UNIVERSIDAD POLITÉCNICA DE MADRID

ESCUELA TÉCNICA SUPERIOR
DE INGENIEROS DE TELECOMUNICACIÓN

MÁSTER UNIVERSITARIO EN
TEORÍA DE LA SEÑAL Y COMUNICACIONES

TRABAJO FIN DE MASTER

Generation and Evaluation of Lexicons Focused on Hate Speech using Deep Neural Networks

David Algarra Medina

2021

TRABAJO DE FIN DE MASTER

Título:	Generación y Evaluación de Léxicos Centrados en Lenguaje de Odio usando Redes Neuronales
Título (inglés):	Generation and Evaluation of Lexicons Focused on Hate Speech using Deep Neural Networks
Autor:	David Algarra Medina
Tutor:	Óscar Araque Iborra
Ponente:	
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 DE FIN DE MASTER

Generation and Evaluation of Lexicons Focused on Hate Speech using Deep Neural Networks

Junio 2021

Resumen

El auge del uso de las redes sociales y de consumir contenido online en los últimos años ha supuesto un cambio en la manera en que la gente usa Internet y se informa. La cantidad de información generada cada día es de magnitudes nunca vistas, donde fluyen todo tipo de mensajes desde todos los puntos de vista. Entre esa información, desde hace unos años preocupa la libertad con la que se propagan mensajes de odio sin ningún tipo de filtro, provocando situaciones donde colectivos o minorías se ven discriminadas, y pudiendo causar daños físicos y mentales en las personas afectadas.

La libertad de expresión permite que cada uno puede expresar sus opiniones libremente, pero este derecho acarrea con unas responsabilidades y es que se puede restringir si el mensaje interfiere con los derechos humanos de otras personas. Mensajes apoyando el racismo, discriminando mujeres o a miembros del colectivo LGTB o propagando mensajes del ideario nazi no están permitidos por las autoridades locales ni por las plataformas de redes sociales.

Este trabajo está enfocado en generar un léxico donde cada palabra contiene lo relacionada o no que está con el lenguaje de odio. Para ello, se utilizan técnicas de Procesado de Lenguaje Natural y de análisis de sentimientos. Primero, se realiza un estudio para analizar sentimientos de palabras, si son positivos o negativos, de una colección de textos. Y en segundo lugar, aplicar las técnicas del problema anterior a un problema para generar un léxico que contenga el lenguaje de odio que las palabras tienen.

Por último, se evalúan dichos léxicos con texto real y comprobar si es capaz de identificar sentimientos o lenguaje de odio en frases o párrafos. Así se demuestra el alcance y el potencial impacto que pueden tener estas herramientas en el futuro.

Palabras clave: Natural Language Processing, NLP, análisis de sentimientos, machine learning, redes neuronales, word embeddings, lenguaje de odio

Abstract

The rise in the use of social media and the consumption of online content in recent years has meant a change in the way people use the Internet and get information. The amount of information generated every day is of unprecedented magnitude, with all kinds of messages flowing from all points of view. Among this information, the freedom with which hate messages are spread without any kind of filter has been a concern for some years now, causing situations where groups or minorities are discriminated against, and may cause physical and mental damage to the people affected.

Freedom of expression allows everyone to express their opinions freely, but this right comes with responsibilities and can be restricted if the message interferes with the human rights of other persons. Messages supporting racism, discriminating against women or members of the LGBT community or propagating messages of Nazi ideology are not allowed by local authorities or social media platforms.

This work is focused on generating a lexicon where each word contains how related or not it is to hate speech. For this, Natural Language Processing and sentiment analysis techniques are used. First, a study is carried out to analyze word sentiments, whether they are positive or negative, from a collection of texts. And second, apply the techniques of the previous problem to a problem to generate a lexicon containing the hate language that the words have.

Finally, we evaluate such lexicons with real text and check whether it is able to identify sentiment or hate language in sentences or paragraphs. This demonstrates the scope and potential impact that these tools can have in the future.

Keywords: Natural Language Processing, NLP, sentiment analysis, machine learning, neural networks, word embeddings, hate speech

Acknowledgement

To my family, my friends and especially to my tutor for the help I have received to complete this project.

Contents

List of Figures

CHAPTER 1

Introduction

This chapter presents the introduction of the project. It is introduced the context of the different concepts treated throughout the study. It will also break down the different objectives followed during the realization of the project. Finally, an overview of the different chapters that compose the document will be given.

Language has been a subject of study since the 15th century BC by Indian grammarians [18], which marks the earliest record that exists today and it was not until the 19th century when it was introduced the modern concept known as linguistics. Linguistics is the scientific study of language [30], and linguists are scientists who apply the scientific method to questions about the nature and function of language. They make observations, test hypotheses, and develop theories about every concept surrounding language.

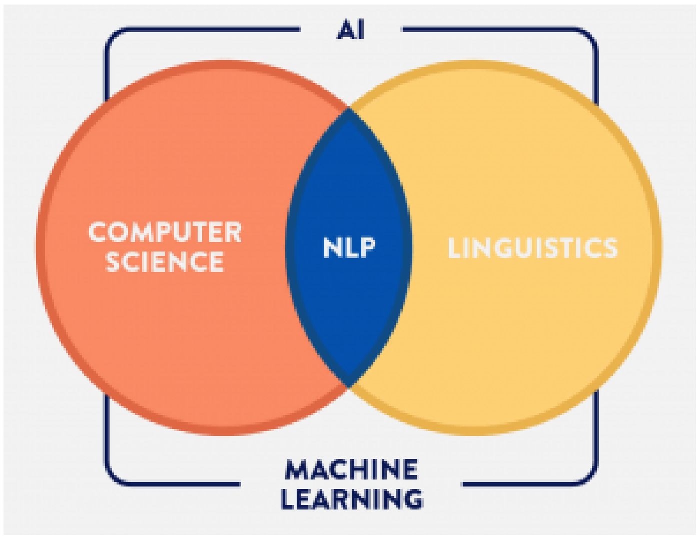


Figure 1.1: Natural Language Processing diagram as part of Computer Science and Linguistics [15]

Later on, it was developed the first computer-based application related to natural language in 1949, the Weaver´s Memorandum, a machine translation application.

From this and other projects was born what is known today as Natural Language Processing, a sub-discipline of Linguistics, Computer Science and Artificial Intelligence as figure 1.1 shows, dedicated to program computers to process and analyze large of natural language data. The task to process human language is highly ambiguous. It is also ever changing and evolving. People are great at creating language and be able to understand language, and are capable of expressing, perceiving, and interpreting very elaborate and detailed meanings in different contexts. At the same time, as humans are great using the language, there are no rules to formally describe and understand the language [43].

While the difficulties are present, the accessibility to computationally powerful machines and large data sets has led to a massive increase in the research of the Natural Language Processing field, as it is shown in figure 1.2 an exponentially rate of growth of NLP published papers since the beginnings of the century, and with it the eruption of new techniques and numerous use cases like for example [1]:

- Email filters. One of the most basic applications. It started as a spam filter, which mails was identified by certain words. Newer applications for example like the Gmail one, identifies the mail into one of the categories of primary, social, promotions and updates to keep the inbox more manageable and highlight the relevant mails.

- Smart assistants. Perhaps the most eye-catching application for how a revolutionary technology has become part of everyday life. Now it is not strange to 'talk' with a machine, this Smart assistants like Apple's Siri, Amazon's Alexa or Google assistant recognize patterns in speech thanks to voice recognition, infer the meaning of the command and provide a useful answer. The success of this technology in the society is remarkable, the 38.5% of US population has used a voice assistant in 2020 and the trend is still increasing [28].

- Predictive text. We are so used to things like autocorrect, autocomplete and predictive text that it is taken for granted when writing on smartphones. Some interesting aspect of this algorithms is that they learn as they are used, so they adapt to every person and every predictor is different from one person to another.

- Language translation. Online translators as Google Translate or DeepL are getting consistently better and present a more and more natural result. They even can detect automatically the input language and translate to the desired language directly, for example when entering a web page in another language.

- Digital Phone calls. When calling a customer support, it may sound that the call is recorded for training purposes, this means that the conversation will be used to train an NLP model so in the future an automated system will answer the customer calls.

The reflection of all the advances made in this field is evident in the technology business. NLP is a key technology and has a bright future as it can be seen in figure 1.3, it shows a 2017 Tractica report on the NLP market that estimates the revenue obtained from the total of NLP software, hardware, and services market from 2016 to 2025. The report forecasts that the revenues will go from a few billions dollars at the time the report was made to almost 25 billions dollars in 2025.

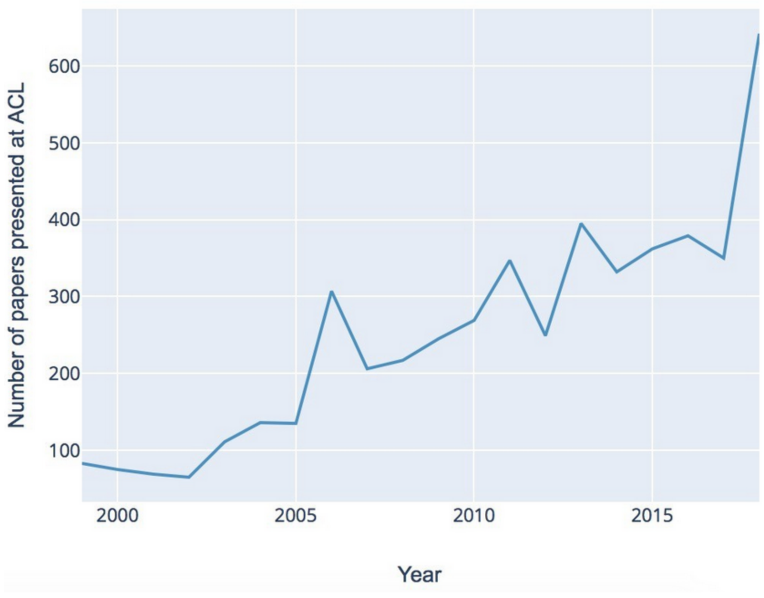


Figure 1.2: Papers published at the ACL conference by years [47]

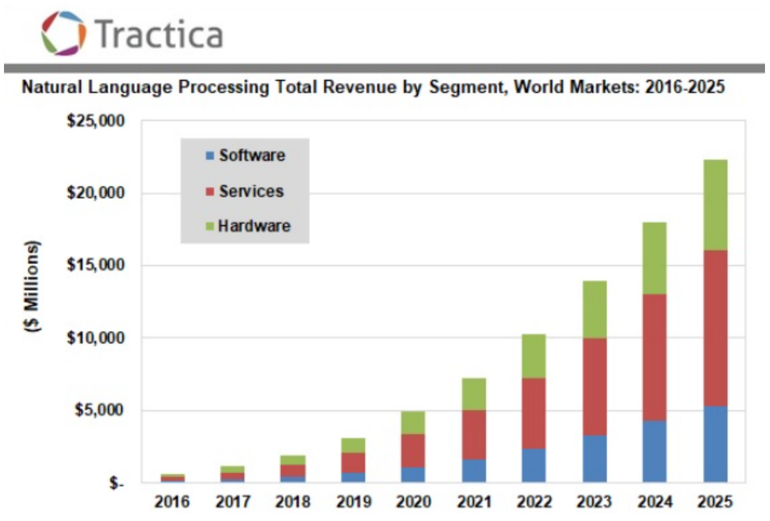


Figure 1.3: NLP revenue by segment, 2016-2025 [11]

1.1 Context

This section presents the context of the project and are summarized the relevant points treated in the project.

1.1.1 Sentiment analysis

Sentiment analysis is a NLP technique that analyzes text and detects subjective information, such as sentiment, opinion or attitude towards a topic. The principal demand comes from companies interested in the opinion related to a certain product or campaign, so if a company is interested in the opinion of its users about a product, it can collect the opinions in the web, process it and apply sentiment analysis algorithms to obtain information if the opinion towards its product is positive or negative.

According to the study [51] almost 7000 published papers related to sentiment analysis in the citation database Scopus were reviewed and they concluded that this research area is one of the fastest growing. Of the nearly 7000 papers, 99% of them appeared after 2004. The first known investigations date back to the World War II, where it was measured the public opinion in the political field. Sentiment analysis boom came in the mid 2000s with the appearance of product reviews available on Internet.

