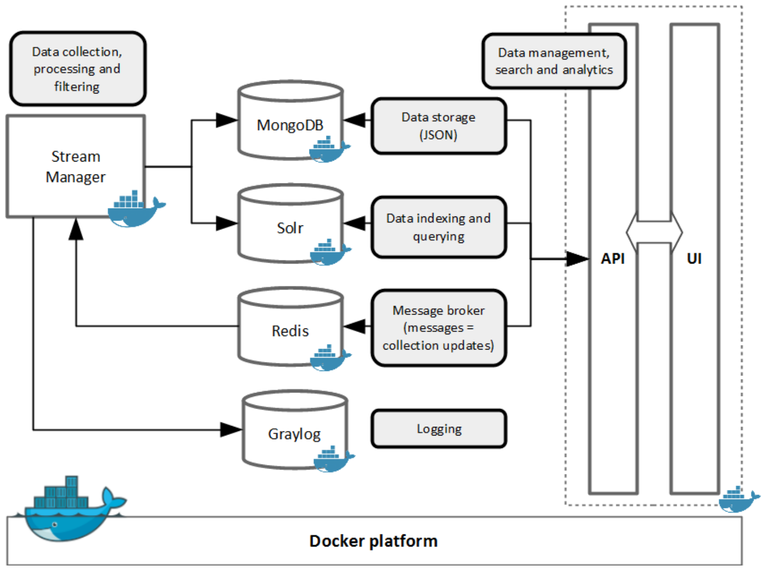

Data collection, processing and filtering
Stream Manager
MongoDB
Solr
Redis
Graylog
Data storage (JSON)
Data indexing and querying
Message broker (messages = collection updates)
Logging
Data management, search and analytics
API
UI
Docker platform

Fig. 2. Overview of framework components (white blocks) and their function (gray).

3 Usage and Deployment

The functionalities of the presented framework can be used by developers through a comprehensive RESTful Application Programming Interface (section 3.1), but it is also possible for non-developers to benefit from its capabilities through a simple to use web-based user interface (section 3.2). In both cases, it is necessary for the framework to be deployed and administered in a selected host environment (section 3.3).

3.1 Application Programming Interface

All the data collection, indexing, aggregation and retrieval capabilities of the framework are exposed to developers through a REST API that is built on PHP and served by an Apache Web server. The methods of the API belong to three categories: a) collection management, b) content retrieval and c) analytics. The first category contains a set of methods to create/update/delete collections and to retrieve collections created by a specific end user. For content retrieval there is the */items/collection-id* method, which used to get the items related to a specific collection. The method has a set of parameters to refine how retrieval takes place, e.g. filtering or sorting. Finally, analytics over a selected collection is exposed through six methods: */statistics*, */timeline*, */users*, */terms*, */countries* and */heatmap/points*. For each of these methods the unique identifier of the collection must be specified. In addition, a set of parameters for the refinement of the collection (e.g. by source or language) is also provided.

3.2 User interface

The developed user interface, built on top of the REST API, consists of three main pages: a) a page where users of the platform can create their own collections according to their information needs (Figure 3), b) a feed view for browsing content (Figure 4), and c) a dashboard view for analytics (Figure 5). In the following, we provide a further description of the feed and dashboard view.

