

Rights, Equality and Citizenship (REC)
Programme of the European Commission
(2014-2020)



Monitoring and Detecting Online Hate Speech

D3.1: MANDOLA Monitoring Dashboard[†]

Abstract: The aim of this document is to present the implementation of MANDOLA monitoring dashboard, as well as to be used as a user manual. A detailed study of the architecture and the hate speech detection analysis is presented with a description on the implementation and usage of dashboard's content. Each of the dashboard's pages is described and the techniques used for better user experience are explained.

Contractual Date of Delivery	September 2016
Actual Date of Delivery	September 2016
Deliverable Security Class	Public
Editors	D. Stefanidis, D. Paschalides, G. Pallis
Contributors	All MANDOLA partners
Quality Assurance	M. D. Dikaiakos

[†] This project is funded by the Rights, Equality and Citizenship (REC) Programme of the European Commission.

The *MANDOLA* consortium consists of:

FORTH	Coordinator	Greece
ACONITE	Principal Contractor	Ireland
ICITA	Principal Contractor	Bulgaria
INTHEMIS	Principal Contractor	France
UAM	Principal Contractor	Spain
UCY	Principal Contractor	Cyprus
UM1	Principal Contractor	France

Document Revisions & Quality Assurance

Internal Reviewers

Revisions

Version	Date	By	Overview
0.1	10 September 2016	D. Paschalides, D. Stefanidis, M. Hernando	MANDOLA Monitoring Dashboard
0.5	22 September 2016	D. Paschalides, D. Stefanidis, M. Hernando, G. Pallis, M. Dikaiakos	MANDOLA Monitoring Dashboard
1.0	29 September 2016	D. Paschalides, D. Stefanidis, M. Hernando, G. Pallis, M. Dikaiakos	MANDOLA Monitoring Dashboard

Table of Contents

DOCUMENT REVISIONS & QUALITY ASSURANCE	3
TABLE OF CONTENTS.....	4
TABLE OF FIGURES	6
1 INTRODUCTION	7
2 MONITORING DASHBOARD ARCHITECTURE	9
3 DATA STREAMS COLLECTION	11
3.1 TWITTER DATA STREAM COLLECTION.....	11
3.2 GOOGLE DATA STREAM COLLECTION	12
3.3 DATA STREAM PROCESSING.....	15
4 DATA ANALYSIS.....	16
4.1 MULTI-LINGUAL CORPUS	16
4.2 HATE-SPEECH DATA ANALYSIS	18
5 DATA STORAGE	21
5.1 DATABASE.....	21
5.2 DATA COLLECTIONS	21
6 API.....	23
7 DASHBOARD	24
7.1 HATE-MAP	24
7.1.1 <i>Heat-map</i>	24
7.1.1.1 Geo-clustering	25
7.1.1.2 Date range bar.....	26
7.1.1.3 Filtering Categories	27
7.1.2 <i>Hot-spot density map</i>	28
7.1.2.1 Hate-rate metric.....	29
7.1.2.2 Date range filtering	29
7.1.2.3 Categories filtering.....	30
7.1.2.4 Country drill-down	31
7.2 STATISTICS	31
7.2.1 <i>Time-line chart</i>	32
7.2.1.1 Zoom functionality	32
7.2.1.2 Data zoom aggregation	33
7.2.2 <i>Language usage chart</i>	34
7.2.3 <i>Hate rate per category chart</i>	35
7.2.4 <i>Hate rate per country chart</i>	35
7.2.5 <i>Hate rate per city chart</i>	36
7.2.6 <i>Countries per category chart</i>	37
7.2.7 <i>Cities per category chart</i>	38
7.2.8 <i>Time-line per category chart</i>	38
7.2.9 <i>Hate strength gauge</i>	39
7.3 RESPONSIVENESS AND MOBILE COMPATIBILITY	39

8 CONCLUSION.....	40
REFERENCES	41
ANNEX 1. API RESOURCES	42

Table of Figures

Figure 1: Mandola Dashboard Architectural Diagram	10
Figure 2: Ucy Framework Architecture	12
Figure 3: General Activities Diagram - Modules Structures And Links Providers	13
Figure 4: Activities Diagram - Services And Tools	14
Figure 5: Activities Diagram - Web Crawling Method.....	14
Figure 6: Kafka Message Queuing.....	15
Figure 7: Hate Filtering.....	16
Figure 8: Heatmap Interface	24
Figure 9: Heatmap Gradient.....	25
Figure 10: Geohash Definition Diagram	25
Figure 11: Heatmap Zoom Out.....	26
Figure 12: Heatmap Zoom In.....	26
Figure 13: Date Range Bar Stable.....	26
Figure 14: Date Range Bar Moved	27
Figure 15: Hotspot Map Interface.....	28
Figure 16: Hotspot Map Gradient	28
Figure 17: Calendar Panel For Date Selection.....	30
Figure 18: Statistics Date/Country Select Bar	31
Figure 19: Timeline Hate-Rate Chart. Axis X Is Date And Axis Y Is Hate Rate (%).....	32
Figure 20: Timeline Zoom In Functionality.....	32
Figure 21: Timeline With Aggregated Results Per Date	34
Figure 22: Hate Rate Pie Chart	34
Figure 23: Disable Language Functionality.....	35
Figure 24: Hate Rate Per Category	35
Figure 25: Hate Rate Per Country	36
Figure 26: Disable Country Functionality	36
Figure 27: Hate Categories Bubble Chart.....	37
Figure 28: Top Three Countries Per Category.....	37
Figure 29: Top Three Cities Per Category.....	38
Figure 30: Timeline Per Category	38
Figure 31: Hate Strength Gauge.....	39
Figure 32: Mandola Dashboard Mobile Interface.....	39
Table 1: Hate Categories Table	18
Table 2: Hatespeech Data Collection Properties	22
Table 3: Colour Representation In Heat Gradient.....	25
Table 4: Heat Map Filtering Category Buttons.....	27
Table 5: Colour Representation In Heat Gradient.....	28
Table 6: Hotspot Map Category Filtering Buttons	30
Table 7: Data Aggregation Conditions.....	33
Equation 1: Naïve Bayes Equation	19
Equation 2: Hate Rate Metric Equation	29

1 Introduction

The term "hate speech" covers all forms of expression which spread, incite, promote, propagate, support or justify every form of hatred, violence, discrimination, segregation, hostility against persons or against a religion or the divine, including, inter alia, intolerance expressed by aggressive nationalism and ethnocentrism, discrimination and hostility against religious groups, minorities, migrants and people of immigrant origin, incitement or threat to commit harm, an offence or a crime, humiliation and offense to the dignity, insult, defamation and harassment. Although the definition lists a number of groups, which are frequently seen to be the targets of hate speech it does not limit the possible targets to these groups alone. This is an 'open-ended' definition (Deliverable 2.1 deals with proposing a definition for "hate speech"), in accordance with the open-ended understanding of discrimination adopted by the European Court of Human Rights. The relevant actions of hate speech should usually have been performed on the ground of the belonging or not belonging of the victim to a real or supposed particular group or on the ground of one of the personal characteristics of the victim, which might be physical (e.g. colour, handicap), psychological, philosophical (e.g. religion, beliefs), or behavioural (e.g. exercise of a worship, belonging to a professional organisation) or have been committed against religion or the divine.

In recent years an ominous picture about online hate speech has started to materialise within cyberspace. Recent polls suggest that as many as four out of five respondents have encountered hate speech online and two out of five have personally felt attacked or threatened [1]. Although it is difficult to get accurate statistics about the spread of hate speech in cyberspace, the picture is becoming increasingly clear: the Internet is alarmingly effective at spreading hate speech – so much so that most Internet users have encountered it at some point. To make matters worse, hate speech usually targets the most vulnerable groups within society: children, minorities, and immigrants – groups that by definition have little capacity to protect themselves, both in the online and the physical worlds. There are two major difficulties in dealing with online hate speech: (i) Lack of reliable data that can show detailed online hate speech trends. (ii) Poor awareness about how to deal with the issue since there is a fine line between hate speech and freedom of speech: the boundaries between "legal" and potentially illegal hate speech are sometimes blurred, and may vary between territories. The same speech might not be illegal in all E.U. countries, and might in some countries be illegal only if some additional circumstances are noticed.

MANDOLA plans to fill this gap by: (i) monitoring the spread and penetration of online hate-related speech in Europe and in member states using big data approaches; (ii) providing policy makers with actionable information that can be used to promote policies that mitigate the spread of online hate speech; (iii) providing ordinary citizens with useful tools that can help them deal with online hate speech; (iv) transferring best practices among member states.

In this deliverable, the multi-lingual monitoring dashboard¹ is presented, which has been developed, in order to offer reliable information about online hate speech enabling users to focus on their geographic region ranging from their city to their country to the entire European Union. The dashboard uses Twitter and Web sites as sources of possible hate-related online content.

The rest of this deliverable is structured as follows: Section 2 presents the architecture of the monitoring system. The data stream collection mechanism is presented in Section 3. Section 4 describes the data processing procedure that takes place. Section 5 presents how data is stored. Section 6 presents the main features of API. Section 7 presents the MANDOLA Monitoring dashboard functionalities and Section 8 concludes this deliverable.

¹ Currently available on <http://mandola.grid.ucy.ac.cy:3000>

2 Monitoring Dashboard Architecture

The monitoring dashboard in MANDOLA handles two types of **data streams collections**. The first one is the **Twitter data stream** and the second one is the **Google data stream**. The data streams are handled through a distributed publish-subscribe messaging system named **Apache Kafka** [2]. Kafka feeds the **hate-speech data analysis** module and also collects a data sample set in order to create the multi-lingual corpus based on **hate filtering** module. The **hate-speech data analysis** module utilizes sentiment analysis tools via the NLTK platform [3] in order to classify content whether it is hate-related speech or not. The multi-lingual corpus, which is used to train the classification model that exists in the **hate-speech data analysis** module, is given by experts, which are called **social scientists**. Social scientists classify the hate-related content in the given Twitter and Google sample set, based on its strength and its categories. The **hate filtering** module is used to conduct automatically an initial filtering of the sample set so that social scientists receive for review more relevant content (i.e., hate-related). A **multi-lingual corpus** for hate speech is given as input to the hate filtering module. The corpus has been initially built from hate databases such as the crowdsourcing database Hatebase [4], and from hate-related sentiment lexicons containing seed words such as the AFINN [5] lexicon. Specifically, Hatebase is a Canadian initiative that its ultimate goal is to build the largest online repository of structured, multilingual, usage-based hate speech. Hatebase consists of hate words and phrases, annotated with their offensiveness strength, their category and their meaning. The multi-lingual corpus is enriched by the input of social scientists. The corpus is given as input to **hate-speech data analysis** module. Section 4 describes the data processing procedure that takes place. When the processing is done, the output is stored in the **hate speech database** (MongoDB). More details about the data storage are described in section 5. The **dashboard** is connected with the **hate speech database** (MongoDB) via an **API** that is used to retrieve data required for the various types of data visualization supported by the Dashboard (heat map, charts etc.). The API functionalities and resources are presented in section 6. Figure 1 depicts the architecture of the MANDOLA Monitoring Dashboard. For the implementation of the dashboard, the Express application framework [6] has been used. Express is a minimal and flexible Node JS application framework that provides a robust set of features for web and mobile development. Express supports a thin layer of fundamental web application features, without obscuring Node JS features, with HTTP utility methods and middleware to support API implementation.

