UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

GRADO EN INGENIERÍA DE TECNOLOGÍAS Y SERVICIOS DE TELECOMUNICACIÓN

TRABAJO FIN DE GRADO

DESIGN AND DEVELOPMENT OF A MONITORING SYSTEM FOR SPANISH POLITICAL TWEETS IN COVID-19 EMERGENCY

Daniel Lledó Raigal

JUNIO 2020

TRABAJO DE FIN DE GRADO

Título:	Diseño y desarollo de un sistema monitorizador de tweets políticos en la emergencia del COVID-19
Título (inglés):	Design and development of a monitoring system for Spanish political tweets in COVID-19 emergency
Autor:	Daniel Lledó Raigal
Tutor:	Carlos A. Iglesias Fernández
Departamento:	Departamento de Ingeniería de Sistemas Telemáticos

MIEMBROS DEL TRIBUNAL CALIFICADOR

Presidente:	—
Vocal:	—
Secretario:	—
Suplente:	—

FECHA DE LECTURA:

CALIFICACIÓN:

UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR DE
INGENIEROS DE TELECOMUNICACIÓN

Departamento de Ingeniería de Sistemas Telemáticos
Grupo de Sistemas Inteligentes

TRABAJO FIN DE GRADO

DESIGN AND DEVELOPMENT OF A MONITORING SYSTEM FOR SPANISH POLITICAL TWEETS IN COVID-19 EMERGENCY

Daniel Lledó Raigal

Junio 2020

Resumen

Los tiempos han cambiado, el discurso político ya no es como antes, ahora se recogen estudios de análisis de datos que informan a los gobernantes sobre cómo proceder ante la multitud y cómo dirigirse a su público, mediante el análisis de los sentimientos. No hay que olvidar que los que votan por ellos son los ciudadanos. En este estudio se monitorizan los tweets de los diferentes partidos políticos que componen el parlamento español y se les aplica un algoritmo de análisis de sentimientos, en el contexto de un tema crítico.

Tras el estudio del Estado del Arte, este proyecto tiene como objetivo diseñar el sistema Docker utilizando un orquestador como Luigi e implementar un Dashboard utilizando tecnologías de Big Data y Visualización como Elasticsearch o Polymer Web W3C Components, que muestran estadísticas sobre la actividad de los partidos políticos representados en el parlamento español en el contexto de Twitter. Estas estadísticas son la actividad dentro de la red, el número de tweets diarios, las tendencias, así como un análisis del sentimiento. Para ello, aplicamos el Análisis de Sentimiento con Técnicas de Procesamiento del Lenguaje Natural.

Estas características del sistema nos permiten medir la actividad de los datos recogidos desde Twitter, obtener datos relacionados con los políticos españoles y analizar de forma más concreta, sobre los sentimientos mostrados en Twitter sobre diferentes términos en la emergencia de COVID-19. En este estudio, intentamos centrarnos en analizar el partido político desde el punto de vista del ciudadano. Las diferentes estadísticas recogidas por el sistema muestran una aparente afinidad ideológica para proyectar sentimientos sobre un tema. En este proyecto, se realiza un gráfico cronológico sobre la actividad y los sentimientos mostrados en los tweets proyectados sobre un tema. Se comprueba la evolución temporal que puede haber tenido la polaridad sobre un tema.

Mostramos las pautas, la actividad, los sentimientos y la posición de un partido sobre diferentes temas aplicando las diferentes tecnologías de análisis de Big Data a los mensajes publicados por las cuentas oficiales en Twitter, desde el punto de vista del ciudadano.

Palabras clave: Sentiment Analysis, Big Data, Politics, COVID-19, Twitter, Elasticsearch

Abstract

Times have changed, political discourse is no longer as it used to be, now data analysis studies are gathered that inform the rulers of how to proceed before the crowd, and how to address their public, through the analysis of feelings. It should not be forgotten that those who vote for them are the citizens. In this study, the tweets of the different political parties that make up the Spanish parliament are monitored, and an algorithm of sentiment analysis is applied to them, in the context of a critical issue.

After the study of the State of the Art, this project has the goal of designing the Docker system using an orchestrator such as Luigi and implementing a dashboard using Big Data and Visualization technologies like Elasticsearch or Polymer Web W3C Components, which show statistics regarding the activity of the political parties represented in the Spanish parliament in the context of Twitter. These statistics are the activity within the network, the number of daily tweets, the extracted topics spoken about, as well as a sentiment analysis. To achieve this, we apply Sentiment Analysis with Natural Language Processing Techniques.

These system characteristics allow us to time the input of data collected from Twitter, get data related to Spanish politicians and analyze in more concrete ways, about the sentiments shown on Twitter about different terms in the COVID-19 emergency. In this study, we try to focus on analyzing the political party from the citizen's point of view. The different statistics collected by the system show an apparent ideological affinity for projecting feelings on a topic. By breaking down the statistics, a chronological graphic carries out on the activity and feelings shown in the tweets projected on a topic. We can check the temporal evolution that the polarity about a concept may have had.

We show the guidelines, the activity, the feelings, and the position of a party on different issues by applying the different technologies of Big Data analysis to the messages published on the social network Twitter, all from a citizen's point of view.

Keywords: Sentiment Analysis, Big Data, Politics, COVID-19, Twitter, Elasticsearch

Agradecimientos

Muchas gracias a toda mi familia y amigos, que han sabido estar ahi en todos estos momentos de la crisis. A mi tutor Carlos Ángel Iglesias, por ayudarme y guiarme durante todo el desarrollo del proyecto. Muchas gracias también a todos los compañeros del Grupo de Sistemas Inteligentes, en especial a Álvaro de Pablo Marsal, por enseñarme a usar estas tecnologías, y no perder la paciencia. Gracias a Victor, Raul, Jose, Pedro y Álvaro, que supieron dar luz en momentos duros hace ya 5 años.

Gracias en general a todos los que me habéis ayudado, creisteis en mi, y habéis hecho esto posible.

Contents

List of Figures

CHAPTER 1

Introduction

1.1 Context

We are in a new era of technology in which people are connected, no matter how far they are [6]. This connection is mostly achieved through the use of social networks, where people show their thoughts, deeds, ideas, behaviors and feelings.
Twitter [7] has become one of the most important social networks in the world of TICS, it has 328 million active users. Twitter has a turnover of more than 2500 million annually with a stock market value of over 10,000 million dollars. Twitter's final users are private citizens, celebrities, journalists, businesses, and organisations; in other words, they can be both individuals and collectives, with aims that are strategic, casual, or a dynamic combination of both [8].

These statements lead to the conclusion that Twitter encompasses all the issues that people are talking about in practically the whole world. The large number of users and the constant flow of information that they are exchanging, make it ideal to monitorize trending topics. This statistics able us to identify the language records used, the type of language, or the feeling that their tweets show on the network [9].
If these data are obtained in an aggregated and limited way, we can reach conclusions such

as the community feeling before an issue or even detect events in a possible location.

There have been published studies that address the different ways in which people are interconnected through relationships based-on follow or retweet the user content [10], analysing the coherence and the feeling or idea that is projected when writing a tweet. To natural language process analysis to detect concepts, ideas or styles at more general levels with aggregate data from many sources, and to draw conclusions that portray a community's way of expressing itself or its response to an incoming topic.

On the one hand, one of these fields where one can distinguish the different groups that, for example, give space to an idea, or tend to share ideas and even give opinions about events in a joint way on social networks such as Twitter is politics.
The political discourse has changed, and from press conferences and meetings, we have moved to an ecosystem in which users themselves can interact with political representatives, and even make judgments for or against what they publish on Twitter. As a result of all this, trends are created that seek to influence the citizen or even, by some users, hoaxes to try to harm the opponent and thus create a bad public opinion about an entity or representative [11].

On the other hand, another field in which Twitter is of vital importance is in immediate information, of all kinds. So much so, that in areas such as health, is of great importance, as can be seen in the pandemic, can get to have first hand information from the entire network of Twitter. Being as it is, a pandemic, and one of the first most important in what leads Twitter life, this network has taken great importance as a means of communication, as it did in the 2009 flu pandemic [12].
As a consequence of this work that Twitter entails, it is of vital importance that political parties inform citizens and give their opinion about what is happening in this era.

Unlike practically everyone else [13], where the differences between the different political organisations have disappeared to provide a truce to avoid tensions and tackle the COVID-19 crisis. In Spain the political war is still active, the polarization on certain issues is more present than ever and this is noticeable in the speeches of the different political parties that are reflected on Twitter [14].
This study will analyze use of the social network by the different political parties. Also we will study the polarity of feelings that the tweets from the different accounts associated to politicians and political parties are having, in relation to the Tweets published during the emergence of COVID-19.
We will also analyze the most commented topics in this whole period, classified by parties and showing the feeling gathered by the system of each one of them.

All these services explained above will be hosted virtually by Docker (Sec. 2.4) containers in a local context. We used in Senpy (Sec. 2.8) the Meaning cloud algorithm to analyse sentiments, the Twitter API to download and follow the activity of the politics accounts, ElasticSearch (Sec. 2.6) to dump all the Data to the Dashboard, made with Sefarad (Sec. 2.5) to visualize data, using Polymer Web Components (Sec. 2.9).

1.2 Project goals

The main objective of the project is to design a system that analyses the tweets published by the official accounts of the political parties represented in the Spanish Parliament. The desired analysis consists of a part in which, through data aggregations, we analyze the activity presented by the different official accounts of the political parties, as well as an analysis with Senpy's API that classifies the polarity of the overall sentiment used in the text.
This analysis is reflected in the different components destined to the analysis of feelings, placing the reference in the temporal period of the publication, or referring to the source from which the tweet comes.

The other part consists of the design of a component that analyzes all the texts, being able to classify them so that we obtain the tweets in which that term is commented. This provides us with the function of filtering by terms, with the consequent ability to analyze the feeling projected in the tweets containing that term.

1.3 Structure of this document

This section comments on the different chapters that make up this document, as well as brief ideas describing each one. The structure is as follow:

Chapter 1 is the introduction of the project, in which we describe the context of the project, as well as the objectives set.

Chapter 2 discusses the different technologies required for the development of the project. It also provides a brief description of the State of the Art.

Chapter 3 describes the architecture of the complete system, as well as the decisions that have been considered in the development of the project.

Chapter 4 provides further details on the use of the system and explains the process of a term analysis.

Chapter 5 breaks down the different conclusions and ideas due to the project, comments on specific problems encountered, as well as a small description of possible future enhancements.

CHAPTER 2

Enabling Technologies

2.1 State of the Art

Twitter has become one of the principal sources of information, and its message format called ’tweets’ allows the science to use it for analytical purposes.

Recent studies have begun to study the emotions and feelings in Twitter’s visual objects such as images and emoticons in Twitter [15], and maybe could be used in the future to complement the analysis of feelings in the text.
The wide range of possibilities of using Twitter makes this social network ideal for analyzing data and users’ feelings. The analysis of the sentiments in an area or city about the political themes in the election season could help as Lorentzen claimed in 2014 [16], analyzing the various relationships between Swedish politics and communications. He concluded that polarization is evident in the retweet and followership networks.
Another political use for social networks was to collect and analyze the opinion of the people in Brexit [6] using Python libraries and Natural Language Processing kit to determine the polarity of the sentiment with a decrease of positive sentiment about Brexit in 2017-2019.

Another investigation was about community based political herd sentiments [17] which used the speeches and motions of a political party with LDA-based topic modeling to derive information about the topics labeled getting the sentiment polarity and demonstrating the relationships between government and opposition with their followers in terms of ideology or like-minded ideas.

As we see, Twitter is a pillar of the current political dimension. It's enormous the quantity of data that is daily processed; the accuracy of collecting the sentiment about a speech sometimes is wrongly generated. Acccording to Katta's research [10], it is proposed a system for predicting election results using an Adaptative Neuro-Fuzzy Inference System, with a neural network and a no Linear Support Vector Machine showing that decreases the complexity and gives more accuracy than existing methodologies.
On certain occasions, we witness how politicians get into arguments, continually disrespecting each other, to expose other politicians or people who disagree with what they preach. In a IEEE article [11], Basaks explains how there are developed several systems to perform the detection of shaming events on Twitter and mitigate the effects of shaming. It divides the tweets with an LDA - based classifier in six types of shaming, obtaining the most presence of negative terms in the texts of this type of tweets.

On the other hand, it's also important the type of language used to approach the audience, as Abdullah's research shows [9]. Abdullah aims to perform sentiment analysis of people's reactions after Trump's primary debates. That analysis consisted of data gathering in Twitter and a classifier for the people emotions about the primary debate. The study shows that sometimes it's not as relevant the sentiment as the principal target of the tweet.

In those articles, we found sources which conclude that many times the politician or entity that tweets, needs many more resources than simple language to reach their audience. Adding that many times the concepts they rely on are often exciting and above all, simple for the citizenry. It seems that with the use of elements that show feelings, and in many cases, with joy, citizens can be more reached and influenced.
Still, in political terms it is concluded that users who are followers and like-minded are more likely to interact with people with different ideas and from other parties, although retweet and follow networks are within the same ideology.

However, there are also more approaches to research open in areas such as public health. According to Sigh et al. [18], they used datasets extracted from Kaggle and UCI repositories and used deep learning to create a classifier of Natural Language to identify new suicide behaviors in Punjabi text.
Also, these approaches can be used to monitorize and predict the spread of an emerging infectious disease, like Ebola. As Kraemer [19] explains in his paper, it is required an invasion model to estimate the probability of invasion between districts. Using a Transmis-

sion Model Selection, they concluded that Ebola Virus spread in Sierra Leona was within national borders suggesting the borders could help more in a humanitarian spread.
Recent studies [20] about the new pandemic disease COVID-19 had created a DataSet to register common responses to the pandemic and how these responses changed and adapted along the time. It helps to understand and improve diagnosis and prevention to avoid this kind of situations

This unusual event has caused the media and social networks to get affected. It has been a great flood of many kinds of information, such as publications of governing announcements, opinions, and of course, political parties. Some recent study [21] shows an analysis of Facebook at the beginning of this pandemic and analyzes the style of narrative that different entities, as described, take when it comes to informing or giving their opinion about this fact.

In this final graduate work, we examined a dataset from Spanish politicians tweets taken from the beginning of March until the end of the pandemic, and we applied a sentiment analysis to analyze the polarity of the tweets by relating and classifying the tweets with the main terms of the pandemic.

2.2 Python

Python [22] is powerful and fast, plays well with other technologies, runs everywhere, is friendly and easy to learn, and is Open Source. Currently, many Internet services use Python, such as Google (since the beginning), Youtube and other platforms that use libraries written in Python.
The most important thing that made us use Python language in our system was the well-playing with other technologies and the versatility to use it everywhere you want and finally the capacity that it has to implement libraries of almost all types of research fields. As far as we are concerned, these libraries provide a suitable environment to build the system and connect the different parts of it:

- **CSV:** One of the most popular formats for exchanging data is the so-called CSV [22] (Comma Separated Values), because the amount of storage and processing you need is meager. CSV module in Python lets us to read and write files with this format and analyze them in our project.

- **JSON:** JSON (JavaScript Object Notation) is the well-known format specified by RFC 7159 [22]. It was created in 1999, but now it is not exclusively built for JavaScript objects, so as to, in 2017, became a language-independent data format [23]. It is implemented in most systems that integrate an API, and, in this project, it is also the format we use for loading data into our Elasticsearch (Sec. 2.6).

- **OS:** This module provides a portable way of using operating system dependent functionality [22]. It has several functions that give us the functionality of managing files and directories (os.open()), file paths (os.path), and reading the lines of the console input with other accessory libraries(fileinput). In brief, it gives us the control of tasks of the machine running the script.

- **sched:** The sched module defines a class which implements a general-purpose event scheduler [22]. This library is used for creating an event scheduler. The purpose is to execute periodic scripts, such as a tweet download.

- **Tweepy:** This library provides the interface for Python scripts to access the Twitter API [24]. Not only does it allow us to authenticate with both OAuth1 y OAuth2, needed by Twitter [25], but it also provides all the methods to manage the requests and responses of Twitter API.

- **Datetime:** The Datetime module supplies classes for manipulating dates and times [22]. It is a widespread library used to give Date or Time objects in scripts of Python.

