

**H2020-ICT-2018-2 /ICT-28-2018-CSA**  
**SOMA: Social Observatory for Disinformation and Social Media Analysis**



## D3.4 Data Intelligence toolkit v1

<b>Project Reference No</b>	SOMA [825469]
<b>Deliverable</b>	D3.4 Data Intelligence toolkit v1
<b>Workpackage</b>	WP3 Observatory set up and Operation
<b>Type</b>	OTHER
<b>Dissemination Level</b>	Public
<b>Date</b>	31/1/2020
<b>Status</b>	Final
<b>Editor(s)</b>	Alice Andreuzzi, LUISS Datalab, LUISS University Luca Tacchetti, LUISS Datalab, LUISS University
<b>Contributor(s)</b>	Emanuele Camarda, LUISS Datalab, LUISS University Noemi Trino, LUISS Datalab, LUISS University Stefano Guarino, LUISS Datalab, LUISS University
<b>Reviewer(s)</b>	Marina Klitsi, ATC
<b>Document description</b>	This deliverable D3.4 introduces the first version of the software implementation of the AI Driven Observatory

## Document Revision History

Version	Date	Modifications Introduced	
		Modification Reason	Modified by
v0.1	08/01/2020	Concept and Structure	LUISS University
v0.2	14/01/2020	First Draft	LUISS University
v0.3	17/01/2020	Second Draft	LUISS University
v0.4	24/01/2020	ATC revision	ATC
v1.0	28/01/2020	Final Version	LUISS University

## Executive Summary

This deliverable D3.4 introduces the first version of the software implementation of the AI Driven Observatory, that puts into practice the outcome of the work performed for Deliverables 3.2 and 3.3. In particular, the report provides an overview of DisInfoNet Toolbox's current functionalities and features. The toolbox is designed to help the SOMA community to understand the dynamics of (fake) news dissemination in social networks. DisInfoNet combines text mining and classification with graph analysis and visualization to offer a comprehensive and user-friendly suite that allows users to (i) track relevant news stories and reconstruct their prevalence over time and space; (ii) detect central debating communities and capture their distinctive polarization/narrative; (iii) identify influencers both globally and in specific "disinformation networks". Software development has been conducted according to the Agile methodology, based on iterative development of users' requirements and epics. The subsequent platform Architecture incorporates the logical structure of the previous functional analysis.

## Table of Contents

<b>1</b>	<b>INTRODUCTION.....</b>	<b>5</b>
1.1	STRUCTURE OF THE DELIVERABLE.....	5
<b>2</b>	<b>HIGH LEVEL DESIGN.....</b>	<b>6</b>
2.1	DISINFONET: A DEFINITION.....	6
2.2	FUNCTIONALITIES.....	6
<b>3</b>	<b>USER REQUIREMENTS .....</b>	<b>8</b>
3.1	PRODUCT BACKLOG, EPICS, USER STORIES .....	8
3.2	DISNIFONET: EPICS.....	8
3.3	DISNIFONET: USER STORIES .....	10
<b>4</b>	<b>HIGH LEVEL ARCHITECTURE .....</b>	<b>13</b>
4.1	CODE STRUCTURE .....	14
<b>5</b>	<b>DATA INTELLIGENCE TOOLKIT V1 .....</b>	<b>15</b>
	<b>WHILE ADDING A NEW QUERY, YOU CAN SPECIFY SOME SYNONYMOUS .....</b>	<b>18</b>
<b>6</b>	<b>CONCLUSIONS.....</b>	<b>20</b>

### List of Figures

Figure 1: DisInfoNet's main pipeline .....	7
Figure 2: High Level Architecture .....	14
Figure 3: DisInfoNet Homepage .....	16
Figure 4: Amazonia focus .....	16
Figure 5: Filters .....	17
Figure 6: Create new filter .....	17
Figure 7: Synonyms.....	18
Figure 8: Classifier.....	18
Figure 9: Cluster selection .....	19
Figure 10: Cluster label and description.....	19
Figure 11: About page .....	20

### List of Tables

Table 1: DisInfoNet Epics .....	10
Table 2: DisInfoNet User Stories.....	13

## List of Terms and Abbreviations

Abbreviation	Definition
API	Application Programming Interface
CSV	Comma-Separated Values
JSON	JavaScript Object Notation
ORM	Object-Relational Mapping
DB	Database
SVG	Scalable Vector Graphics
JPEG	Joint Photographic Experts Group
PNG	Portable Network Graphics

# 1 Introduction

This document intends to collect the analyses carried out to design the DisInfoNet prototype. From the first phase of collecting user requirements, we moved on to writing the Product Backlog, accompanied by Epics and User Stories.

Starting from the functional analysis conducted, a high level architecture was designed that describes the technical components of the prototype and the main functions. Finally, we moved on to the back end and front end development of the Data Intelligence toolkit v1 (only some of the functionalities have been implemented).

## 1.1 Structure of the Deliverable

The current report is structured as follows.

- Section 2 provides a complete overview of DisinfoNet high level design and main functionalities. The section describes the general purposes of the toolbox and the tools already implemented in order to reconstruct relevant news stories on social media over time and space as well as to identify central debating communities and specific “disinformation networks” and actors.
- Section 3 is devoted to the analysis and collection of user requirements for the development of the DisinfoNet platform, conceived according to a “scrum” approach, that accounts for product backlog, epics’ definition, prioritized user-stories.
- Following the functional analysis, Section 4 presents the high-level architecture of the platform, based on a classic three levels architecture composed by a database, a backend and a frontend. The same section presents the code-structure.
- The final section of the report illustrates the first version of the Data Intelligence Toolkit.