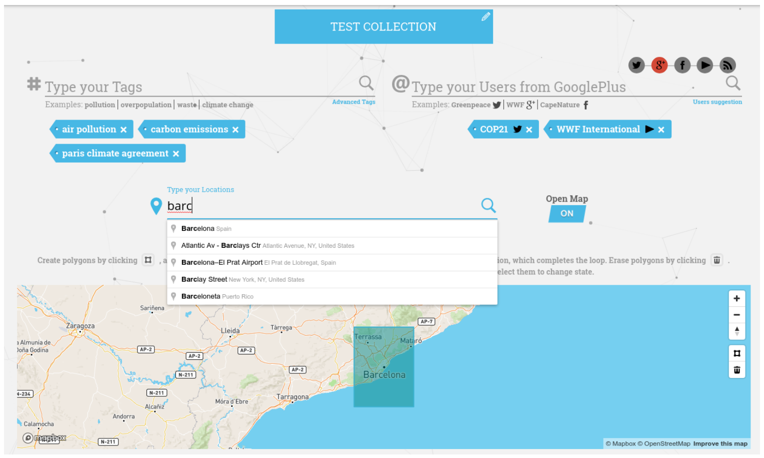
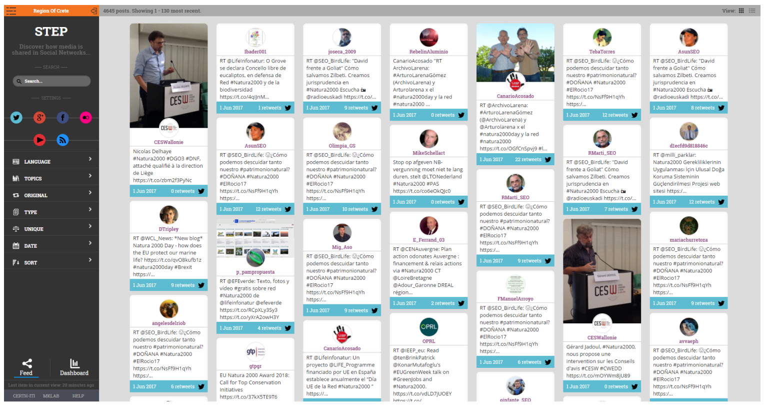

TEST COLLECTION
Type your Tags
Examples: pollution | overpopulation | waste | climate change
Advanced Tags
air pollution
carbon emissions
paris climate agreement
Type your Users from GooglePlus
Examples: Greenpeace | WWF | CapeNature
Users suggestion
COP21
WWF International
Type your Locations
barc
Barcelona Spain
Atlantic Av - Barclays Ctr Atlantic Avenue, NY, United States
Barcelona–El Prat Airport El Prat de Llobregat, Spain
Barclay Street New York, NY, United States
Barceloneta Puerto Rico
Open Map
ON
Create polygons by clicking
ion, which completes the loop. Erase polygons by clicking
elect them to change state.
Zaragoza
Lleida
Tàrrega
Terrassa
Mataró
Barcelona
Tarragona
Sariñena
Móra d'Ebre
Alcañiz
Andorra
Calamocha
© Mapbox © OpenStreetMap Improve this map

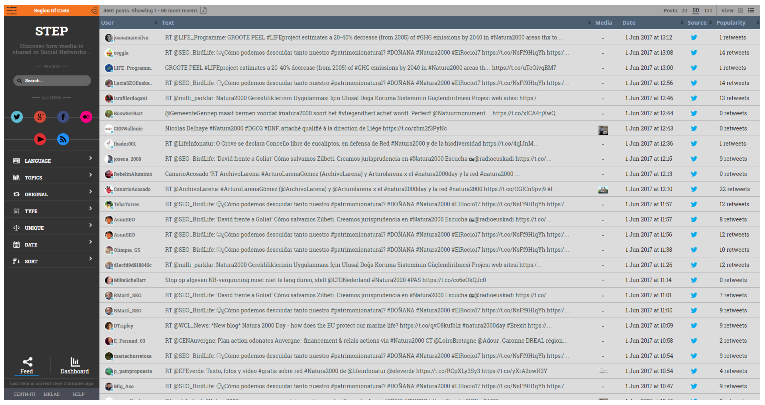
Fig. 3. Creation of a new collection by specifying keywords, accounts and locations of interest. The resulting queries are combined using an OR operator.