2.3 Twitter API REST

Twitter [7] is what's happening now. So it is that there are a lot of tweets per minute in all the world, uploading information that can be accessed by Twitter's API. Twitter's developer platform provides many API products, tools, and resources that enable you to harness the power of Twitter's open, global, and real-time communication network [25]. Twitter has divided the API service offer, limiting some characteristics. The distinct levels are:

- **Standard API**: Limited speed rates downloading tweets information, limited requests for a 30-day window. Manage and pull public information. Retrieve trends. It is required to apply by email for an account and with no commercial-purposes. It is the most limited product Twitter offers, but it is free. It provides access to the Streaming API and the API REST, with a maximum tweet age of 7-days. We can obtain all the parameters of the user tweets and the information statistics of the tweet.

- **Premium API**: Allows more characteristics in Search Tweets API, extends the limit of month-requests and day-limit tweets.

- **Enterprise API**: Special Access Requirement to enter the API. No limits about requests per month or per day. Allows to enter in Search Tweets functions. It is only available by Enterprise Licenses.

- **Ads API**: Integrate programmatically with the Twitter Ads platform to manage the ads of your company.

The whole different types of access the API needs the Developer Account, which has to be required to Twitter Inc by a form. To access the Premium API, you need to pay a subscription with the products you would want in terms of the number of requests or the age of the tweet requested. Twitter API REST is limited for Standard API, and it only allows tweet or timeline request with a 30-day old.

2.4 Docker

Developing apps today requires so much more than writing code. Multiple languages, frameworks, architectures, and discontinuous interfaces between tools for each lifecycle stage create enormous complexity [26]. Docker [26] is a framework that allows us to virtualize several services and to create a network between them to communicate and offer a multi-services system.

Docker makes the workflow much easier, isolating each part of the programming app, and deploying it on a container, making it more portable and easier to run everywhere. The framework host in localhost ports the multiple services and reference each other by ids, communicating them by defining a network with its entries and outputs.

Containers are standardized units of software. The app is virtualized in the Docker network, connecting each service, such as a dashboard with an ElasticSearch (Sec. 2.6) server.
Finally, Docker associates each virtual part with the system directories and assigns the memory for each container. In conclusion, it makes the service portable and executable in everywhere.

2.5 Sefarad

Sefarad [27] is an environment developed to explore, analyze, and visualize data.
It is an application developed to explore data by making SPARQL queries to the endpoint you choose without writing more code. You can also create your own cores if you have a big collection of data. To view your data you can customize your own widgets and visualize it through them [1].
The environment is divided into modules with its own function.

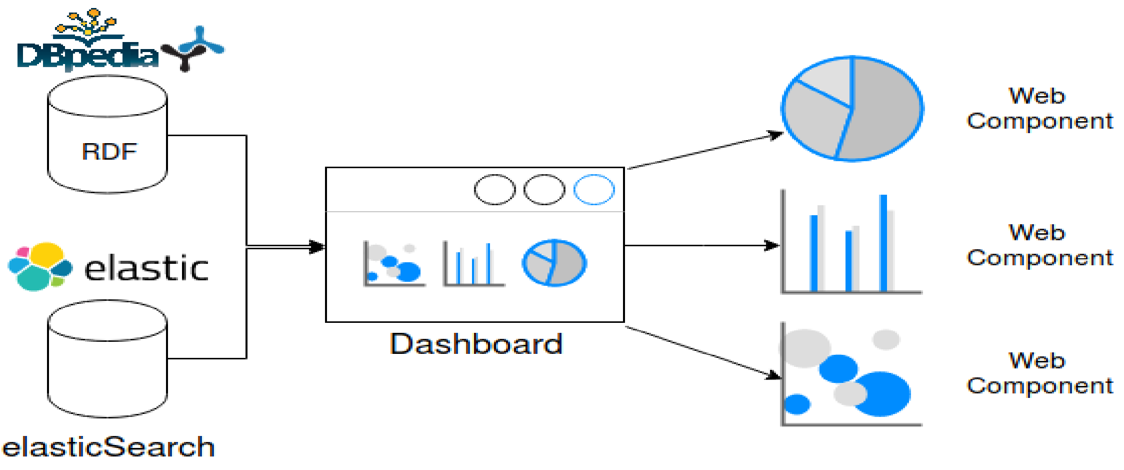


Figure 2.1: Sefarad's Architecture [1]

- **Visualization:** The primary function of this module is to represent the data processed by the system and draw charts taking different aggregations of data to visualize interesting topics. These graphics are divided into other components like Polymer Web Components or Google Chart Components

- **Elasticsearch:** Represents our system database, and due to its vast capacity to aggregate data grouping by terms. In Sefarad, its role is to be in charge of the layer persistence app. It deploys the aggregated data to fulfill the component requirements.

This framework allows us to connect the persistence services, together with the visualization services. Thanks to this connection, and the integration capacity of Polymer Web Components, we create the ideal environment to design interactive dashboards.

2.6 Elasticsearch

Elasticsearch [28] is a distributed, open-source analytics and analysis engine for all types of data, including textual, numeric, geospatial, structured, and unstructured.
It was created to be the open-sourced Apache Lucene's alternative. Some features make Elasticsearch a suitable environment to use for index information. It was implemented in Java, and it provides an HTTP/JSON API REST, meaning that URLs expose all the resources, and the responses are JSON format. This environment has the following characteristics [29]:

- **Well-distributed**: It has been designed to be scaled vertically and horizontally both.
- **Query DSL**: Query DSL (Domain Specified Language) is a JSON interface provided by Elasticsearch that enables the ability to create complex queries with no much effort with its language, "painless", made especially for querying in Elasticsearch indexes.
- **Schemaless**: It is not required to create a schema with field names and data types to index the information on Elasticsearch. Elasticsearch created a schema itself, parsing all the values given onto a schema model.

Its most known characteristic is that it is a reverse index searcher, so it allows us to structure and aggregate the information in a form that lets us to search inside massive objects with specific terms. Elasticsearch is capable of index much information and aggregate it grouping by specific terms.
These mentioned characteristics empower our objectives, because, with its aggregation capacity, we can join the data with common characteristics, and analyze the statistics of each concept.
Obviously, it is used to analyze concepts focusing on each political party as a common characteristic, and thanks to the aggregation, order them in two-level aggregations, such as date-author, which let us to classify data in date and author terms.

2.7 Luigi

Luigi [30] is a Python package that helps you to build complex pipelines of batch jobs. It was used by Spotify, GIPHY, and Groupon to develop their systems. It handles dependency resolution, workflow management, visualization, handling failures, command line integration, and much more.

This package creates a secure environment, addressing all the entries and outputs of the different system tasks, chaining them and managing the achievement of the different parts of the system, handling errors printing them in the console. Luigi has some entities to manage this system processes:

- **Target**: It is the object which Luigi uses to represent files. They could be on a hard disk or a checkpoint. For example, to locate entry data for the system, such a JSON with tweets.
- **Task**: This is the system point where the implementation of logical sentences is taken. The task class defines a job to run. Some useful functions are provided to implement the workflow of the system:
  - **requires()**: Several parts of the code have to return entities, which are used for other parts of the code. This function indicates which scripts must be completed before executing this task. App workflow of these associated tasks won't continue until the required entities are returned
  - **run()**: Here is where are developed the logical functions that characterize this task. The result of these functions are presented in the output() function
  - **output()**: Defines the location of the result of the task.
- **Parameter**: The Parameter class is used to create Tasks, associating them with some kinds of identifiers. For example, I want all the tweets of the day; this implies that once a day, we must do a download job to have the tweets of that day. If I have to identify that task, we use a Date Parameter. To sum up, it is the class that allows us to create variables in Luigi's environment.

In conclusion, Luigi helps us to schedule many tasks that our system has to execute. The data plumbing is addressed with the date when data is collected. This container permits us to connect the different services such as Senpy, Elasticsearch and visualization web components of the dashboard.

2.8 Senpy

Senpy [2] is a framework to build sentiment and emotion analysis services providing functionalities for:

- **Senpy Common Interface**: It provides an API and tools, making it much more manegeable, changing the sentiment algorithm with only a parameter in the query request.

- **Senpy Plugin**: Depending on the plugin you choose, you have in the HTTP response the sentiment analysis of the text introduced. The plugin can be easily changed, allowing us to choose a different analysis thanks to the Senpy build.

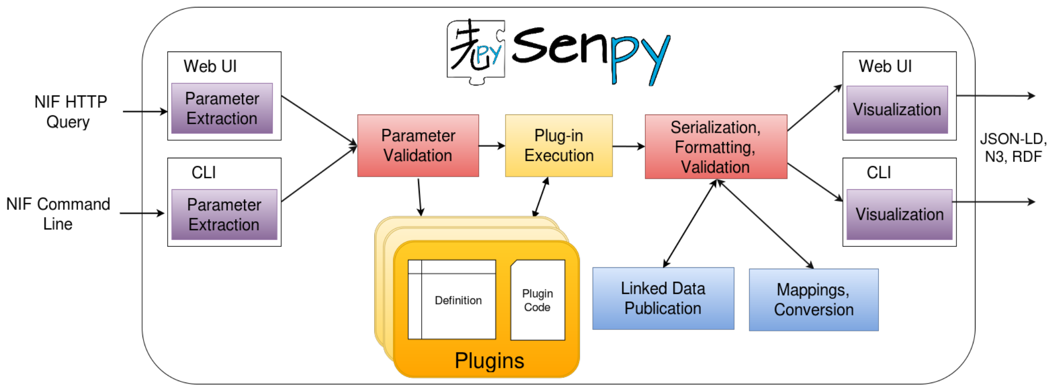


Figure 2.2: Senpy's Architecture [2]

Developing text analysis services must implement some typical tasks, such as parameter extraction and format conversion visualization of the results. In this project, we use the meaning-cloud plugin of Senpy, which returns a request with the analysis of the opinion of each sentence of the text, indicating the polarity of the feeling, between positive, negative, and neutral. This polarity of the feelings must be indexed in the Semantic Web, for this requirement, we have to use Ontologies.

The Semantic Web relies heavily on formal ontologies to structure data for comprehensive and transportable machine understanding [31]. To associate sentiments with opinions, we have to use ontologies; they can associate feelings and their polarity with the style of natural language writing.

Marl [3] is an ontology standardized by W3C RDF technology, an open web standard used to annotate and describe subjective opinions expressed in a software system.
Classify according to terms, entities, subjects, or feelings that can be contextualized in the web data, thus allowing the creation of an interoperable format with other systems that can reach comparisons between them. These functions are the ones that the Marl ontology tries to complete:

- Enable to publish, linked and indexed data about opinions.

- Deliver schemas to improve the dissemination of a standard model able to express opinions and feelings.
- Interconnect opinions,to get a network of generated opinions and feelings.

These types of standards are an initiative of W3C that tries to produce metadata on the web to express concepts in a way that is readable by all users of the network, capable of being indexed for computer systems.
Likewise, this ontology uses other schemes available on the net, such as the Provenance [3] Working Group, which tries to formalize a system to indicate the origin of the data being delivered to the algorithm.

This ontology is also in charge of analyzing the phrase and its language, identifying adjectives, entities to which the opinion is directed, adding Marl's capacities to indicate the polarity of the feeling to which it is directed, we obtain a global structure of the ontology in the following way.

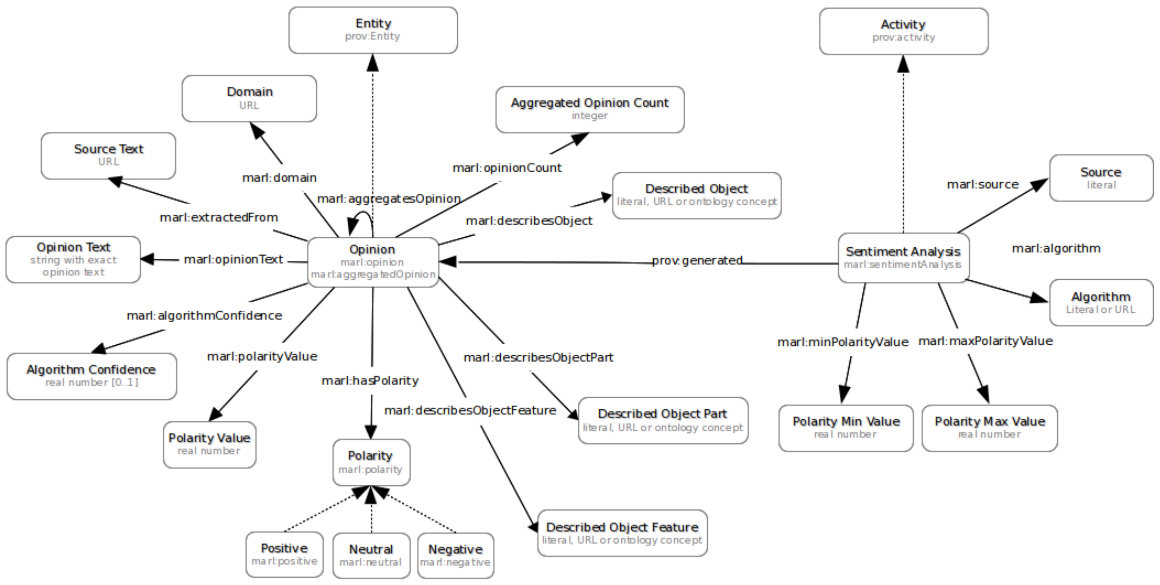


Figure 2.3: Marl Ontology's Architecture [3]

In short, after the Senpy API process, it returns a JSON [23] file in the response. In this file, we find the presence of several opinions, if any. In the case of having it, an opinion is auto-generated, which contains the aggregated results of all the text. These results, observed in a disaggregated way, contain the general one, the respective marginal ones, as well as a vector with all the recognized entities.

This algorithm also collects the different entities that contain the text, such as users with @ or trends with hashtags. Concluding, this last characteristic could be used to monitor the sentiment of each trend expressed by a political party.

2.9 Web Components

Web Components are rendered in a Dashboard, which shows the different statistics that our systems elaborates in the computation tasks. They are building blocks, composed by: The main tasks of computation are executed by the last technologies exposed. Regarding Sefarad 2.5 module' requirements, there are computation tasks, and visualization tasks, Polymer Web Components [32] are in charge of this visualization tasks. With Web Components, you can create and share custom elements that work on any site, inter-operate seamlessly with the browser's built-in elements, and play nicely with frameworks of all kinds.

- **Shadow DOM Elements**: This feature was reserved for browser creators. It is the software that allows us to create a separate Custom Element for the rest
- **HTML Templates and Imports**: Creates a sketch of the component logic and visualization with HTML5 labels, and you can import some external libraries with HTML Imports
- **Custom Elements**: A Custom Element works in modular form and foreign of your software system. It is an independent, customized, and efficient element.

In this development, we use Polymer 3, a specialized library pack. Polymer 2.4 is built on top of Web Components, a collection of standards being approved by the W3C, as it can be seen in the Figure.

As we aforementioned, Polymer enables the creation of Custom Elements using the Shadow DOM. Thanks to the HTML5 labels and JavaScript ES6 language, the Polymer Components are responsive and well-defined. These components can share data between them using data-binding getting the aforementioned characteristics. Breaking your app up into right-sized components helps make your code cleaner and less expensive to maintain [32].

The library in which the Dashboard is developed is Polymer, which allows us to create our web elements that allow us to design at a high level. It is designed to provide the programmer with an intuitive interface.
The placement and views of the element are represented with HTML language, while the

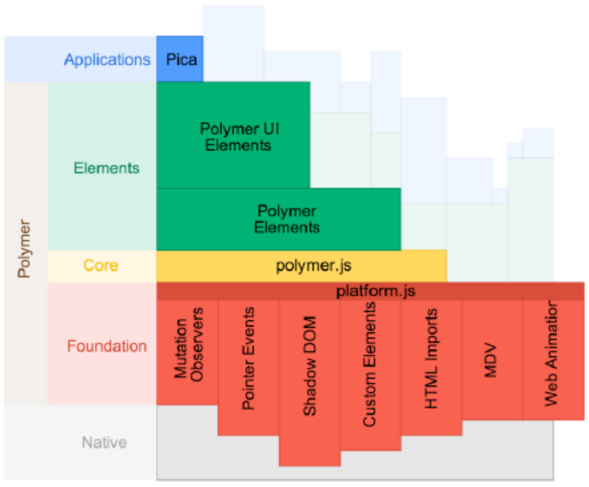


Figure 2.4: Polymer Standard Architecture [4]

logic and connection with other elements is made through JavaScript. Numerous features are available in this library:

- Simplification when creating custom elements.
- The computation of its class attributes, which allows access to them in more easily.
- The class functions are treated as attributes, which allows to use animations or call-backs to draw in a more natural way
- The components are interconnected, they are responsive, and you can access and modify values of their style from other components

These characteristics imply that we can create functional unitary components on their own. Therefore, we can control and specialize in global functions by each element, thanks to the information flow in the display components. We achieve this flow between the parts thanks to the properties. These properties are object attributes that modify global variables, producing a global change in the parent component.