## 2 High Level Design

### 2.1 DisinfoNet: a Definition

DisInfoNet Toolbox aims at supporting users of the SOMA verification platform in understanding the dynamics of (fake) news dissemination in social media and tracking down the origin and the broadcasters of false information. It combines text mining and classification with graph analysis and visualization to offer a comprehensive and user-friendly suite that allows users to:

- i. track relevant news stories and reconstruct their prevalence over time and space;
- ii. detect central debating communities and capture their distinctive polarization/narrative;
- iii. identify influencers both globally and in specific “disinformation networks”.

DisInfoNet is developed by Data Lab for web users, media experts, journalists and researchers to analyze Twitter discussions in a simple but rigorous way. More specifically, the toolbox help users to identify messages, news, comments and, in general, contents that are false, tendentious or propagandistic; it helps the members of the vast community around SOMA to understand the dynamics of dissemination of this content on social media, identifying users who have a particularly active role in this process and possibly reconstructing the news sources. These are the main features:

- a scraper for data collection from Twitter;
- a filter for the selection, within a set of documents, of texts that report or treat or comment on a news item of interest;
- a tool for extracting a graph - oriented to users or hashtags - from Twitter data;
- a toolbox for graph analysis, equipped with community detection tools, centrality and identification of hubs and influencers, classification of users based on their role in intra- and inter-community connections;
- a software to perform topic modeling on a corpus of documents;
- a classifier for understanding the polarization of a given message or user against a topic of interest.

### 2.2 Functionalities

DisInfoNet implements a pipeline designed to enable journalists and fact-checkers with no coding expertise assessing the prevalence of disinformation in social media data. This pipeline, depicted in Figure 1, consists of three main tools, described in the following sections. It can be fed with a CSV file or the (possibly GZ-compressed) JSON file returned by the Twitter search or streaming API. One of DisInfoNet’s main features is the ability to extract and examine both keyword co-occurrence graphs and user interaction graphs induced by a specific set of themes of interest, thus providing valuable insights into the contents and the actors of the social debate around disinformation stories.

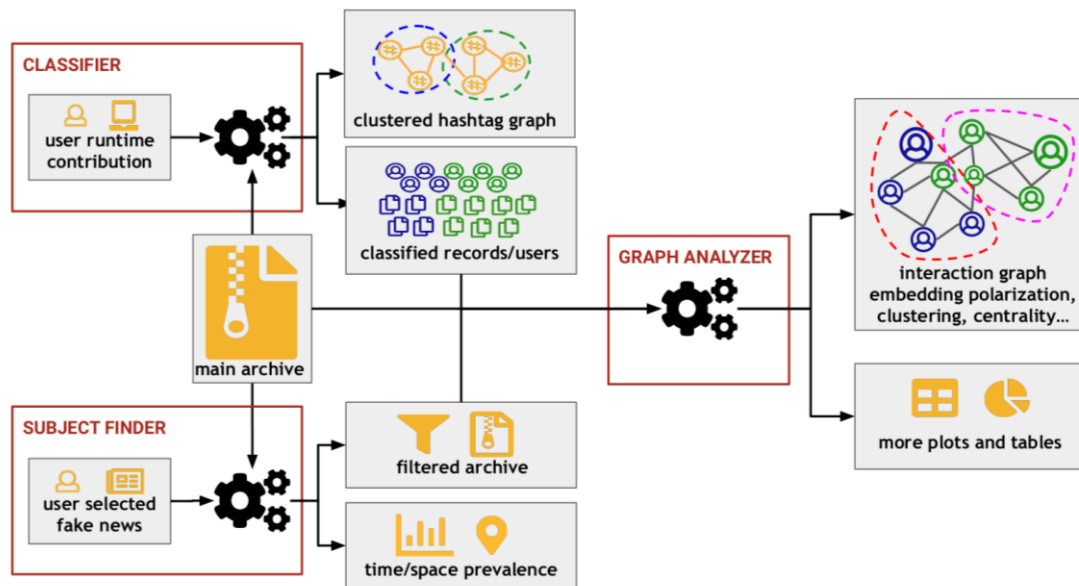


Figure 1: DisInfoNet's main pipeline

**Data Collection:** DisInfoNet provides a dashboard for collecting data from selected social media and news items. The current data collection tool is a tweepy-based Python scraper for Twitter data that accepts a list of keywords and returns a variable-length list of tweets containing at least one of such keywords.

**Classifier:** is based on a semi-automatic “self-training” process”, in which a list of hashtags associated with two (or more) classes of interest are used to automatically extract a training set. The classifier works according to a three-step process:

1. It builds and cluster a **keyword co-occurrence graph**<sup>1</sup>, that presents the user with an excerpt of the keywords associated with the obtained classes.
2. It provides the **user** with the possibility to **prune** of central (high pagerank) yet generic and/or out-of-context keywords, detrimental to clustering.
3. After performing **community detection** on the graph, the user is presented with an excerpt of these clusters for manual inspection and she/he can pick and label any of these communities as representative of a specific class.

As a result, the classifier can use as many as 10/100/100× more hashtags for classification than with any realistic fully manual approach, without sacrificing accuracy and possibly bringing to light previously unknown highly discriminative hashtags. When only two classes are used (eg. pro/anti, yes/no), the obtained classification can be extended to users by averaging over the classification of all tweets produced by a specific user.

The **Graph Analyzer** provides users with graph mining and visualization. It first extracts a directed user interaction graph, wherein two users (e.g., authors) are connected based on how often they interact (e.g., cite each other). It then computes a set of global and local metrics, including: distances, eccentricity, radius and diameter; clustering coefficient; degree and assortativity; PageRank, closeness

<sup>1</sup> The hashtag graph is extracted from the available dataset by connecting any two hashtags that co-occur at least once in the same tweet and weighing the edge by the total count of such co-occurrences.

and betweenness centrality. It also partitions the graph into communities, relying on the Louvain or Leading Eigenvector algorithms, and applies the Guimera`-Amaral cartography.