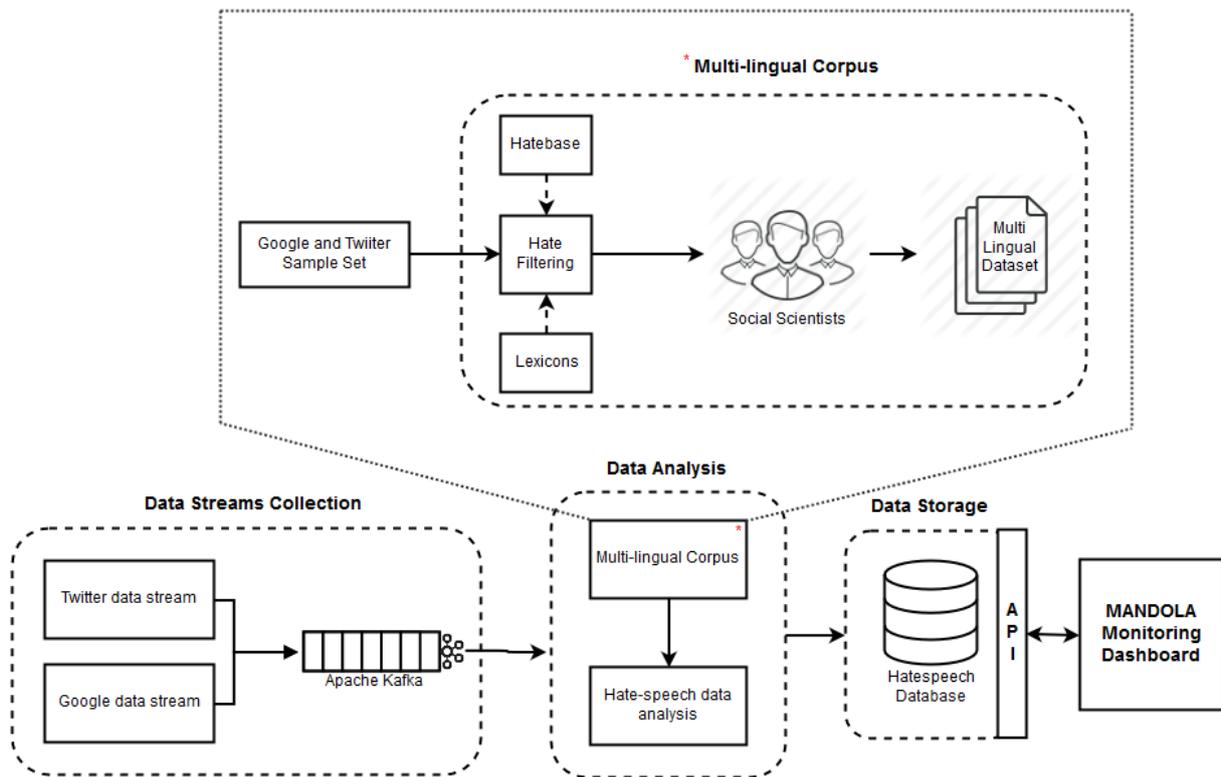


Figure 1: MANDOLA Dashboard Architectural Diagram

3 Data streams collection

The Data collection engine consists of two sub-modules that are responsible for collecting data from Twitter [7] and Google API [8]. This is done for purely research purposes in the MANDOLA project. The processing and storing is in line with article 7(1)(2) of the Personal Data Protection (Protection of the Human) Laws of 2001 to 2012 in Cyprus (N. 138(I)/2001 as it was modified with N. 37(I)/2003 and N105(I)/2012), which was submitted to the Cyprus Data Protection Commissioner on 18th December 2015. Specifically, no sensitive data is stored during the processing. The only information stored is a) the hate processing output, b) the date that it was published or updated, c) the language and d) the location. The location is converted in geo-hash with accuracy reduced to the level of city.

3.1 Twitter data stream collection

Most Online Social Network (OSN) platforms provide their data with streams based on IP where each machine collects data within a certain threshold. For the Twitter data collection, we used an advanced framework developed by UCY. This framework [9] is able to retrieve millions tweets per day, through an intelligent distribution mechanism in the most efficient way, compared to the state-of-the-art.

Twitter API policy restricts a single application of performing a large number of requests. In order to make authorized calls to Twitter API, each application must obtain an access token on behalf of a Twitter user. The framework (depicted in Figure 2) uses a Map-Reduce-like approach to overcome this limitation by partitioning tokens into a large number of small instances, greater than the available nodes, with some being replicated for performance objectives. Specifically, the framework asks from users to authorize a number of applications in order to gain access on the OSN API and performs the retrieval requests. This “crowd crawling” procedure increases the number of tokens that can be used during the retrieval process. The master is responsible for the workload distribution and the resource allocation of each instance. The master has knowledge about the whole system’s state and maintains the resources based on its needs. When an instance requires more resources, it sends a request to the master.

The Twitter API has two different flavors: *RESTful and Streaming*. The RESTful API is useful for getting things like lists of followers and those who follow a particular user. The Streaming API works by making a request for a specific type of data — filtered by keyword, user, geographic area, or a random sample — and then keeping the connection open as long as there are no errors in the connection.

UCY framework is able to efficiently: *i) collect data in asynchronous manner from OSN data storage servers using the RESTful API, ii) collect the OSN data stream in real-time using the Streaming API*. For the asynchronous data collection procedure, a distributed execution on two different instances has been implemented, which stores the data on a centralized server. During the collection procedure an instance retrieves all the properties of the requested users. During the experimental setting, two instances resulted to a dataset of more than one million complete Twitter user profiles for a single day. The Streaming API

collects data streams in real-time and the framework redistributes the load with respect to the keywords and the geographic area. This approach overcomes the restriction of collecting only 1% of the total Twitter stream through the Twitter API. The framework architecture is depicted in Figure 2.

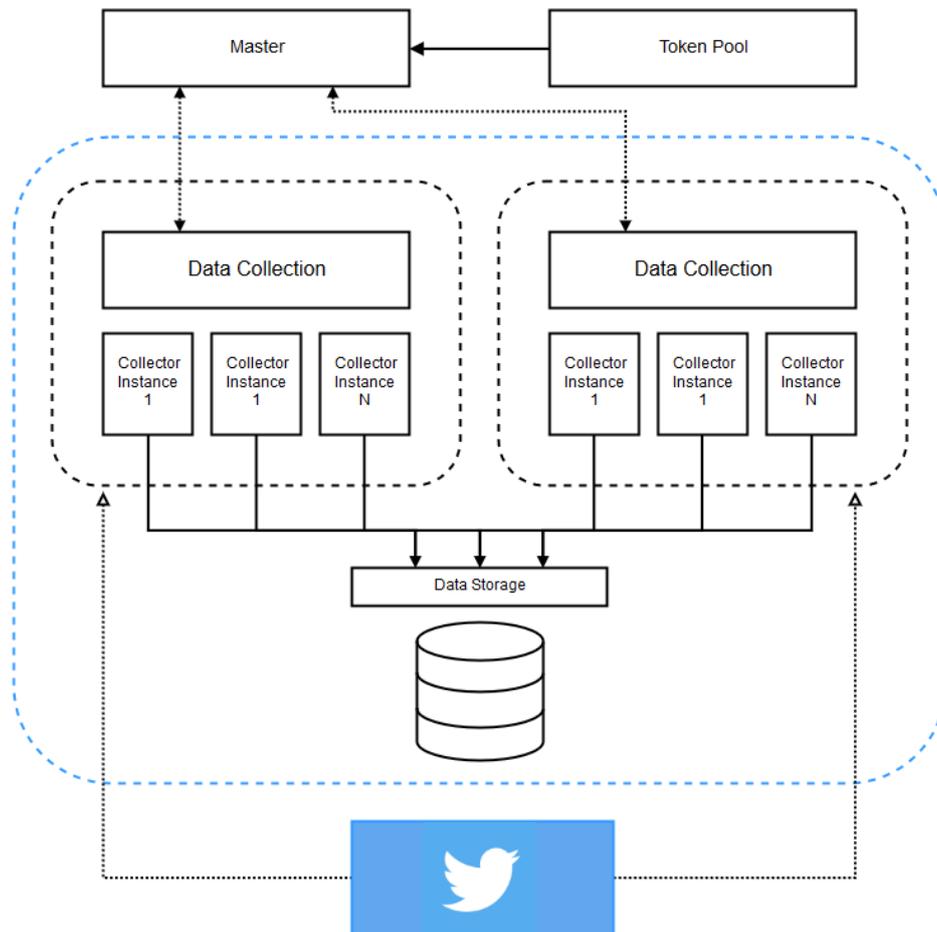


Figure 2: UCY Framework Architecture

Experimental results have shown that UCY framework is able to efficiently retrieve about 11 million tweets per day. This allows us to have a large collection of tweets without purchasing the Tweets from GNIP² (Twitter's data reseller).

3.2 Google data stream collection

The Google data collection is based on a meta-search engine that provides the content and information for in-depth analysis of possible hate-related speech. The following diagram (depicted in Figure 3) describes the general design of the module.

² <https://gnip.com>

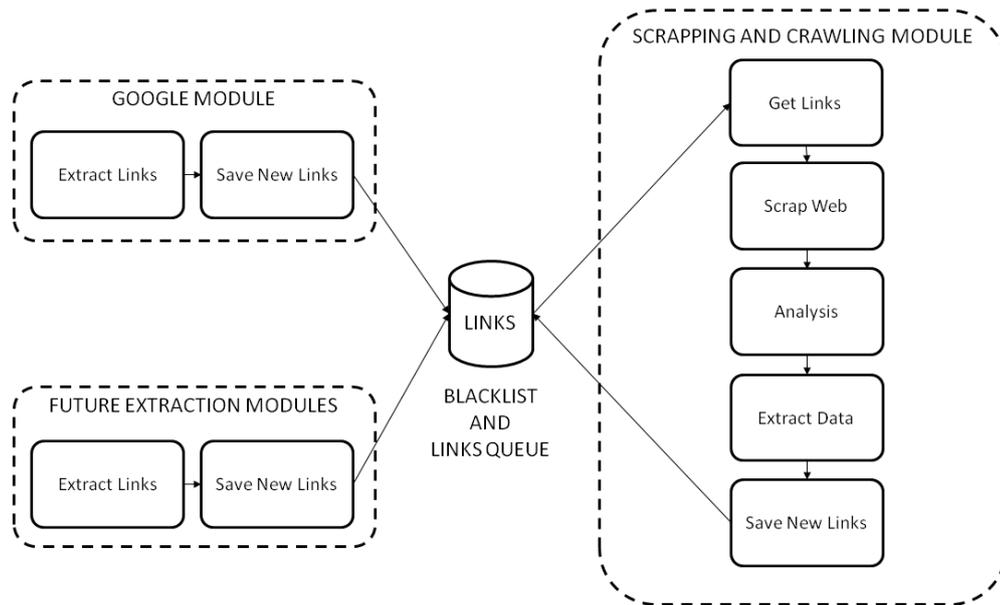


Figure 3: General Activities Diagram - Modules Structures and Links Providers

It has been designed to be able to easily add new modules (not only Google Search Engine), so as to provide more links from other search engines in the future. It captures all the visited links, non-visited links and the blacklisted web pages or domains. This helps to avoid repeating the visit to the same web page within a small period of time as well as optimize the search of web pages or domains which have been classified as “uninteresting”.