There are currently a large number of pre-designed elements on the web, which we can use in our project, as well as modify as we wish. From only functional components, responsible for the data dump, such as Ajax components. To elements, pre-designed by Google, such as Google Charts. This Google Charts able us to make many graphics by merely indicating the desired in the parameters. All these pre-made elements must be done with Material Design rules and effects, to make them more elegant and responsive.

CHAPTER 3

Architecture

3.1 Introduction

In this chapter, we cover the design phase of this project, as well as implementation details involving its architecture. Firstly, we present an overview of the project, divided into several modules. This is intended to offer the reader a general view of this project architecture. After that, we present each module separately and in much more depth.

The whole system is virtualized in virtual containers; for this, we use the tools provided by Docker (Sec. 2.4), and thus be able to configure the different services in a modular way. From a more technical point of view, our system bases its operation on the different characteristics provided by the different implementations of our services, programming them to adapt to our needs.

In the first instance, we would find the tweet capture system, which is responsible for downloading the tweets, apply a format, and dump them into a folder with files by date. Luigi, the process manager, is in charge of running the sentiment analysis subsystem and adding this information to the model by attaching it as a JSON (Sec. 2.2) variable. This JSON is indexed later in the Elasticsearch (Sec. 2.6) server, thanks to the pipelines provided by Luigi.

Finally, to create elements to monitor a data set we use Sefarad, generating a Dashboard with all its responsive web components well implemented. Concluding, the summary from a broader system perspective is illustrated in the following figure.

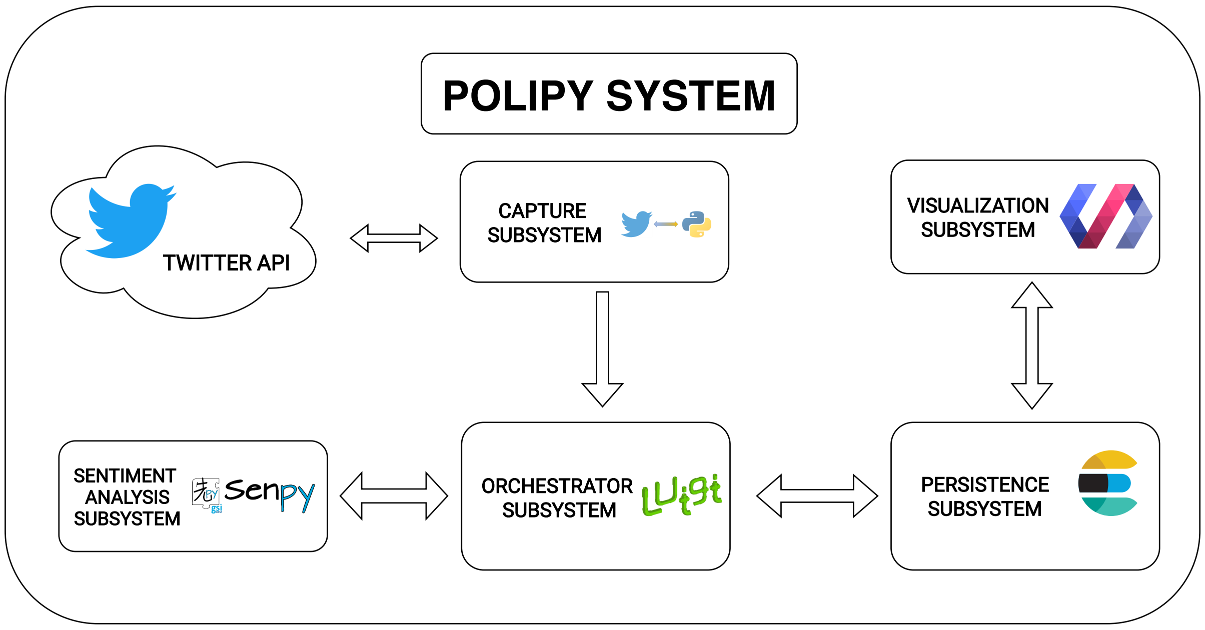


Figure 3.1: System's Architecture

As we can see in Fig. 3.1, the system has a defined flow, by which, the data are collected from the twitter API in a JSON format. Through the different python libraries, we create the JSON data model customized for the system previously broken down. This data model is collected by the Luigi orchestrator, which obtains the object's sentiment analysis through the use of the Senpy API. After the analysis, Luigi takes care of the object's persistence, hosting it in the ElasticSearch index. Finally, all this data is visualized in an aggregated form in the Dashboard, illustrating in different ways the information collected, being able to draw conclusions from it.

3.2 Docker Subsystem

The system is fully virtualized in Docker containers, that exchange data by connecting to a Community network. Services such as Luigi (Sec. 2.7), Elasticsearch (Sec. 2.6), and Dashboard (Sec. 2.9) are connected to ports 8082, 9200, and 8080, respectively.

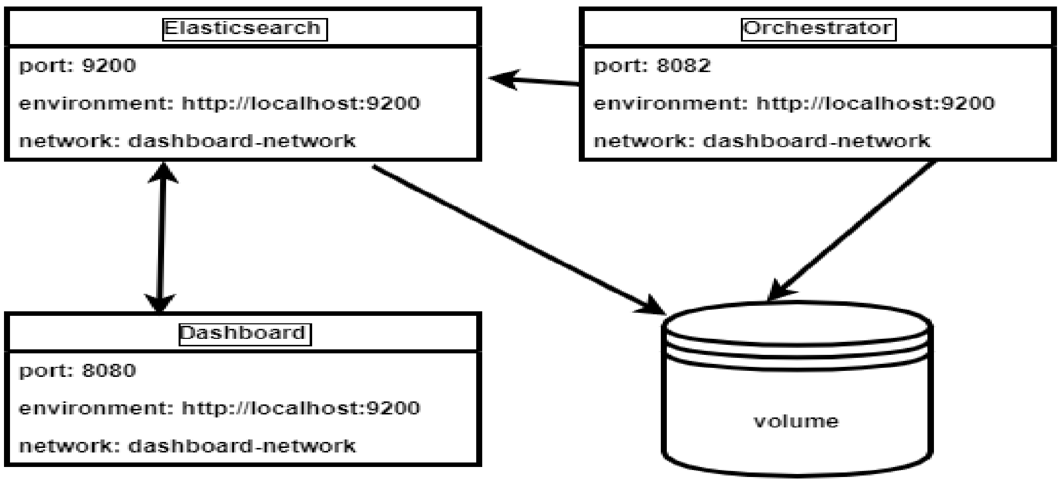

Elasticsearch
port: 9200
environment: http://localhost:9200
network: dashboard-network
Orchestrator
port: 8082
environment: http://localhost:9200
network: dashboard-network
Dashboard
port: 8080
environment: http://localhost:9200
network: dashboard-network
volume

Figure 3.2: Docker Subsystem's Architecture

As we can see in Fig. 3.2, the different services that we use in our system indicate in their Dockerfile the different configurations that are necessary, as well as the different dependencies to install, necessary for the first operation of the service.
We define an environment, as well as a bridge-type network in which all the services are interconnected. Likewise, we define the different connections with the hosting system that the virtual system should have, to access the different hosting functions that the system has, as well as to create and modify files in the system's execution; we define a volume that connects directly with a folder of the host.

Finally, to mount the system image and virtualize it we use the docker-compose command, visualized in Fig. 3.3.

```
sudo docker -compose up --build
```

Figure 3.3: Docker-compose Command

3.3 Capture Subsystem

The tweet capture system is based on two modules within the same script, which is executed periodically every four hours.
In this system, the Twitter API REST (Sec. 2.3) is accessed through the credentials pro-

vided by the developer account. Also, due to the characteristics of the API method used (user-timeline), it is necessary to save the id of the last tweet downloaded. The modules would be:

- **Configuration:** Load in variables the data to configure the API call, such as the different API access tokens, as well as the necessary instances to access the service. Also, the different ids of the last downloaded tweet are loaded to avoid duplicates associated with each user.
  Finally, after downloading the data, we update the different ids of the tweets of each user to avoid duplicates.
- **Download:** All the tweets of the users we had previously loaded are downloaded, introducing in the since-id parameter the id of the last tweet of the user and omitting retweets with include-rts to False. As the script runs every four hours, we check if there is already a JSON created with that same date. If so, we load the file in a variable and put the new tweets to the ones we had previously.

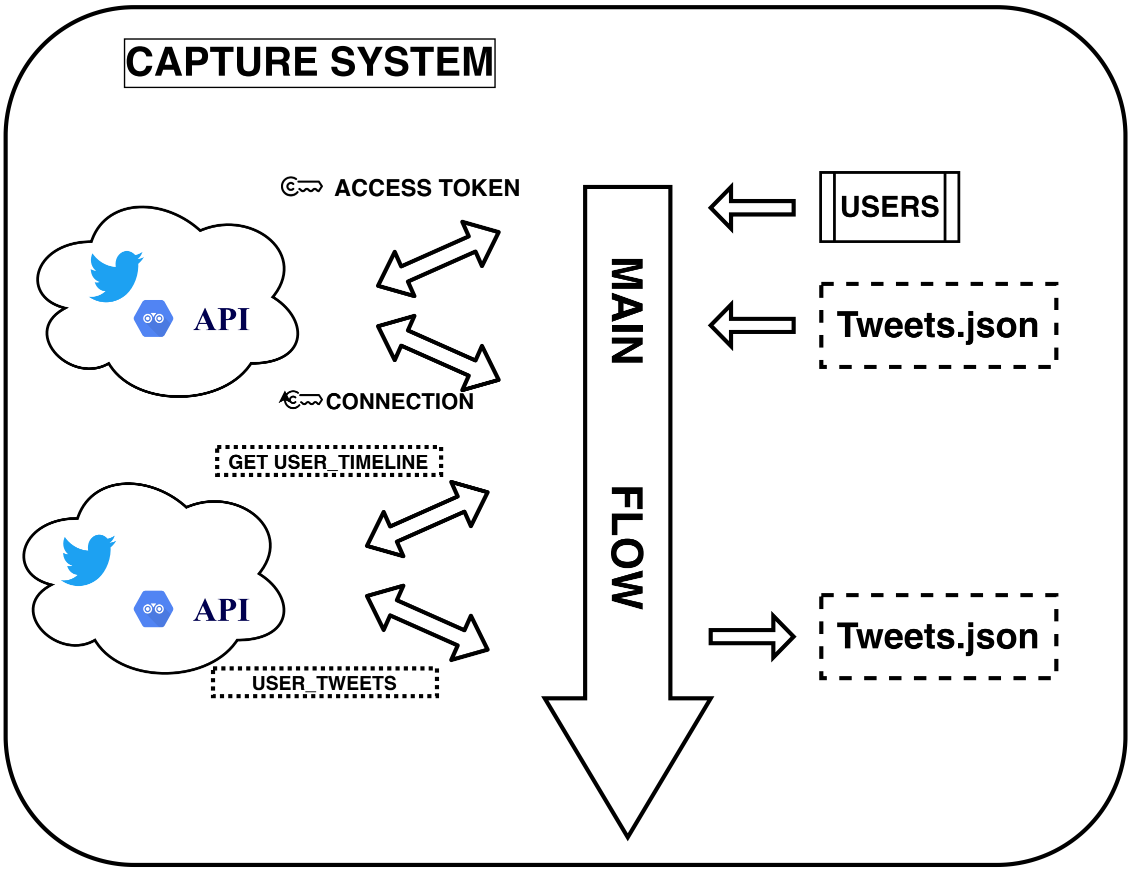


Figure 3.4: Capture Subsystem's Architecture

As we can see in Fig. 3.4, we download the tweets in two phases. Firstly, it is necessary to establish the connection with the API. To do this, we set the parameters of the request containing the information with the API key provided by the Twitter developer account.

At first, the idea was to use the Twitter Stream API, but seeing the limitations that Twitter imposed on its API as well as the low cadence with which the official accounts of the parties tweet, we decided to use the REST API with the "get_user_timeline" method, with more limitations in the antiquity of the tweets. Still, instead, it brings a more gentle flow of tweets per party. When choosing this option, we had to take into account several parameters that the method accepts. On the one hand, as we are studying the feeling that the authors project on the tweets, the parameter "include_rts" we initialize to false, to avoid tweets from other people's accounts in the response delimiting the case study.

Another problem that this method implies is the variability of the time_line. This variability can cause, as a consequence, the capture of duplicated tweets. To avoid duplicates, in this method, we use the since_id parameter. This parameter has to be initialized with the id of the last tweet that we capture from the same author. That's why we configure an entry as a cache of the capture system, so that, once the download of the only author's tweets is completed, it saves in an external instance, the id of the last collected tweet. We use a CSV file to store some important values, we will keep the different names that we will use to tell the API the user to download data, as well as the id of the last tweet downloaded for future uses of the method.

Also, it is necessary to load in a variable from a CSV the different users from which we want to obtain information and we dump the tweets collected that day in a variable, we push new tweets at the end of the object, so we don't lose any information. We charge an id that indicates the last tweet that we downloaded from the user. Through a loop, with the connection made, we make requests to the API REST for each of the users. We extract the fields of the response, and they are wrapped in an object and added to a JSON. This object contains the tweets published by all users on that day.

- **TweetId:** Locator to uniquely identify the tweet.
- **AuthorId:** Indicates the author's id, beneficial to add the different data per author.
- **Date:** Indicates the day, month, and year of the tweet. Useful to add the data chronologically.
- **Text:** Contains in a String the tweet, as well as the emoticons used in the message.

In this project, we obtain the whole set of tweets published by the official accounts of the political parties that make up the Spanish parliament. These tweets were published during the period of the state of health emergency produced by COVID-19, we only get account tweets, with no cites or answer, so we only get original written tweets.

All tweets will have this format before being analyzed by Senpy (Sec. 2.8). After that analysis, the final data model will have an additional field, in which we will save the analysis made by Senpy's API. In conclusion, the different objects are dumped into a JSON file thanks to the dump method that dumps the variable as a JSON and save it in the indicated directory with the date as its name.

3.4 Orchestrator Subsystem

Luigi (Sec. 2.7) provides us with a task orchestrator necessary for the sequential execution of the software system that allows the analysis of the feelings of the tweets, storing the information in the Elasticsearch (Sec. 2.6) index. It is necessary to import an additional library, which allows us to access the Elasticsearch index.

When designing the orchestrator, one must bear in mind that its functions encompass almost all the system's logic. Knowing the capabilities offered by the orchestrator Luigi, we had to create Tasks that are responsible for collecting the tweets collected by the capture system, using the subsystem sentiment analyzer with the tweets collected, and load and index them in a database associated with the entire system.

The different tasks that the service performs interconnect all the different parts that make up the subsystem discussed above. All these sentences that sequence the operation are reflected in the tasks.py file inside the orchestrator's container.
Several classes are defined within the system to perform each of the tasks that the orchestrator oversees:

- **Main:** This is the initial task, and it is in charge of defining the parameters that are injected into the tasks. The date of the task is defined, and Luigi's parameters are injected, such as the name of the index, the kind of document stored, in this case, is a tweet.
- **SearchAndStore:** It is a library that has the function of providing a valid target to access an endpoint to manage Elasticsearch indexes. A series of parameters are configured:
  - Index: Name of the Elasticsearch index

  - Doc-type: Name with the type of document to be stored
  - Purge-existing-index: We define it as False to avoid that the previous index is deleted when the system is restarted, creating an ecosystem of persistences of the stored data.

- **Analyzer:** It is in charge of Senpy's API calls, and processing the tweet for the upload to ES, we add a new field inside the tweet that contains the response of the Senpy's API.
  This API, as explained before, requires the indication of the plugin to use in each case. In this research, we use the Meaning-Cloud plugin, in which we obtain an analysis of the polarity of the feeling.

  Later we store a new field that keeps the polarity of the feeling, extracted from the global analysis of the tweet, to be able to add later the totality of the data by feelings. For each element that is analyzed, we convert it into a JSON that would contain all the analyzed tweets of that day.

  The document with all the results is saved in a local folder, defining the absolute path of the document. Luigi also, prepares the output with the output() method.

- **Streamer:** This element is in charge of receiving the parameter of the date in which JSON is required and prepares the output with the output() method indicating the absolute path of the element we want to return.
  This returned file contains the JSON corresponding to the tweets collected by the tweet capture subsystem.