**Temporal distribution by class:** The figure provides the one-day rolling mean of each obtained class in the considered period of time, compared with the overall trend.

**Spatial distribution by class:** The figure provides the geographic distribution of tweets, based on the percentage of geotagged tweets.

**Hashtag graph:** The graph plots the hashtags, highlighting clusters and vertex sizes (using pagerank).

### 3 User Requirements

The phase of analysis and collection of user requirements for the development of the Disinonet platform has followed an approach known as “scrum”. Scrum is the most popular agile framework for software development, and it is characterized by iterative development, self-organized and empowered team, time boxed meeting. In detail, following the scrum methodology we draft the product backlog, define the epics, finalize the user stories.

#### 3.1 Product Backlog, Epics, User Stories

In its simplest definition, the Product Backlog is a list of all the features that must be implemented in the project. In Scrum it plays the role of specification of the requirements. The items making up the Backlog are technical or user-centric in nature, written in the form of a user story, the analogue of use cases in a context of UML based analysis of requirements.

- A User Story defines a user's need with respect to an object or context of use. A hypothetical user story model can be the following: **As a [subject], I want [have to [action], so that [goal]**
  - *Subject:* the user story owner. It can be the user but it is advisable to specify it (eg "administrator", "logged in user", "unauthenticated visitor"). In this way the story is easier to understand and frame in the appropriate context.
  - *Action:* what the actor wants to do. If it is a mandatory action, the prefix “have to” have to be used, otherwise the prefix "I want".
  - *Goal:* what the actor wants to achieve by carrying out the action described by the story.

#### 3.2 Disinonet: Epics

Below is the detail of the epics identified for the design of the Disinonet prototype. Epics are divided by area: overview component, data collecting component, analysis component, visualization component. The functional analysis shown below is the complete analysis for the design of the DisInfoNet prototype. However, in the first release of January 2020 (Data Intelligence toolkit v1), only some of the features designed are developed and available.

1	Overview Component
1.1	Upon first access, the user receives information relating to the platform and its main partners: Luiss Data Lab, Aletheia, Soma
1.2	The user can view the graphs of the latest searches



1.3	Upon login, the user can decide to perform analysis on existing datasets
1.4	The user upon login can create a new analysis

<b>2</b>	<b>Data Collecting Component</b>
2.1	The user must have a twitter developer account with the relative access keys which must be shared on the platform on first access. The access keys linked to the user will be recorded in the db for future access
2.2	The user can create a new search by filling in the filter fields
2.3	The user can specify synonyms for each word entered in the query
2.4	The user can decide not to collect retweets, replies
2.5	The launched searches go back as far as possible with the possibility of inserting the start date and proceed forward automatically if not stopped

<b>3</b>	<b>Analysis Component</b>
3.1	The search results can be downloaded in csv
3.2	General search results can be further filtered by one or more TAGs (exact words, hashtags or phrases) and each exact word, hashtag or phrase can be synonyms. The result can be downloaded in csv.
3.3	The results of the general search can be elaborated through the classifier, which proposes some hashtag clusters for the communities. The user can decide to keep the clusters and assign him a LABEL or skip the clusters that do not interest him. The clusters will instruct other data, training sets.

<b>4</b>	<b>Visualization Component</b>
4.1	The user can create geolocated maps of all the overall tweets collected
4.2	The user can decide to apply filters to the visualization - select words to give different color to the map
4.3	The user can create geolocated maps of the tweets already analyzed with the TAG filter if available
4.4	The user can create geolocated maps of the tweets already analyzed with the LABEL filter if available
4.5	The graphs can be downloaded in png and svg
4.6	The user can create a network chart for all the overall tweets collected
4.7	The user can decide to apply filters to the visualization - select words to give different color to the network graph
4.8	The user can create a network chart of the tweets already analyzed with the TAG filter if

	available
4.9	The user can create a network chart of the tweets already analyzed with the LABEL filter if available
4.10	Charts can be downloaded in svg and png
4.11	The user can create a time trend graph for all the overall tweets collected
4.12	The user can decide to apply filters to the visualization - select words to assign different color to the time trend graph
4.13	The user can create a time trend graph of the tweets already analyzed with the TAG filter if available
4.14	The user can create a time trend graph of the tweets already analyzed with the LABEL filter if available

Table 1: DisInfoNet Epics

### 3.3 Disnifonet: User Stories

The Epics were broke out into prioritized user stories (high/low). The "status" column identify which user stories have already been implemented at the time of this deliverable (Data Intelligence Toolkit v1).

ID	Area	User Stories	Priority	Status
1	<b>Overview Component</b>	By accessing the Homepage of the DisInfoNet platform, the user finds a description with a brief summary of the prototype	HIGH	DONE
2		By accessing the homepage of the DisInfoNet platform, the user finds the results of the last search of the Data Lab. The results can be browsed in interactive mode.	HIGH	DONE
3		Through the DisInfoNet logo it is always possible to return to the homepage	HIGH	DONE
4		By accessing the Homepage, the user can decide to start a new search by clicking on the "start a search" button, also present as an item in the "new search" menu.	LOW	
5		By clicking on the button and / or on the menu item "start a search" the authentication window opens.	LOW	
6		The user finds 3 menu items: "start a search", "About us" and "Login" (characterized by an icon)	LOW	
7		By clicking on "about us" you will land on an explanatory text	HIGH	DONE
8		By clicking on "Login" you will land on the authentication page	LOW	