The meta-search engine is composed of a set of services and tools (Figure 4). There are external services like Hadoop API, Google API, detectlanguage.com, alexa.org and MANDOLA API and internal services like a link database, where the visited and blacklisted links are stored. The web scrapping and crawling module is developed with the Scrapy Framework³. Furthermore, some Python modules such as *RE* (Regular Expression library) and *w3lib* (to manage HTML webpage content) were used.

Google module receives a list of hate terms from Hadoop API service and uses Google's API to search for those terms. All links received are stored into the links database. The web crawling module fetches those links from the database and each result is scrapped to obtain the web content (clean of HTML tags), the URL and metadata like language, location and timestamp. Some of them are not extracted directly, but they need a previous short analysis or data processing. For instance, some metadata are defined by the external services, namely detectlanguage.com for language detection and alexa.org for location site information. In addition, each analysed web page is used like a resource to scrap new links (web crawling or web spidering). After the scrapping of the web page is done, the output is sent to the MANDOLA Monitoring Dashboard via its API endpoint.

³ <https://scrapy.org/>

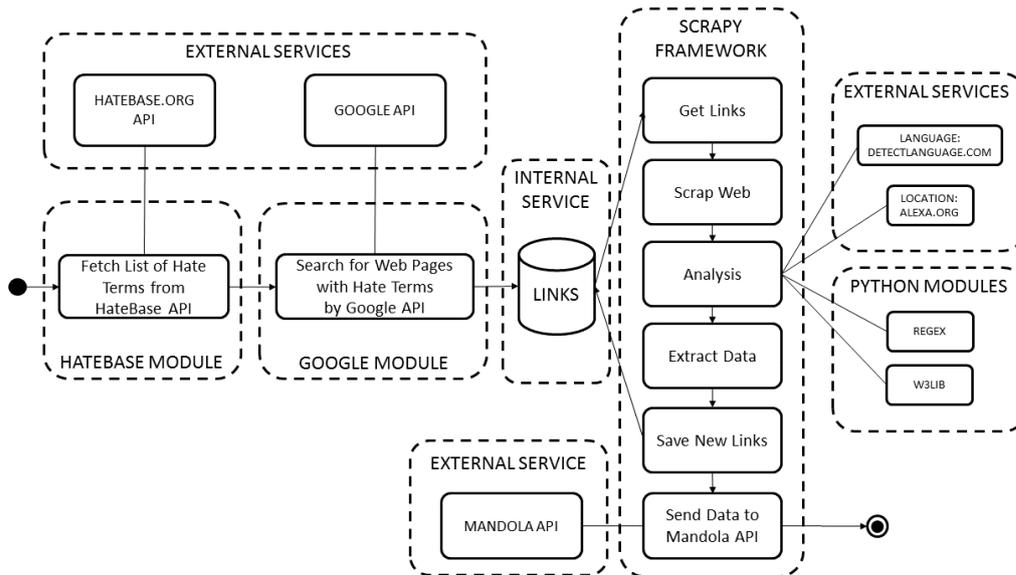


Figure 4: Activities Diagram - Services and Tools

As stated before, the web crawling method has been developed using the Scrapy Framework in order to take advantage of some features like the control of robot.txt and connection features.

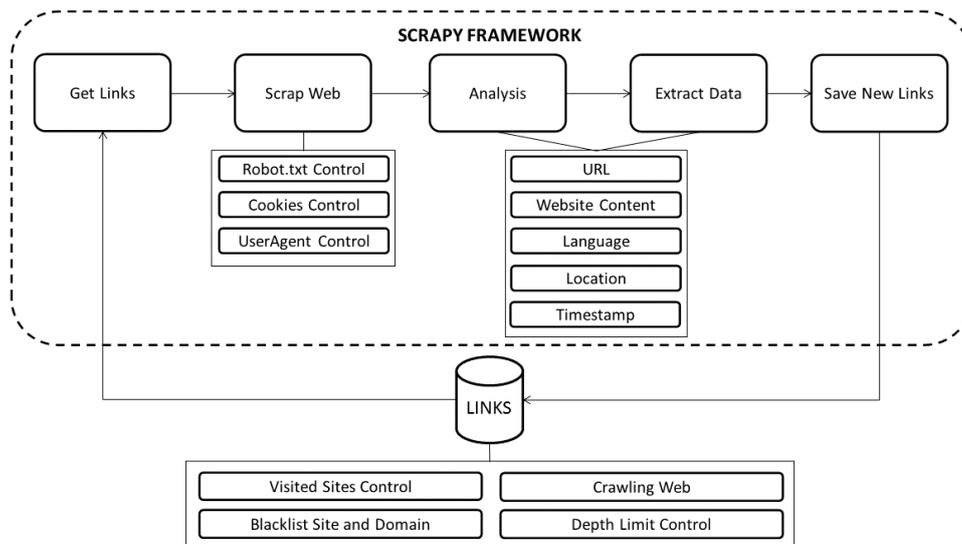


Figure 5: Activities Diagram - Web Crawling Method

Specifically, this method works by taking all the links of a website and storing them in a database. Through this database the system is able to control the visited web pages, filter the pages which are based on a blacklist (site or domain level) and control the depth limit for obtaining links.

3.3 Data Stream Processing

The concept of stream processing has been around for a while and most software systems continuously transform streams of inputs into streams of outputs. In the dashboard's case, we used Kafka, a distributed messaging system for large-scale stream processing.

In Kafka, the logical collections of messages (called topics) decouple producers, which are the sources of data, and consumers, which are the applications that process, analyze, and share data. Topics are partitioned for throughput and scalability. Partitions make topics scalable by spreading the load for a topic across multiple servers. Producers (Twitter and Google) are load balanced between partitions and consumers can be grouped to read in parallel from multiple partitions within a topic for faster performance. Partitioned parallel messaging is a key to high performance at scale. Another key to high performance at scale is minimizing the time spent on disk reads and writes. Compared with older messaging systems, Kafka eliminates the need to track message acknowledgements on a per-message, per-listener basis. Messages are persisted sequentially as produced, and read sequentially when consumed. These design decisions mean that non-sequential reading is rare, and allow messages to be handled at very high speeds. Experimental results have shown that Kafka performance scales linearly as servers are added within a cluster.

In MANDOLA Monitoring Dashboard, Kafka message broker (as depicted in Figure 6) receives tweets and Web pages as input and passes them to the hate-speech data analysis module. Google and Twitter are the producers which receive the input from the data streams and wrap it in a message, sending it to the centralized broker (called zookeeper) which provides distributed synchronization and adding it to the queue. Then the analogous consumer extracts the message from the queue and passes it as an input to the **hate-speech analysis** module, which process it, discards the message and stores the output in the hate-speech database.

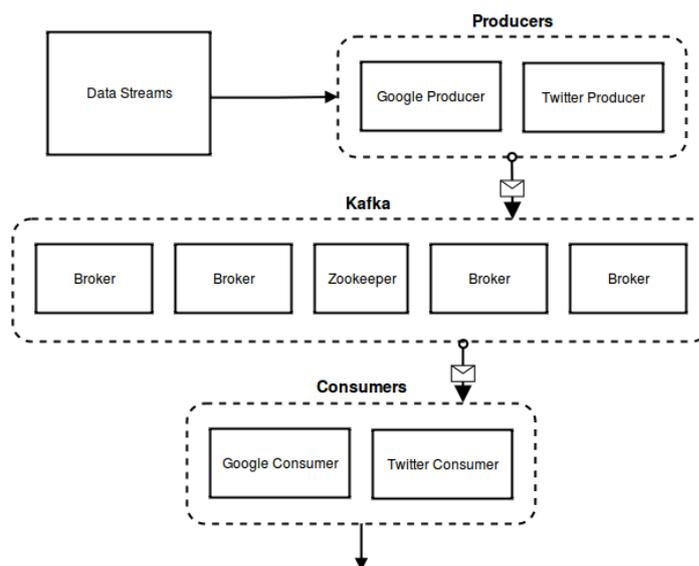


Figure 6: Kafka Message Queuing

4 Data Analysis

4.1 Multi-lingual Corpus

Within the context of MANDOLA project, a multi-lingual corpus was produced, which contains hate-related words. For the multi-lingual corpus creation, as a first step we used a seed set of words from the crowdsourcing database Hatebase and from AFINN, which is a valence sentiment lexicon.

The next step is to enrich the corpus taking into account the collected sample data sets (Twitter and Google). Social scientists' task is to study the collected sample set of Tweets and Web pages and identify which of them include hate-related content. However, due to the large volume of collected data sets and the fact that the majority of the collected tweets and Web pages do not include hate-related speech, the data sets that are sent to social scientists have been preprocessed by a lightweight filtering mechanism (Figure 7) in order to identify whether or not the data may contain hate-related content.

This filtering procedure consists of the following steps: text preprocessing, abbreviation extraction, language detection, text encoding (i.e., sentence and word tokenization, stop-word removal, punctuation removal, word stemming and noise removal) and sentiment analysis tasks through the NLTK platform⁴. NLTK is a leading programming framework that provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, semantic reasoning and wrappers for industrial-strength NLP libraries. So far, the filtering procedure has been developed for the following languages: English, Greek, Spanish and French. In the next paragraphs, the workflow to create a multi-lingual corpus is described.

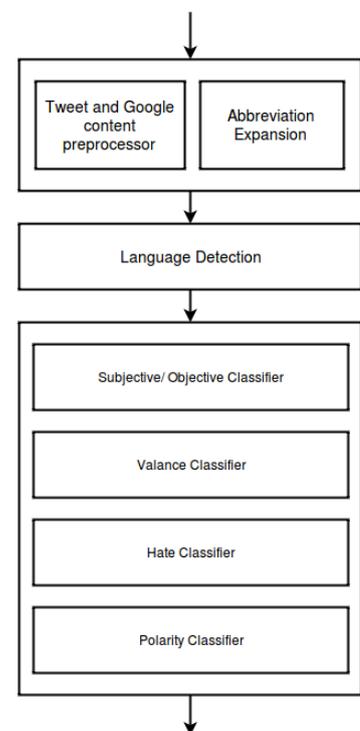


Figure 7: Hate filtering

Twitter and Google pre-processor

The first step of the filtering procedure is to strip the input content from any non-linguistic feature such as URLs (e.g. <http://www.google.com>), hashtags (e.g. #MANDOLA) and “user mentions” (e.g. @MANDOLA_project). This procedure removes the noise from the content and improves the quality of the results. The hashtags usually consist of concatenated words that cannot be separated by machine; this results to the expansion of the vocabulary with non-informative words. The same applies to the “user mentions”.

⁴ Natural Language Toolkit - <http://www.nltk.org/>

Input	@LosAngeles Los Angeles is a city in USA. #LARocks
Output	los angeles is a city in usa.

Abbreviation Expansion

The next step is the abbreviation expansion, which is based on the dictionary which has been created by parsing the Web site <http://slangit.com/>. This site maps abbreviations used in today's on-line chatting to their definitions. The goal of this step is to replace abbreviations with their unabbreviated meanings (e.g. lol = laughing out loud). This improves the quality of the results for the hate filtering by presenting any hate-related words hidden in the abbreviations.

Language Detection

After the first two steps of the hate filtering workflow, the linguistic content goes through the language detection that classifies the content of the input to its respective language. NLTK [9] provides the libraries to tokenize the content into a list of words by splitting all the punctuation in separate tokens and compute the language probability depending on which stop words (provided by NLTK) are used. The language is detected by computing the intersection between the content word list and each language's stop-word list; the language with the most stop-words in a sentence is considered as the default one.