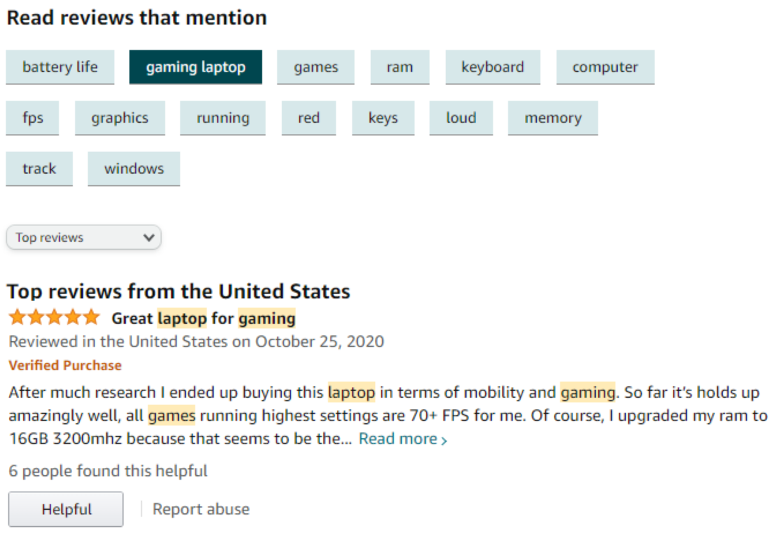
Nowadays there are more opinions than ever, people express themselves openly on any subject and in many ways, opinions can be found in Facebook posts, tweets, online blogs, recommendation web pages, etc. And it is not only that the quantity of reviews are increasing, it is the importance of them for users, according to the study [36], products that have multiple reviews can have a conversion rate up to 270% more than a product with no reviews. And results soar for high priced products with up to a 380% increase in the conversion rate, and precisely those products accumulate more reviews than low-prices ones.

Companies know that reviews are a very important resource. Having a good reputation is important to get new users interested in their product. And if the needs of the new customers are satisfied and the experience is positive, they will review positively across the Internet and engage new users, so a loop is made where fulfilling user demands is the engine that makes it work. Ultimately, it has a positive impact on users since a company will effort to make a good product in order to grow.

Then organizations are interested in knowing what the comments say because with that information they can improve their product, such as adjusting the marketing target, exploring new opportunities and understanding better the customers in general. But with the large volume of information it can be difficult to identify the relevant content on users' opinions.

That's when sentiment analysis comes into play, it can manage a lot of information and give an insight into the most relevant topics. Companies apply sentiment analysis in various sectors, like:

In social media and brand monitoring, it allows companies to know about a certain topic and users opinions in real time, for example if there is a bug or if the web is down, the issue will be mentioned in social media immediately. Another use is that they can know how a campaign is working, for example, Nike launched a campaign where the company positioned itself on a controversial issue. It was an anti-racism campaign and it received compliments from users and even from big celebrities and caused a huge impact with millions of visualizations. This time the marketing strategy was a success but for example in the case of El Corte Ingles campaign [58], it received numerous comments in social media where users associated the picture of a child's shoes with suicide. The company rectified and ended up cancelling the campaign.



Top reviews from the United States

Great laptop for gaming

Reviewed in the United States on October 25, 2020

Verified Purchase

After much research I ended up buying this laptop in terms of mobility and gaming. So far it's holds up amazingly well, all games running highest settings are 70+ FPS for me. Of course, I upgraded my ram to 16GB 3200mhz because that seems to be the... Read more ›

6 people found this helpful

Figure 1.4: Review of a laptop at Amazon.com

In marketing, if a company wants to know the users' feelings towards a product, instead of conducting a survey, it can collect the reviews in e-commerces like Amazon and evaluate if the comments are positive or negative. Also, it can detect what users are more worried about, what aspects they focus more when buying a product, etc. The figure 1.4 shows a review of a gaming laptop at Amazon, the sentiment analysis can be applied users' comments, but in this example Amazon uses an additional NLP technique, it separates the reviews in different categories depending on what they talk about like gaming laptop, battery, ram, graphics, etc. So the sentiment analysis can be applied to each one of the categories and it can reveal the reason of negative comments if they have a common category, or what it is liked the most about the product.

Sentiment analysis can be applied in a wide range of applications, it can detect sentiment in a product, a service, a brand and even movie reviews. It can also indicate political sentiment, the paper "Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment" [64] demonstrates that political sentiment obtained in tweets can be associated to political parties and candidates.

1.1.2 Hate speech

The main point of this project is hate speech analysis. According to Cambridge Dictionary [10], hate speech is "public speech that expresses hate or encourages violence towards a person or group based on something such as race, religion, sex, or sexual orientation". Although one may think it is thing of the past, antisemitism, Islamophobia, racism against black and Asian people, discrimination against women is happening now in our world.

One of the main reasons of the increase of hate speech is how social media works, recommendation systems of social media like Facebook or YouTube, will reinforce those publications that generate more impressions, this could be the number of likes, visualizations, comments or times it is shared. The ultimate goal of this social media is to keep users in the platform as long as possible, because the main source of income is advertisement, so as the duration of the session increases, more ads are shown to users and therefore revenue increases. Users will extend their sessions as long as recommendations shown are interested for them.

This model seems inoffensive, but it is partly to blame for some of the phenomena that have taken place in recent years, especially on the political aspect. The reinforce of content that causes the greater impact causes that if the user interacts with a content related to a certain topic, the recommendation system will show more content related to that topic, and as more interest is shown about a topic, the recommendations tend to be more extreme. For example, if a person shows interest in animals and environment, recommendations will tend to show content about veganism or searching about jogging will lead to recommendations about running ultramarathons [48]. This is an example to illustrate how it works. But what happens about political content, now society it is more polarized than in the past, as it is shown in figure 1.5, the trend in the past decades is that people from different political sides, are more ideologically divided than before.

Figure 1.6 shows the problem of social media reinforcing extreme content, it causes echo chambers to be created. Echo chamber in this area, refers to the idea of a group of users that share an ideology or a belief, and only content that reinforce their common identity is transmitted and the rest of perspectives are ignored. In the example, a hashtag is tracked over a two day period. It shows the interactions of users that use the hashtag where every

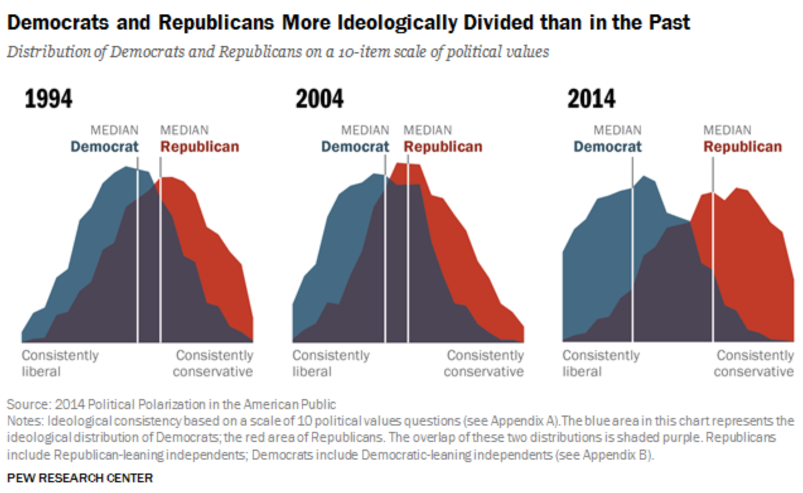


Figure 1.5: Ideologically distribution of republicans and democrats over three decades [56]

node is a user and every line is a communication between two users, and there is a clear separation between two groups, the liberal group on the right and the conservative group on the left. They are massive groups where interactions between users that share ideology is big, however there is no communication between groups only a few lines that can be considered negligible. This confirms that users are not debating their opinions, they are reinforcing their ideas.

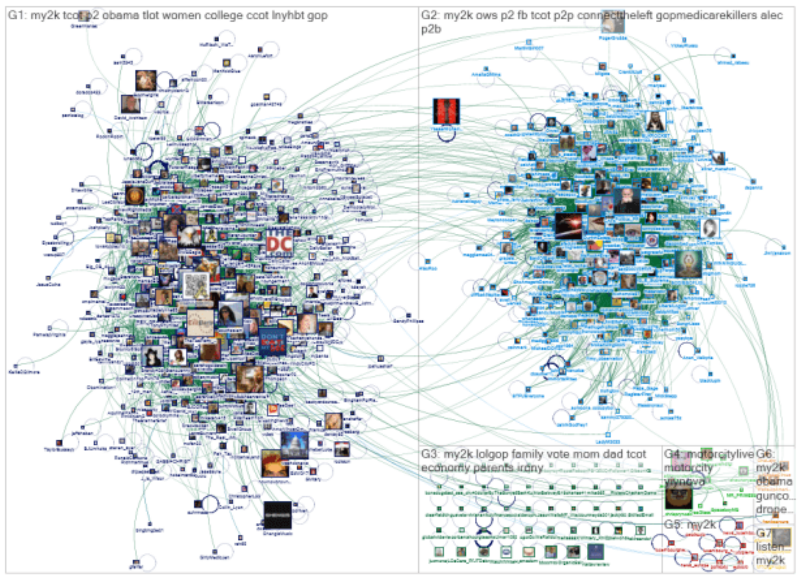
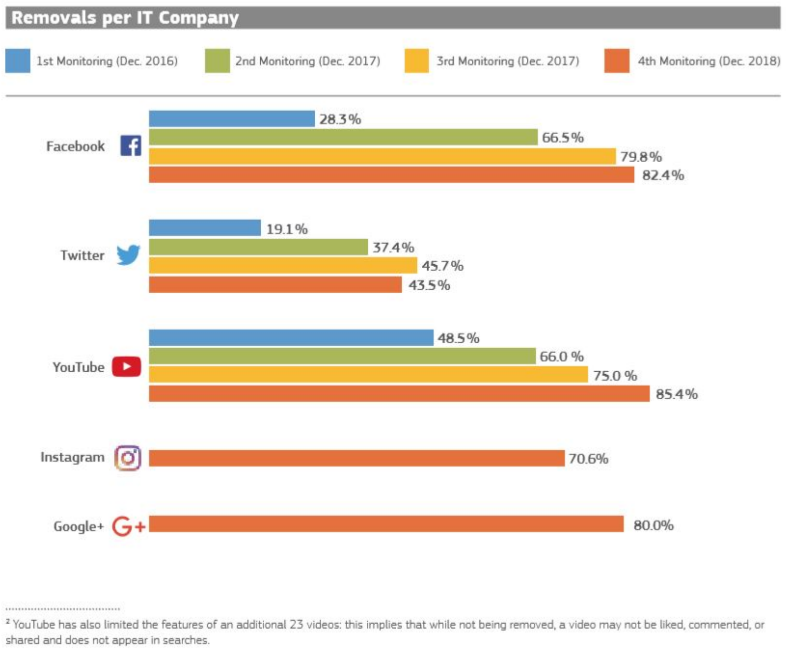

G1: my2k tcot p2 obama tlot women college ccot lnyhbt gop
G2: my2k ows p2 fb tcot p2p connecttheleft gopmedicarekillers alec p2b
G3: my2k lolgop family vote mom dad tcot economy parents irony
G4: motorcitylive motorcity wynova
G5: my2k
G6: my2k obama gunco drone
G7: listen my2k

Figure 1.6: Visualization of two groups using same hashtag over a two-day period in Jan 2013. [54]

This echo chambers have allowed spaces where hate speech is transmitted, the message is reinforced and a part of society has taken it for granted. These messages are not in isolated environments, they have made their way into the politics of the most developed countries and can provoke dangerous situations for some collectives. In United Kingdom, after Prime Minister Boris Johnson's statement referring to veiled Muslim women as "letterboxes", it provoked religious hatred and Islamophobic incidents increased by 375% [2]. In United States, President Donald Trump can be accused of spreading hate speech when referring to the Covid-19 as the "Chinese virus", therefore associating the virus with ethnicity. The consequences are an increase of anti-Asian hate incidents in the U.S. during the pandemic [33].