9		The SOMA logo is present on the platform ( <a href="https://www.disinfobservatory.org/">https://www.disinfobservatory.org/</a> )	HIGH	DONE
10	<b>Data Collecting Component</b>	The user logs in with username (email) and password	LOW	
11		If you do not have the credentials, you can request the credentials through a specific call to action	LOW	
12		Password recovery must be possible	LOW	
13		Entering the private area, the user can choose whether to analyze existing datasets or activate a new data collection	LOW	
14		If the user chooses to start a new data collection, they are asked for the access keys linked to the Twitter Developer user they owned ( <a href="https://developer.twitter.com/en/application/use-case">https://developer.twitter.com/en/application/use-case</a> )	LOW	
15		When creating a new collection, the user must necessarily assign a title to it (with the addition of a possible description)	LOW	
16		The user must fill in the fields relating to the search rules: these fields are the filters of the search that will be launched	LOW	
17		The filters that the user can fill in are: All of these words, this exact phrase, any of these words, none of this words, this hashtag, written in (language), near this place, advanced rules (filter retweets - filter reply ). People: to this account, mentioning accounts, from these accounts.	LOW	
18		The user can also decide to specify synonyms for each word entered in the search query	LOW	
19		The user can select the dates delimiting the search (minimum search start date-maximum end date). If the user does not select the minimum date, the search goes back in time as much as possible.	LOW	
20		If the user does not stop the search, it proceeds forward in time	LOW	
21	<b>Analysis Component</b>	Once the user believes he has completed a search, he can download the results via a csv file. The file contains all the results without distinction whatsoever.	LOW	

22		The user may decide to filter the general search. In this case, before downloading the csv file can fill in specific fields such as: words, hashtags, or exact sentences. In this case the search can be filtered as if it were a pivot table. Once filtered, the specific file can be downloaded by the user. This filter is called TAG.	HIGH	DONE
23		The user can decide to deepen the research carried out to identify one or more communities.	HIGH	DONE
24		If the user wants to deepen the identification of the communities, he can click on the "Classify the dataset" button.	HIGH	DONE
25		Once activated, the classifier proceeds to classify the dataset according to the precompiled algorithm	HIGH	DONE
26		Once the processing is complete, the user can decide to: attach a label to the clusters obtained from the classification, skip the clusters that he deems not useful or unnecessary. This filter is called LABEL	HIGH	DONE
27		The user can decide to use clusters as a training set basis for further data analysis	HIGH	DONE
28	<b>Visualization Component</b>	Once the analysis is completed, the user can decide to browse the results according to the different components and according to the research carried out	HIGH	DONE
29		The user can view: maps, network graphs, time trends	HIGH	DONE
30	<b>Visualization Component - maps</b>	When accessing the homepage, the user finds some geo-localized maps with the datasets of the latest searches	HIGH	DONE
31		The user can create geolocated maps of all the overall tweets collected	HIGH	DONE
32		The user can decide to apply filters to the visualization - select words to assign different colors to the map	LOW	
33		The user can create geolocated maps of the tweets already analyzed with the TAG filter if available	HIGH	DONE
34		The user can create geolocated maps of the tweets already analyzed with the LABEL filter if available	LOW	
35	<b>Visualization component</b>	The user, when accessing the homepage, finds some network graphs with the datasets of the latest searches	HIGH	

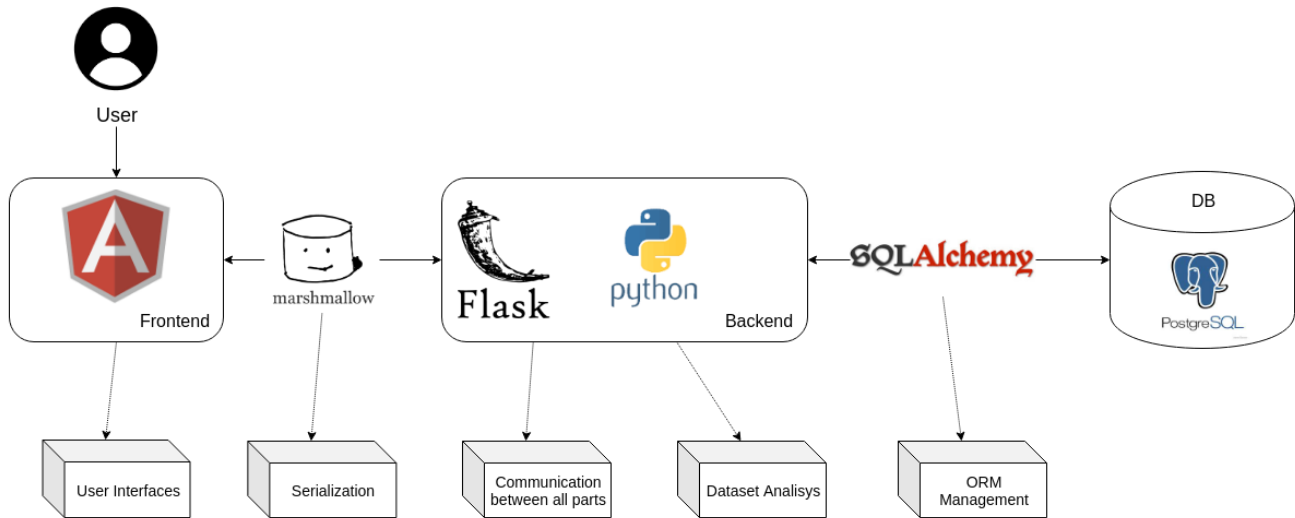
	<b>network graph</b>			DONE
36		The user can create a network chart for all the overall tweets collected	HIGH	DONE
37		The user can decide to apply filters to the visualization - select words to give different color to the network graph	LOW	
38		The user can create a network chart of the tweets already analyzed with the TAG filter if available	HIGH	DONE
39		The user can create a network chart of the tweets already analyzed with the LABEL filter if available	HIGH	DONE
40	<b>Visualization component - highchart trend</b>	The user, when accessing the homepage, finds some temporal trends with the datasets of the latest searches	HIGH	DONE
41		The user can create a time trend graph for all the overall tweets collected	HIGH	DONE
42		The user can decide to apply filters to the visualization - select words to assign different color to the time trend graph	LOW	
43		The user can create a time trend graph of the tweets already analyzed with the TAG filter if available	HIGH	DONE
44		The user can create a time trend graph of the tweets already analyzed with the LABEL filter if available	LOW	