Input	los angeles is a city in usa.
Output	["los", "angeles", "is", "a", "city", "in", "usa"]

English	Spanish	French	Greek
3 ["a", "is", "in",]	2 ["los", "a"]	0	0

Sentiment Analysis

Subjective/Objective Classifier. A subjective sentence expresses feelings, views, or beliefs. With sentence level subjectivity each sentence in a given document is analyzed and checked to be subjective. When necessary, the subjective sentence can be further classified as being of positive or negative semantic orientation. Subjective classifier calculates the probability that the input is either objective or subjective. Hate-related speech has orientation to opinionated and arguing topics, thus subjective sentences [10] are more likely to be hate-related than objective ones. In this framework, the subjectivity classifier library from NLTK is used, which has been trained using subjectivity clues from the Multi-Perspective Question Answering (MPQA)⁵ corpus.

Valence Classifier. Valence classifier provided by NLTK uses AFINN, a list of English words rated for valence with score between -5 and 5. These words have been manually labeled. Experiment results [11] have shown that negative valence is likely to promote hatred or

⁵ http://mpqa.cs.pitt.edu/lexicons/subj_lexicon/

discrimination against a person or a group. By filtering out the positive valence content, the corpus is reduced in order to balance hate and not hate content.

Hate Classifier. The hate classifier is based on a Hatebase sentiment lexicon containing all the possible Hatebase terms in addition to their hate score from 1 to 10. The terms provided by Hatebase are hate specific for several categories such as ethnicity, nationality, gender etc. and their usage often expresses hatred against the referent. If the hate classifier produces score greater than 1, then it is considered as a sighting of hate related words and marks the input as hate suspicious.

Polarity Classifier. The polarity classifier is based on VADER – Valence Aware Dictionary and Sentiment Reasoner classifier provided by NLTK. VADER is a lexicon-based sentiment analysis that performs well on text originating from social media [12]. Through this classifier the negative sentiments are kept since hate speech has negative meaning.

Manual Classification by Social Scientists

The output from the hate filtering procedure is given to social scientists. Social scientists rate the texts for hate-speech strength and hate-speech category (see Table 1). Afterwards, a kappa statistics test is performed to define the actual agreement between 2 ratings. This statistical model takes into account also the probability of 2 ratings to agree by chance. After kappa statistics, the example sets where the ratings agree in less than 80 percent are discarded because 80 percent can be considered a strong agreement. The output of this process will create a ground truth dataset which will be utilized as a training dataset for the classifier.

Table 1: Hate Categories Table

Hate Category	Target Example
Ethnicity	chinese people, indian people, paki
Nationality	nigga, black people, white people
Religion	religious people, jewish people
Gender	pregnant people, cunt, sexist people
Sexual	orientation gay people, straight people
Disability	retard, bipolar people
Class	ghetto people, rich people

4.2 Hate-speech Data analysis

To determine whether an input is considered hate-speech or not is a standard classification problem, which can be solved by training a text classifier with a corpus consisting of hate and not hate speech. The corpus used is the one collected and enriched from the social scientists. The corpus is in several languages from the Member States and thus the classifier must be language agnostic and must perform multi-lingual classification. Performing this classification on the input from Twitter and Google data streams, the classifier must have real time classification support without reducing the quality of the results. Furthermore, considering that a hate-related input can be labeled with different hate categories, such as religion, nationality and ethnicity, the classifier must also support multi-label classification,

with the hate categories being the labels.

Several classification approaches, such as Support Vector Machines, Decision Trees and Naïve Bayes, have been studied in order to identify the most appropriate one for MANDOLA Monitoring Dashboard. Naïve Bayes has been selected as the most suitable classification algorithm for the hate-speech classification. Naive Bayes is a simple multiclass classification algorithm with the assumption of independence between every pair of features. Naive Bayes can be trained very efficiently. It works well for both numerical and textual data and it is easy to implement. Within a single pass to the training data, it computes the conditional probability distribution of each feature given label, and then it applies Bayes' theorem to compute the conditional probability distribution of label given an observation and uses it for prediction. Experimental results have shown that the Naive Bayes classifier converges quicker than discriminative models like logistic regression.

In order to train the classifier, a balanced training dataset (called the ground truth data set) has been built from the sample set of tweets and Google pages that have been evaluated by social scientists. Initially, the hate corpus is converted into a word frequency table. Afterwards, a likelihood table is created by finding the probabilities of words in a sentence. Then the Naïve Bayesian equation (Equation 1) is used to calculate the posterior probability (denoted by P) for hate and non-hate class (denoted by c) for each Web content (denoted by x). For the category classification, each category has its own classifier trained in the same way as the hate-related classifier using the enriched data from the social scientists.

$$P(c | x) = \frac{P(x | c)P(c)}{P(x)}$$

$P(c | X) = P(x_1 | c) \times P(x_2 | c) \times \dots \times P(x_n | c) \times P(c)$

Equation 1: Naïve Bayes Equation

The same training approach has been applied for all the Member State languages that MANDOLA Monitoring Dashboard supports. These languages are specified by the social scientists input. Specifically, each language has its own classifier and, via the language detection module, the input is guided to its respective language classifier. The pseudocode of Naïve Bayes Algorithm is described below:

Naïve Bayes Algorithm	
Input = {D: Docs, C: Classes}	
Output = {score}	
TrainNB(C, D)	
1	V = ExtractVocabulary (D)
2	N = CountDocs (D)
3	for each c in C do
4	N _c = CountDocsInClass (D, c)
5	prior[c] = N _c / N
6	text _c = MergeTextOfAllDocsInClass (D, c)
7	for each t in V do
9	T _{ct} = CountTokensOfTerm (text _c , t)
10	for each t in V do
11	cond_prob[t][c] = CalculateLikelihood (T _{ct} , V, T, c)
12	return V, prior, cond_prob
ApplyNB(C, V, prior, cond_prob, d)	
1	W = ExtractTokensFromDoc (V, d)
2	for each c in C do
3	score[c] = prior[c]
4	for each t in W do
5	score[c] * = cond_prob[t][c]
6	return max(score)

Taking into account that a large amount of data should be processed, the classification process is sped up via the Apache Spark Streaming⁶ open-source framework. Apache Spark framework includes a suite of machine learning algorithms, called MLlib⁷. Specifically, MLlib supports the multinomial Naive Bayes. Within that context, each observation is a stream (Tweet or Web page) and each feature represents a term whose value is the frequency of the term. In MANDOLA Monitoring Dashboard's case, Apache Spark MLlib's implementation of Naive Bayes classifier is used for classifying the tweets and Web pages in real-time.

⁶ <http://spark.apache.org/streaming/>

⁷ <http://spark.apache.org/mlib/>

5 Data storage

The hate-speech data storage takes into consideration data privacy policies by protecting the user's identification from the information stored. So no sensitive information is stored in the hate-speech database and the data processing is done on the fly storing only data statistics calculated from the hate-speech data analysis module.

5.1 Database

The data (see section 5.2) are stored in MongoDB⁸ database, a NoSQL database which provides flexibility, scalability and performance required from applications such as the MANDOLA Monitoring Dashboard. MongoDB was selected from the various schemaless databases (e.g. CouchDB, Cassandra) because of the potent combination with Node JS, where they both employ the usage of JavaScript and JSON. Thus, Node JS can have direct access to MongoDB and retrieve the data faster and easier than other databases.

MongoDB also provides several indexes that can be used to improve query performance. MANDOLA Monitoring Dashboard utilizes both single and geospatial indexes. The first one is automatically generated for every data collection in MongoDB and uses the *_id* field. The second one is applied on the countries and cities collections in order to find the country and city that the input is located. By applying the geospatial index, a query on the applied collection can use *\$geoWithin*, *\$geoIntersects* and *\$near* operations which are required in order to discover in which country and city a point belongs to.

5.2 Data Collections

There are 4 basic collections of data which are stored in the hate database: hatespeech, hatebase, countries and cities. Hate speech stores its document in geojson format in order to make it easier for executing spatial queries and processing the results. Each document is considered as a Point in geojson and has several properties which are presented in Table 2:

Property	Description
hatescore	A number between 0 and 1, representing the hate strength of the tweet or Google page content, with 1 being the most hateful and 0 being the least.
posted_on	The date in UTC format of when the specific data was posted or last updated.
language	The language in which the hate speech is written.
country	The country from which was posted.
city	The city from which was posted.
tags	An array of the hate categories that have

⁸ <https://www.mongodb.com/>

	been detected in the input. The probable categories are ethnicity, nationality, gender, religion, sexuality, class, disability and history.
geohash	The encoded location of the data that is used in the grouping for the map visualization as well as for the protection of the user personal information.

Table 2: Hatespeech Data Collection Properties

The cities and countries collections contain geojson data⁹ that are used in the spatial queries for detecting the city and country of the input data, as well as calculating the statistics and visualizing them in the map. The Hatebase collection contains the parsed lexicon from Hatebase and is used by the hate-speech data analysis module to process the inputs and calculate the hate score.

Country and city detection. This procedure is required in order to place each text to a specific region rather than a point. By doing so, data for a specific country or city can be analyzed without using the coordinates pointing to a certain user, which is restricted. This is done by executing a geospatial query¹⁰ in MongoDB. The query returns the country and city that the given coordinates intersect with.

<pre>var country = db.countries.find({ 'geometry': { \$near: { \$geometry: { type: 'Point', coordinates: [[lng, lat]] } } } }).limit(1)[0];</pre>	<pre>var city = db.cities.find({ 'geometry': { \$near: { \$geometry: { type: 'Point', coordinates: [[lng, lat]] } } } }).limit(1)[0];</pre>
--	--

The geospatial query executes between the input's coordinates that are temporarily provided and the geojson collections of countries and cities. If the given coordinates intersect with the polygon created by a country's or city's geojson, then it is marked as originated from the specific country or city.

Coordinates encoding. The coordinates are stored in hate-speech database but not in the same format as they are received, in order to visualize the hate-speech data to a heat map. The accuracy of the coordinates is reduced to the point where there is no possibility that the user can be identified. This is done by removing decimals from the latitude and longitude of each coordinate. Thus, reducing the decimals of the coordinates of the hate speech's location down to 3, the location is stored without exposing the user's personal information and possible identity discovery.

⁹ <http://www.naturalearthdata.com/>

¹⁰ <https://www.mongodb.com/blog/post/geospatial-performance-improvements-in-mongodb-3-2>

6 API

The MANDOLA Monitoring Dashboard uses an API to retrieve data and calculations from hate-speech database. The *RESTful API* was developed using the Express's HTTP middleware and routing. Using the API, a developer can retrieve the data from the hate-speech database. In this section the API's main features are presented. MANDOLA Dashboard API has been developed as an independent module that can also be used by any 3rd party with authentication. Its main features consist of the retrieval and calculation of hate rate percentages, using the different factors.

The main functionalities are listed below:

- Calculating the usage percentage of each language in hate speech.
- Calculating the usage percentage of each category in hate speech.
- Calculating each language usage in each category in hate speech.
- Calculating the percentage of hate speech in each country.
- Calculating the percentage of hate speech for each category in each country.
- Calculating the average hate strength.
- Retrieving the encoded locations of hate speech.

The API's resource table is shown in the **Annex 1**.