Companies and organizations are aware of this problem and a number of actions have been taken. In 2016, the European Commission together with IT companies Google, Facebook, Twitter and Microsoft introduced a code of conduct a set of commitments to fight the spread of hate speech online in Europe [6]. Since the announcement, the content removed from the social media platforms has vastly increased, in 2016 about 28% of illegal hate speech content was removed while in 2018 the rate increased to about 80%, as it can be seen in figure 1.7.



² YouTube has also limited the features of an additional 23 videos: this implies that while not being removed, a video may not be liked, commented, or shared and does not appear in searches.

Figure 1.7: Removals of hate speech content in main social media platforms [60]

1.2 Project goals

This project has two main goals that cover different analysis, from sentiment analysis to hate speech analysis.

- First objective is to adapt socialsent tool in order to develop the following objectives with it. Socialsent framework is written in Python 2 which is deprecated and some of the libraries used ceased development years ago. The challenge is to make the necessary changes so the code can run with current software.
- Second objective is to generate a sentiment lexicon using sentiment analysis techniques from the socialsent framework, a pre-trained word embedding model and a large dataset. The result is a lexicon where each word contains a polarity score indicating how positive or negative is. The framework allows to choose between state of the art sentiment analysis algorithms, so two lexicons are generated, with SentProp and densifier algorithms. For this purpose, the dataset chosen is the Imdb review dataset which contains reviews from the Internet movie Database separated as negative or positive reviews.
- Third objective is the evaluation of the lexicon in a real scenario. The dataset is split into train and test sets, so in this case the lexicon obtained will be tested to identify if the sentiment of a sentence, or in this case a review, is positive or negative. As the true result is known, metrics and statistics can be extracted to measure the performance.
- On the same basis as the second objective, the fourth objective is to generate a hate speech lexicon using sentiment analysis techniques from the socialsent framework. It is a similar problem, where it is used a different approach and the dataset employed is different. Again two different algorithms are used and two lexicons are generated where each word score reflects how much hate speech content the word has. For this goal, the dataset used is a collection tweets, annotated if they are offensive, hate speech or neither. The problem is repeated twice for hate speech tweets and for offensive tweets.
- The fifth and last objective is the same as third objective but in this case for hate speech analysis. The procedure followed is the same and statistics are obtained when evaluating the results of the hate speech lexicons on real tweets.

1.3 Structure of this document

The remaining of this document is structured as follows:

Chapter 2 introduces the theoretical concepts on which the technologies used along the project and it presents the state of the art in natural Language Processing, word embeddings and sentiment analysis and hate speech analysis.

Chapter 3 describes the architecture of the project, how it is structured and which modules compound it.

Chapter 4 describes the methodology followed to develop the lexicons of the different problems.

Chapter 5 presents the different evaluation methods and the results obtained.

Chapter 6 sums up the conclusions of the project.

CHAPTER 2

State of Art

This chapter gives a theoretical view of current technologies in the field of Natural Language Processing, sentiment analysis and hate speech analysis, focused on the systems developed in this project. Also, it will cover the theoretical framework of the different algorithms, programming languages, technologies and tools used in the project.

2.1 Machine Learning

Machine Learning is a discipline within the field of Artificial Intelligence (AI) [31] and is the enabling technology for most of the projects described in the current chapter. It is, likewise, the source from which the rest of the theoretical explanations in this chapter of state of the art are based.

Machine learning is defined according to the Cambridge dictionary as *the process of computers changing the way they carry out tasks by learning from new data, without a human being needing to give instructions in the form of a program* [14]. The learning phase begins with observations or data, such as examples, direct experience or instructions, then it looks for pattern in data to make better decisions in the future based on the data provided. The primary goal is to enable computers to learn a model automatically without the need or help of a human and adjust actions accordingly.

Within the discipline of ML there are multitude of different applications and technologies. All of them fall into one of the three categories that are shown in figure 2.1, supervised learning, unsupervised learning or reinforcement learning. This project tackles the use of both supervised and unsupervised learning.

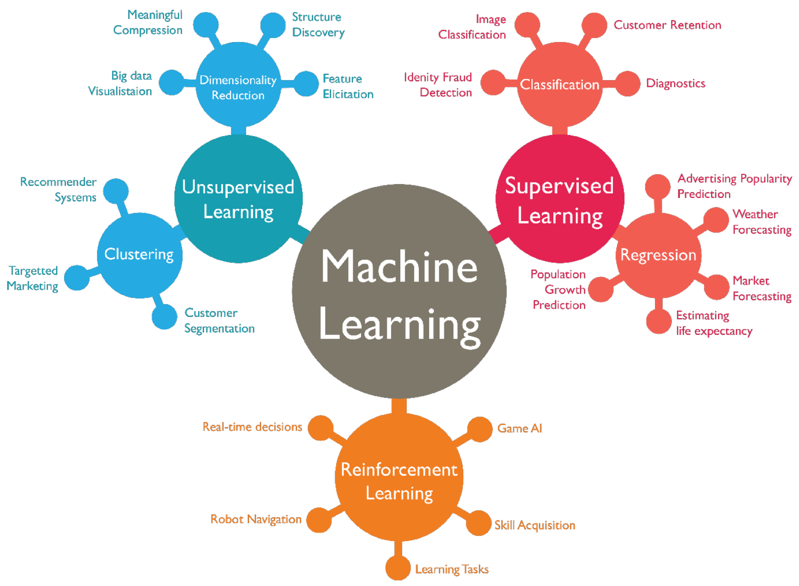


Figure 2.1: Machine Learning Categories [61]

2.1.1 Supervised Learning

Supervised Learning is the most common application. One of its most relevant characteristics is that in the training phase the input data is already labeled for a particular output.

This way the algorithm learns a function f that maps input values X to a output variable y, in short $y = f(X)$. The figure 2.2 shows a diagram flow of the process.

The objective is that when the mapping function f is trained, if a new input is shown to the system it returns the desired output.

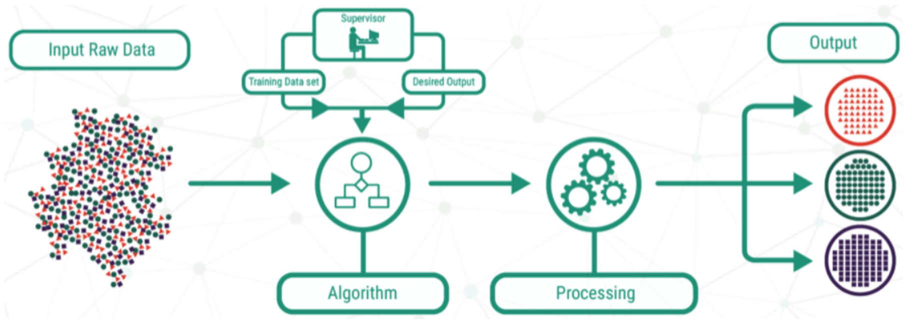


Figure 2.2: Supervised Learning diagram [62]

Supervised learning can be divided into two types of problems, regression and classification problems.

- Regression. A regression problem is when the output variable is a real value, for example 'price' or 'weight'.
- Classification. A classification problem is when the output is a categorical variable, for example 'red' or 'blue'.

2.1.1.1 Logistic Regression

Logistic regression is a simple classification problem despite the name may indicate a regression problem. The key is that it has the same fundamental as regression problems but instead of fitting a straight line to the data, it fits a sigmoid function, figure 2.3. The main characteristic of this function is that the maximum value in the y-axis is 1 and the minimum value is 0.

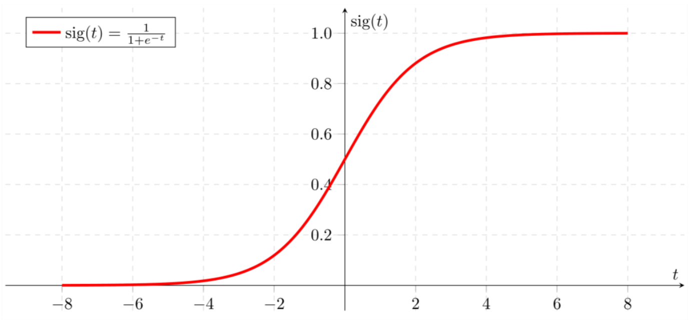


Figure 2.3: Sigmoid function [63]

This function allows to make a class discrimination, small values tend to go to '0' and big values will result in '1'. This way, applying a classification threshold, when the sigmoid function is applied to the input data, if the value is below the threshold it will be classified as class 0 and in the other case as class 1. Figure 2.4 shows a geometric separation as computed after training a logistic regression algorithm., where the points are the input data, the colour is the class they belong to, and the dotted line is the learned logistic regression threshold.

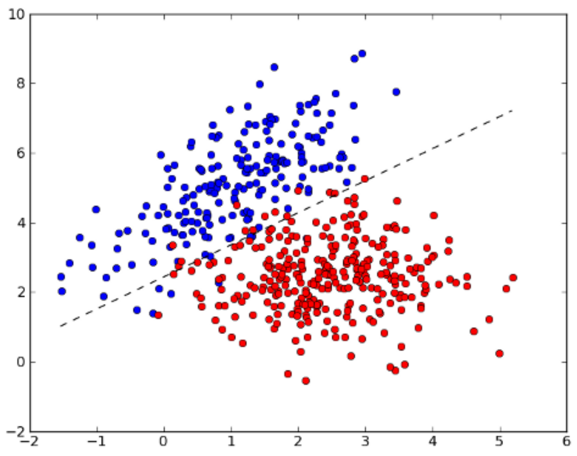


Figure 2.4: Logistic regression output example [23]

2.1.1.2 Evaluation

It can be noticed in figure 2.4 that most of the points of the same class are correctly discriminated by the threshold but nevertheless some data points are on the opposite side of the threshold and this would result in a incorrect classification. On the basis of this concept, each data classification falls into one of this terms:

- True Positive (TP) is when the model correctly predicts the input sample as the positive class.
- False Positive (FP) is when the model incorrectly predicts the input sample as the positive class.
- True Negative (TN) is when the model correctly predicts the input sample as the negative class.
- False Negative (FN) is when the model incorrectly predicts the input sample as the negative class.

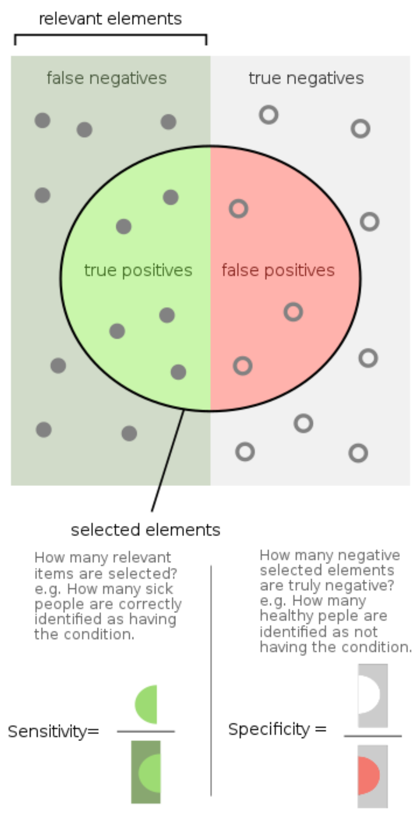

relevant elements
false negatives
true negatives
true positives
false positives
selected elements
How many relevant items are selected? e.g. How many sick people are correctly identified as having the condition.
How many negative selected elements are truly negative? e.g. How many healthy peple are identified as not having the condition.
Sensitivity=
Specificity =

Figure 2.5: Sensitivity and specificity [32]

Having the realization of a set of samples categorized as the four possible outcomes, TP, FP, TN, FN, now it can be applied a metric that measure the performance of the classifier.

To measure the performance of a classification problem, several statistics can be extracted, each one representing some interesting aspect of the classifier. The role of sensitivity and specificity can be visually appreciate in the figure 2.5.