As we can see in Fig. 3.5, this module interconnects most of the services that make up the system. The flow execution is done in such a way that some tasks depend on others, using the requires method, we indicate the order of the complete flow. The capture system provides the Streamer with the modeled data collected from the Twitter API. This object is passed to the Analyzer, which analyzes the text sentiment, and returns the analyzed object.
This object is passed to the task in charge of data persistence, once here, the object is indexed in Elasticsearch. At the moment we get confirmation that everything went well, the Main task is completed and ends the flow of the orchestrator.

It is essential to highlight, as can be seen in Fig. 3.5, the work carried out by the sched.py module. This module consists of a daemon script which is in charge of invoking the orchestrator Luigi periodically using the command "-m luigi –local-scheduler –module tasks Main". The execution date is passed as a parameter to this command so that it is possible to assign an identifier within the own scheduler that Luigi has.

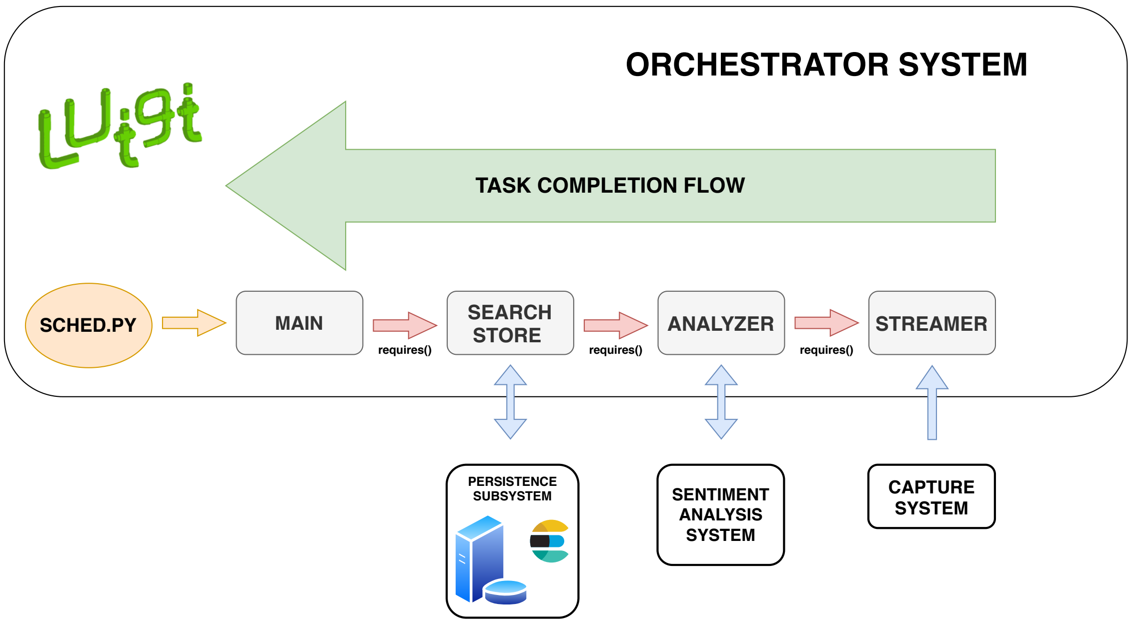


Figure 3.5: Orchestrator Subsystem's Architecture

This module, as we have just described, takes care of all the logical parts of the system. It models the information by creating a specific format for the system and applies the sentiment analysis.

This environment is ideal, since being the most critical part, the error is admissible, preventing the execution flow from being cut off. Maintaining at all times a control over the system thanks to the traces that Luigi provides us.

3.5 Sentiment Analysis Subsystem

The Sentiment Analysis Subsystem is composed by the APIs used in the develoment, as well as the multiple tasks used to connect to them, and save the response. The service used for this application is the one provided by the Senpy and Meaningcloud APIs.

Initially, the idea was to create a container that would host an image of Senpy's service and thus host it in a modular way. Unfortunately, this was not possible, because the Meaning Cloud algorithm was not available within the service image, and it was this one that we had to use, since it accepts tweets in Spanish, Catalan and English, unlike other plugins such as sentiment-140 that does not accept tweets in specific languages such as

Catalan. Due to all the above, we use the Senpy service API.

The service uses the architecture already presented above (Sec. 2.8), and using meaningcloud plugin, applying it to the CORE Senpy will return a JSON with the analysis, with all the opinion mining analysis.

With that JSON, in this project, we are going to use the polarity of opinion aggregation, which gives us an overview of the polarization of opinion projected in the text. This aggregate opinion gives us back the polarity of feeling within the possibilities. Likewise, we also collect the data provided by the service, about the presence of entities, such as objects, institutions, or persons. All this analysis is added to the system's data model, allowing the quantification of emotions as one more parameter of our Dashboard.

In this project, we access the Senpy API, using the Meaning Cloud plugin previously explained, and in the body of the request, we send the text we have extracted from the tweet. Due to the richness of languages within the different users, we have chosen for the analysis, in the language option, it is automatically configured to be identified automatically. With Senpy's API, it is necessary to introduce in a query the text of the tweet to be analyzed, as well as the plugin to analyze and the key provided by Meaning Cloud. Through the response provided by the service, we classify the tweets. So, those tweets that the service does not accept don't contain the data sought from the response, and the system consequently discards them.

The analysis is carried out by extracting opinions from the text, using the previously explained ontology, if they exist, we are returned with the entities and subjects spoken about in the text. As well as a global polarity of the whole text is carried out counting the polarity presented in each opinion and computed as the sum of the whole body.

3.6 Persistence Subsystem

Within the Persistence Subsystem, we find the service in charge of the data persistence, Elasticsearch (Sec. 2.6); which, as we saw in the Docker System (Sec. 3.2), is hosted at port 9200. There, we can find the interface that shows us the state of the Elasticsearch server. We have to manage an index in which we store the documents, in this case, tweets. All the documents are stored in JSON format. As Elasticsearch is a service that hosts elements produced by others, it is necessary to indicate employing a route the endpoint to which the other services are directed.

When implementing it, some problems arose due to the deployment of Elasticsearch in

the virtual machine. This problem consisted of the system limiting the service deployment by memory blocks.
There were two possible solutions. The first, more temporary, was to modify the allowed blocks with the command *"sudo sysctl -w vm.max_map_count = 262144"*. On the other hand, we were looking for a permanent solution, so we modified the system file "etc/sysctl.conf", with the previous sentence modifying the value of *"vm.max_map_count"*. We chose this last option due to the permanence.

One of the advantages of Elasticsearch implementation is that it allows us to create aggregations of the stored data. These aggregations are groupings of stored data The grouping key is based on a common characteristic among them, such as a feeling, or a date.

We use these aggregations to show groupings of data depending on the parameter selected in the graph. Another possible use is in the variation of the data that can be visualized through filters.
These aggregations, in the same way, that they can accept one key referred to a value of the document, they can also accept two hierarchical ones between them. It could be used to show the chronological feeling or the number of tweets per author, depending on the evolution in time.

3.7 Visualization Subsystem

This system consists of the Dashboard service provided by Sefarad (Sec. 2.5) and the different Web Components (Sec. 2.9) that make it up, as well as the connections to Elasticsearch's client and the aggregations injected into the HTML document. In this system, we display the data that allow us to analyze the different authors and the most used terms in the data set.

The structure of the Dashboard service is composed of different Web components developed with Polymer previously explained (Sec. 2.9). On the one hand, we have the Dashboard. It is treated as a component, as well as all the elements that compose it.
On the other hand, we have the configuration of all the connections between the Dashboard and the ES server. These connections are known as aggregations. We also include here the filters that are used in the Dashboard to manage the entry or modification of the displayed data.

3.7.1 Aggregations

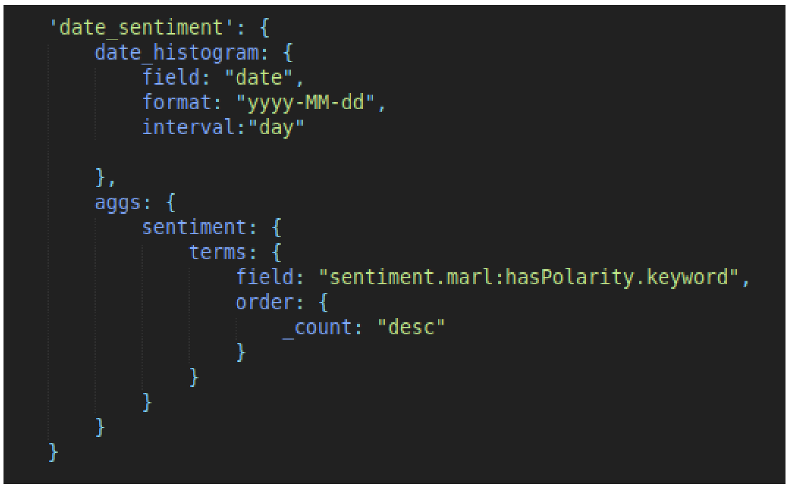
Elasticsearch (Sec. 2.6) provides an accessible environment for storing and managing data, creating indexes with the ability to classify your data with aggregations.
An aggregation can be seen as a unit-of-work that builds analytical information over a set of documents [33]. We can aggregate the data around a value or text string and group it according to the different terms we can obtain.

There are many different ways to group the data. Taking into account the emergency period in which we are, we believe it is convenient to analyze the validity in time of the concepts and the source of information. It is essential to analyze the feelings, observe their temporal evolution, and associate them with the authors who express them.
When we are in a political environment, it is convenient to compare the different attitudes shown among them. Therefore, adding and showing the feelings of the author is another adequate way of adding the data.

In this study, we implement a series of aggregations depending on the graph we want to draw. To encompass all the requirements mentioned, we need some series of aggregations to be implemented to meet the needs of this project, without forgetting the information available to us. In the data model, we have the text of the tweet, the date in day format, the author of the tweet, an analysis with all the data returned by the Senpy API. The polarity of the sentiment, the entities, and the topics collected compose Senpy analysis.
With this data, we implement the following aggregations in the data entry to the Dashboard:

- **AuthorId:** This aggregation is responsible for counting all the tweets per author. This way, we obtain the activity per user in the total tweets.
- **Date-AuthorId:** Classifies the tweets by date, and within each date, the number of tweets from each author per day. In other words, this is the previous aggregation, dividing the activity of each user by date, thus being able to establish a temporal evolution in the activity of the authors of the tweets.
- **Sentiment:** Groups the tweets in terms of the polarity of the feeling shown by the author in the actual set of tweets. With this aggregation, we analyze the percentage of each sentiment polarity in the set of tweets provided.
- **Date-Sentiment:** As in the Date-AuthorId aggregation, this aggregation has the function of joining the data by date, and show the general distribution of sentiment.
- **Sentiment-Author:** Very similar to the previous one, it shows us the percentage of each polarity of feeling for each author.

The Fig. 3.6 illustrates an example of aggregation. As we can see, it groups the data according to date, as the primary key. Within the aggregation, we can sub-aggregate the data; in this case, this secondary key is the polarity of sentiment. With this aggregation, we would obtain a chronological evolution of the feeling of the data group.



```
'date_sentiment': {
    date_histogram: {
        field: "date",
        format: "yyyy-MM-dd",
        interval:"day"

    },
    aggs: {
        sentiment: {
            terms: {
                field: "sentiment.marl:hasPolarity.keyword",
                order: {
                    _count: "desc"
                }
            }
        }
    }
}
```

Figure 3.6: Date-Sentiment Aggregation

Finally, it is essential to note that these aggregations work on the data provided by the Dashboard element. As it is explained later, to give it that responsive and interactive feature, this element accepts some filters. These filters, making use of the aggregations and the data provided by Elasticsearch, modify the initial data entry set for the other elements that make up the Dashboard.

3.7.2 Dashboard

This is the part of the visualization system that allows us to create graphics or other elements to show the information that is dumped from the Elasticsearch (Sec. 2.6) server. The table which shows all the statistics collected and the data parsed after the analysis is the Dashboard. This Dashboard is a Polymer (Sec. 2.9) element, which is implemented in the index HTML of our project, together with the elastic-client object. This ES client is the only parameter that our Dashboard receive.

The Dashboard is the main element of our visualization system, as we said before it is also a Polymer component, whose objective is to design the placement and style of the elements imbued in it, as well as the necessary interconnections between them.

The placement of the different components, and the overall design of the page is pre-defined with a Bootstrap [34] template. Bootstrap follows the standards set by the Google development team with Material Design. Polymer has a large community behind it, including Google, which with its Material Design standard, has created libraries of pre-defined components, functional as a unit. It is the case of the paper-tabs and iron-pages, which allows us to create a navigation bar and a system of tabs, respectively, to make more efficient use of space.

These components are imported through the bower command, and once downloaded, and placed in the root of the element, they are fully functional and ready to be implemented in the system. In the Dashboard, the necessary client for the connection with Elasticsearch's server is installed, allowing the connection with the data persistence subsystem.

The components that compose the Dashboard serve to show graphically the data classified by Elasticsearch's client aggregations and are susceptible to any change, automatically updating their views and rendering them according to the data change.

3.7.3 Material Search Widget

Material Search is the first component located on the Dashboard. It is a search bar, in which you enter a phrase or a word. The word is pushed as a filter, obtaining only the tweets with a text match of the input word.

This element prepares the text input, which initializes the Dashboard query variable, causing the filters to be updated, returning the objects that have those terms inside. It is a handy element since we can extract the statistics by word matches, or associate a feeling to a word. We can see the graphic design of the element in the Fig. 3.7.

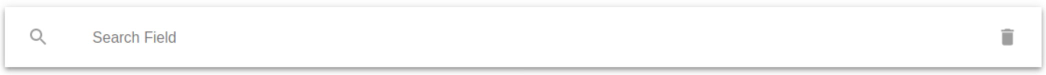


Figure 3.7: Material Search Component

This element applies a filter and only allows the ones that match with the query, although at first sight it seems simple, is one of the most important, since it allows us to connect the semantic sense of what a discourse says, with the analysis of the global picture. This characteristic is vital in this project since the language in politics is inherent, and the ability to connect it with feelings, and associate aggregate data, allows us to deduce an author's position on a term.

3.7.4 Number Chart

This is the simplest web component of the visualization system. It consists of a marker indicating the number of tweets in the current dumped dataset, as well as a progress bar in which we can see the percentage of tweets of the current dataset concerning the total number of tweets of the system.
This element accepts as parameters:

- **Data:** All the data representing the current set of tweets.
- **Total:** It represents the total number of tweets in Elasticsearch's server.
- **Aggkey:** String key passed as a parameter to indicate the aggregation that this component accepts.
- **Title:** Title of the component.
- **Stylebg:** Parameter to indicate the id that we associate in the CSS to apply a style to the component box.

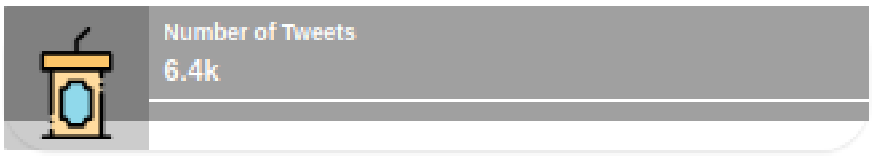


Figure 3.8: Number Chart Component

We can see Number Chart Component design in Fig. 3.8. It is another useful element because it shows us the number of tweets chosen at all times about the entire set collected and analyzed by the system, indicating the number of tweets filtered, and showing the proportion regarding the total set of tweets.

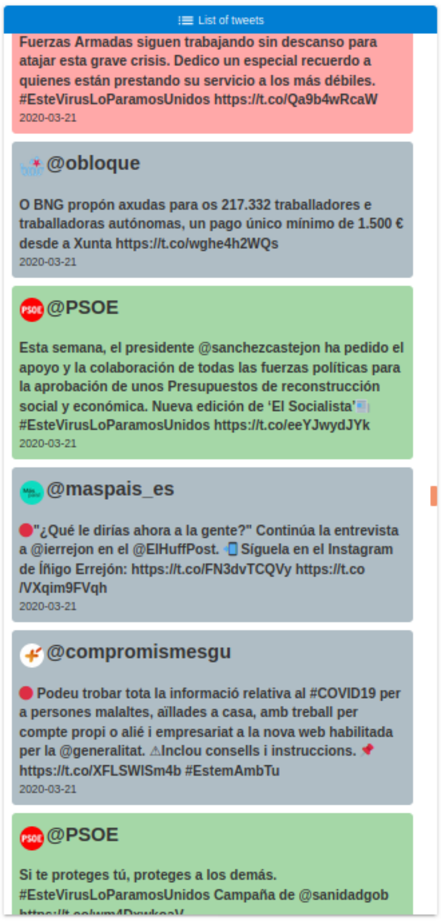
3.7.5 Tweet Chart

The Tweet-Chart component is the last of the no-predefined Polymer elements (Sec. 2.9). Its function is based on being a list, scrollable, in which you can visualize the tweets of the current dataset of the Dashboard.