7 Dashboard

The User Interface (UI) of MANDOLA Monitoring Dashboard's design and functions have been based on several other examples of monitoring dashboards, e.g. Google analytics, Gavagai, Tableau. There are three crucial parameters for MANDOLA Monitoring Dashboard functionality. The first one is the date range that the user expects to see the hate-speech analysis, the second one is the country and the third one is the categories to which the user wants to filter the results.

The interface uses bootstrap's grid system for responsiveness and compatibility with mobile devices. Each page is rendered by Node JS via Jade, which is a Node JS templating engine. In the next subsections, implementation details are provided.

7.1 Hate-map

Hate-map consists of two map visualizations, a heat-map and a hot-spot density map. The first one shows a heat-map visualization of the hate speech in a way where the hate strength (heats gradient gets warmer) is aggregated based on the hate in the specific regions. The second one uses a statistical analysis of the data per region in order to define areas of high occurrence versus areas of low occurrence and presents the data in terms of statistical confidence.

7.1.1 Heat-map

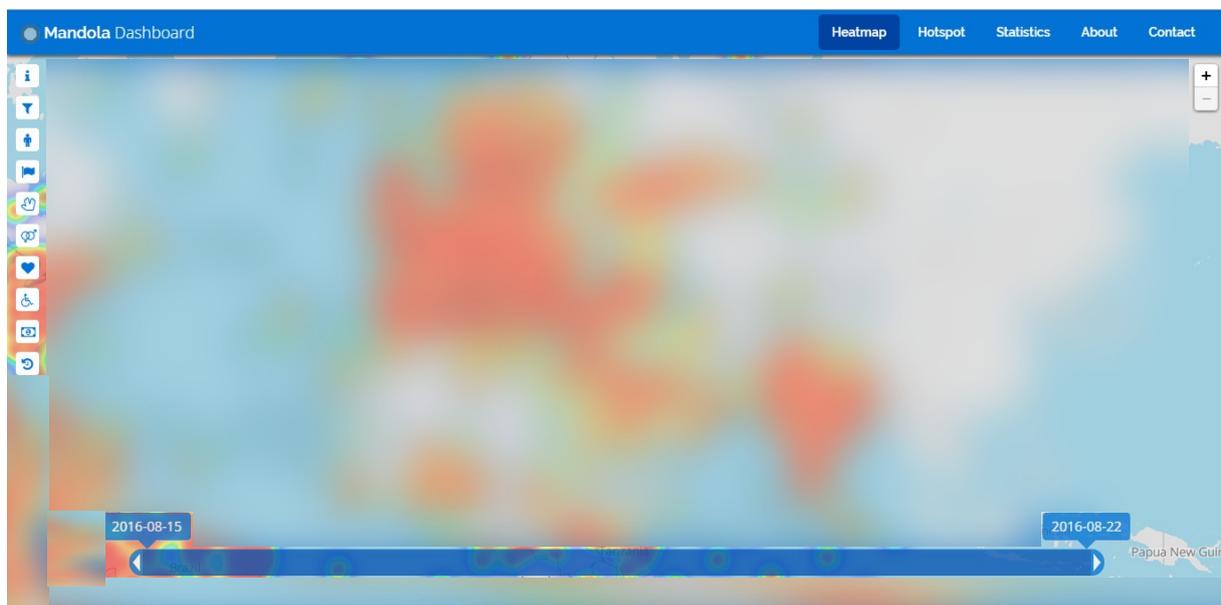


Figure 8: Heatmap Interface (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

Hate-map is a global heat map visualization approach. It uses leaflet¹¹, which is an open source interactive map JavaScript library.

¹¹ <http://leafletjs.com/>

The heat represents the percentage of hate-speech in the area. The percentage is characterized by the standard heat color range (Figure 9), with blue being the least and red the most possible hate-speech percentage.

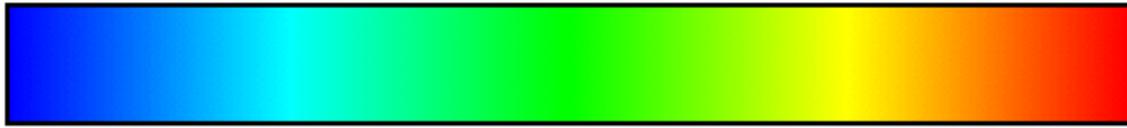


Figure 9: Heatmap Gradient

Hate-speech percentage	Color
Below 30%	Blue
Between 30% and 50%	Cyan
Between 50% and 70%	Green
Between 70% and 80%	Yellow
Between 80% and 90%	Orange
Above 90%	Red

Table 3: Colour Representation in Heat Gradient

7.1.1.1 Geo-clustering

The heat is drawn on the map with simpleheat¹², a lightweight JavaScript library for drawing heat maps with Canvas. Handling and presenting all the data to the user takes a toll on the user’s browser in terms of performance and memory. The browser’s iteration through all the data, in addition to the heat drawing, might lead to a slower performance, or even to the browser’s crash. To prevent this and make the user experience smoother, geo-hashes [13] were used to cluster the data.

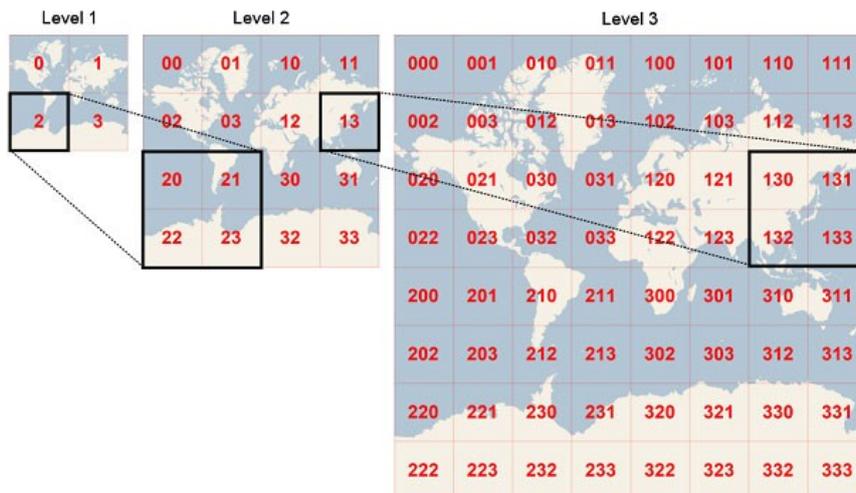


Figure 10: Geohash Definition Diagram

¹² <https://github.com/mourner/simpleheat>

Geo-hashes are a representation of coordinates, with each character recursively describing a quadrant on the map. The first character of a geo-hash describes a quadrant on the map; the second character describes a quadrant of the previous character's quadrant etc. (Figure 10).

This results to the reduction of each quadrants size, until the required resolution is reached. For example, consider these two geo-hashes: *tnxtuku9vxfr*, *tnxtu0v6vxgr*. To determine if these two coordinates are related in the same group on the map's scale the prefix *tnxtu* is used. The prefix is used as the clustering criterion in order to perform a data aggregation in MongoDB, using the specified grouping factor. The grouping factor is the number of characters that will be used from the prefix to calculate the clusters. For the grouping factor the map's zoom level is used. Leaflet provides 12 zoom levels, but dashboard's hate-map stops at level 3. So if the user is at level 3, which contains the whole globe, then he can view a more generic image of the hate-speech percentage. If the user changes to 12 which is a city specific zoom, then the user can view a more detailed image of hate-speech. By applying this geo-clustering technique, the heat drawing issue is addressed.



Figure 11: Heatmap Zoom Out (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)



Figure 12: Heatmap Zoom In (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

7.1.1.2 Date range bar

At the bottom of the hate map there is a horizontal bar representing the bounds of the dates that are possible for filtering. The user can drag the date labels to select the desired range and a request to the date API will be executed to present the results on the map, using the geo-clustering technique. The range selected can be dragged to filter the same number of days in a different year or month. The date request is repeated with the different parameters and the results are presented on the hate map.



Figure 13: Date Range Bar Stable



Figure 14: Date Range Bar Moved

7.1.1.3 Filtering Categories

The filtering categories can be applied by the filter button at the top-left corner of the hate-map. By pressing the filter button, a list of the buttons representing the possible categories appears. Table 4 presents the existing buttons with their functionality.

Table 4: Heat map Filtering Category Buttons

Button	Name	Functionality
	Filter	Shows the possible filter buttons.
	Ethnicity	Filters the hate-speech with ethnicity based content.
	Nationality	Filters the hate-speech with nationality based content.
	Religion	Filters the hate-speech with religion based content.
	Gender	Filters the hate-speech with gender based content.
	Sexual	Filters the hate-speech with sexual based content.
	Disability	Filters the hate-speech with disability based content.
	Class	Filters the hate-speech with class based content.
	Historic	Filters the hate-speech with historic based content.

	Information	Shows a panel explaining the possible filter buttons.
	Zoom in/out	By pressing – the map zooms out, by pressing + the map zooms in.

7.1.2 Hot-spot density map

Hotspot map is a hate map visualization approach in a global scale. It uses Highmaps, which is an open source interactive map JavaScript library and part of Highcharts¹³.

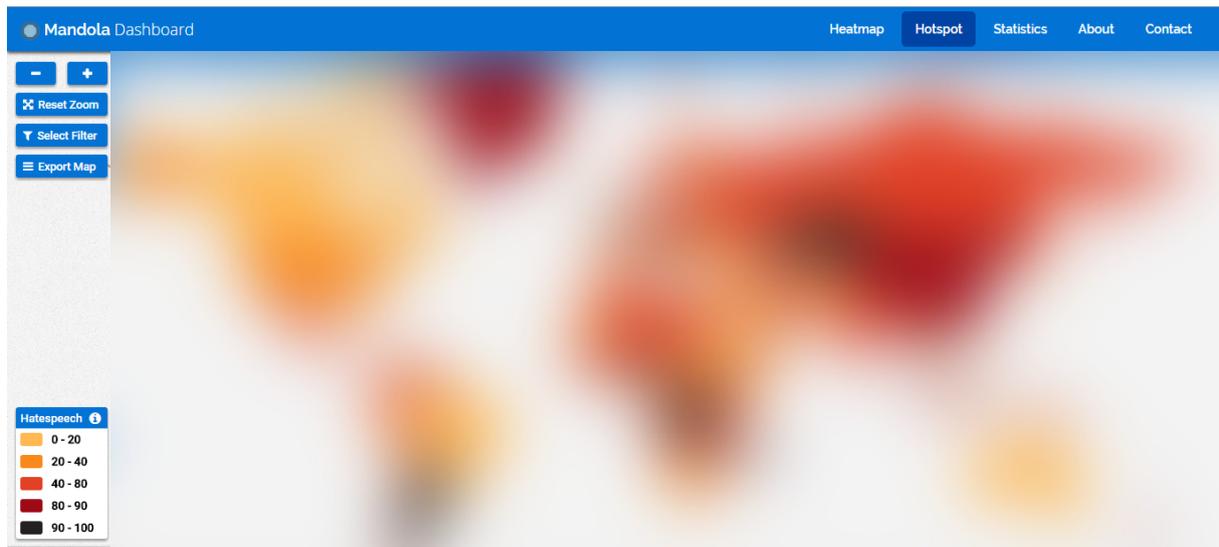


Figure 15: Hotspot Map Interface (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

The heat represents the percentage of hate-speech in the area. The percentage is characterized by the standard heat color range below, with texas-rose being the least and black the most hate-speech percentage possible.