Table 2: DisInfoNet User Stories

## 4 High Level Architecture

The high-level architecture for the platform features different software components that incorporate the same logical subdivision proposed in the functional analysis phase.

DisinfoNet has a classic three levels architecture composed by database, backend and frontend. **Flask** is used to manage the interaction between the backend and the other components; **SqlAlchemy** is used under Flask to manage the ORM problem and interaction with database. For persistence level a relation-entity database (**Postgresql**) is used. The frontend logic is written in **javascript** through the framework **AngularJs**. Backend and frontend communicate through **RestApi** and exchange data in json format. **Marshmallow** is used for the serialization of data.



**Figure 2: High Level Architecture**

## 4.1 Code Structure

About the file structure of the application, we have the following main directories:

- disInfoNet: contains all the logic of the application;
- data: contains the physical dataset and all the support (dumped) data structures;
- result: contains all graphics and table
- utils, text\_analyzer, graph\_analyzer: contain support libraries

The main directory of the project contains the following files:

- runserver.py: starts the application;
- manage.py: manages the database through the option create\_db, delete\_db, seed\_db --seedfile <path\_to json\_file>;
- <dataset>\_config.ini: configuration file for insert dataset automatically generated;

The backend logic it's implemented inside the .py files in the directory disInfoNet. In particular:

- \_\_init\_\_.py: initialize the application loading the settings from setting.py and starts the application;
- controllers.py: contains the routing of backend's method;
- models.py: define the entity model of the application's classes;
- schemas.py: define the serialization of entity in the database;

The front end it's implemented inside the directories disInfoNet/templates and disInfoNet/static:

- templates/ contains all main page html files; frontend follows the Single Page Application model;
- static/partials contains all the static html page injected inside the index.html through the routing's logic define with AngularJs;
- static/js: contains all javascript code with the logic of frontend application;
- static/lib: all support libraries for frontend;
- static/img: all static image;

- static/css: all css styles files.

About the logic of fronted, the application it's so divided:

- app.js: initialize the application and define the routing through the AngularJs' module 'ngRoute';
- controllers.js: implements the logic of all pages through its corresponding controllers;
- directive.js: implements some custom html directives;
- filters.js: implements some custom filters for displaying texts;
- services.js: implements some utilities function for javascript logic, such as custom ajax methods.

Backend exported methods:

- /saveFilter
- /saveClassifier
- /deleteClassifier/<dataset\_id>
- /getAllDataset
- /getAllFilters/<dataset\_id>
- /getLabels/<dataset\_id>
- /getHystogram/<dataset\_id>/<int\_list:ids>
- /getTimeDistribution/<dataset\_id>/<int\_list:ids>
- /getMap/<dataset\_id>/<int\_list:ids>
- /classifier/takeTopHashtag/<dataset\_id>
- /classifier/takeCommunities/<dataset\_id>
- /classifier/createGraphs/<dataset\_id>
- /getInteractionGraph/<dataset\_id>/<filter\_category>

The software code of the application is available on GitLab, with authorization, at the following link: <https://gitlab.com/s.guarino/disinfofet>.

## 5 Data Intelligence Toolkit v1

The following section provides an overview of the main features already developed for the Data Intelligence Toolkit.

### Homepage: overview

A brief introduction to the DisInfoNet prototype is available on the homepage accompanied by the main graphs (time trends, network graphs and geolocalized maps) of the analysis already made by the Luiss Data Lab within the SOMA project. In particular, there are two starting datasets already available for consultation: **Referendum\_november** and **Amazonia**.