- Accuracy. It is the most intuitive statistic, it tells the ratio of correctly predicted label.

$$Accuracy = \frac{TP + TN}{TP + FP + FN + TN} \tag{2.1}$$

- Precision. How much were correctly classified as positive out of all positives.

$$Precision = \frac{TP}{TP + FP} \tag{2.2}$$

- Recall or sensitivity. Both statistics refers to the True Positive rate. It measures the ratio between how much were correctly identified as positive to how much were actually positive.

$$Recall = Sensitivity = \frac{TP}{TP + FN} \tag{2.3}$$

- Specificity . Can also be refered as True negative rate. It measures the ratio between how much were correctly identified as negative to how much were actually negative.

$$Specificity = \frac{TN}{TN + FP} \tag{2.4}$$

2.1.2 Unsupervised learning

Some problems requiring a machine learning solution consists in a set of data that does not have a corresponding target label. In this situation, unsupervised learning is a technique that does not require the supervision of the user and it is left on its own to discover patterns and information that was previously undetected. The disadvantage of this applications, it is difficult to evaluate their performance by any metrics because it is unlabelled data, so it is important how it is applied and interpret the results accordingly to each application.

Data obtained across Internet or other sources are primarily unlabelled data, because labelled data needs human intervention. Unsupervised learning is then needed to obtain information when it comes to a large data set where human intervention is unmanageable or the information is treated in real time. The main applications of unsupervised learning are clustering and association.

2.1.2.1 Clustering

Clustering is the most common of unsupervised learning, it consist in finding common patterns or structure in data and find natural groups.

Figure 2.6 is an example of clustering, where data is separated into three clusters, red, blue and yellow. Elements that are classified into the same group are as similar as possible, and the least similar with data of other groups.

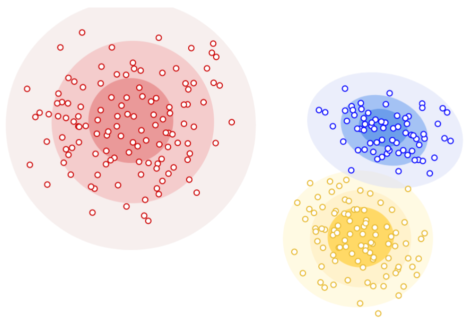


Figure 2.6: Clustering example [4]

There are multiple types and algorithms to clusterize data:

- Hierarchical clustering. In this method a hierarchy of clusters is build. The most common type is agglomerative hierarchical clustering, each data point is a cluster on its own, then pairs of clusters that are more similar are merged into a higher hierarchy, and it is done iteratively until there is only one cluster left. This hierarchical relationships can be visually appreciated by a dendogram.

- K-means clustering. This algorithm cluster data into k groups or clusters. A large k will create smaller clusters with more granularity while a small k will create large clusters with less granularity. The objective is to minimize the distance between each data point and its respective cluster centroid.

2.1.2.2 Association

Association algorithms make it possible to discover associations between a large data set. For example, e-commerces like Amazon, can discover associations between products by analyzing the purchase history of customers. So if an association is found like if people that buy the object X also buy the object Y, then when a user purchase an item, the recommendations will show items that are usually bought together.

2.1.2.3 Dimensionality Reduction

Dimensionality reduction is a technique to transform high dimensional data to lower dimension while maintaining meaningful properties of the data. The advantage to reduce dimensionality is an increase in algorithms performance due to it needs less computational power to train a model It also can be beneficial to remove the noise of data and keep only relevant information. Two of the most used techniques to reduce dimensionality are:

- Singular Value Decomposition is based on the property that a matrix A can be decomposes into:

  $$A = USV^T$$

  The matrix diagonal of S contains what is called the singular values, and the value of each value corresponds to the variance explained. The reduction is achieved removing the singular values that does not contribute to the explained variance.

- Principal Component Analysis. It is considered the main technique for this purpose. It basically projects a high dimensional data into a lower dimensional space. To calculate that lower space, PCA calculates the linear combination of the original data so the variance along the new axis is maximum.

2.2 Neural networks

Neural networks or artificial neural networks are computing systems inspired by the biological neural networks that forms animal brains.

Neural networks are formed by a set of nodes, also called neurons which model biological neurons. Like in the human brain, neurons can transmit a signal to another neuron by the synapse. The synapse is a junction between a neuron's axon which act as transmission line, and the dendrites of the next neuron. The artificial neuron receives an input or a set of inputs, real numbers, and the output is a non-linear function of the weighted inputs, and that will as the input of another neuron. Figure 2.7 shows the components of a biological neuron and the corresponding model of an artificial neuron. And the mathematical equation of the output of the neuron is equation 2.5, where y is the output, x_i and w_i is the input and the weight of the neuron i respectively, and B is the bias.

$$y = f(\sum_{i=0}^{n} w_i * x_i) + B \tag{2.5}$$

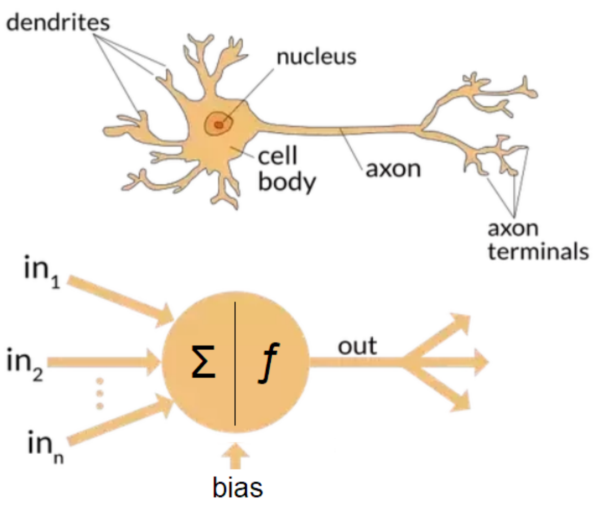


Figure 2.7: A biological and an artificial neuron [55]

Neural networks are a discipline within machine learning, and therefore is a type of Artificial Intelligence. The main difference with machine learning is how algorithms learn and how data is presented to the system. Machine Learning requires always structured data and it learns to fit a function based on the algorithm proposed, there are many of them, logistic regression, SVM, random tree, etc., however neural networks does not rely on an proposed function, it relies on a series of layers of neurons interconnected with each other and it does not need human intervention, raw data can be the input of the network and it will learn on its own a the data representation in a hierarchical structure, extracting complex abstractions from data. For example, if the input of a neural network is a picture, it will extract in the first hierarchy some basic features like straight lines, after that it will extract second order features like curves or some shapes and the complexity advances through hierarchies.

The structure of a neural network can be seen in figure 2.8, it is composed by layers of three types:

- Input layers. The function of this layer is to introduce data to the system. It can be any type of data, images, audio, text, etc. The nodes does not have inputs, they only forward data to the next layer.
- Hidden layers. This layers has both input and output. The number of hidden layers can be modified, when there are more than one hidden layer it is considered deep learning. They receive an input, calculate the transformation of the input, and send it to the next layer. They are called hidden because they are transparent to the user, from outside it is only seen the input and the output of the system.
- Output layer. It is the last layer, it only has inputs and generates the output of the network.

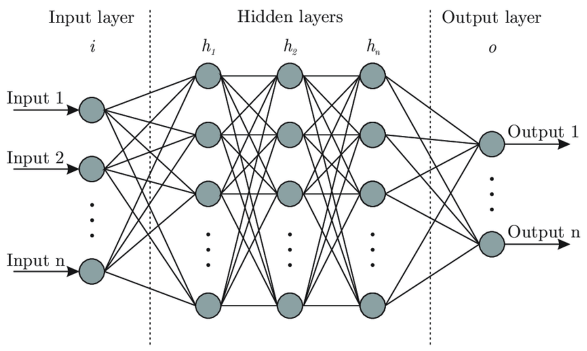


Figure 2.8: Artificial neural network architecture [37]

The network shown in the figure 2.8 is a multi layer perceptron network, but there are other types, these are the main ones:

- Multi layer perceptron network. It is the simplest neural network, every node is connected to all nodes in the next layer, information travels in one direction, from the input to the output passing from all the middle layers.
- Convolutional neural network. The main characteristic of convolutional neural networks is the input is a three dimensional array of neurons, so it is able to capture

spatial and temporal dependencies. They are useful for computer vision or image processing.

- Recurrent neural networks. This class of networks uses the output as input, so it has information of previous record to train the network.

2.3 Natural Language Processing

Natural Language Processing (NLP) is a field within Artificial Intelligence that allows computers the ability to understand and process human's language. It is not an easy task to teach a machine to interpret the language they way humans do. In the end, our way of expressing differs from one person to another, it depends on the local context of everyone, like the geographical location, age even gender, language is alive and it evolves as the society do, and in addition to that, some resources like the tone, the selection of words adds some type of information, like for example the irony, that have to be collected by the machine in order to truly understand human language.

Also, another problem is that the huge quantity of data that is generated daily, via newspapers, Twitter, conversations, etc is unstructured data. This data can not be separated into its features and added to a data set with the classical structure with a row for every entry and a column for each feature, so this makes the problem unmanageable and hard to manipulate. Thanks to successive advances in this field since the second half of the 20th century, solutions are coming and a large number of technologies and applications have emerged that helps to deal with this kind of unstructured data.

The increase in the available computational power and the accessibility to large datasets has let NLP applications to be used in a variey of fields,for example some relevant use cases:

- Applications of language translation, like Google Translate or DeepL.
- Filter to detect which mails are spam or not by analyzing the text of the mail, used in mail services like Yahoo and Google ones.
- Voice driven interfaces like Amazon's Alexa or Siri's Apple use NLP to identify the objective of the vocal prompt.
- Sentiment analysis has numerous applications, like identify the opinion in social media about a product.
- Predictive text and auto-correction used on smartphone keyboards.

As it has been mentioned, NLP it is a difficult task and data is not treated as it is generated. There are some techniques that help to manipulate data before passing it to a model. Some of the most common techniques are:

Tokenization Tokenization is the technique to separate an input text into small pieces called tokens. Tokens are usually words, but it can also characters, sub-words or even n-grams, which are contiguous sequences of n items in a sentence, and it usually removes some characters like punctuation. For example, the sentence:

"*Everything is possible*"

it is converted to

["*Everything*", "*is*", "*possible*"]

Following with the common case that every token is a word, the delimiter to get each word is the blank space. But this issue is more complex than it may seem, if the delimiter is the white space then compound words are non-sense if they are separated, for example roller coaster, hot dog or ice cream. Another issue is that removing punctuation can bring some complications, for example in English it is common to see the apostrophe for possession or contractions, like how the word $aren't$ should be treated. These questions are language-specific are there no better solution.

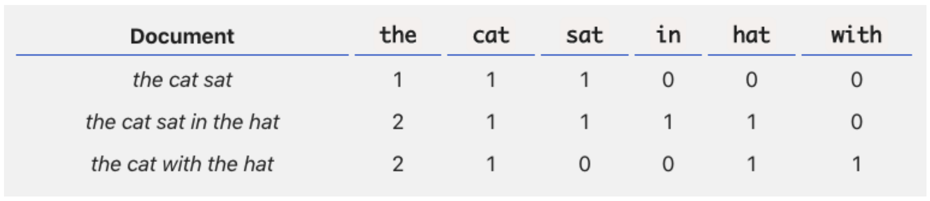
Stop words removal Stop words removal is technique to remove words like articles, pronouns or prepositions, such as 'the', 'an', 'a', because they appear with a lot of frequency in a text and they do not add relevant information to the NLP objective.

The list of stop words is not fixed, and different applications can use some stop words or other projects can choose to not ignore any word. For example, in sentiment analysis removing the word 'not' can change the overall sentiment of the sentence.

Bag of words Bag of words is a commonly used model that describes the occurrence of words within a document. It creates a occurrence matrix that for each document note down which words have appeared, without taking into account any information about the order or structure of words, the figure 2.9 shows an example of Bag of Words. The concept is that if documents have similar words occurrence they may have similar content.

With the occurrence matrix it is known the vocabulary words and a measure of the presence of them for every document. Now this structure can be used for training a model having as input every document and the features are the vocabulary words.