Figure 16: Hotspot Map Gradient

Table 5: Colour Representation in Heat Gradient

Hate-speech percentage	Color
Below 20%	

¹³ <http://www.highcharts.com/>

Between 20% and 40%	
Between 40% and 80%	
Between 80% and 90%	
Between 90% and 100%	

7.1.2.1 Hate-rate metric

To show the heat that represents the hate-speech in the map, firstly, the hate-rate is calculated. The hate-rate of a country/city can be calculated using the formula presented in equation 2. Considering that each country has different population, the number of hate-related content is not a reliable source to measure the hate-speech percentage. By calculating the average hatescore, which corresponds to the hatefulness of the content, and multiplying it with the country's hate-related content number, a more reliable measure is achieved.

Hate-rate(A)	Hate-rate of country/city "A"
Hatespeech_Content(A)	#num of hatespeech content of country/city "A"
Most_Hatespeech_Content()	#num of hatespeech content of the country/city with the most hatespeech content
Average_Hatescore(A)	average hatescore of country/city "A"

$$\text{Hate-rate(A)} = \frac{\text{Hatespeech_Content(A)} * \text{Average_Hatescore(A)}}{\text{Most_Hatespeech_Content()}}$$

Equation 2: Hate Rate Metric Equation

After the calculation of the hate-rate of all countries/cities, data normalization is performed. Finally, the data value is converted to a percentage based on the total hate-speech rate.

7.1.2.2 Date range filtering

By pressing the *Select filter* button at the left side of the hate-map a modal will appear. Then, by clicking the date range input, a panel with two different calendars appears (one for the beginning and one for the ending date), where the user can select the desired date range. By clicking *Apply* the data is then filtered, and presented in the hotspot map.

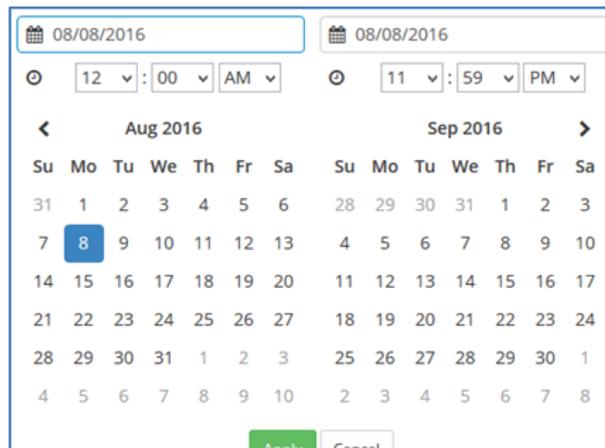


Figure 17: Calendar Panel for Date Selection

7.1.2.3 Categories filtering

The categories filtering can be applied by the button *Select filter* at the left side of the hate-map. By pressing the filter button, a list of the buttons representing the possible categories appears. Table 6 presents the existing buttons with their functionality.

Table 6: Hotspot map Category Filtering Buttons

Button	Name	Functionality
	Select Filter	Shows the possible filter buttons.
	Ethnicity	Filters the hate-speech with ethnicity based content.
	Nationality	Filters the hate-speech with nationality based content.
	Religion	Filters the hate-speech with religion based content.
	Gender	Filters the hate-speech with gender based content.
	Sexual	Filters the hate-speech with sexual based content.
	Disability	Filters the hate-speech with disability based content.

	Class	Filters the hate-speech with class based content.
	Historic	Filters the hate-speech with historic based content.
	Zoom in/out	By pressing – the map zooms out, by pressing + the map zooms in.
	Reset Zoom	Resets the zoom.
	Export Map	The exporting module allows users to download the chart as PDF, PNG, JPG or SVG vector images. It also allows printing the chart directly without distracting elements from the web page.
	Info	Shows a panel explaining the metric that is used in hotspot map. It also shows the size of the data being processed.

7.1.2.4 Country drill-down

The user can view a general map with all countries for the specified date range provided, but can also view a more detailed map for each country by clicking the desired country. A request is executed for a specified range and a specific country and the results will be loaded in the new map. The user can revert back to the previous zoom level, by clicking the *Back* button on the left side of the map.

7.2 Statistics

Statistics shows the data visualization of several metrics using the API calls. These visualizations consist of several chart and timeline types that were created using Highcharts. The Statistics page shows a general picture of the global hate-speech status, but can also present the results for a specific country. Both of these features can be filtered with a date range, just like the hate and hotspot map page.

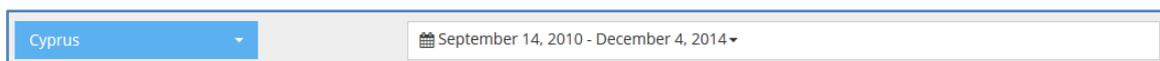


Figure 18: Statistics date/country Select Bar

By pressing the country select box, the user can select a country from the ones populating the *countries* collection in MongoDB. Then the results are filtered and presented. By clicking

the date range input, a panel with two different calendars (Figure 17) appears (one for the beginning and one for the ending date), where the user can select the desired date range. By clicking *Apply* the data is filtered and presented. In the right side of the calendar panel there are some shortcut selection to the most used date ranges. The default selection is *Custom range* where the calendars appear.

7.2.1 Time-line chart

The time-line chart is a representation of the hate-speech status in time, with the Y axis representing the percentage of hate-related speech and the X axis representing the date range.

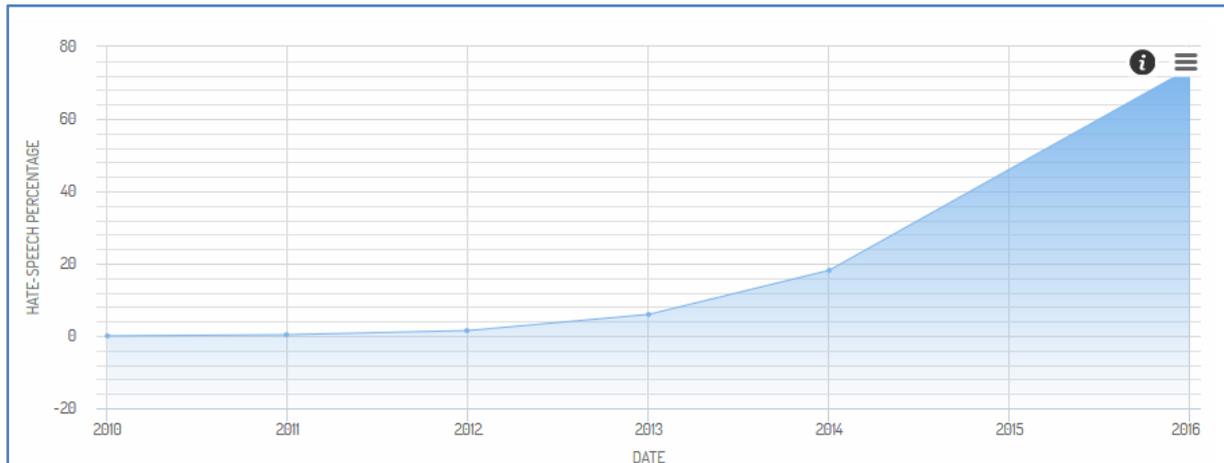


Figure 19: Timeline hate-speech percentage chart. Axis X is date and axis Y is hate-speech percentage (%)

7.2.1.1 Zoom functionality

The user can view a general picture of the results for the specified date range provided, but can also view a more detailed presentation for inner ranges by dragging and zooming the required range in the graph. By doing so a request is executed for the newly specified range and the results are loaded in the graph. The user can revert back to the previous zoom level by clicking the *Reset zoom* button on the top right corner of the graph.

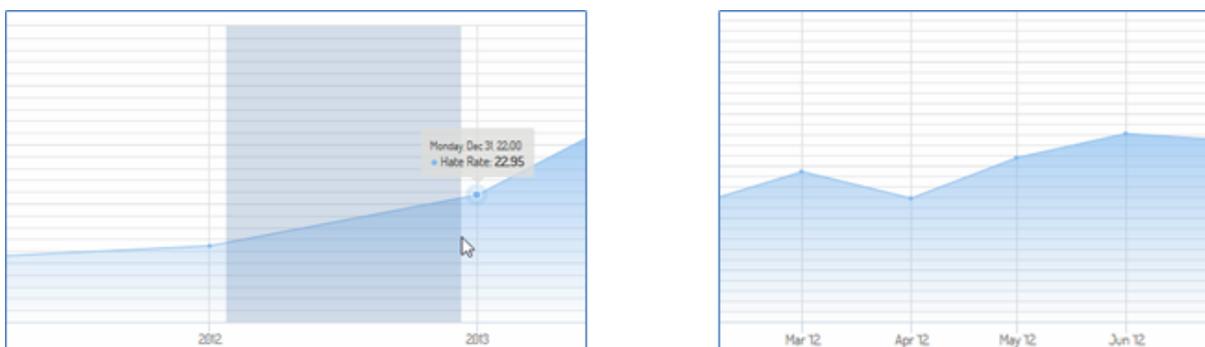


Figure 20: Timeline Zoom in Functionality

This functionality uses a date zoom aggregation technique, which is described in detail below.

7.2.1.2 Data zoom aggregation

Another issue similar to the heat drawing in hate-map (see 7.1.1.1), in terms of browser performance and memory load, is the time-line data representation in statistics. Drawing in all the possible dates in the specified range is not a possible due to browser's processing power limitations. Thus, a similar method to the one used for the location clustering was applied using timestamps. A time-stamp is the representation of a date and time for a certain event in a sequence of numbers. By calculating the difference of the maximum and minimum timestamps, a threshold can be set to present the analogous data for each request. Specifically, if the difference is less than or equal to 2 days, the results are presented in hours, if is greater than 2 days and less than or equal to 2 months, then the results are presented in days, etc. The following conditions used for the calculation of the difference and the condition¹⁴.

Condition	Representation	Format
$\leq 2 * 24 * 60 * 60 * 1000$	In hours	%Y-%m-%d %H
$> 2 * 24 * 60 * 60 * 1000$ && $\leq 62 * 24 * 60 * 60 * 1000$	In days	%Y-%m-%d
$> 62 * 24 * 60 * 60 * 1000$ && $\leq 730 * 24 * 60 * 60 * 1000$	In months	%Y-%m
$> 730 * 24 * 60 * 60 * 1000$	In years	%Y

Table 7: Data Aggregation Conditions

Based on the value of the condition, the analogous date format is selected and used as a grouping factor in MongoDB. The results are the formatted representation of the date with the total amount of hate-speech in that range and the average hate-score. The results are highly reduced without losing their quality and are presented to the user. A zooming event is set to the time-line chart that executes the same request with the selected range as timestamps and loads the data of the specified range with more details.

¹⁴ The conditional values are calculated by multiplying the milliseconds, seconds, minutes, hours and days of the required values. For example, 2 days are 2 days * 24 hours * 60 minutes * 60 seconds * 1000 milliseconds.