The tweets are shown in pre-designed templates, modifying the View depending on the polarity of the analyzed tweet feeling, through JavaScript functions modifying the color value in the CSS file. As shown in Fig. 3.9, the tweet chart shows all the filtered tweets of the chosen set. This component receives a series of parameters:

- **Data:** All the tweets of the current set are passed as a parameter.
- **Title:** Component title.
- **Filters:** The current Dashboard filters are passed as a parameter.

Also, these elements are clickable and take us to the website where the original tweet is located.



List of tweets

Fuerzas Armadas siguen trabajando sin descanso para atajar esta grave crisis. Dedico un especial recuerdo a quienes están prestando su servicio a los más débiles. #EsteVirusLoParamosUnidos https://t.co/Qa9b4wRcaW
2020-03-21

@obloque

O BNG propón axudas para os 217.332 traballadores e traballadoras autónomas, un pago único mínimo de 1.500 € desde a Xunta https://t.co/wghe4h2WQs
2020-03-21

@PSOE

Esta semana, el presidente @sanchezcastejon ha pedido el apoyo y la colaboración de todas las fuerzas políticas para la aprobación de unos Presupuestos de reconstrucción social y económica. Nueva edición de 'El Socialista' #EsteVirusLoParamosUnidos https://t.co/eeYJwydJYk
2020-03-21

@maspais_es

"¿Qué le dirías ahora a la gente?" Continúa la entrevista a @ierrejon en el @ElHuffPost. Síguela en el Instagram de Íñigo Errejón: https://t.co/FN3dvTCQVy https://t.co /VXqim9FVqh
2020-03-21

@compromismesgu

Podeu trobar tota la informació relativa al #COVID19 per a persones malaltes, aïllades a casa, amb treball per compte propi o alié i empresariat a la nova web habilitada per la @generalitat. ⚠Inclou consells i instruccions. 📌 https://t.co/XFLSWISm4b #EstemAmbTu
2020-03-21

@PSOE

Si te proteges tú, proteges a los demás. #EsteVirusLoParamosUnidos Campaña de @sanidadgob

Figure 3.9: Tweet Chart Component

3.7.6 Google Chart

Google chart tools are powerful, simple to use, and free [35].
The Google Chart [35] element is a library in itself; the customization options it has make

it a handy tool for developing charts of all kinds. In our extensive Dashboard, we make use of several Google Charts, which, depending on the specific aggregation of each chart, shows statistics related to the data set.

The versatility of these components lies in the great variety of charts that we can create with this element, and even, if required, modify them as if it were our component to adapt it to our needs. This is where the different aggregations explained above (Sec.) are implemented, which will allow us to aggregate the data around different keys.
Google Chart components are interactive; that is, they are capable of modifying the first interface through the use of filters. These filters are selected depending on the graphic we find and constitute a template when dumping data in the general Dashboard.

Like the other components, these are also developed with the Polymer philosophy, and implemented in the same way. Google Charts receive a series of parameters in the HTML object and the functions, stored as variables and properties of the element.

It is relevant to highlight the method in charge of updating the data, dataChanged(), which provides the data in the format accepted by the Google Chart component. There is an option to indicate the columns and rows separately, but in this project, we use the table format.
We consider this method as the best because you can add any data, even of several dimensions, to show multiple data. Consulting Google's documentation, we look for which is the table format of each type of chart we want to draw.
When implementing a Google Chart, it can receive the following parameters:

- **Data:** The data of the current set of tweets is passed as a parameter.
- **Field:** The name of the aggregation to be used in this chart is injected.
- **Filters:** The global filters are passed as a parameter, if they exist.
- **Type:** Indicates the type of graph to use, for example, 'foot' or 'column,' the chart would generate a sector diagram and a histogram with columns.
- **Param:** Passes as a parameter some element necessary for the graph if any.
- **Icon:** String with the path of the icon for the title.
- **Options:** Rules are implemented in CSS format in the case of style, or roles are added within the data. These roles represent characteristics of the component, such as color, formatting. Depending on the graphic to be used, the possible options change.

In particular, the most crucial parameter is the field one, since it defines the kind of aggregation we are going to use. Along with the aggregation, the type of graphic also is of vital importance since it defines the possible design options that can be applied to the component.

At the same time, it is vital to take into account the format of the input data.
One of the general actions when implementing the Google chart design has been to associate the different colors and names of the parties within the chart.
In general, we have solved this problem by adding a column to the data with the label role: style , then adding a value in the object with the color in HEX of each match. But, when designing pie charts, they did not accept the current format, so we have to extract each color value, and pass it by using the options function.

Concerning to the type of data provided to the Google Chart there are graphics made for each aggregation. Here is where the aggregations explained above (Sec. 3.7.1) are implemented, which allows us to aggregate the data around different keys. According to the type of aggregation we can have, we classify the system's Google Charts in:

- **AuthorId:** This widget receives in the field the parameter AuthorId, and it's of type footer. In this component, the different weights each author represents in the whole set of tweets. This way, you can see the percentage of activity for each type of user, such as is shown in Fig. 3.10.

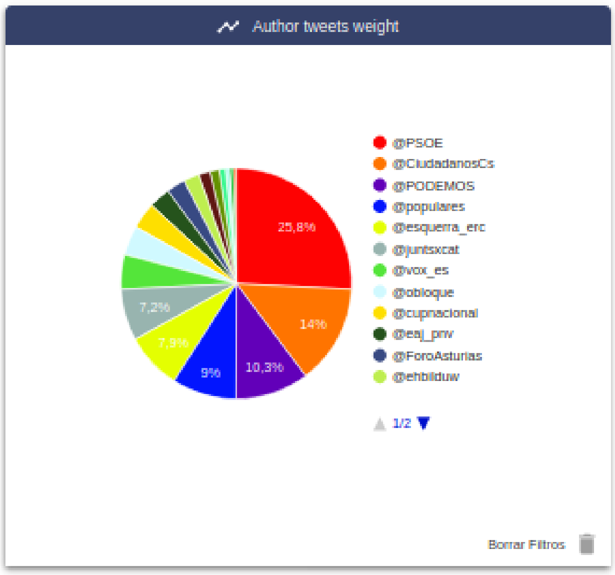


Figure 3.10: Author Activity Component

- **Date-AuthorId:** This element receives in the field the name of the date aggregation, and the graph is of type Line. The information shown corresponds with the number of tweets from each author per date. You can evaluate the temporal evolution of each political party's activity in a general point of view or regarding a specific term. An example of this component is shown in Fig. 3.11.

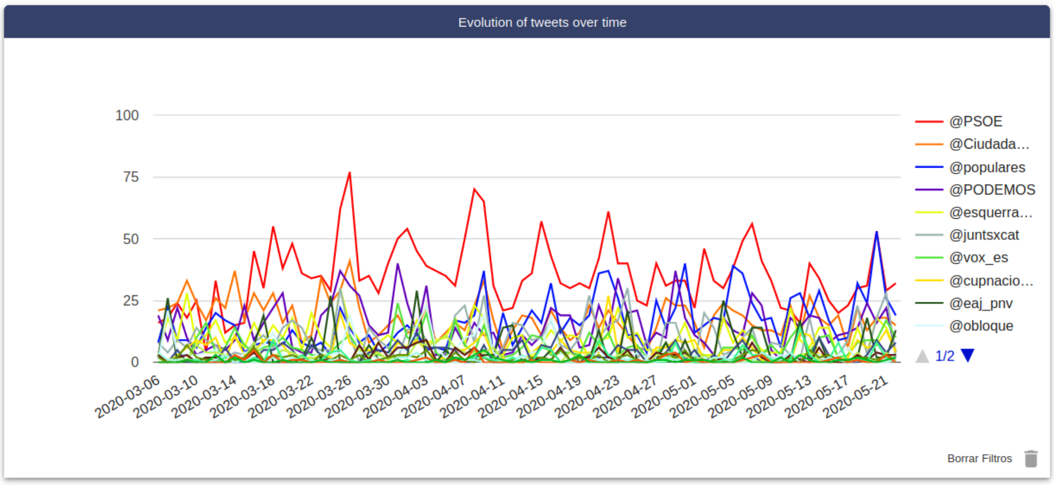


Figure 3.11: Date-AuthorId Component

- **Sentiment:** This graph receives in its field the sentiment aggregation, and the graph is of the foot type. The information shown corresponds to the total percentage of tweets with each type of sentiment polarity in the tweet set selected. In Fig. 3.12 is illustrated an example of this widget.

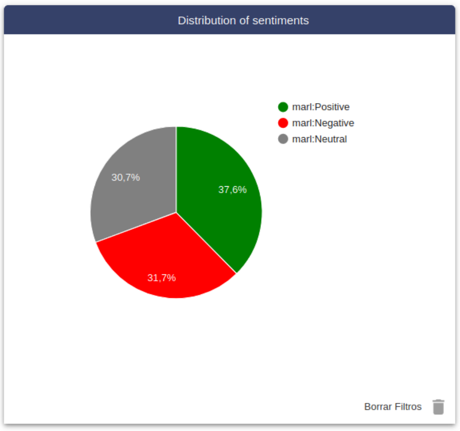


Figure 3.12: Sentiment Chart Component

- **Date-Sentiment:** This widget receives in its field parameter the date-sentiment aggregation that was explained above (Sec. 3.7.1), and the graph is of column type, like is shown in Fig. 3.13. In this graphic, we apply the colors according to the type of sentiment. We also configure the options so that the columns appear in stack mode, showing the proportion of each feeling per date.

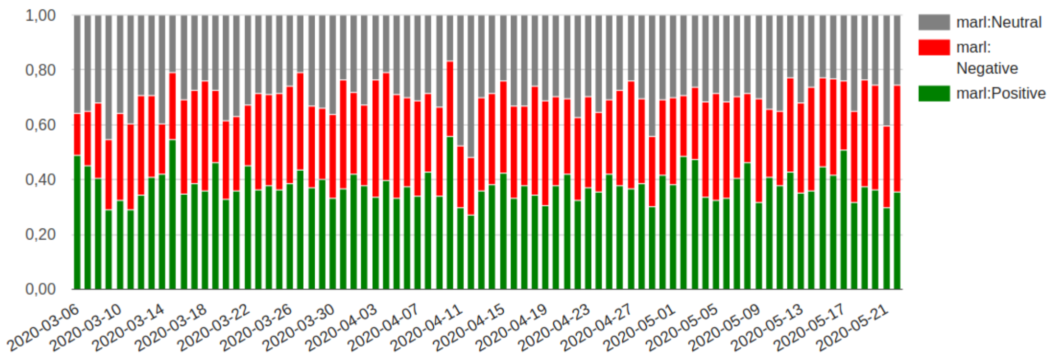


Figure 3.13: Date-Sentiment Component

- **Sentiment-Author:** This Google chart is very similar to the previous one. It receives as field the sentiment-author aggregation, and it is a combo chart. It shows the different sentiment percentages per match within the selected tweets. As can be seen in Fig. 3.14, this widget shows us the different sentiment polarities presented by each author in the whole tweet set selected.

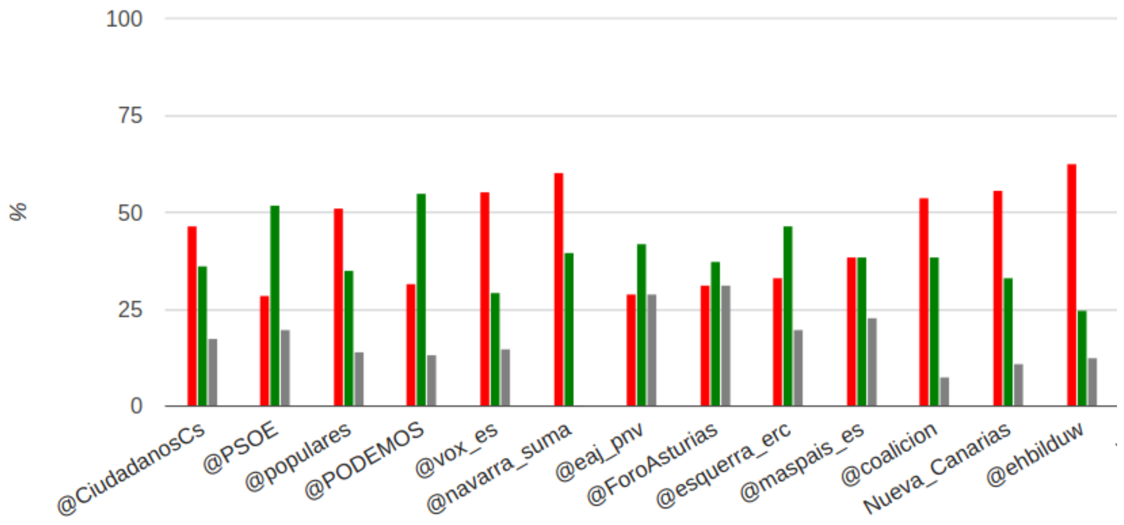


Figure 3.14: Sentiment-Author Component

These aggregations, implemented, allow us to know hierarchically different data about the authors. The presence of components that perform these aggregations enables us to understand trends and distributions of the various data graphically.

3.7.7 Filters

Filters are one of the most critical parts of the Dashboard. Their function is to interconnect the different elements of the Dashboard, generating a global response to a modification of the filters.

They are in charge of imposing rules or conditions to the aggregations we talked about at the beginning, limiting the valid tweets, and reloading the complete view of the Dashboard. These characteristics make the system work even though it is modular, achieving the responsiveness characteristic that Polymer requires. [graphic of how the drawing changes] The method is to extract some parameters from the selected data, such as the key to encircle the data and see only the positives. At the moment we want to return to the complete data, we have to press the button Clear Filters, which calls an internal function, updates the global filters, and reloads the initial view with the empty filters. The following Fig. 3.15 illustrates a possible change in the layout of the dashboard, produced by applying a filter, in which only the tweets published by a particular account are shown.

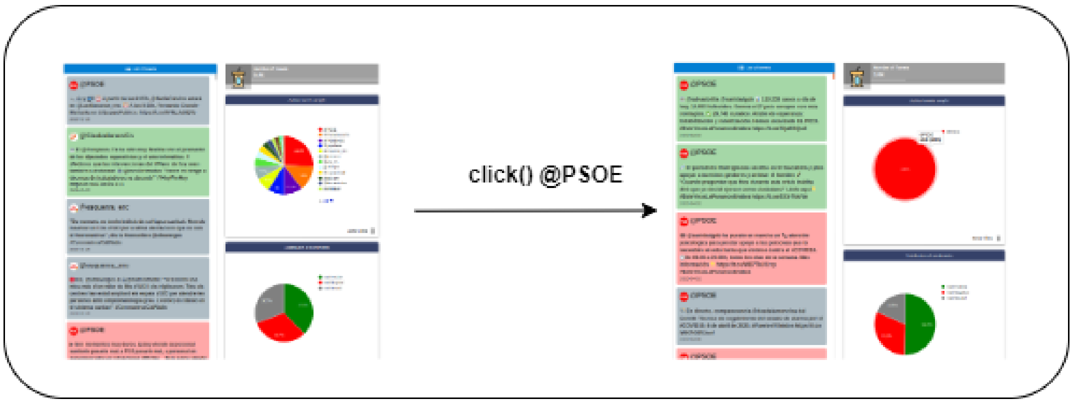


Figure 3.15: Filter Effects on Dashboard

Collecting all the explained components the whole view of our dashboard would be composed by Fig. D.1 and Fig. D.2. In Fig. D.1 we have an activity layout, and in Fig. D.2 there is a sentiment analysis, regarding the authors and the days as was explained in (Sec. 3.7.6).

CHAPTER 4

Case study

4.1 Introduction

In this chapter we are going to describe a selected use case. This description will cover the main whole features, and its main purpose is to completely understand the functionalities of whole components, and how to use it. The analysis of feelings when expressing oneself can indicate the feeling or the intention with which the interlocutor emits the information.

As we commented before (Sec. 2.1), the application of sentiment analysis in politics is not new, from the study of networks in electoral times or public opinion on a given topic. In this project, we applied sentiment analysis to different political parties, indicating the polarity of sentiment in their tweets at the time of COVID-19. Different concepts and methods of analysis provided by the system are presented. Likewise, we carry out an analysis of the terms most present in the pandemic political discourse.