The feed view presents the list of media items collected in the form of a stream. Each item in the view features relevant metadata, namely the publication time, username and social media platform. The feed can be presented in both gallery/grid and list form (Figure 4). The user can also search for items based on free text queries and use a number of filters and sorting criteria. In particular, the following filters can be applied: a) source (social media platform), b) language, c) topics, d) original (show retweets/shares or not), e) type (text item or item with embedded multimedia), f) unique (remove near-duplicate content), g) date. In addition to filtering, the collections of items can be further customized based on explicit user feedback. The end user can rate the items retrieved for a collection on a scale of 1-5, with 1 corresponding to “irrelevant” and 5 to “relevant”. In addition, users may choose to exclude selected items and/or users along with their published items.

The dashboard view offers a variety of metrics and widgets that depict summary views over the collected data, enabling users to gain valuable insights about their topic of interest. The visualizations are dynamic by leveraging the same set of filters as the ones in the feed view, and allow users of the tool to inspect the media that are behind these statistics. In more detail, the dashboard consists of the following widgets:



(a) Feed gallery view.



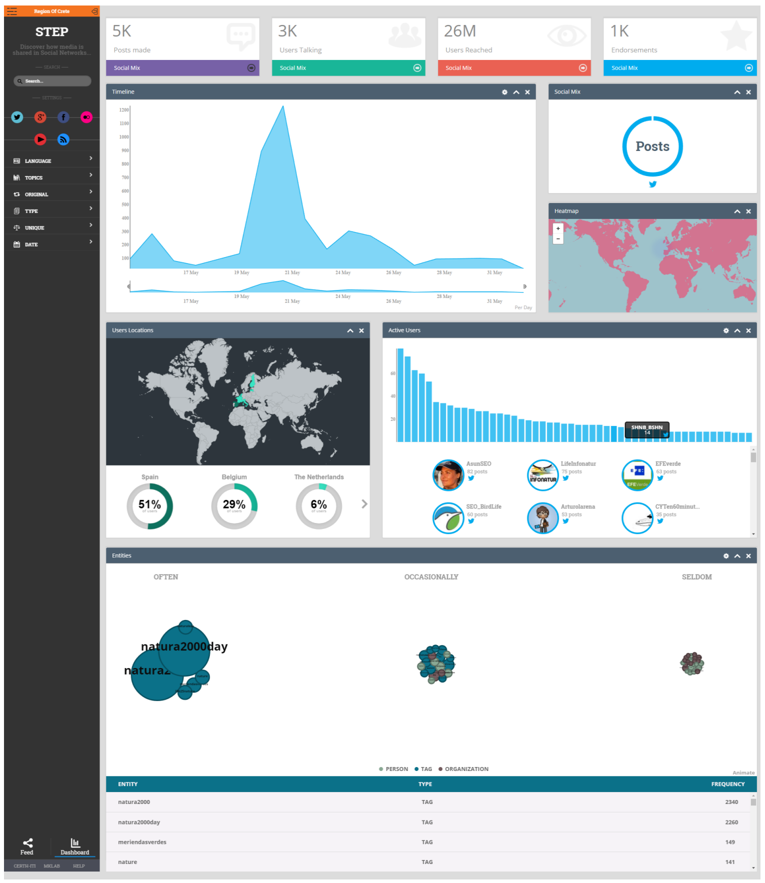
4651 posts. Showing 1 - 50 most recent.