This model is the most basic one but it can be added some complexity to punctuate the presence of a word, taking some considerations to improve the performance of the model. For example, the technique “Term Frequency — Inverse Document Frequency” (TFIDF) weights the punctuation of a word depending in how frequent is the word across all documents. So words like ’the’ that is frequent will get punished and unique or rare words will be rewarded.



Document	the	cat	sat	in	hat	with
the cat sat	1	1	1	0	0	0
the cat sat in the hat	2	1	1	1	1	0
the cat with the hat	2	1	0	0	1	1

Figure 2.9: Bag of Words example [67]

2.4 Word embedding

A word embedding is the learned representation of a word where words that represent a similar context have similar representations and different words have a separated representation. Most word vector methods rely on the distance, cosine distance, or angle between pairs of words to determine the similarity between them. For example, if one were to ask an embedding model which are the closest words that to ‘Spain’, it should be obtained words like ’France’, ’Italy’, ’Portugal’ or similar.

Word embeddings are a class of techniques where individual words are represented as real-valued vectors in a predefined vector space, usually of tens or hundreds of dimensions. The benefits of this representations are key if a large corpus of text is trained because it will encode efficiently every word, in contrast with the hundreds or millions dimensions a one-hot encoding will require, with one dimension per word [38].

In 2013, Word2Vec [52] was developed by Mikolov at Google and it is one of the most popular choices to use in a NLP application, mainly because it is computationally efficient and has public loaded pre-trained models everybody can download. Word2Vec is a family of model architectures and two methods are proposed to approach the embedding learning:

- Continuous Bag-of-Words Model. The model consists on the prediction of the word given the context. The context is the words that surround the target word, a number of words before and after it. The target word is also called middle word, and in this model the order of words is not important, that’s wy it is called bag of words.

- Continuous Skip-gram Model. The model predicts a number of words given a target word. It predicts words before and after and the number depends on the window size.

Also Mikolov [53] introduced a new evaluation scheme that proved vectors learned captured the syntactic and semantic regularities in language and that each relationship is characterized by a relation-specific vector offset. For example, for the male/female relationship, the analogy "king is to queen as man is to woman" should be encoded in the vector space by the vector equation king − queen = man − woman. This proposed evaluation introduces a new paradigm where models are tested to produce meaning in various dimensions, thus capturing the idea of multiple grouping of the distributed representations.

2.4.1 GloVe Embedding

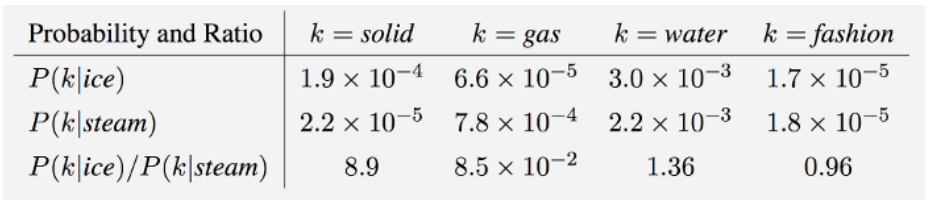
The statistics of word occurrences in a corpus is the primary source of information of unsupervised methods for learning word representations, the question still remains as to how meaning is generated from these statistics, and how the resulting word vectors might represent that meaning. Under this idea GloVe is created.
GloVe are the initials of Global Vectors, a word representation model developed by Pennington [57] at Stanford University, which its objective is to create a model that is able to leverage statistical information of the global corpus and induces semantic and syntactic properties of words. For this purpose is based on:

- Matrix Factorization Methods. This method is the base of LSA [41], Latent Semantic Analysis, one of the pioneers algorithm to generate low-dimensional word representations. These methods utilize low-rank approximations to decompose large matrices that capture statistical information about a corpus.

- Shallow Window-Based Methods. Included in this methods is the Word2Vec model, where word representations are learned using local context windows. The disadvantage of this methods is that they do not operate on the global statistics of the corpus, they rather scan context windows and have demonstrated the capacity to learn linguistic patterns.

The GloVe model is based on the information of the matrix of the word-word co-ocurrence counts, that is denoted by X, and X$_{ij}$ contain the number of times the word j occurs in the context of the word i. Let $X_i = \Sigma_k X_{ik}$ be the number of times any word appears in the context of word i and $P_{ij} = P(j \mid i) = X_{ij}/X_i$ the probability that word j appear in the context of word i.

Let´s illustrate with an example how some aspects of meaning can be induced from the co-occurrence probabilities. The figure 2.10 shows the probabilities of two terms related



Probability and Ratio	$k = solid$	$k = gas$	$k = water$	$k = fashion$
$P(k\|ice)$	1.9×10^{-4}	6.6×10^{-5}	3.0×10^{-3}	1.7×10^{-5}
$P(k\|steam)$	2.2×10^{-5}	7.8×10^{-4}	2.2×10^{-3}	1.8×10^{-5}
$P(k\|ice)/P(k\|steam)$	8.9	8.5×10^{-2}	1.36	0.96

Figure 2.10: Co-ocurrence probabilities and ratio for words ice and steam with some context words [8]

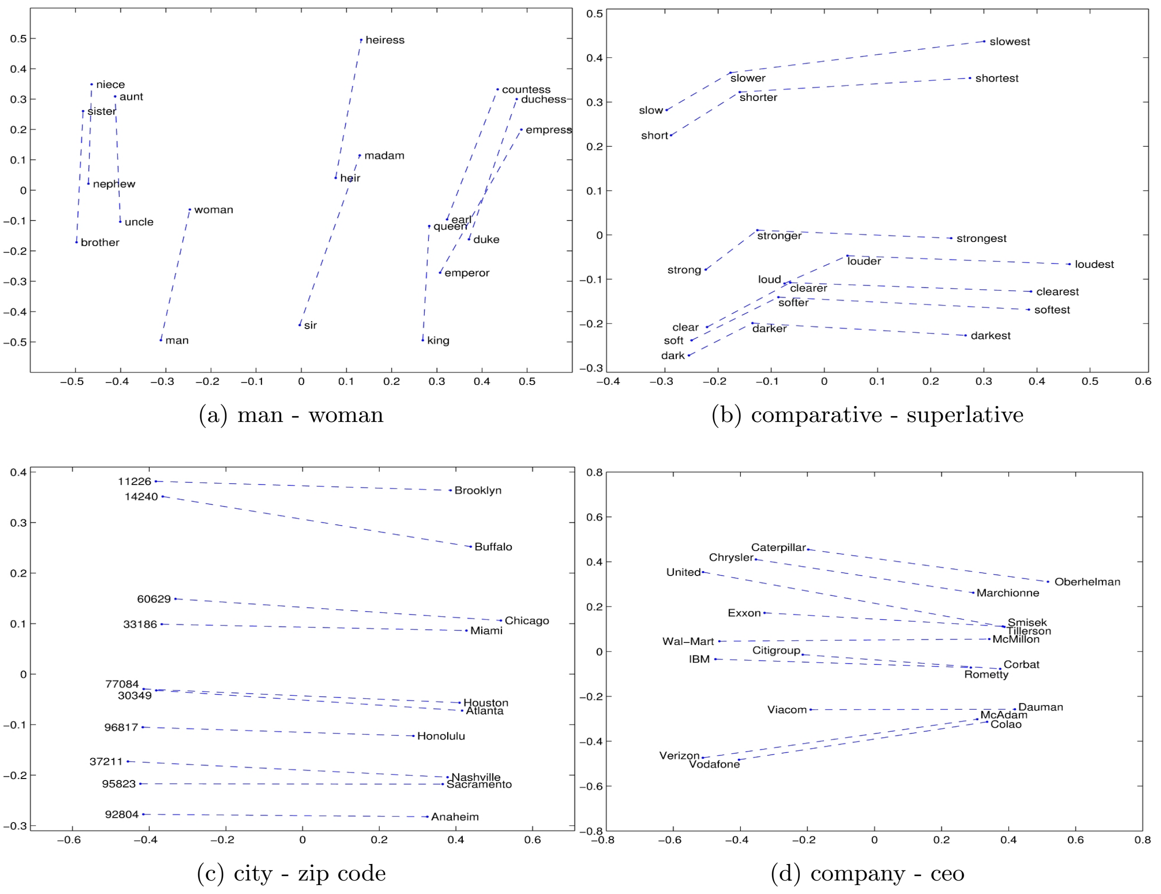
to thermodynamics, ice and steam, with four target words, solid, gas, water and fashion. In addition, the last row shows the ratio of their co-occurrence probabilities, which is the statistics that reveals the relationship between these words. For words k related to ice but not steam, say $k =$ solid, the expected ratio P_{ik}/P_{jk} will be large. And for the other case, words k related to steam but not ice, say $k =$ gas, the ratio should be small. For words k like water or fashion, that are either related to both ice and steam, or to neither, the ratio should be close to one. Comparing this with the raw probabilities, this feature reveals which are relevant words(solid and gas) and which are irrelevant words(water and fashion). This proves that the ratio of probabilities encodes a kind of meaning of, in this case, the thermodynamic phase.

The proposal of GloVe is a log-bilinear model with a weighted least-squares with the objective to learn word vectors such that their dot product equals the logarithm of the words' probability of co-occurrence. Knowing that the logarithm of a ratio equals the difference of logarithms, this objective associates the logarithm of ratios of co-occurrence probabilities with vector differences in the word vector space. BAs it have said, the ratios encode some form of meaning, this information gets encoded as vector differences as well. For this reason, the resulting word vectors perform very well on word analogy tasks.

$$J = \sum_{i,j=1}^{V} f(X_{ij})(w_i^T \tilde{w} + b_i + \tilde{b}_j - logX_{ij})^2 \tag{2.6}$$

The equation 2.6 is the proposed model where V is the size of the vocabulary, $w \in R^d$ are word vectors, $\tilde{w} \in R^d$ are separate context word vectors, b_i and $\tilde{b}_j$ are added bias for symmetry restoring purposes and f is the weighting function that address the problem of rare co-occurrences adding noise or irrelevant information, so this cases are weighted with a small value and normal co-occurrences are weighted by 1.

The intrinsic relationship between words can not be given by a number, or at least if it is compared across various dimensions of meaning, in that case it is necessary a vector to illustrate that the difference of word vectors associates some semantic or patterns languages. In the figure 2.11 it is shown that the concept of gender is acquired by the word vector because the vectors of the difference between man - woman, are similar to the vectors of king - queen, sir - madam or sister - brother. And it is the same for other analogies like the comparative/ superlative form, the ownership of a company or the zip code.



(a) man - woman

(b) comparative - superlative

(c) city - zip code

(d) company - ceo

Figure 2.11: Visualization of word vector differences for different contexts [8]

2.5 SentProp

SentProp is the framework contained in the application SocialSent [45] developed by Hamilton at Stanford University, a collection of code and datasets for performing domain-specific sentiment analysis. In particular it is developed SentProp, the algorithm for inducing domain-specific sentiment lexicons from unlabeled text.

The algorithm includes the building of a high quality word vector representation of 300 dimensions, based on the Singular Value Decomposition of the co-occurrence matrix, but this is an optional step because it can be used any embedding, such as GloVe. Then, given a set of word embeddings, a weighted lexical graph is created where each word represents a node in the graph and its edges are connected to the k nearest neighbors according to the cosine distance, as we can see in equation 2.7.

$$E_{i,j} = arccos(-\frac{w_i^T w_j}{\|w_i\| \|w_j\|}) \tag{2.7}$$

2.5.1 Label Propagation

Once the lexical graph is created, the next step is to propagate sentiment labels over the graph using a random walk method [66] from a defined seed set. The seeds are a small set of manually selected words, around 8 or 10 words, that are the best representatives of the class it is about to learn. For example, for the positive sentiment the seeds could be good, excellent, delightful, etc and for the negative sentiment the seeds could be bad, horrible, hate, etc.
With the random walk method, a word's polarity score for a seed set is proportional to the probability of a random walk from the seed set hitting that word, as it can be seen in figure 2.12.