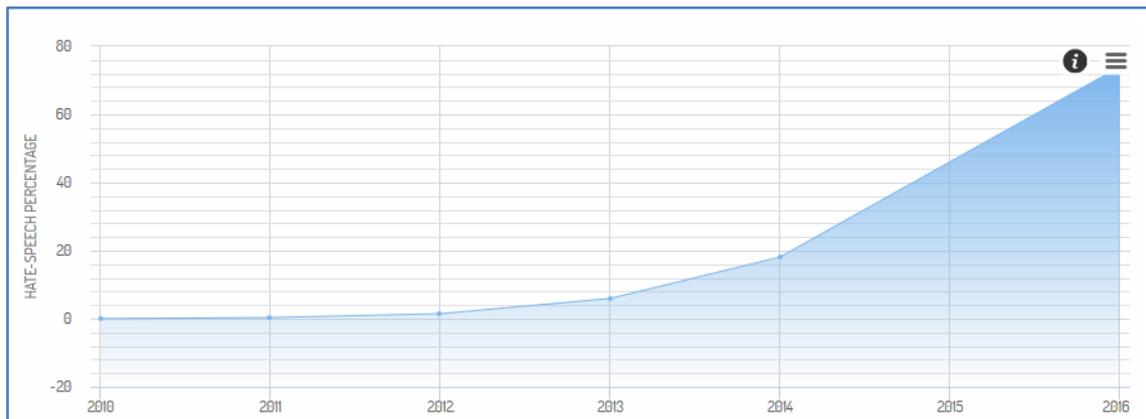


Figure 21: Timeline with aggregated results per date

7.2.2 Language usage chart

The language usage pie chart shows the top ten languages in utilization of hate-speech. The metric is done by calculating the hate-speech count for each language and finding out the percentage of that count against the total count of hate-speech during the predefined period of time. After this calculation is done, the ten languages with the most utilization are collected and presented through the chart.

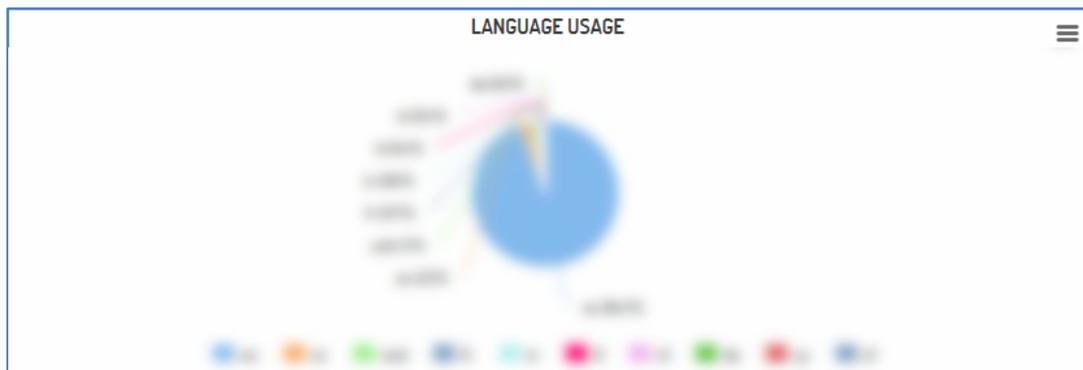


Figure 22: Hate-speech Percentage Pie Chart (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

The user can disable a language, to compare the other language's utilization, by clicking the desired language. Then the section of the pie chart representing the clicked language disappears and the percentages change in order to describe the remaining languages.

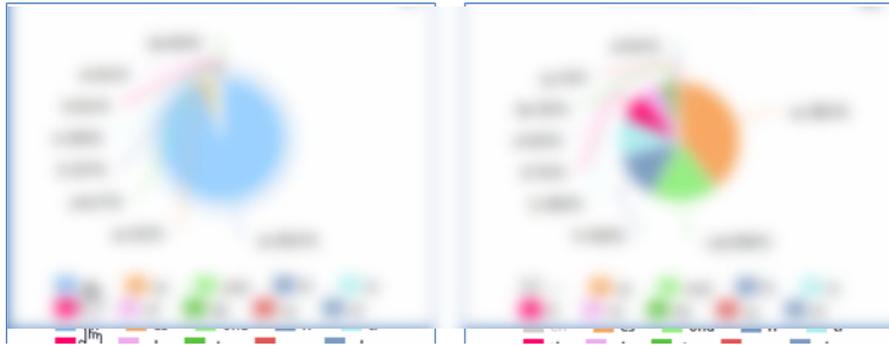


Figure 23: Disable Language Functionality (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

7.2.3 Hate-speech Percentage per category chart

The utilization percentage of each category is presented, globally or partially, for the selected country. The Y axis represents the categories defined and the X axis represents the total utilization percentage of each category in the hate-relate speech during the selected period.

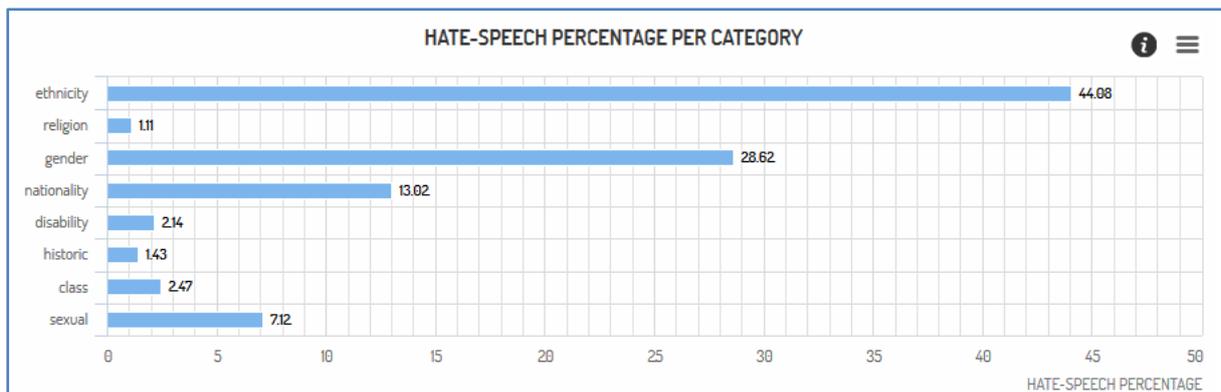


Figure 24: Hate-speech percentage per Category

The calculation of each category utilization is done by calculating the hate-speech count for each category and finding out the percentage of that count against the total count of hate-speech during the predefined period.

7.2.4 Hate-speech Percentage per country chart

This pie chart depicts the top ten countries in utilization of hate-speech when the option “general” is selected in the country select box. The metric is done by calculating the hate-speech count for each country and finding out the percentage of that count against the total count of hate-speech for the given date range. After this calculation is done, the 10 countries with the most utilization are collected and presented through the chart.

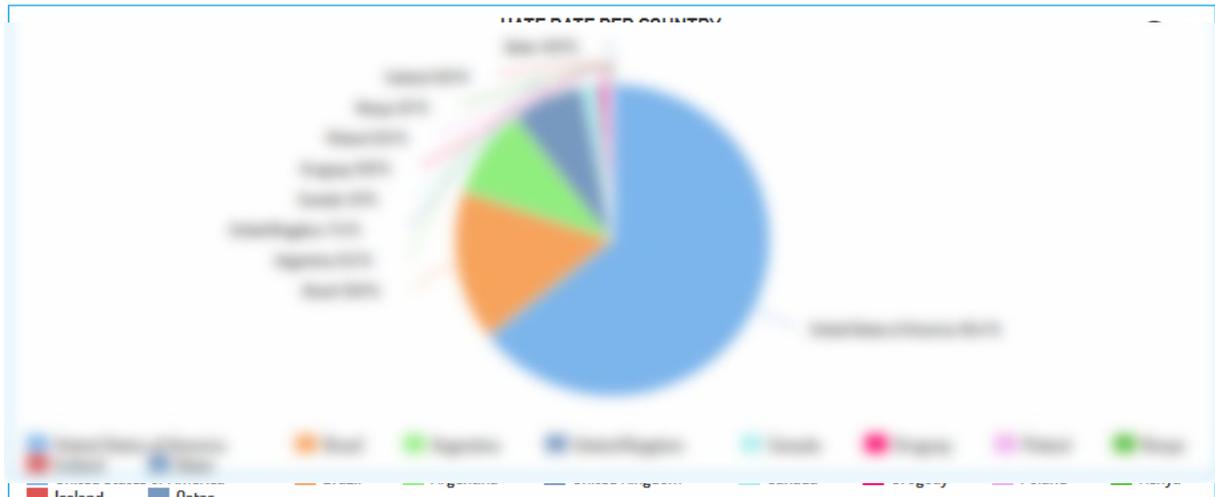


Figure 25: Hate-speech Percentage per Country (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

Similar to the language usage pie chart, the user can disable a country by clicking it. This will help him in comparing the remaining countries, in terms of hate-speech utilization. Then the section of the pie representing the clicked country disappears and the percentages are changing in order to describe the remaining countries.

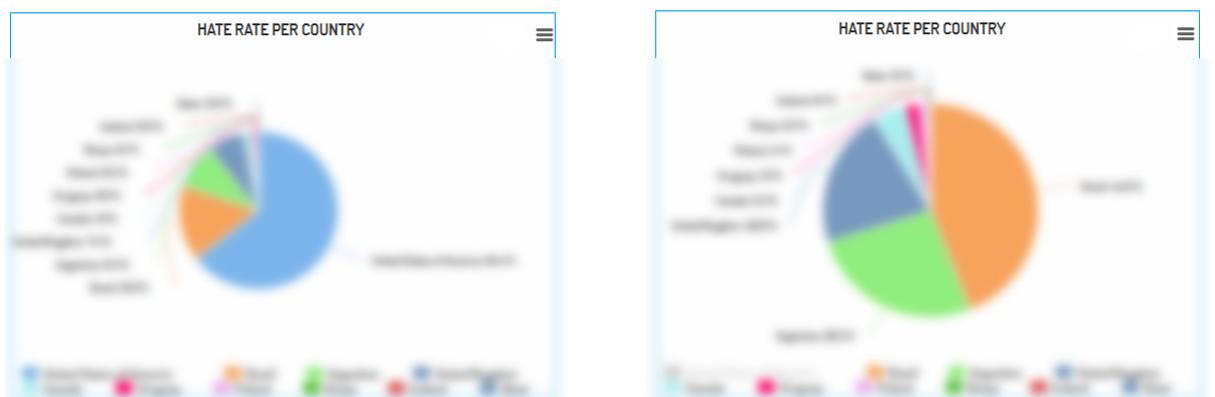


Figure 26: Disable Country Functionality (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

7.2.5 Hate-speech Percentage per city chart

This is a map representation of the selected country with circles of different radius and color. The color of the circle represents the category of hate-speech, and the radius represents the utilization of that category in the specified location. This chart uses the same geo-clustering technique as the hate-map, with the addition of calculating the average hate-score of each cluster, as well as the total count of hate-speech. The clusters radius is calculated by multiplying the average hate-score with the total number of hate-speech in the cluster.

Similar to the previous charts, the user can also disable a category in order to remove its clusters and show more specific results considering the remaining categories.

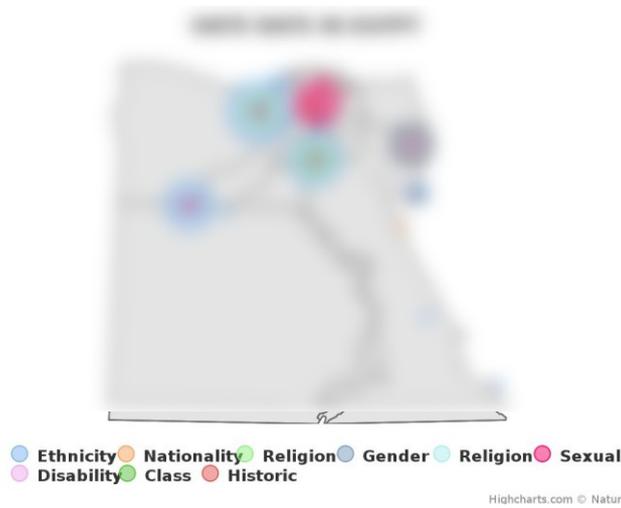


Figure 27: Hate Categories Bubble Chart (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

7.2.6 Countries per category chart

This bar chart shows the top 3 countries in utilization of each category. The Y axis represents the categories and the X axis the total utilization percentage of the hate-speech. It is visible in the general page of the statistics where no country is selected.