4.2 Data Set

In the overall data set, **10,932** tweets have been collected and published by the official accounts of the political parties that make up the Spanish parliament. The tweets were published in the period **from March 6 to May 22**, a total of **78 days**. The average activity presented by the global group of parties in the whole period is **140,16 tweets per day**. In the period, the record for tweets posted by a single match in a day is @PSOE with 77 tweets on March 26. Analyzing the reality at that time, we observe an increase of COVID-19 cases of 8.3k positive and 655 more deaths than the previous day. On March 26, Spain was at the peak of daily infections with a global count of 4,089 dead and 57,786 positive.

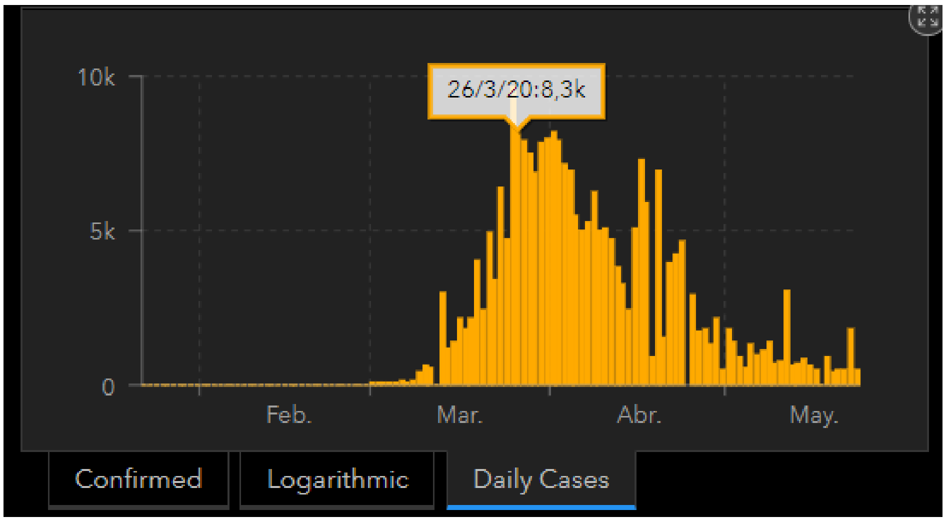


Figure 4.1: Spain Daily Cases Curve [5]

It is possible to associate in a deductive way the increase of activity by the account of one of the parties that compose the government in those days.

Concerning the distribution of publications, we found several differences in the activity presented by each author displayed in Table 4.2.

Tweet Distribution by Authors									
User	@PSOE	@CiudadanosCs	@populares	@PODEMOS	@esquerra_erc	@juntsxcat	@vox_es	@cupnacional	Others
Nº Tweets	2.741 tweets	1.417 tweets	1.296 tweets	1.154 tweets	802 tweets	774 tweets	519 tweets	432 tweets	1797 tweets
(%) Tweets	25,1%	13,0%	11,9%	10,6%	7,3%	7,1%	4,7%	4,0%	16,4%

Table 4.1: Tweet Distribution by Authors

When tweets were taken, Twitter took great importance due to the rules of the state of alarm. The political discourse had to adapt and therefore spread through the social networks. Observing the global count of activity presented by most of the authors in Table 4.2, we can deduce that the highest publication activity is presented by the parties, most of which have a sizeable electoral representation.

4.3 Displaying tools

According to the different graphs shown of the aggregated data, we can show the activity statistics by account, as well as the temporal progression of the activity, or the sentiment, among others. All these graphs, with the functioning of the data filters, explained in the previous section, allow us to obtain reversible functionalities.

With these web components, we can select the concept, feeling, day, or author on which we want to filter, leaving different interpretations of the data, varying the point of view.
As we can see in the following tables, we illustrate the different data collected regarding the activity presented by each of the parties.

With the dashboard, we can visualize in a filtered and ordered way the whole data set of the system. Some components allow us to select data sets that share a characteristic, providing specific functions to the user:

- **Activity and feelings by the author:** The overall activity and feelings in the entire data set are those initially displayed at the start of the service. In the case of wanting to obtain the data and feelings associated with an author, we select the one chosen in the sector diagram, updating the table completely.
  If we want to continue filtering within the author's tweets, we have several options. By a lay do we could introduce a term, and observe the feeling that the previously selected author has had in the tweets in which the word is present. On the other hand, we can obtain in the tweets box filtering by the feeling that the author has had in the global tweets set.

  Finally, we can also observe the variation of the tweets during the temporary period, as well as the associated feeling. We can even deduce the evolution of the feeling projected in the tweets during the whole period. In the case of placing events in parallel, we could extract the polarity of each author on each event.

- **Data statistics by feeling:** If we want to filter to know the general feeling of the set of tweets, we select the desired one in the dashboard.

Automatically, this is updated, showing only the tweets with the chosen feeling so that we can analyze the activity and contribution to the total by the different parties within the chosen feeling. Thanks to the reciprocity, we can filter the tweets by concepts and with the component that shows us the activity of the authors, to know who has had more importance and who has tweeted more the word, with the global feeling chosen.

- **Global feelings by concepts:** We have the function of knowing the statistics of the tweets that include a term in the text. We could know the temporality of an event thanks to the figures that show us the activity of the matches within the global period.

4.4 Data Analysis

After analyzing the entire set of tweets, we can extract a series of conclusions, about the different uses that we can give to the system, classifying our data in terms of activity in the network, the general feeling, or concerning a concept, that has a particular author, and the visibility of the concepts in a temporary way.

Firstly, an analysis of the total number of tweets published allows us to know a series of data. Generally, the most active author is the @PSOE, with 25.4% of the total published tweets, followed by @CiudadanosCs with 13.3% and @PODEMOS with 10.8%.

Regarding the sentiment analysis made to all the authors, we can see in the Table 4.2 the distribution of the total sentiment in the most outstanding authors and the average for each of them, being the authors shown, those who have used more than sentiment regarding the own set of tweets.

More Prominent Authors by Sentiment				
Sentiment	**Global Average**	**1º**	**2º**	**3º**
Positive	37,6%	@PSOE (49,12%)	@prcantabria (49,12%)	@PODEMOS (45,90%)
Negative	31,7%	@vox_es (53,40%)	@navarra_suma (50,00%)	@ehbildu(45,39%)
Neutral	30,7%	@compromis_mesgu (70,4%)	@esquerra_erc (67,12%)	@cup_nacional (65,93%)

Table 4.2: More Prominent Authors by Sentiment

Likewise, we could analyze the feelings per day, obtaining a global computation with which it is easy for us to analyze the variation of the feeling projected by an author or in a general way like Fig. 4.2.

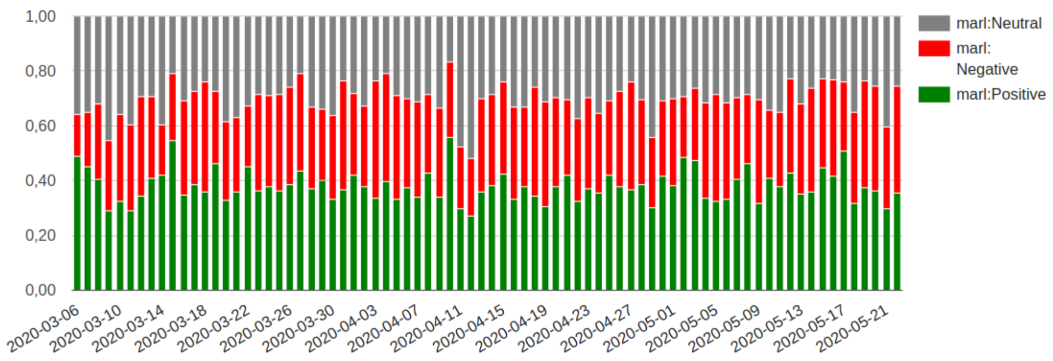


Figure 4.2: Chronological General Sentiment Distribution

As with the authors, we also can analyze the feeling projected on concepts. With the other components, we classify them by days, authors, and validity of the concept in the period. Regarding the study of terms, being able to analyze the feelings shown by the different parties, we highlight a series of terms used in a majority way, and how the feeling varies, depending on the party we analyze.
We are in the COVID-19 pandemic. It is essential to highlight that, due to the confinement, many of the political formations use the social networks as a means of communication.

We highlight a series of more critical terms within the period. These terms had a more significant presence in the global political discourse; thus, they are more susceptible to a general analysis involving different authors. In view of the period in which we find ourselves, political and health terms have been taken into account, which in turn have become quite famous in the overall period. The different causes and analysis of the words are related in the annex, respectively for these terms. we have analyzed "gobierno" (Sec. C.4), "país" (Sec. C.6), "España" (Sec. C.2), "COVID19" (Sec. C.1), "crisis" (Sec. C.3), "estado de alarma" (Sec. C.5). We have also included one of the most widely used slogans of the nation's government, "#EsteVirusLoParamosUnidos" (Sec. C.7). A more detailed description of each of the analyses performed on the different terms can be found in the Appendix. C.

In this study, we have carried out an analysis of the most published terms. We indicate the most outstanding percentages concerning the polarity of feeling projected by the parties about "gobierno" in Table 4.3.

Also, there is an overall comparison of the different accounts that included the word "gobierno" in Fig. 4.3. We can observe a clear tendency as a side to ideology concerning the feelings projected on the word "gobierno", observing a positive polarity, in the case of

Data Collected by term "gobierno"			
Activity → 1.787 Tweets		**Most Active Period** → 11/03 - 21/05	
Global Sentiment	**Positive 41,0%**	**Negative 42,1%**	**Neutral 16,9%**
Term most used by @CiudadanosCs (24,72%)			
More Positive	@PODEMOS (55,06%)	@PSOE (51,73%)	@eaj_pnv (43,39%)
More Negative	@navarra_suma (60,41%)	@vox_es (54,36%)	@populares(49,14%)

Table 4.3: Data Collected by term "gobierno"

the governing parties, as well as a negative polarity of feeling, the further away the parties position themselves ideologically.

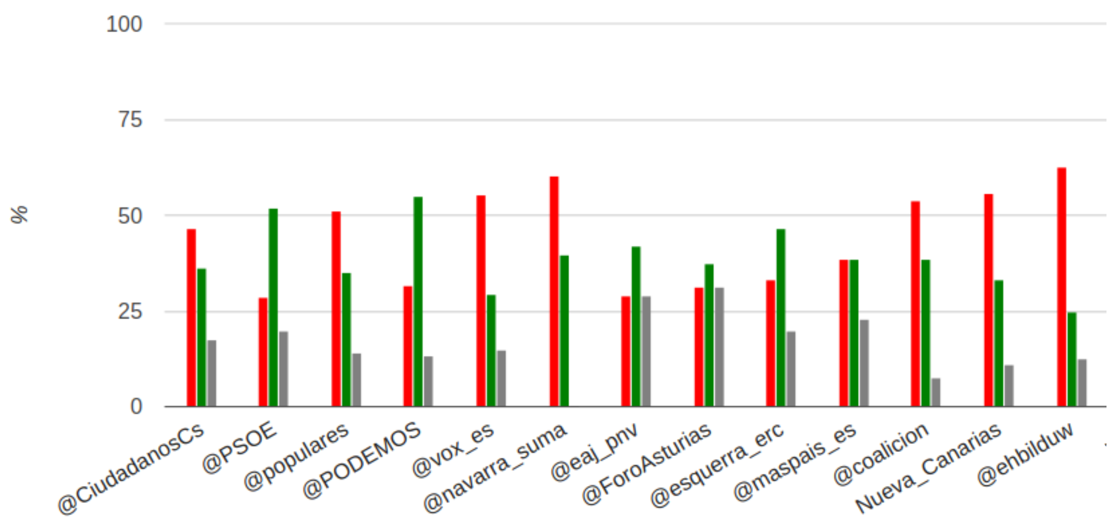


Figure 4.3: Global sentiments about "gobierno"

In the following analysis, we can observe the data associated with the concept "estado de alarma". This analysis allows us to observe the temporal activity of the term in the network debate directly in Fig. 4.4. These publication intensity peaks coincide with the events broken down in Table 4.4, corresponding to the declaration and successive extensions of the alarm status. Also, Table 4.4 presents a breakdown of the essential percentages regarding feelings and activity on the part of the authors of the term "estado de alarma".

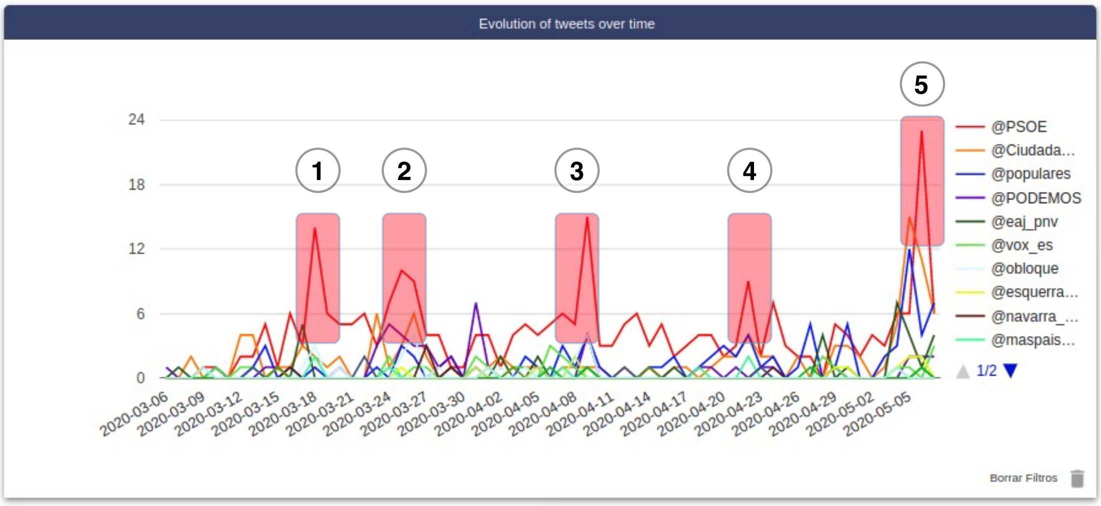


Figure 4.4: Temporal activity of “estado de alarma”

Alarm Sttextbf Events	
1	Alarm State Declaration → 14/03/2020
2	1st Alarm State Extension → 27/03/2020
3	2nd Alarm State Extension → 10/04/2020
4	3rd Alarm State Extension → 24/04/2020
5	4th Alarm State Extension → 11/05/2020

Table 4.4: Alarm State Events

These data correspond to the number of tweets that use the term, the period when was most used. Finally, we breakdown the most prominent authors with the most outstanding sentiment. All of them show polarization, which coincides, to no small extent, with the ideological one, regarding support for the government.

As can be seen, there is an increase in activity in the final phase. From these data, we can deduce that this variation is due to the loss of government support [14], and the consequent more considerable debate in the network.

In several of the terms analyzed in this project, we can observe an illogical confrontation in almost all popular terms, an example of this is displayed in Fig. 4.5. Being able to identify

Data Collected by term "estado de alarma"			
Activity → 650 Tweets		**Most Active Period** → 2/05 - 7/05	
Global Sentiment	**Positive 36,90%**	**Negative 39,5%**	**Neutral 23,5%**
Term most used by @PSOE (42,5%)			
More Positive	@eaj_pnv (47,06%)	@PODEMOS (44,89%)	@PSOE (37,68%)
More Negative	@esquerra$_{e}rc(62,50\%)$	@navarra_suma (57,14%)	@vox_es (51,85%)

Table 4.5: Data Collected by term "estado de alarma"

subgroups at a hierarchical level, as it could be government and opposition, extracting a clear tendency of polarization. In Fig. 4.5 it is shown for each author the proyected sentiment about "España". This fact, feeds the theory that only concepts are politicized and associated with a party, which can lead to prejudices or misunderstandings caused by fanaticism.