User	Text	Media	Date	Source	Popularity
joaoamarosilva	RT @LIFE_Programme: GROOTE PEEL #LIFEproject estimates a 20-40% decrease (from 2005) of #GHG emissions by 2040 in #Natura2000 areas thx to...	-	1 Jun 2017 at 13:12		1 retweets
ceggla	RT @SEO_BirdLife: ☹¿Cómo podemos descuidar tanto nuestro #patrimonionatural? #DOÑANA #Natura2000 #ElRocio17 https://t.co/NsFf9H1qYh https:/...	-	1 Jun 2017 at 13:08		14 retweets
LIFE_Programm.	GROOTE PEEL #LIFEproject estimates a 20-40% decrease (from 2005) of #GHG emissions by 2040 in #Natura2000 areas th... https://t.co/uTeOivqBM7	-	1 Jun 2017 at 13:00		1 retweets
LuciaSEOEuska..	RT @SEO_BirdLife: ☹¿Cómo podemos descuidar tanto nuestro #patrimonionatural? #DOÑANA #Natura2000 #ElRocio17 https://t.co/NsFf9H1qYh https:/...	-	1 Jun 2017 at 12:56		14 retweets
israfilerdogan1	RT @milli_parklar: Natura2000 Gerekliliklerinin Uygulanması İçin Ulusal Doğa Koruma Sisteminin Güçlendirilmesi Projesi web sitesi https:/...	-	1 Jun 2017 at 12:46		13 retweets
Bosw8erBart	@GemeenteGennep maait bermen voordat #natura2000 soort het #vliegendhert actief wordt. Perfect! @Natuurmonument... https://t.co/xICA4rjXwQ	-	1 Jun 2017 at 12:44		0 retweets
CESWallonie	Nicolas Delhaye #Natura2000 #DGO3 #DNF, attaché qualifié à la direction de Liège https://t.co/zbm2f3PyNc		1 Jun 2017 at 12:43		0 retweets
Ibader001	RT @LifeInfonatur: O Grove se declara Concello libre de eucaliptos, en defensa de Red #Natura2000 y de la biodiversidad https://t.co/4qIJnM...	-	1 Jun 2017 at 12:36		1 retweets
joseca_2009	RT @SEO_BirdLife: 'David frente a Goliat' Cómo salvamos Zilbeti. Creamos jurisprudencia en #Natura2000 Escucha 📻@radioeuskadi https://t.co/...	-	1 Jun 2017 at 12:15		9 retweets
RebelinAluminio	CanarioAcosado 'RT ArchivoLarena: #ArturoLarenaGómez (ArchivoLarena) y Arturolarena x el #natura2000day y la red #natura2000 ...	-	1 Jun 2017 at 12:13		0 retweets
CanarioAcosado	RT @ArchivoLarena: #ArturoLarenaGómez (@ArchivoLarena) y @Arturolarena x el #natura2000day y la red #natura2000 https://t.co/OGfCnSpvj9 #l...		1 Jun 2017 at 12:10		22 retweets
TebaTorres	RT @SEO_BirdLife: ☹¿Cómo podemos descuidar tanto nuestro #patrimonionatural? #DOÑANA #Natura2000 #ElRocio17 https://t.co/NsFf9H1qYh https:/...	-	1 Jun 2017 at 11:57		12 retweets
AsunSEO	RT @SEO_BirdLife: 'David frente a Goliat' Cómo salvamos Zilbeti. Creamos jurisprudencia en #Natura2000 Escucha 📻@radioeuskadi https://t.co/...	-	1 Jun 2017 at 11:57		8 retweets
AsunSEO	RT @SEO_BirdLife: ☹¿Cómo podemos descuidar tanto nuestro #patrimonionatural? #DOÑANA #Natura2000 #ElRocio17 https://t.co/NsFf9H1qYh https:/...	-	1 Jun 2017 at 11:56		12 retweets
Olimpia_GS	RT @SEO_BirdLife: ☹¿Cómo podemos descuidar tanto nuestro #patrimonionatural? #DOÑANA #Natura2000 #ElRocio17 https://t.co/NsFf9H1qYh https:/...	-	1 Jun 2017 at 11:38		10 retweets
d1ecfd9d818846c	RT @milli_parklar: Natura2000 Gerekliliklerinin Uygulanması İçin Ulusal Doğa Koruma Sisteminin Güçlendirilmesi Projesi web sitesi https:/...	-	1 Jun 2017 at 11:26		12 retweets
MikeSchellart	Stop op afgeven NB-vergunning moet niet te lang duren, stelt @LTONederland #Natura2000 #PAS https://t.co/co6eOkQJc0	-	1 Jun 2017 at 11:14		0 retweets
RMarti_SEO	RT @SEO_BirdLife: 'David frente a Goliat' Cómo salvamos Zilbeti. Creamos jurisprudencia en #Natura2000 Escucha 📻@radioeuskadi https://t.co/...	-	1 Jun 2017 at 11:01		7 retweets
RMarti_SEO	RT @SEO_BirdLife: ☹¿Cómo podemos descuidar tanto nuestro #patrimonionatural? #DOÑANA #Natura2000 #ElRocio17 https://t.co/NsFf9H1qYh https:/...	-	1 Jun 2017 at 11:00		9 retweets
DTripley	RT @WCL_News: *New blog* Natura 2000 Day - how does the EU protect our marine life? https://t.co/qvO8kufb1z #natura2000day #Brexit https://...	-	1 Jun 2017 at 10:59		9 retweets
E_Ferrand_03	RT @CENAuvergne: Plan action odonates Auvergne : financement & relais actions via #Natura2000 CT @LoireBretagne @Adour_Garonne DREAL région...	-	1 Jun 2017 at 10:58		2 retweets
mariachurretoza	RT @SEO_BirdLife: ☹¿Cómo podemos descuidar tanto nuestro #patrimonionatural? #DOÑANA #Natura2000 #ElRocio17 https://t.co/NsFf9H1qYh https:/...	-	1 Jun 2017 at 10:54		9 retweets
p_pampropuesta	RT @EFEverde: Texto, fotos y vídeo #gratis sobre red #Natura2000 de @lifeinfonatur @efeverde https://t.co/RCpXLy3Sy3 https://t.co/yXrA2owH3Y		1 Jun 2017 at 10:54		4 retweets
Mig_Aso	RT @SEO_BirdLife: ☹¿Cómo podemos descuidar tanto nuestro #patrimonionatural? #DOÑANA #Natura2000 #ElRocio17 https://t.co/NsFf9H1qYh https:/...	-	1 Jun 2017 at 10:47		9 retweets