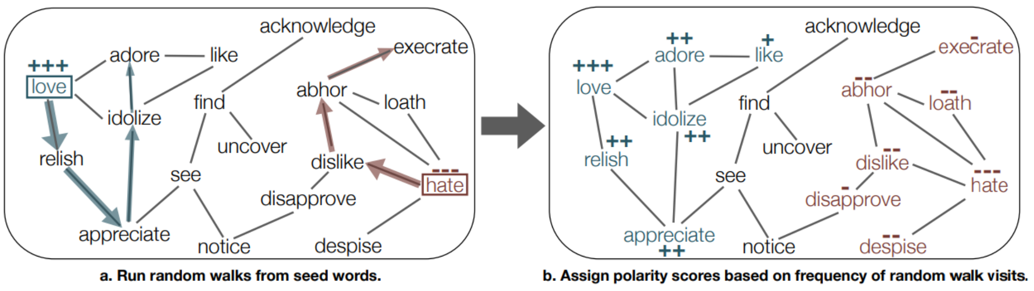


Figure 2.12: Visual summary of the SentProp algorithm. [45]

The equation 2.8 calculates $p \in R^V$ iteratively, a word-sentiment score for every word in the corpus, by means of T that is calculated after constructing a symmetric transition matrix from 2.7 by computing $T = D^{\frac{1}{2}} E D^{\frac{1}{2}}$. s is a vector with values set to $\frac{1}{S}$ in the corresponding values of the words of the set S and with zeros in all other entries. The β term regulates the similarity of neighbor labels for higher β values, or ensure that correct labels are on seed words for lower β values..

the corresponding values of the set S and with zeros in all other entries. The β term regulates the similarity of neigbor labels for higher β values, or ensure that correct labels are on seed words for lower β values.

$$p^{(t+1)} = \beta T p^{(t)} + (1 - \beta)s \tag{2.8}$$

Finally, to obtain the polarity score for a word w_i, the random walk is computed iteratively for both seed sets and it is obtained a positive score $P^P(w_i)$ and a negative score $P^N(w_i)$. Then combine these values into a positive-polarity score as $P\text{-}^P = \frac{P^P(w_i)}{P^P(w_i)+P^N(w_i)}$

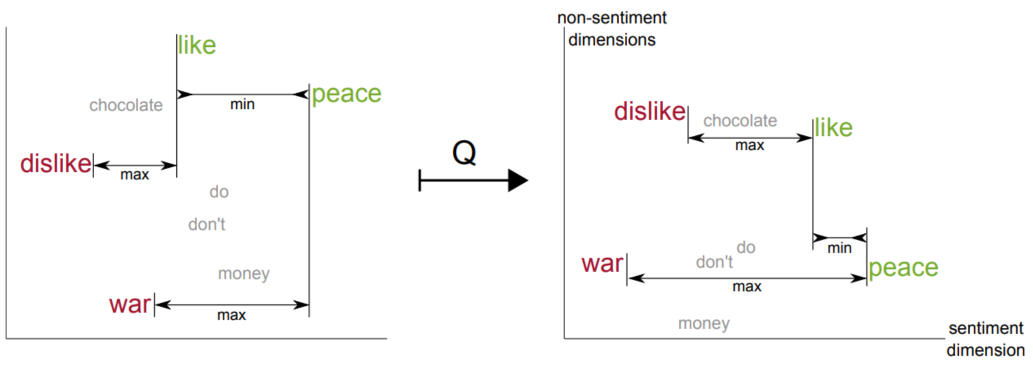


Figure 2.13: Original word embedding space(left) and transformed embedding space(right). The function Q minimize distance between words of the same group and maximize distance between words of different groups [59]

2.6 Densifier

Densifier [59] is a word embedding proposal with the main characteristic that the embedding is compressed in an ultradense vector that contains all the relevant information.
The objective of Densifier is to find the ultradense subspace E_u that contains all the relevant information for any given task. The method learns an orthogonal transformation of the original space E_o, reducing the dimensions by a factor of 100, making the training phase of the embeddings faster and computationally cheaper. The orthogonal transformation moves non-task related information outisde of the orthogonal complement, so the quality of the representations increase because noise or non-relevant information is reduced in E_u.
In the training phase, the objective is to find the orthogonal matrix $Q \in R^{dxd}$ that transforms the original word embedding space into a space in which the information of interest is represented by a small number of dimensions.
To satisfy that the transformation of space it is getting the necessary information, two optimization problems are solved:

- Separating words of different groups. It is maximized the distance between words that have contrary label. Let a word w be labeled as $l(w)$ with +1 or -1, the objective is to maximize equation 2.9 where e_w is the original embedding and P is a identity matrix for the dimensions of the ultradense subspace.

$$\arg\max_{Q} \sum_{v,w} \|P * Q(e_w - e_v)\| \tag{2.9}$$

subject to Q being a orthogonal matrix and for the set of word pairs (v, w), $l(w) \neq l(v)$ holds.

- Allign words of the same group. In contrast with the previous objective, now it is seek to obtain that the distance between the set of word pairs (w, v) that comply with $l(w) = l(v)$ it is minimized as equation 2.10 shows.

  $$\arg\min_{Q} \sum_{v,w} \|P * Q(e_w - e_v)\| \tag{2.10}$$

  subject to Q being a orthogonal matrix.

The objectives followed by equation 2.9 and 2.10 can be visually represented in figure 2.13. In the case of creating a lexicon, Densifier is trained to produce a one-dimensional ultradense space, where the return is that each word in the data set has an associated real value indicating the relationship strength with the type of information of interest.

2.7 Sentiment Analysis

Sentiment analysis is a discipline within Natural Language Processing which identifies the sentiment of text, normally detecting if it is positive, negative or neutral. The positive or negative emotion is a subjective perception and it is not based on objective arguments. Therefore different strategies and methods use different approaches to address the problem of identifying sentiment in a text. There are two main ways:

- Subjectivity/Objectivity Identification. The identification of subjectivity / objectivity consists of classifying a sentence or a piece of text into subjective or objective. However, it must be borne in mind that this type of analysis poses difficulties. The main challenge is that the context of a word or phrase affects to the meaning of itself.

- Feature/Aspect-Based Identification. Identification by features/aspects allows determining different opinions or feelings in relation to different aspects of a text. Unlike the identification of subjectivity/objectivity, feature/aspect based identification provides a more detailed view of opinions and feelings.

There are numerous of studies implementing a solution to detect sentiment over a text, the amount of data all over Internet is large, the problem is that an implementation needs to know the performance of the algorithm, and that can only be done with labelled data. Fortunately, there are several data sets with labels [24]:

- Imdb dataset. It is the one used in this project and it contains 50000 movie reviews from the Internet Movie Database classified as positive or negative.

- Yelp Review dataset. It contains more than 500000 Yelp reviews. There are two versions, a binary classification, positive or negative, and a fine-grained of five classes classification, where 0 is very negative, 5 is very positive and between are all the intermediate values.

- Subjectivity dataset. It contains 10000 sentences labelled as subjective or objective.

- Stanford Sentiment Treebank. It contains more than 200000 phrases of movie reviews with binary classification or fine-grained classification.

- Sentihood. It contains more than 5000 sentences and this dataset has an additional perspective. It is an aspect-based sentiment, so in addition to the sentiment, each sentence indicates the aspect target or multiple aspect targets.

- SemEval. It is a popular semantic evaluation analysis workshop done every year. The first editions analyzed emotions on tweets, or to specific topics, every year new tasks are explored, for example in the SemEval-2019, the 13th workshop edition the tasks were among others [25]:

  - EmoContext: Contextual Emotion Detection in Text.
  - Hyperpartisan News Detection
  - HatEval: Multilingual Detection of Hate Speech Against Immigrants and Women in Twitter.
  - OffensEval: Identifying and Categorizing Offensive Language in Social Media.
  - RumourEval 2019: Determining Rumour Veracity and Support for Rumours.
  - Suggestion Mining from Online Reviews and Forums.

2.8 Hate speech analysis

It is of great importance to detect hate speech on social media platforms and other online sites because it can cause serious implications on real life of the people or collectives affected. The main motivation to identify correctly hate speech content is to stop the spreading of hate speech, powered by recommendation systems of social media platforms that favor the polarization of users.

Hate speech is spread in sites like Reddit, the biggest forum on the Internet, this platform is a collection of subforums where each one has a different theme. There are all kinds of issues, and some of them are particularly notable for their hate speech content. Some of them like the subreddit rIncels belonged to the manosphere, a section of the Internet where masculinity is promoted and are opposite to movements like feminism, and are misogynists. Reddit took action and banned nearly 7000 subreddits in a campaign to remove hate speech content from its platform, among them it was banned the subreddit rthe_Donald, the pro-Trump community, and the result has been a reduction of 18 percent in hateful content [65]. A remarkable example was Tay [12], a Microsoft bot which intention was to learn to interact with humans at Twitter, the result was that it learned to be racist, sexist and neo-nazi only in one day and Microsoft had to cancel the project inmediately. That confirmed the amount of hate speech there is in the platform despite the fact that Twitter terms do not tolerate hate speech content. This has generated controversy, when Twitter suspended the account of former U.S. president Donald Trump [20] and the Spanish political party Vox [42] claiming they broke Twitter's conditions by inciting hatred.

The task has to deal with several issues. First, the identification of hate speech is subjective. It can be present in various forms and the targets are numerous. It is difficult to establish a common criteria to define what is hate speech or what can be considered as just offensive speech, and when there is no hate speech, so each data set can differ because of the lack of a common agreement. Another issue is how some regulations can limit the actions to remove hate speech. In the U.S. free speech is a human right recognized by the U.S. constitution and it limit the actions taken to eliminate hate speech on social media platforms.

Research in this field is extensive, essentially classification of text problems to identify if there is hate speech or not. There are several approaches, with binary classification, fine-grained classification with 5 classes, or even some works where it detects the hate speech and it can extract the aspect of the text, like racism or sexism. They rely on sentiment analysis technologies, but changing the objective and the training parameters so the target is to identify hate speech instead of positive or negative sentiment.

The data sets mainly used in the literature are [50]:

- Davidson dataset. It contains 24802 tweets, classified as hate speech 5.77% of them, as offensive 77.43% and neither the rest. The tweets' label has been annotated thanks to contributions by collaborators.

- Burnap dataset. It contains data form various sources classified as sexual orientation, race, disability and religion.

- Waseem dataset. It contains 16000 tweets classified as racist, sexist and neither. The data is biased as the proportions are 1972k, 3383k and 11559k tweets for each category respectively.

- Founta dataset. It contains 80000 tweets annotated with 5 different classes, abusive, hateful, speech, spam and normal.

- Warner dataset. It collects data from Yahoo News Group and the American Jewish Society. It contains 9000 entries, classified as anti-semitic, anti-black, anti-Asian, anti-woman, anti-Muslim, anti-immigrant and other hate(anti-gay and antiwhite). This data is not public.

- Qian dataset. It collects conversations from several subreddits, a total of 22324 comments made on 5020 conversations. The conversations are binary classified as hate speech or not hate speech.

2.9 Enabling technologies

This project uses open-source, free to use technologies for the development of the programs. This section will introduce the main technologies used.

2.9.1 Python

Python [29] has become the de facto programming language for machine learning. Python is an interpreted high-level general-purpose programming language. It was created with the philosophy of a code with high readability. It uses significant indentation and it is object-oriented.

Part of the success of Python is due to the capacity of import modules and packages from third parties. This has led to reference packages used in most of machine learning projects, and some of them are going to be introduced in this chapter.

Python 2.0 was released in 2000 and introduced important new features, then Python 3.0 was released in 2008 and was a major upgrade and it is not compatible with previous versions, so Python 2 codes can not be run on Python 3. In this project it is used both Python 2 and Python 3 versions.

The main disadvantage of the programming language is the emphasis on readability and usability makes it not as fast and efficient as others languages like C#.

2.9.2 Conda

Conda [5] is an open source environment and package management system that was created for Python programs but it can works on R, Node.js, Java and other application stacks. It can create, save, load and switch between different environments, so if a different version of a package is needed, conda is an environment manager for this purpose, it can can install and update packages and their dependencies. It is also multi-platform so it is available on Windows, macOs and Linux.