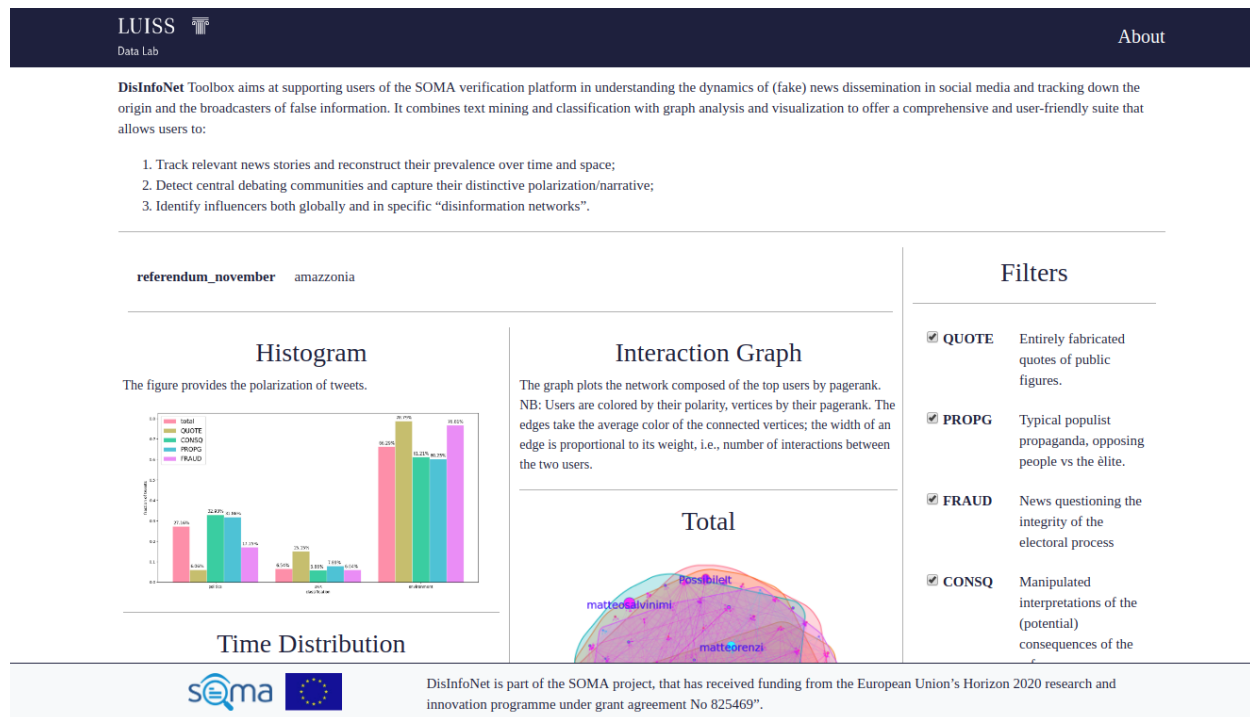


Figure 3: DisInfoNet Homepage

By choosing between referendum\_november and amazonia, it is possible to view the correspondent analyses.

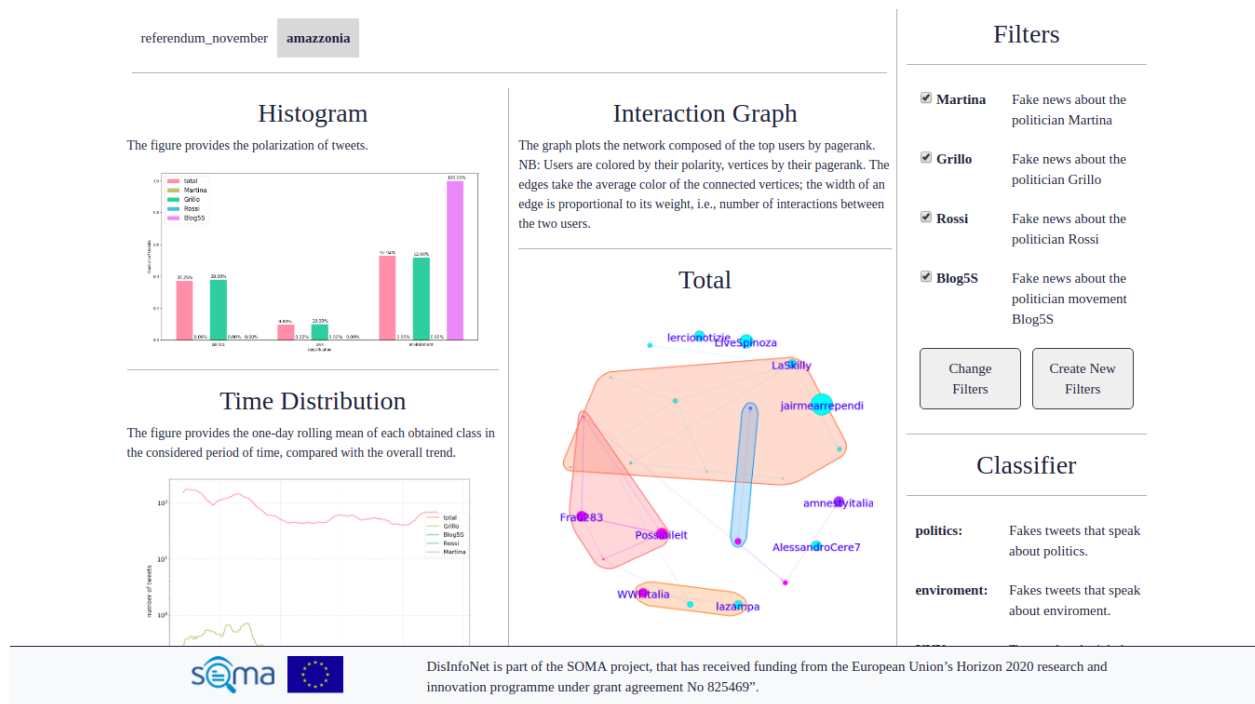
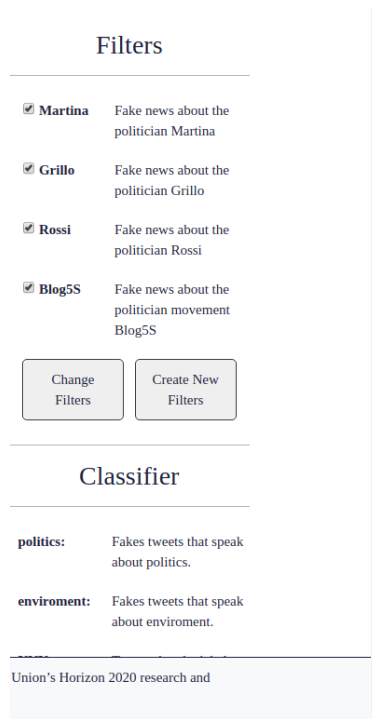