(b) Feed list view.

Fig. 4. Snapshot of feed view in two forms.

– Numbers of posts made, users talking, users reached and endorsements, and platform contribution pie chart.
– Heatmap based on the exact location (latitude, longitude) of geo-tagged items in the collection (typically a small subset of all items).
– User location at the level of a country, which is automatically detected from the location text field in users' profiles.
– Timeline visualization based on the number of items over time. The granularity of the depiction may be set to hour, day or week.
– Top N users with most posts in the collection. N can be set between 10 and 200. Avatar, username and total number of posts for each user is presented along with the link to their social profile pages.
– Top N entities in terms of frequency. N can be set between 10 and 200. Entities are organized in three frequency categories (often, occasionally, seldom) and three types (person, tag, location).



ENTITY	TYPE	FREQUENCY
natura2000	TAG	2340
natura2000day	TAG	2260
meriendasverdes	TAG	149
nature	TAG	141

Fig. 5. Snapshot of Dashboard view.

3.3 Deployment to Local Environment

For the deployment of social media monitoring tool as a service that can be locally administered, we opted for the use of Docker[18]. Each of the components of the tool is deployed and run in a separate Docker container[19]. Figure 2 depicts all the components on top of Docker and the communication between them. For MongoDB and Solr, the official Docker images are used. For the Stream Manager, we built an image that clones a stable release of the module from its GitHub repository and generates an executable jar using maven. The REST API and user interface is built and deployed as a separate Docker image on top of Apache web server. Finally, the signaling of collection-related events to the MongoDB is handled by a Redis instance, while the aggregation and storage of produced logs by a Graylog instance[20], both of which are also deployed on top of Docker.

[18] `https://www.docker.com`
[19] AVailable on: `https://github.com/MKLab-ITI/mmdemo-dockerized`
[20] `https://www.graylog.org/`

Using Docker Compose[21], we are able to define all the components in a single configuration file expressed in YAML[22]. That configuration file needs limited edits to be adapted to users' local environment. The only requirement prior to deployment is to set the paths of directories in the local file system for MongoDb, Solr, etc. Optionally, service ports can be edited to values other than the default ones, e.g. port 80 in Apache web server. In addition to the YAML configuration file, a separate configuration file associated with the Stream Manager needs to be edited to enable the use of social media APIs, by providing the necessary user credentials for each platform.