2.9.3 Machine Learning Technologies

2.9.3.1 Jupyter notebook

Jupyter [21] is a free open-source, interactive web application to create, edit and share documents than contain live code, equations, images, data visualization and markdown text. It is widely used for data science because of its readability and serves for educational purposes as it can include many ways of information to illustrate the code. It can support over 40 programming languages like Python, R, Scala, Julia, etc.

2.9.3.2 Numpy

Numpy [17] stands for Numerical Python. It is the reference library to work with arrays in Python and it is the de-facto standard. It offers comprehensive mathematical functions, matrices, random number generators, linear algebra routines, Fourier transforms, etc.

This library overcomes the shortcomings of python by bringing the computational power or languages like C or Fortran.

It is the fundamental basis of important libraries related to data science or other numerical innovative libraries, like for example Dask, CuPy, Sparse, PyTorch, TensorLy, scikit-learn and scipy among others.

2.9.3.3 Pandas

Pandas [19] is a widely used library of Python for creating fast, flexible, and expressive data structures. It analyzes and manipulates data and it is based on two structures, series and dataframes. The main functions of pandas are:

- Identify easily missing data, represented ad NaN.
- Merge and join data sets intuitively.
- Split, apply and combine operations on data sets for aggregate or transform data.
- Align data automatically, data can be labeled else pandas can automatically align it.
- Manage multiple labels in a hierarchical way.
- Intuitive insert and delete of columns.
- Reshape and pivot data sets.
- Specific functions for time-series like data range generation, moving windows statistics, etc.

2.9.3.4 Scikit-learn

Scikit-learn is a powerful library built on numpy, scipy and matplotlib for data science purposes. It offers a whole suite of tools for each phase of a machine learning project.

The main module of scikit-learn are estimators. It provides many built-in machine learning algorithms and models, they are called estimators and can be fitted to some data. It provides estimators for both supervised and unsupervised learning like linear models, support vector machines, nearest neighbors, decision trees, naive bayes, clustering and density estimation among others.

For the earliest phases of the project, it offers pre-processors and transformers. Transfomers and estimators can be combined into a single object called pipeline, this way it is combined the pre-process of the data with the fitting process. The advantage of the pipeline is that simplifies the process and avoids data leakages of training data to test data or viceversa.

After have been fitted, it needs to be evaluated, for that purpose scikit-learn has methods to split the data into train and test splits, but it can more advanced techniques like cross-validation. There are many metrics offered to observe the performance of the estimator, like accuracy, precision, recall, f1 score, etc.

2.9.3.5 GSITK

Gsitk [34] [35] is a library developed by the Intelligent systems group (GSI) is a library on top of scikit-learn that facilitates the development process on NLP machine learning driven projects. It uses numpy, pandas and related libraries and it is designed to be compatible with scikit-learn's Pipeline. It was developed by the Intelligent systems group (GSI).

gsitk makes easy and fast to write a pipeline that manages datasets, get features, apply classifiers and evaluation techniques. It has the following components:

- Datasets. It offers a suite to download, process and work with some of the most important NLP datasets. With the common framework, it eases projects using the same datasets.

- Pre-processing. It has various funcionalities for pre-processing text tasks. The pre-processor can be based on regular expressions parsing English text or a less-efficient based on Nltk. It has a special module to process tweets and deal with the special characters and expressions of Twitter texts. This special module is the reason why this library has been used in this project.

- Feature extraction. Implementations of a number of feature extraction techniques. The techniques implemented are Word2VecFeatures, Doc2VecFeatures, SIMON and SSWE.

- Classifiers. It implements classifiers that are compatible with scikit-learn predictors. There are two types: LexiconSum and VaderClassifier.

- Evaluation. A number of functionalities are offered to aid in the task of performing evaluation in the field of Sentiment Analysis. It is divided in two types: basic evaluation and advanced evaluation.

2.9.3.6 Gensim

Gensim [7] is a free open-source Python library focused on NLP tasks. It was designed to process raw, unstructured text data using unsupervised machine learning algorithms. These algorithms are among others Word2Vec, FastText, Latent Semantic Indexing (LSI, LSA, LsiModel), Latent Dirichlet Allocation (LDA, LdaModel) and they automatically discover the semantic structure of documents by examining statistical co-occurrence patterns, then it can be expressed in the new, semantic representation.

Gensim is built to be fast, computationally efficient and optimized. It is the fastest library to train vector embeddings. It can use data-streamed algorithms as it has no limitations on RAM size. Its performance is proven with the use of gensim for thousand of companies and research citations.

2.9.3.7 Nltk

Natural Language Toolkit [16] offers a suite of Python libraries for symbolic and statistical NLP. It is a complete library that can perform almost all operation for text like classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers.

It also has more than one hundred built-in corpora and trained models. This library ultimately is a all-in-one for natural language processing purposes.

2.9.3.8 Keras

Keras [13] is an open-source that that provides a Python interface for artificial neural networks. It is designed to be powerful, modular, easy to use, easily interpretable and minimize the number of user actions required for common use cases.

It is the most used deep learning framework and it is meant to be build everywhere. However it needs to be build along other libraries that handles low-level computations like TensorFlow or Theano.

2.9.3.9 TensorFlow

TensorFlow [26] is an end-to-end open-source platform for machine learning developed by Google and it offers a suite of tools to build and deploy machine learning models.

TensorFlow is meant to be used and trained no matter the language or the platform, it can be on mobiles, servers, web. It can load and preprocess data, build and train models using Keras without sacrificing speed or performance and they can be trained on CPU, GPU or TPU. It offers different models for different levels of abstraction to work with the Keras API and makes it easy to deploy a machine learning model.

It is one of the most used libraries in data science with an active community of developers, researchers open to contribute and collaborate, and some of the top companies use it like Google, Airbnb or Intel.

CHAPTER 3

Architecture

This chapter presents the general architecture of the project, with the different blocks that compose it and with the connections between the different components involved on the development of the project. An introduction of each of the different modules is given.

The architecture of the project, from a high level of abstraction is described in figure 3.1, it can be seen as a black box, where the input is a collection of texts, some operations are made inside a tool, and it delivers two outputs, two lexicons each one from a different algorithm, the sentprop lexicon and the densifier lexicon.

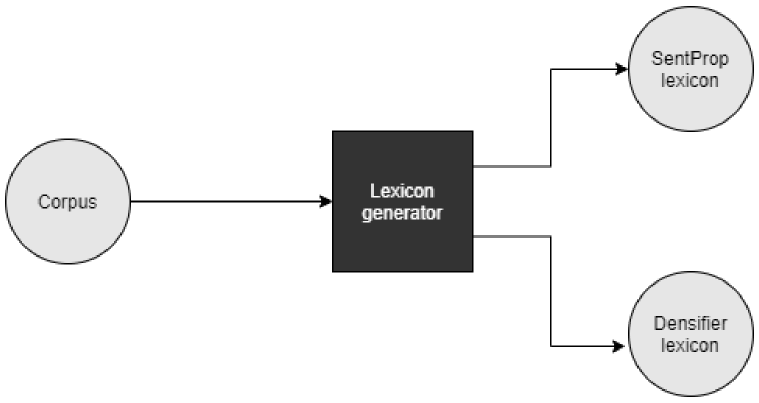


Figure 3.1: High level abstraction of the project

3.1 Input data

The input of the program as it have been said is a collection of texts, all arranged in one text file, where each line corresponds to a text, this could be a Imdb review or a tweet.

Before data appears in that format, a series of data pre-processing techniques are applied to the raw data. First of all, raw data have a different structure, it is not a single file with all the texts, each dataset has a different organization so texts are read and written one by one into a blank file with each line a new text. In addition, data is processed to discard non relevant information, so the pre-processing applies the following techniques:

- Tokenization. Split the whole text into small tokens, in this case each token corresponds to a word.
- Lower case. All words are put in lower case. This way ensure is the same token if the same word appears in uppercase and lowercase throughout the corpus.
- Stop words removal. Remove the low information words like articles and pronouns, in order to focus on the important information. Both python libraries nltk and gsitk can perform this function.
- Punctuation removal. Remove punctuation signs like dots and commas.

These techniques are common for both problems, in addition to that, for the Twitter dataset, it is applied a special process related to tweets. This is a function of the gsitk suite, it deals with the special symbols that appears in Twitter, like the '#' for hashtags and '@' for usernames. An example of a tweet after applying the gsitk function:

@YoungJeezy got me feelin like #trappin ain't dead. Got me ready to call the #guala! #Salute da truth! #SeenItAll #SeenItAllTheAutobiography

results in

<user>got me feelin like <hastag>trappin ain't dead. got me ready to call the <hastag >guala! <hastag>salute da truth! <hastag >seenitall <hastag >seenitalltheautobiography

After all the pre-processing, text is split into two sets, train and test split. In the case of the Imdb dataset this is not necessary because data is already separated as train and test. For the Twitter dataset, it is used a 80/20 proportions for the training and test sets respectively.

3.2 Lexicon generation

Now, it will be analyzed the black box as it is expressed in figure 3.1. A detailed view of the operations inside this module are explained and can be seen in figure 3.2. The main functions used in this module are provided by the socialsent framework, like the lexicon generations or the creation of embeddings, the code is open-source and it is available on the github page of the developer [44].

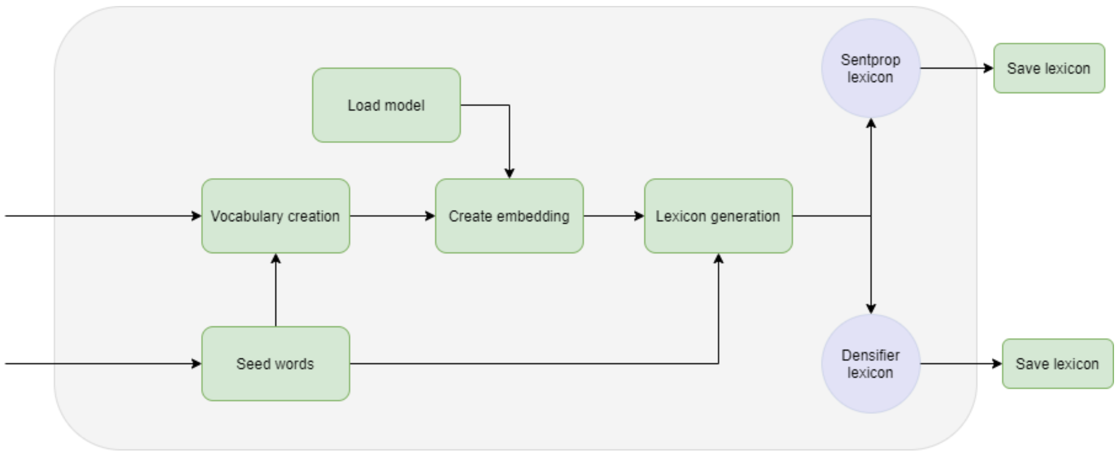


Figure 3.2: Detailed view on the lexicon generation module

Vocabulary creation. As it has been said, input data is a unique text file with every line a different text. The first step is read the file line by line obtaining the text, tokenizing the sentence and getting all the words that appear on it. Then a list is created with all the words that appear in the corpus, but every word is unique, only appears once. This is the vocabulary of the problem.

Seed words. Word seeds are in a document where each line is a seed. Actually, there are two documents, one for positive seeds and other one for negative seeds. Files are read and word seeds obtained, and they are included in the vocabulary of the corpus.