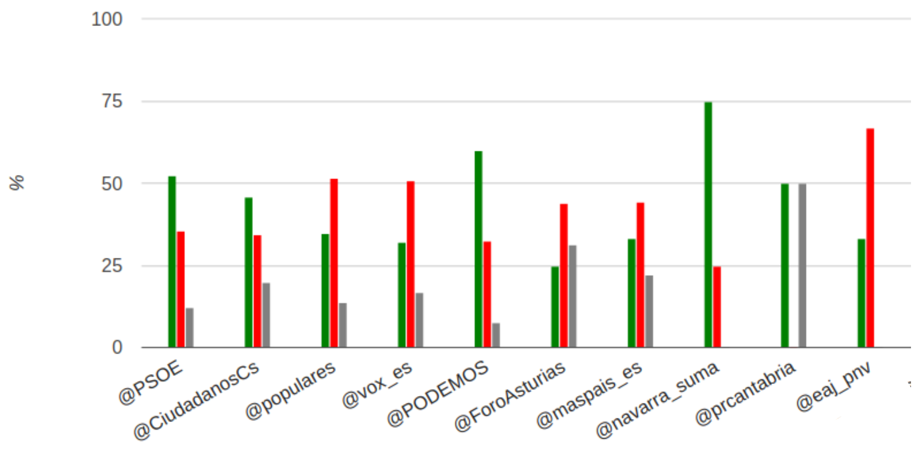


Figure 4.5: Global sentiments about "España"

We have only observed a real union between all the parties in the concept 'crisis', which could be due to the double meaning possible between economic crisis and health crisis. We show the global response for this term in Fig. 4.6

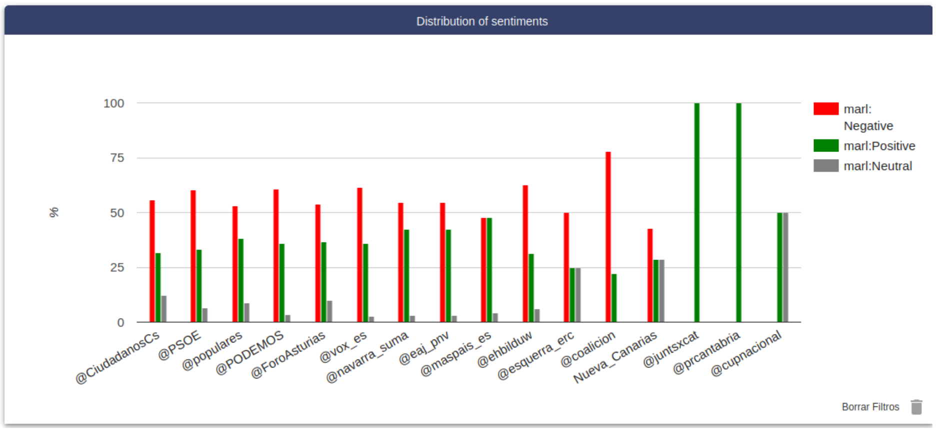


Figure 4.6: Global sentiments about "crisis"

In short, we can get to know the percentage of the polarity of sentiment in the term during the period of publications. Also, we can even know the source and trends that were followed in the event.

CHAPTER 5

Conclusions and future work

As conclusions to this document, we highlight a series of conclusions about the data that we have found remarkable. It also describes the obstacles encountered in the development of the project. Finally, it exposes possible improvements to be implemented in the system as future work.

5.1 Conclusions

This situation has caught everyone off guard. No government would have liked to govern in such a situation. Even so, we believe it is important to emphasize that their work is based on managing the state and ensuring the rights and enforcing the laws of the entire citizenry.

The state of alarm began on March 16, 2020, resulting in the confinement of the vast majority of the population in their homes, which caused an increase in the use of social networks, and therefore, the activity of political discourse on the network. The use of the network as a political means is evident, and the possible use we project depends on the tools we have. In almost all the different searches carried out on different terms, a polarization can be observed, which, beyond being logical, is similar to the projected feelings on a particular subject.

As an essential aspect to highlight, we can observe similarities in the different accounts on the same topic, attending to ideological criteria. It can be deduced that different parties with a similar ideology show the same polarity of feeling, in a grouped way, and opposed to that of the opposition, on a concept or public issue on the net. This analysis shows the innate projection that was discussed in the State of the Art (Sec. 2.1) when it comes to finding and protecting your allies within the electoral panorama. One of the leading indicators is found in the concept of the herd. This concept explains the differentiated polarization we find between parties of different ideologies.

One of the characteristics observed at the time of the project's development was the progressive change that the discourses had, each time distancing more from a conciliatory position. This could lead to a discourse that is more than political, provoked by the misfortune of the emergency experienced and the discontent. , in global terms, a tendency towards negative sentiment is observed at almost all times in their political speeches to the most liberal parties. In the case of the governing parties as a whole, there is a clear tendency towards positive sentiment.

In the same way, it is vital to highlight the analyses by parties regarding a concept, observing an apparent intensity in the accounts' activity in the moments of political debate. In some graphs, we observe a clear positive projection, and an increase in the activity of the network, in the moments in which these events take place. We must not forget certain functionalities quite useful when collecting information, such as the timing of an event, and the overall contribution by the political environment, create viral trends that end up becoming election campaign strategies.

To conclude, it is vital to highlight the use of this platform as a method of analyzing opinions, web activity, and emotions projected in the message of politicians. This system associates the projected feeling, and we can visualize divisions between different ideological groups. Likewise, we also see similarities with general concepts that imply a good or lousy projection, as far as feeling is concerned.

5.2 Achieved goals

To complete the requirements that we have detailed at the beginning of the document, we present the different resolution methods of the different processes exposed, in turn, at the beginning of the document (Sec. 1.2):

- **Tweet capture using Twitter API:** Thanks to Tweepy, we can connect to the Twitter API, and apply our required JSON format. The tweets are stored in a JSON, with a global format for the integration of different environments.
- **Automate the system with Luigi:** Creation of a series of processes and pipelines. To provide the system with automation when analyzing the feelings and processing the information, as well as delivering it to the persistence server, implemented with Elasticsearch.
- **Implementation of personalized aggregations:** Grouping of the data, varying the keys of union between them, allowing us to know different points of view of the set of information.
- **Associate political speeches with sentiments:** Using sentiment analysis techniques, we associate the discourse published on the networks with an author, and using a search engine, we associate the general polarity of the tweet with the concept, being able to extract the feeling inherent in it.
- **Analyzing political reality in an emergency:** It is a fact that politicians are there to manage the state and largely prevent harm to citizens in a situation like this. In this project, we often observe confrontation. It is evident that there is a polarization that exists in front of terms that should not be politicized.

5.3 Problems found

During the development of the monitoring system, we encountered problems of various kinds:

- **Twitter API limitations:** Twitter offers API services that are limited and priced. In this project, we make use of the Standard API, which offers us a limited number of requests per day, a maximum number of tweets per request, and a maximum number of requests in a time window of 1 month.
  These limitations made the object of study of tweets was reduced only to the general official accounts of each party. These limitations prevented the analysis of more accounts. We could improve the area with the leading representatives, or the official accounts of the different ministries.
- **Language limitations:** Sentiment analysis, as we have seen, leads to the use of an API that analyses sentiment with Natural Language Processing techniques. The

service provided by the API only accepts a series of languages in the input text. Therefore all those tweets written in Basque, Galician, and Valencian had to be discarded, leaving only Spanish and Catalan as possible.

- **Learning new technologies and programming methodologies:** For the development of this project, research was not enough. It is necessary to learn from different web technologies, such as Polymer (Sec. 2.9) or Sefarad (Sec. 2.5), and even methodologies to carry out good practices when developing in different services, interconnected in the same environment.

5.4 Future work

This section includes the different tasks and improvements that could be carried out in the future to continue research in this field. These improvements, above all, complement the analysis carried out, and add other possible dimensions of analysis, as well as other components to the display system.

- **Expansion of the total accounts** from which we extract the analysis tweets allowing us to perform a more specific analysis within each party, and obtain similarities and differences between personal accounts.
- **Design and implementation relation plots** in which we could observe the relations within the network, identify internal flows of the information in the network, and integrate it with some verification service and find sources of hoaxes.
- **Development and training** of a classifier with a linear discriminant algorithm (LDA). This allows us to know the most outstanding terms, topics, and concepts within the entire set of information.

APPENDIX A

Impact of this project

This appendix reflects, quantitatively or qualitatively, on the possible impact in different respects. We consider four possible dimensions of impact on the project, social, economic, political, environmental, and finally, we will take into account some ethical implications as a result of the project.

A.1 Social impact

As we commented in Section 2.1, the use of Twitter as a social media is a fact, and this project can have consequences when it comes to an understanding of the discourse of the main political parties. As well as an analysis of specific terms, alluding to subjective concepts. These concepts depend on the attitude or feeling with which they are addressed in the social network.

This project provides a tool for citizens to analyze the behaviors that took place in the COVID-19 pandemic. The political leaders in this situation took much more importance, which is why the analysis of these organizations is more necessary than ever, letting us see on many occasions what is vital for each party.

A.2 Economic Impact

The project refers directly to the various official accounts of the political parties represented in parliament. It is important to remember that the subsidies and aids granted to political parties depend on the party's greater or lesser presence in public office.

This fact is directly related to the popularity they have among the citizens. Therefore, this project breaks down different attitudes towards terms and statistical data on the activity presented by the parties' official accounts. All this information, in the hands of the citizens, can refine the general voting decision, directly impacting the internal economy of each party.

A.3 Environmental Impact

As Twitter is a medium in which proposals can be debated, with this project, we can measure the activity of environmental concepts in the different discourses offered by the accounts. This fact creates debate on the network and, by measuring the policies and discourses made by the party and carrying out a sentimental analysis, we can find out its position on the subject. This position could affect by influencing the popularity of the party and promoting the creation of measures for the environment.

Thanks to the versatility that this project has for the infinity of concepts that we can compare. It is possible to know the political organization's opinion regarding environmental issues or problems discussed in the network.

A.4 Ethical Implications

Many times the goal of Big Data studies is citizenship. In this project, the Big Data is born as an objective to provide a more technical analysis of the different political parties about the attitude they print in their speeches and debates.

Today, freedom of the press hangs by a thread of commercial, business and hidden interests. This project tries to offer an impartial vision, applying sentiment analysis algorithms in an equal way, to all parties, in order to try to understand the opinion of the parliamentary group in the coronavirus crisis.

APPENDIX B Economic budget

This appendix details an adequate budget to bring about the project...

B.1 Physical resources

This section presents the budgets derived from physical resources for the development of this End-of-Grade work. For the development of the prototype, we have used Ubuntu 18.04 virtualized with VirtualBox. The host system had the following characteristics:

- **CPU: Intel Core i7 8th Gen**
- **RAM: 16 GB**
- **SSD: 200 GB**
- **HDD: 1000 GB**

It does not mean that development in another environment was not possible, since VirtualBox allows us to develop with Ubuntu on any system, but the power of the host system has been beneficial for indexing data.

B.2 Human resources

This section details the various costs associated with the human resources required for this project.

We assume that the project has been carried out entirely by one person. This project has taken five months to complete. Taking into account that this is the work of one student, and the average fee received per scholarship in this body is 450€ per month.

The cost associated, in a symbolic way, to the realization of the project, would amount to 2250€.

B.3 Licenses

This section details the budgets associated with the licenses required to develop with the software necessary for the project.

In this case, all the technologies used in this project are Open Source, which means that no cost is derived from the project realization. Including the Twitter API, from which we will use your standard product, which is also free.

APPENDIX C

Analysed Terms

In this chapter, we show the method by which we develop a complete analysis of a term. As well as, a series of analysis made to the terms mentioned above (Sec. 4.4). We have broken down the terms on which we have focused the analysis of the environment.

C.1 Term "COVID19"

This term is one of the most important of the period, it is the name of the disease produced by the new Coronavirus.
As we can see in the Fig. C.1, @CiudadanosCs is the user who has used it most with a 31.5% presence. Likewise, the most positive and most negative authors are broken down, respectively.

Data Collected by term "COVID19"			
Activity → 894 Tweets		**Most Active Period** → 10/03 - 21/03	
Global Sentiment	**Positive 31%**	**Negative 31%**	**Neutral 38%**
Sentiment	**1º**	**2º**	**3º**
More Positive	@maspais_es (83,33%)	@navarra_suma (66,72%)	@PODEMOS (57,14%)
More Negative	@vox_es(68,4%)	@obloque (65%)	@ehbildu (61,5%)
Term most used by @CiudadanosCs (42,5%)			

Table C.1: Data Collected by term "COVID19"

C.2 Term "España"

We chose this term because of the scope of the system. We considered it to be an essential and recurring word.The Table C.2 shows that the author with the most activity is PSOE, with 35.5% of total activity presented by the term.

Data Collected by term "España"			
Activity → 642 Tweets		**Most Active Period** → 10/03 - 20/05	
Global Sentiment	**Positive 44,9%**	**Negative 39,9%**	**Neutral 15,3%**
Term most used by @PSOE (35,5%)			
More Positive	@PODEMOS(60,15%)	@PSOE (52,19%)	@CiudadanosCs (45,78%)
More Negative	@populares(51,51%)	@vox_es (50,85%)	@maspais_es(44,44%)

Table C.2: Data Collected by term "España"

C.3 Term "Crisis"

The disease caused by the new coronavirus was a crisis in almost all areas, from health, economic, and social. Politicians had to use it in their speeches. In the Table C.3 we can see that the author with the most activity is CiudadanosCs, with 31,8% of total activity presented by the term.

Data Collected by term "crisis"			
Activity → 642 Tweets		**Most Active Period** → 18/03 - 31/03	
Global Sentiment	**Positive 34,9%**	**Negative 56,8%**	**Neutral 8,3%**
Term most used by @CiudadanosCs (31,8%)			
More Positive	@maspais_es(47,82%)	@eaj_pnv (42,42%)	@navarra_suma (42,17%)
More Negative	@ehbildu(62,5%)	@vox_es (61,53%)	@PODEMOS(60,71%)

Table C.3: Data Collected by term "crisis"

C.4 Term "gobierno"

By the time the tweets were picked up, the political atmosphere was already tense. The successive extensions of the state of alarm provoked an intense debate in the networks about the government's measures taken or not. In the Table C.4 we can see that the author with the most activity is CiudadanosCs, with 24,7% of total activity presented by the term.

Data Collected by term "gobierno"			
Activity → 1.787 Tweets		**Most Active Period** → 11/03 - 21/05	
Global Sentiment	**Positive 41,0%**	**Negative 42,1%**	**Neutral 16,9%**
Term most used by @CiudadanosCs (24,72%)			
More Positive	@PODEMOS (55,06%)	@PSOE (51,73%)	@eaj_pnv (43,39%)
More Negative	@ehbildu(62,50%)	@navarra_suma (60,41%)	@vox_es (54,36%)

Table C.4: Data Collected by term "gobierno"

C.5 Term “estado de alarma”

As well as with term “gobierno”, the successive extensions of the state of alarm provoked an intense debate in the networks about the government’s measures about the state of alarm, taken or not. In the Table C.5 we can see that the author with the most activity is PSOE, with 42,5% of total activity presented by the term.

Data Collected by term “estado de alarma”			
Activity → 839 Tweets		**Most Active Period** → 2/05 - /05	
Global Sentiment	**Positive 39,10%**	**Negative 38,6%**	**Neutral 22,3%**
Term most used by @PSOE (40,4%)			
More Positive	@PODEMOS (49,18%)	@eaj_pnv (47,06%)	@CiudadanosCs (41,14%)
More Negative	@Nueva$_canarias(57,14\%)$	@vox_es (54,28%)	@populares (49,23%)

Table C.5: Data Collected by term “estado de alarma”

C.6 Term “país”

As with the term ’Spain’, one of the most recurrent themes in the political speech was the term ’country’, yet the significant change of roles within the prominent positions is remarkable. In the Table C.6 we can see that the author with the most activity is CiudadanosCs, with 31,8% of total activity presented by the term.

Data Collected by term “país”			
Activity → 642 Tweets		**Most Active Period** → 18/03 - 31/03	
Global Sentiment	**Positive 46,10%**	**Negative 31,10%**	**Neutral 22,80%**
Term most used by @PSOE (31,10%)			
More Positive	@ehbildu(70%%)	@CiudadanosCs (55,81%%)	@PODEMOS(55,17%)
More Negative	@populares (51,28%)	@vox_es (50,00%)	@eaj_pnv (48,12%)

Table C.6: Data Collected by term “país”

C.7 Term "#EsteVirusLoParamosUnidos"

As one of the slogans created due to the current situation, we thought it was convenient to analyze it to observe its statistics. The intensive use in the early parts of the pandemic is remarkable. In the Table C.7 we can see that the author with the most activity is PSOE with a 90,8% of total activity presented by the term.