4 Limitations and Future Work

There are two main limitations in the usage of the presented framework. A first limitation is due to API usage restrictions imposed by the social media platforms. For example, Twitter restricts the number of allowed calls in a time window of 15 minutes[23]. Other platforms have similar restrictions but with different time granularities. As the number of collections and associated queries increases, it is necessary to decrease the request rate for each query. This creates a delay in the discovery of new content, especially for queries (keywords) that exhibit high activity. Moreover, a limitation pertains to the large scale of the collected content items, which may easily reach the order of tens of millions. As the association of content to collections is dynamically resolved based on the Solr indexing mechanism, the response time of the tool is adversely affected by the size of Solr index. This is more pronounced in the case of analytics, which are calculated on the fly using the faceting mechanism of Solr.

To improve the scalability of the framework, we plan to investigate and develop means of distributing the query and indexing load over multiple nodes. In particular, we plan to use SolrCloud for partitioning the Solr index into multiple shards and replicas. In addition to the inherent fault tolerance of that setting, an obvious advantage is that queries and index updates can be directed to different nodes, improving load balancing and decreasing response time. In terms of functionality, we aim at two further improvements: a) more sophisticated relevance models that go beyond the simple heuristic rules that are currently used for filtering, b) support for additional social media platforms such as Sina Weibo and VKontakte.

Acknowledgements. This work was supported by the STEP (H2020-649493) and hackAIR (H2020-688363) projects, funded by the European Commission.

[21] `https://docs.docker.com/compose/`
[22] `http://yaml.org/`
[23] `https://dev.twitter.com/rest/public/rate-limiting`

References

1. Kaplan, A. M., Haenlein, M.: Users of the World, Unite! The Challenges and Opportunities of Social Media. Business Horizons 53(1), 59–68 (2010)
2. Aiello, L.M., Petkos, G., Martin, C., Corney, D., Papadopoulos, S., Skraba, R., Goker, A., Kompatsiaris, Y., Jaimes, A.: Sensing Trending Topics in Twitter. IEEE Transactions on Multimedia 15(6), 1268–1282 (2013)
3. Mejova, Y., Weber, I., Macy, M.W.: Twitter: a Digital Socioscope. Cambridge University Press (2015)
4. Fan, W., Gordon, M.D.: The Power of Social Media Analytics. Communications of the ACM 57(6), 74–81 (2014)
5. Papadopoulos, S., Schinas, E., Mironidis, T., Iliakopoulou, K., Spyromitros-Xioufis, E., Tsampoulatidis, I., Kompatsiaris, I.: Social multimedia crawling and search. IEEE CS Special Technical Community on Social Networking E-Letter 1(3) (2013)
6. Diplaris, S., Papadopoulos, S., Kompatsiaris, I., Goker, A., Macfarlane, A., Spangenberg, J., Hacid, H., Maknavicius, L., Klusch, M.: Socialsensor: sensing user generated input for improved media discovery and experience. Proceedings of the 21st International Conference on World Wide Web (WWW '12 Companion), 243–246, ACM (2012)
7. Andreadou, K., Papadopoulos, S., Apostolidis, L., Krithara, A., Kompatsiaris, Y.: Media REVEALr: A Social Multimedia Monitoring and Intelligence System for Web Multimedia Verification. Proceedings of the Pacific-Asia Workshop on Intelligence and Security Informatics, 1–20, Springer (2015)
8. Schinas, M., Papadopoulos, S., Kompatsiaris, Y., Mitkas, P. A. Visual event summarization on social media using topic modelling and graph-based ranking algorithms. In Proceedings of the 5th ACM on International Conference on Multimedia Retrieval, 203–210, ACM (2015)