Figure 4: Amazonia focus



## Filter

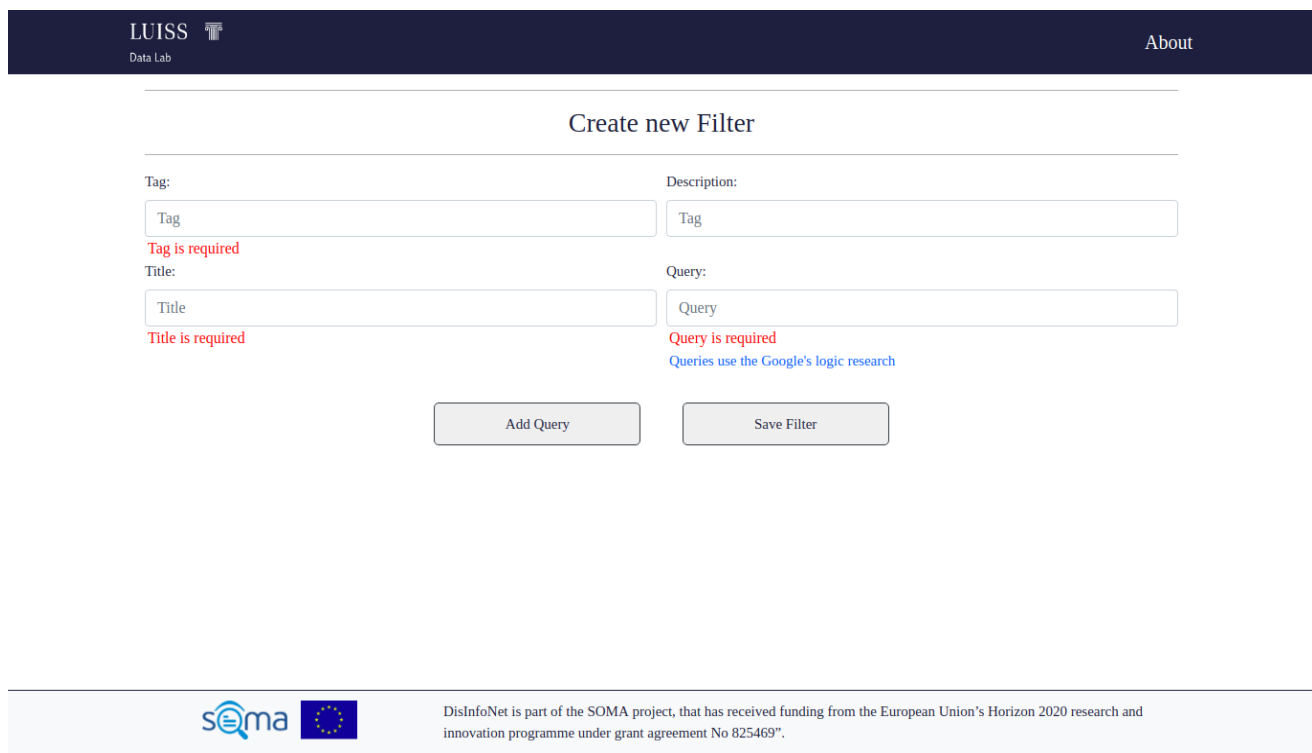
In the column at the top right of the homepage you can find out which tags are already set within the chosen dataset.



The screenshot shows two sections: 'Filters' and 'Classifier'. The 'Filters' section lists four active filters, each with a checked checkbox and a description: 'Martina' (Fake news about the politician Martina), 'Grillo' (Fake news about the politician Grillo), 'Rossi' (Fake news about the politician Rossi), and 'Blog5S' (Fake news about the politician movement Blog5S). Below the list are two buttons: 'Change Filters' and 'Create New Filters'. The 'Classifier' section shows two categories: 'politics:' (Fakes tweets that speak about politics.) and 'enviroment:' (Fakes tweets that speak about enviroment.). At the bottom, there is a text input field containing 'Union's Horizon 2020 research and'.

**Figure 5: Filters**


The analyst / researcher can decide to create a new filter in the dataset



The screenshot shows the 'Create new Filter' form. It has a dark blue header with 'LUISS Data Lab' and 'About' links. The form has two columns: 'Tag' and 'Description'. The 'Tag' column has a text input field with 'Tag' and a red error message 'Tag is required'. The 'Description' column has a text input field with 'Tag' and a red error message 'Query is required'. Below the 'Tag' input is a 'Title' input field with 'Title' and a red error message 'Title is required'. Below the 'Description' input is a 'Query' input field with 'Query' and a red error message 'Query is required'. Below the 'Query' input is a blue text link 'Queries use the Google's logic research'. At the bottom are two buttons: 'Add Query' and 'Save Filter'. The footer contains the 'soma' logo, the European Union flag, and text stating: 'DisInfoNet is part of the SOMA project, that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825469'.

**Figure 6: Create new filter**

LUISS



Data Lab

About

Create new Filter

Tag:

Tag is required

Title:

Title is required

Description:

Query:

[Queries use the Google's logic research](#)

Synonyms

verde:

Add Query

Save Filter

### Figure 7: Synonyms

In the lower right column of the homepage it is possible to know which classifications are already present in the displayed dataset. The analyst / researcher can decide to create a new classification in the dataset.

About

Data Lab

## Create new Classifier

**Classifier** is based on a semi-automatic “self-training” process”, in which a list of hashtags associated with two (or more) classes of interest are used to automatically extract a training set. The classifier works according to a three-step process:

1. It builds and cluster a keyword co-occurrence graph, that presents the user with an excerpt of the keywords associated with the obtained classes.
2. It provides the user with the possibility to prune of central (high pagerank) yet generic and/or out-of-context keywords, detrimental to clustering.
3. After performing community detection on the graph, the user is presented with an excerpt of these clusters for manual inspection and she/he can pick and label any of these communities as representative of a specific class.