Data Collected by term "#EsteVirusLoParamosUnidos"			
Activity → 802 Tweets		**Most Active Period** → 14/03 - 03/04	
Global Sentiment	**Positive 51,90%**	**Negative 30,50%**	**Neutral 17,6%**
Term most used by @PSOE (90,81%)			
Publication Average → **40,10** daily tweets			

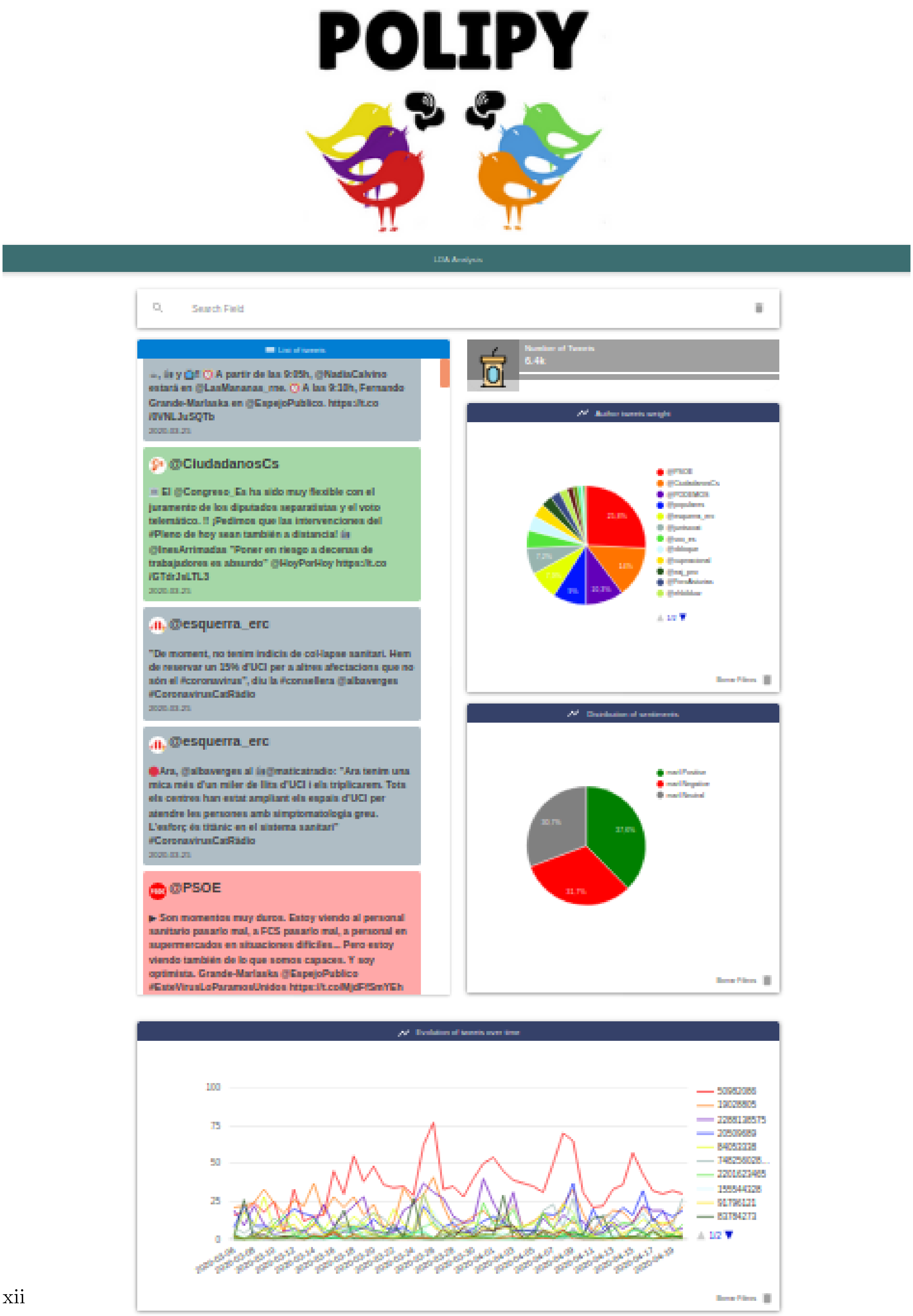
Table C.7: Data Collected by term "#EsteVirusLoParamosUnidos"

Unlike the other analyses, in Fig. C.7 we do not point out outstanding authors, but we do point out their broad spectrum of activity carried out by a single party, in which we can observe an average of 40 tweets a day, which contained the term "#EsteVirusLoParamosUnidos".

APPENDIX D

Dashboard

This chapter shows how is Polipy Dashboard and its different tabs. And how it changes if a term is introduced.



…, … y …! A partir de las 9:05h, @NadiaCalvino estará en @LasMananas_rne. A las 9:10h, Fernando Grande-Marlaska en @EspejoPublico. https://t.co/0VNLJuSQTb
2020.03.21

@CiudadanosCs

El @Congreso_Es ha sido muy flexible con el juramento de los diputados separatistas y el voto telemático. ‼ ¡Pedimos que las intervenciones del #Pleno de hoy sean también a distancia! @InesArrimadas "Poner en riesgo a decenas de trabajadores es absurdo" @HoyPorHoy https://t.co/GTdrJeLTL3
2020.03.21

@esquerra_erc

"De moment, no tenim indicis de col·lapse sanitari. Hem de reservar un 15% d'UCI per a altres afectacions que no són el #coronavirus", diu la #consellera @albaverges #CoronavirusCatRàdio
2020.03.21

@esquerra_erc

🔴Ara, @albaverges al @maticatradio: "Ara tenim una mica més d'un miler de llits d'UCI i els triplicarem. Tots els centres han estat ampliant els espais d'UCI per atendre les persones amb simptomatologia greu. L'esforç és titànic en el sistema sanitari" #CoronavirusCatRàdio
2020.03.21

@PSOE

▶ Son momentos muy duros. Estoy viendo al personal sanitario pasarlo mal, a FCS pasarlo mal, a personal en supermercados en situaciones difíciles... Pero estoy viendo también de lo que somos capaces. Y soy optimista. Grande-Marlaska @EspejoPublico #EsteVirusLoParamosUnidos https://t.co/MjdFfSmYEh

Figure D.1: Dashboard Page 1

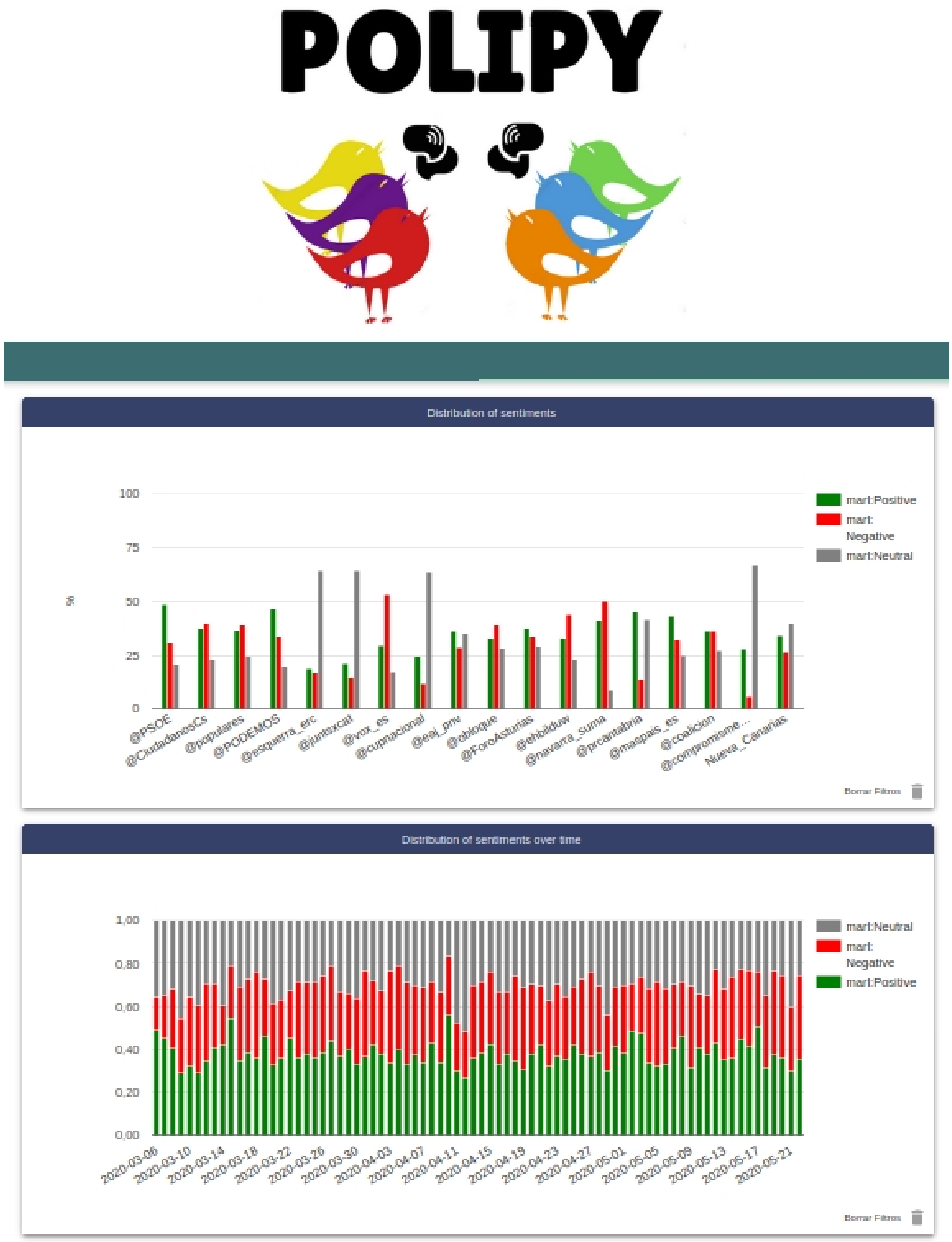


Figure D.2: Dashboard Page 2

Bibliography

[1] GSI UPM. Sefarad docs. `https://github.com/gsi-upm/Sefarad`, 2018.

[2] J. Fernando Sánchez-Rada, Carlos A. Iglesias, Ignacio Corcuera-Platas, and Oscar Araque. Senpy: A Pragmatic Linked Sentiment Analysis Framework. In *Proceedings DSAA 2016 Special Track on Emotion and Sentiment in Intelligent Systems and Big Social Data Analysis (SentISData)*, pages 735–742, Montreal, Canada, October 2016. IEEE.

[3] Adam Westerski J. Fernando Sánchez-Rada. Marl ontology specification. `https://www.gsi.upm.es/ontologies/marl/`, 2016.

[4] Gabriel Cartwright. Polymer: Future modularization of web. *SegueTech*, Jul 2014.

[5] Gardner L. Dong E, Du H. An interactive web-based dashboard to track covid-19 in real time. *Lancet Infect Dis*, Feb 2020.

[6] Muntazar Chandio and Melike Sah. *Brexit Twitter Sentiment Analysis: Changing Opinions About Brexit and UK Politicians*, pages 1–11. 01 2020.

[7] Wikipedia. Twitter. `https://es.wikipedia.org/wiki/Twitter`, 2013.

[8] Katrin Weller, Axel Bruns, Jean Burgess, Merja Mahrt, and Cornelius Puschmann, editors. *Twitter and society*, volume 89. P. Lang, 2014.

[9] Malak Abdullah and Mirsad Hadzikadic. Sentiment analysis of twitter data: Emotions revealed regarding donald trump during the 2015-16 primary debates. In *2017 IEEE 29th International Conference on Tools with Artificial Intelligence (ICTAI)*, pages 760–764. IEEE, 2017.

[10] Padmaja Katta and Nagaratna Parameshwar Hegde. A hybrid adaptive neuro-fuzzy interface and support vector machine based sentiment analysis on political twitter data. *International Journal of Intelligent Engineering and Systems*, 12(1):165–173, 2019.

[11] Rajesh Basak, Shamik Sural, Niloy Ganguly, and Soumya K Ghosh. Online public shaming on twitter: Detection, analysis, and mitigation. *IEEE Transactions on Computational Social Systems*, 6(2):208–220, 2019.

[12] Wasim Ahmed, Peter A Bath, Laura Sbaffi, and Gianluca Demartini. Novel insights into views towards h1n1 during the 2009 pandemic: a thematic analysis of twitter data. *Health Information & Libraries Journal*, 36(1):60–72, 2019.

[13] E. Twitter. `https://es.wikipedia.org/wiki/Twitter`, 2013.

[14] MADRID LEONOR MAYOR ORTEGA. Sánchez pierde apoyos en el congreso para la tercera prórroga del confinamiento. *La Vanguardia*, April 2020.

[15] Ziad Al-Halah, Andrew Aitken, Wenzhe Shi, and Jose Caballero. Smile, be happy :) emoji embedding for visual sentiment analysis. In *The IEEE International Conference on Computer Vision (ICCV) Workshops*, Oct 2019.

[16] David Lorentzen. Polarisation in political twitter conversations. *Aslib Journal of Information Management*, 66, 05 2014.

[17] Anjali Bhavan, Rohan Mishra, Pradyumna Prakhar Sinha, Ramit Sawhney, and Rajiv Shah. Investigating political herd mentality: A community sentiment based approach. In *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics: Student Research Workshop*, pages 281–287, 2019.

[18] Jaspreet Singh, Gurvinder Singh, Rajinder Singh, and Prithvipal Singh. Morphological evaluation and sentiment analysis of punjabi text using deep learning classification. *Journal of King Saud University - Computer and Information Sciences*, 2018.

[19] MUG Kraemer, N Golding, D Bisanzio, S Bhatt, DM Pigott, SE Ray, OJ Brady, JS Brownstein, NR Faria, DAT Cummings, et al. Utilizing general human movement models to predict the spread of emerging infectious diseases in resource poor settings. *Scientific reports*, 9(1):1–11, 2019.

[20] Christian E Lopez, Malolan Vasu, and Caleb Gallemore. Understanding the perception of covid-19 policies by mining a multilanguage twitter dataset. *arXiv preprint arXiv:2003.10359*, 2020.

[21] Yelena Mejova and Kyriaki Kalimeri. Advertisers jump on coronavirus bandwagon: Politics, news, and business. *arXiv preprint arXiv:2003.00923*, 2020.

[22] Various. Python home. `https://python.org`, 2001.

[23] Wikipedia. Json. `https://es.wikipedia.org/wiki/JSON`, 2001.

[24] Various. Tweepy. `http://docs.tweepy.org/en/latest/`, 2009.

[25] Twitter Developer. Twitter api. `https://developer.twitter.com/en/docs`, 2009.

[26] Docker. Twitter api. `https://www.docker.com/`, 2013.

[27] J. Fernando Sánchez-Rada, Alberto Pascual-Saavedra, Enrique Conde-Sánchez, and Carlos A. Iglesias. A Big Linked Data Toolkit for Social Media Analysis and Visualization based on W3C Web Components. In Henderik A. Proper Claudio A. Ardagna Dumitru Roman Robert Meersman Hervé Panetto, Christophe Debruyne, editor, *On the Move to Meaningful Internet Systems. OTM 2018 Conferences. Part II*, volume 11230 of *LNCS*, pages 498–515, Valletta, Malta, October 2018. Springer-Verlag.

[28] Elasticsearch B.V. Elasticsearch. `https://www.elastic.co/es/what-is/elasticsearch`, 2010.

[29] Bharvi Dixit, Rafal Kuc, Marek Rogozinski, and Saurabh Chhajed. *Elasticsearch: A Complete Guide*. Packt Publishing Ltd, 2017.

[30] Various. Luigi. `https://luigi.readthedocs.io/en/stable/`, 2013.

[31] Alexander Maedche and Steffen Staab. Ontology learning for the semantic web. *IEEE Intelligent systems*, 16(2):72–79, 2001.

[32] The Polymer Project Authors. Polymer web components. `https://www.polymer-project.org/`, 2018.

[33] Elastic. Elastic docs. `https://www.elastic.co/guide/en/elasticsearch/reference/7.6/search-aggregations.html`, 2018.

[34] Bootstrap Team. Bootstrap. `https://getbootstrap.com/docs/4.4/about/overview/`, Aug 2011.

[35] Google. Google charts docs. `https://developers.google.com/chart`.

[36] Oscar Araque. Design and Implementation of an Event Rules Web Editor. Trabajo fin de grado, Universidad Politécnica de Madrid, ETSI Telecomunicación, July 2014.

[37] J. Fernando Sánchez-Rada. Design and Implementation of an Agent Architecture Based on Web Hooks. Master's thesis, ETSIT-UPM, 2012.

[38] Omnicore Agency. Twitter statistics. `https://www.omnicoreagency.com/twitter-statistics/`, 1999.

[39] V Subramaniyaswamy, R Logesh, M Abejith, Sunil Umasankar, and A Umamakeswari. Sentiment analysis of tweets for estimating criticality and security of events. In *Improving the Safety and Efficiency of Emergency Services: Emerging Tools and Technologies for First Responders*, pages 293–319. IGI Global, 2020.