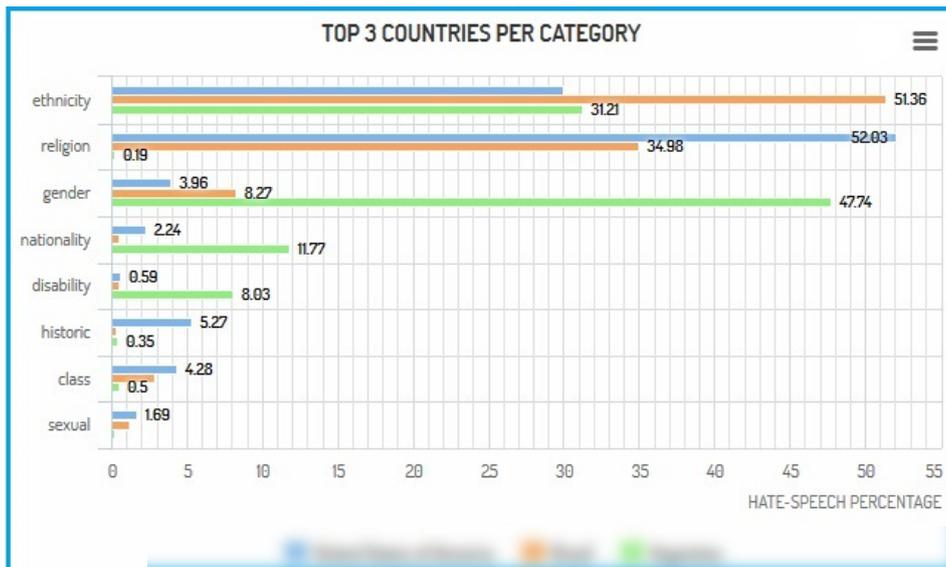


Figure 28: Top Three Countries per Category (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

The hate-speech percentage is given by calculating the hate-speech count for each category in each country, and finding out the percentage of that count against the total count of hate-speech during the predefined period in each country. After this calculation is done, the results are sorted in descending and the top three countries are presented.

7.2.7 Cities per category chart

This bar chart shows the top 3 cities in utilization of each category, in the selected country. The Y-axis represents the categories and the X-axis the total utilization percentage of the hate-speech. The calculation of hate-speech percentage is similar to the one used in *Countries per Category* bar chart in 7.2.6.



Figure 29: Top Three Cities per Category (Image is blurred intentionally due to the sensitivity of the statistical data visualized through the dashboard)

7.2.8 Time-line per category chart

The category timeline chart is the same representation as the general time-line chart, but separates the different categories. It uses similar date aggregation technique as the general timeline chart, but without the zooming functionality due to reduce performance by the use's browser because of the multi-line redrawing.

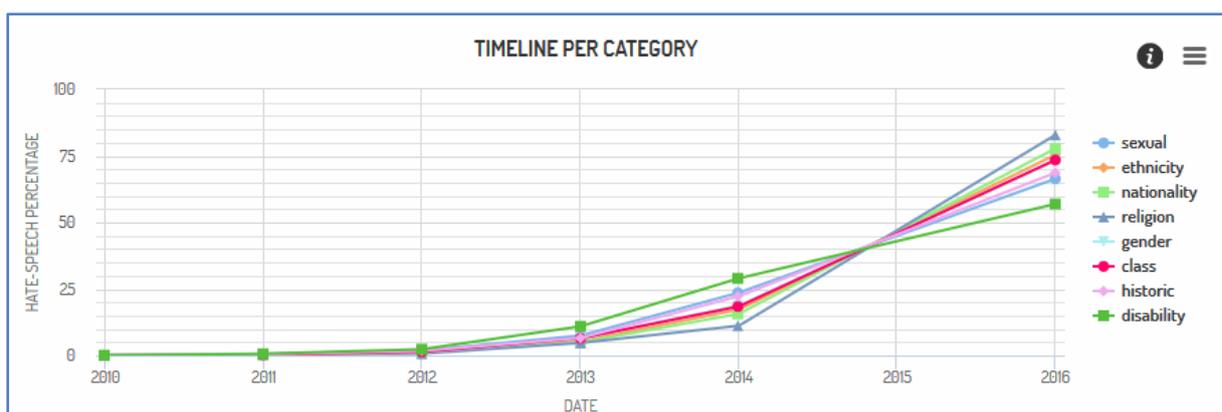


Figure 30: Timeline per Category

Similar to the other charts, the user can disable a category by clicking it, in order to compare the remaining categories. The X axis depicts the aggregated range based on the date

difference, as described to the first chart. The Y axis corresponds to the percentage of hate-related speech growth during that period of time.

7.2.9 Hate strength gauge

This is a gauge representation of hate strength¹⁵ in the specified date range and country. It has 3 stages of color based on the percentage of hate strength. The first one is 10% with the color green, the second is 60% with the color yellow, and the third one is 90% with the color red. Green state shows a non-critical state of hate speech usage, yellow shows a state with warning and finally red shows a dangerous state in terms of hate speech usage.

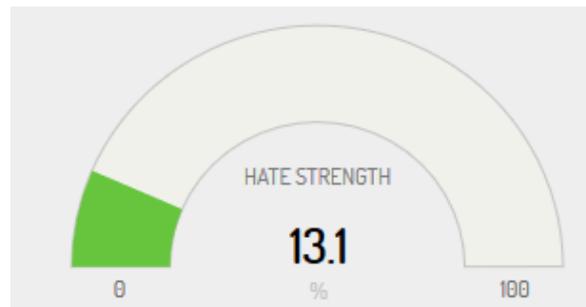


Figure 31: Hate Strength Gauge

7.3 Responsiveness and Mobile Compatibility

As stated above, the monitoring dashboard is based on the bootstrap's grid system to be compatible in mobile and other portable devices. Below is a presentation of the dashboard's view from a mobile device.



Figure 32: MANDOLA Dashboard Mobile Interface

¹⁵ Hate strength is the average hate score during the selected period of time and in the selected region.

8 Conclusion

In this deliverable a detailed description of the MANDOLA monitoring dashboard is provided and the different available data visualizations are described. The architecture of the dashboard and the basic user functionalities provided by the API are documented. Furthermore, an analysis on the data collection, processing and storing is described and documented, explaining the processing workflow from the two main data streams (twitter and Google pages) to the storing in the database.

References

- [1] "Countering hate speech online," 28 11 2012. [Online]. Available: <http://eeagrants.org/News/2012/Countering-hate-speech-online>.
- [2] Apache, "Apache Kafka," Apache, [Online]. Available: <http://kafka.apache.org/>.
- [3] "Natural Language Toolkit," NLTK, [Online]. Available: <http://www.nltk.org/>.
- [4] "Hatebase," Mobicracy, Sentinel Project for Genocide Prevention, 2008. [Online]. Available: <https://www.hatebase.org/>.
- [5] F. Å. Nielsen, "A new ANEW: Evaluation of a word list for sentiment analysis in microblogs," *arXiv:1103.2903 [cs.IR]*, p. 6, 15 3 2011.
- [6] E. M. Hahn, in *Express in Action Writing, Building and Testing Node.js applications*, 2016.
- [7] Twitter, "Twitter Development," Twitter, [Online]. Available: <https://dev.twitter.com/overview/documentation>.
- [8] Google, "Google API Explorer," Google, [Online]. Available: <https://developers.google.com/apis-explorer/#p/>.
- [9] H. Efstathiades, G. Pallis and M. D. Dikaiakos, "A framework for Twitter Data Collection," 2016.
- [10] J. Wiebe, R. Bruce, M. Martin, T. Wilson and M. Bell, "Learning Subjective Language," *Computational Linguistics*, vol. 30, pp. 277-308, September 2004.
- [11] T. Leader, B. Mullen and D. R. Rice, "Complexity and Valence in Ethnophaulisms and Exclusion of Ethnic Out-Groups: What Puts the "Hate" Into Hate Speech?," *Journal of Personality and Social Psychology* 96(1):170-82, 2009.
- [12] C. Hutto and Eric Gilbert, "VADER: A Parsimonious Rule-based Model for Sentiment Analysis of Social Media Text," *Georgia Institute of Technology*, p. 10, 2014.
- [13] K. Lee, R. K. Ganti, M. Srivarsa and L. Liu, "Efficient Spatial Query Processing for Big Data," *Georgia Institute of Technology*, p. 4.
- [14] Kumar, Shamanth, Morstatter, Fred, Liu and Huan, *Twitter Data Analytics*, Springer, 2014.

Annex 1. API Resources

This annex represents the urls to access the specified resource, as well as the type of request that it requires with an example of it. It supports all of the main functionalities described in section 6, with some additions in order for the monitoring dashboard to work.

Resource	Type	Example	Description
/api/city/:country	GET	/api/city/Cyprus?from=2010-09-14%2012:00:00&to=2014-12-04%2012:00:00&tags=ethnicity,nationality,religion	Returns the hate speech data for each city of the specified country.
/api/countries	GET	-	Returns the geojson data for all the countries.
/api/date/earliest	GET	-	Returns the date of the earliest tweet processed.
/api/date/latest	GET	-	Returns the date of the latest tweet processed.
/api/hatestrength	GET	/api/hatestrength?from=2010-09-14&to=2014-12-04	Returns the average hate strength value.
/api/list/countries	GET	-	Returns all the countries and their code.
/api/list/categories	GET	-	Returns an array with all the categories.
/api/language	GET	/api/language?from=2010-09-14&to=2014-12-04&batch=10	Returns the analysis of the language usage.
/api/category	GET	/api/category?from=2010-09-14&to=2014-12-04	Returns the analysis of the category usage.
/api/country	GET	/api/country?from=2010-09-14&to=2014-12-04&batch=10	Returns the analysis of the country hate speech rate.
/api/category/city	GET	/api/category/city?from=2010-09-14%2012:00:00&to=2014-12-04%2012:00:00&batch=3®ion=United%20Kingdom	Returns the analysis of the category usage per city of the specified country.
/api/country/category	GET	/api/country/category?from=2010-09-14&to=2014-12-04&batch=3	Returns the analysis of the category usage per country.
/api/location	GET	/api/location?from=2010-09-14%2012:00:00&to=2014-12-	Returns the clustered locations and analysis of

		04%2012:00:00®ion=United%20Kingdom	hate speech.
/api/timestamp	GET	/api/timestamp?from=1284411600000&to=1417644000000	Returns analysis of hate speech in timestamp range.
/api/timestamp/category	GET	/api/timestamp/category?from=1284411600000&to=1417644000000	Returns analysis of hate speech per category in timestamp range.
/api/location/category	GET	/api/location/category?from=2010-09-14%2012:00:00&to=2014-12-04%2012:00:00®ion=United%20Kingdom	Returns the clustered locations and analysis of hate speech per category.