As a result, the classifier can use as many as 10/100/100× more hashtags for classification than with any realistic fully manual approach, without sacrificing accuracy and possibly bringing to light previously unknown highly discriminative hashtags.

## Top Hashtag

These are the top 30 hashtag. Now you can clean the dataset deleting any hashtag that not interest you. Click on “Finish” when you're done.

<input type="checkbox"/> #iovotono	<input type="checkbox"/> #referendum	<input type="checkbox"/> #referendumcostituzionale	<input type="checkbox"/> #bastaunsi
<input type="checkbox"/> #renzi	<input type="checkbox"/> #photo	<input type="checkbox"/> #no	<input type="checkbox"/> #iodicono
<input type="checkbox"/> #riformacostituzionale	<input type="checkbox"/> #costituzione	<input type="checkbox"/> #4dicembre	<input type="checkbox"/> #ioivotosi
<input type="checkbox"/> #ottoemezzo	<input type="checkbox"/> #nostalgia	<input type="checkbox"/> #pd	<input type="checkbox"/> #trendingtopic
<input type="checkbox"/> #m5s	<input type="checkbox"/> #bastaunsi	<input type="checkbox"/> #si	<input type="checkbox"/> #matteorisponde
<input type="checkbox"/> #italia	<input type="checkbox"/> #brexit	<input type="checkbox"/> #trump	<input type="checkbox"/> #futuro
<input type="checkbox"/> #deluca	<input type="checkbox"/> #kills	<input type="checkbox"/> #1w11ʼ	<input type="checkbox"/> #renziacasa
<input type="checkbox"/> #cdd	<input type="checkbox"/> #dimartedì		

Finish



DisInfoNet is part of the SOMA project, that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825469".

### Figure 8: Classifier

The researchers can analyse the top 30 hashtag and clean the dataset deleting any hashtag that does not interest them. Then, they obtain the main clusters for the filtered dataset and they can choose the two main clusters she wants to study.

As a result, the classifier can use as many as 10/100/100× more hashtags for classification than with any realistic fully manual approach, without sacrificing accuracy and possibly bringing to light previously unknown highly discriminative hashtags.

Cluster selection

These are the top cluster of dataset. Choose 2 cluster for classification and specify labels for them. Click on "Finish" when you're done.

☐ ["ioivotono","referendum","iodicono","ottoemezzo","renziacasa"]

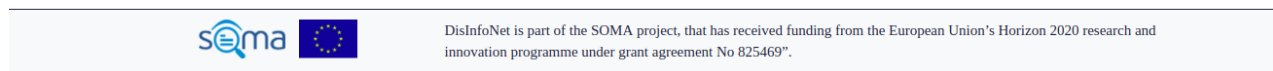
☐ ["bastaunsi","ioivotosi","bastaunsi","si","ioivotosi"]

☐ ["costituzione","no","si","italia","riforma"]

☐ ["referendumcostituzionale","4dicembre","referendum4dicembre","votono","votasi"]

☐ ["brexit","trump","eu","italy","news"]

☐ ["renzi","pd","accozzaglia","governo","cuperlo"]



**Figure 9: Cluster selection**

She is asked to give a label and a description to the clusters she chooses.

Cluster selection

These are the top cluster of dataset. Choose 2 cluster for classification and specify labels for them. Click on "Finish" when you're done.

☐ ["ioivotono","referendum","iodicono","ottoemezzo","renziacasa"]

☒ ["bastaunsi","ioivotosi","bastaunsi","si","ioivotosi"]

☐ ["costituzione","no","si","italia","riforma"]

☐ ["referendumcostituzionale","4dicembre","referendum4dicembre","votono","votasi"]

☐ ["brexit","trump","eu","italy","news"]

☐ ["renzi","pd","accozzaglia","governo","cuperlo"]

☒ ["bastaunsi","ioivotosi","bastaunsi","si","ioivotosi"]
 

Label

Description

☒ ["costituzione","no","si","italia","riforma"]
 

Label

Description

DisInfoNet is part of the SOMA project, that has received funding from the European Union's Horizon 2020 research and innovation programme under grant agreement No 825469.

**Figure 10: Cluster label and description**

## About

The About page, accessible from the header, provides useful information on the Luiss Data Lab and SOMA project.



## About us

**DisInfoNet** Toolbox is developed by [LUISS Data Lab](#), a Research Center based at the Department of Political Science of **LUISS** Guido Carli. The center carries out basic and applied research in the field of Big Data and digital transformation, relying on a multidisciplinary perspective.

**DisinfoNet** is powered by [SOMA](#) “Social Observatory for Disinformation and Social Media Analysis”, a H2020 Project aimed at supporting, coordinating and guiding the efforts of researchers, fact-checkers and journalists contrasting online and social disinformation.



**Figure 11: About page**

## 6 Conclusions

The goal of this deliverable is to describe the main features of the DisinfoNet tool. Starting from what has been described in the document D3.3: Data Intelligence toolkit description the activity has seen the study of algorithms and architectural models to be implemented in the definition of the tool. Through the use of the Scrum framework, functional requirements, platform navigation flows, user experience and user interface, logical architecture have been defined. Through the use of example datasets, the first work has been realized, which represents the basis to start testing the prototype.

On the basis of these results, development will continue with the following objectives:

- possible correction of errors in the development code;
- functionality testing;
- bug fixing;
- graphical update of the platform;
- improvement of UX/UI components;
- testing of new input datasets;
- testing with external users;

As a result of the results obtained, new versions of the software update will be released, able to meet the functional requirements and to guarantee a high operating performance of the DisinfoNet toolkit.