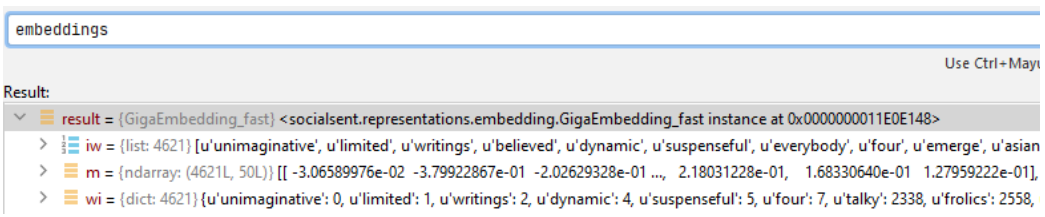
Load pre-trained model. The gensim library has a built-in function to load known pre-trained embedding models, they are models with proven accuracy and commonly used in NLP applications. Figure 3.3 shows how to load a model using gensim library, in this example it is loaded the model 'glove-wiki-gigaword-50'

```
>>> import gensim.downloader
>>> model = gensim.downloader.load('glove-wiki-gigaword-50')
```

Figure 3.3: Load a pre-trained model with the gensim library

Create embedding. In this step, it is used all the data has been created in the earlier steps. The objective is to obtain an object that contains all the words from the vocabulary with the corresponding representation value from the embedding model. So it is used the data from corpus plus the positive and negative seed words as vocabulary and the model loaded to obtain the values. An embedding object is created as figure 3.4 that it is structured as follows:

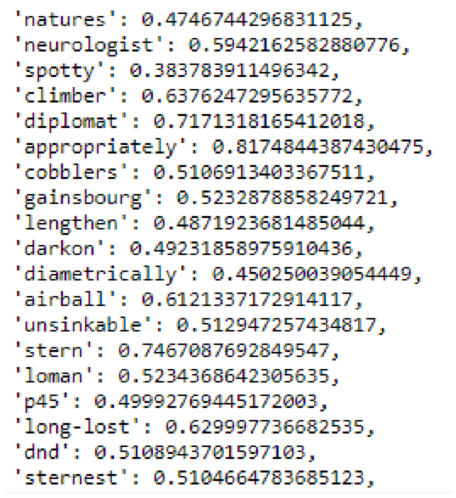
- iw. It is a list that contains all the words from the vocabulary, including the seed words.
- m. It is an array with the same entries as the length of iw. It contains the word embeddings of each word.
- im. It is a dictionary with the word as key and an index as value.



```
result = {GigaEmbedding_fast} <socialsent.representations.embedding.GigaEmbedding_fast instance at 0x000000011E0E148>
  iw = {list: 4621} [u'unimaginative', u'limited', u'writings', u'believed', u'dynamic', u'suspenseful', u'everybody', u'four', u'emerge', u'asian
  m = {ndarray: (4621L, 50L)} [[ -3.06589976e-02  -3.79922867e-01  -2.02629328e-01 ...,   2.18031228e-01,   1.68330640e-01   1.27959222e-01],
  wi = {dict: 4621} {u'unimaginative': 0, u'limited': 1, u'writings': 2, u'dynamic': 4, u'suspenseful': 5, u'four': 7, u'talky': 2338, u'frolics': 2558,
```

Figure 3.4: Example of an obtained object embeddings

Lexicon generation. Finally, lexicons can be generated with the socialsent functions to run the algorithms of sentprop and densifier. For both functions, the inputs are the embedding object previously calculated and the positive and negative seed words. After the execution, two lexicons are generated, one for each algorithm. Figure 3.5 shows a sample of a lexicon, which is a dictionary where words are keys and values the corresponding polarity score.



```
'natures': 0.4746744296831125,
'neurologist': 0.5942162582880776,
'spotty': 0.383783911496342,
'climber': 0.6376247295635772,
'diplomat': 0.7171318165412018,
'appropriately': 0.8174844387430475,
'cobblers': 0.5106913403367511,
'gainsbourg': 0.5232878858249721,
'lengthen': 0.4871923681485044,
'darkon': 0.49231858975910436,
'diametrically': 0.450250039054449,
'airball': 0.6121337172914117,
'unsinkable': 0.512947257434817,
'stern': 0.7467087692849547,
'loman': 0.5234368642305635,
'p45': 0.49992769445172003,
'long-lost': 0.629997736682535,
'dnd': 0.5108943701597103,
'sternest': 0.5104664783685123,
```

Figure 3.5: Sample of a lexicon

Save lexicon. The last step is to save lexicons as json file. As they are dictionaries, this operation it is natural as json files' text contains key-value with in brackets just as python dictionaries. Lexicons are saved to disk for future uses such as evaluation.

CHAPTER 4

Methodology

This chapter presents the methodology followed for the development of the technical part of the project, explaining the procedure for the two use cases, sentiment analysis and hate speech analysis, including the software requirements, an overview of the datasets used, how it is data treated, and the sentiment analysis techniques employed.

4.1 Datasets

Two different datasets have been used for each of the problems, an Imdb dataset for the sentiment analysis problem and a Twitter dataset for the hate speech analysis.

4.1.1 Imdb Dataset

The Imdb dataset [49] contains 50000 movie reviews from the Internet Movie Database, they are classified as positive or negative reviews. It is balanced so 50% of the reviews are positive and 50% negative. In addition, to ensure not having hidden patterns, no more than 30 reviews per movie are allowed. Only highly polarized reviews are considered, negative reviews have a score less than 4 out of 10, and positive reviews have a score more than 7 out of 10, therefore ignoring neutral comments.

The dataset is already split into train and test sets, each set is balanced and contains half of the samples, so the train test have 12500 positive reviews and 12500 negative reviews, and the test set the same. The dataset is divided into folders and it has the following folder structure:

```
IMDB dataset
|
├─ train
|   ├─ positive
|   |
|   └─ negative
|
└─ test
    ├─ positive
    |
    └─ negative
```

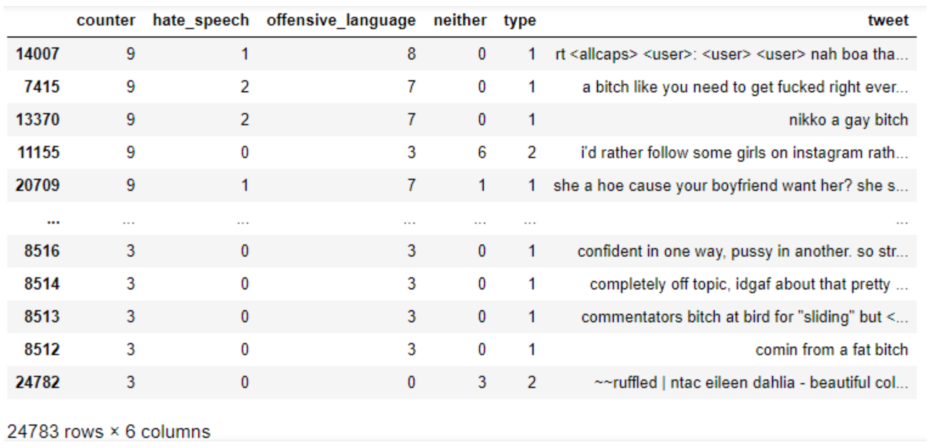
4.1.2 Twitter Dataset

The Twitter dataset [40] is a collection of 25000 tweets categorized as hate speech, offensive or neither. The procedure begins obtaining tweets that contains some words that users have identified as hate speech according to hatebase.org. They obtained a huge amount of tweets and took a random sample of 25000 tweets. These tweets were manually annotated by CrowdFlower workers as hate speech, offensive but not hate speech, or neither offensive nor hate speech. Each twit label is set with the majority decision. Finally, a 5% of tweets are classified as hate speech, a 76% as offensive and the remainder neither offensive nor hate speech. This dataset is unbalanced and majority belongs to the offensive class.

The dataset comes in a csv file, with the following columns as it can be seen in figure 4.1:

- Counter. Total annotations by CrowdFlower workers.
- Hate_speech. Times the tweet is classified as hate speech.
- Offensive_language. Times the tweet is classified as offensive.
- Neither. Times the tweet is classified as neither offensive nor hate speech.
- Type. The classification of the tweet according to the majority vote. The encoding is:
  - 0: Hate speech
  - 1: Offensive language
  - 2: Neither
- Tweet. The tweet in question.

With this classification, two problems are proposed, one with offensive data as negative class and neither data as positive class, and other problem with hate data as negative class and neither data as positive class.



	counter	hate_speech	offensive_language	neither	type	tweet
14007	9	1	8	0	1	rt <allcaps> <user>: <user> <user> nah boa tha...
7415	9	2	7	0	1	a bitch like you need to get fucked right ever...
13370	9	2	7	0	1	nikko a gay bitch
11155	9	0	3	6	2	i'd rather follow some girls on instagram rath...
20709	9	1	7	1	1	she a hoe cause your boyfriend want her? she s...
...	...	...	...	...	...	...
8516	3	0	3	0	1	confident in one way, pussy in another. so str...
8514	3	0	3	0	1	completely off topic, idgaf about that pretty ...
8513	3	0	3	0	1	commentators bitch at bird for "sliding" but <...
8512	3	0	3	0	1	comin from a fat bitch
24782	3	0	0	3	2	~~ruffled | ntac eileen dahlia - beautiful col...

24783 rows × 6 columns

Figure 4.1: Sample of the Twitter dataset

4.2 Pre-trained models

Word embeddings are the representation of words into a space vector. There are several techniques and algorithms to obtain those representations, but other option is to directly import an already trained embedding model.

Some institutions have released word embeddings obtained with their algorithms like word2vec or GloVe, developed by Google and Stanford University respectively, the advantages of using these pre-trained models are that they are well-known word embeddings with proven efficiency in a range of tasks [57], like word analogy or word similarity tasks. So state of the art word embeddings can be used saving the time and computational power trainings require. In fact, some models like word2vec contains about 100 billions words[9].

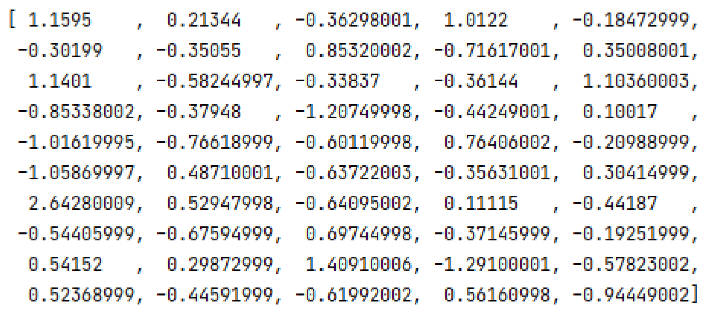
The python library gensim, comes with several pre-trained models easily accessible. Iit can be used any of the list, where the number at the end of each name indicates the number of dimensions the embeddings have.

- fasttext-wiki-news-subwords-300
- conceptnet-numberbatch-17-06-300
- word2vec-ruscorpora-300
- word2vec-google-news-300
- glove-wiki-gigaword-50
- glove-wiki-gigaword-100
- glove-wiki-gigaword-200
- glove-wiki-gigaword-300
- glove-twitter-25
- glove-twitter-50
- glove-twitter-100
- glove-twitter-200

In this project it is used the model 'glove-wiki-gigaword-50' for the sentiment analysis, and as an example we can see the embedding of the word 'spain' in figure 4.2, where it is composed of a 50 dimensional vector. For the hate speech analysis it is used the 'glove-twitter-100' model.

Gensim library has a built-in function to get the most similar words, and following with the example, the 10 words that are more similar to 'spain' with its respective indicator of similarity are:

- portugal, 0.8874139785766602
- italy, 0.8616417646408081
- argentina, 0.8610804080963135
- brazil, 0.8340150117874146



```
[ 1.1595    ,  0.21344   , -0.36298001,  1.0122    , -0.18472999,
 -0.30199   , -0.35055   ,  0.85320002, -0.71617001,  0.35008001,
  1.1401    , -0.58244997, -0.33837   , -0.36144   ,  1.10360003,
 -0.85338002, -0.37948   , -1.20749998, -0.44249001,  0.10017   ,
 -1.01619995, -0.76618999, -0.60119998,  0.76406002, -0.20988999,
 -1.05869997,  0.48710001, -0.63722003, -0.35631001,  0.30414999,
  2.64280009,  0.52947998, -0.64095002,  0.11115   , -0.44187   ,
 -0.54405999, -0.67594999,  0.69744998, -0.37145999, -0.19251999,
  0.54152   ,  0.29872999,  1.40910006, -1.29100001, -0.57823002,
  0.52368999, -0.44591999, -0.61992002,  0.56160998, -0.94449002]
```

Figure 4.2: Embedding of the word ‘spain’ in the ‘glove-wiki-gigaword-50’ model

- spanish, 0.8168432116508484
- costa, 0.8029966354370117
- france, 0.7909148931503296
- rica, 0.7809268832206726
- republic, 0.7761150002479553
- ecuador, 0.7706073522567749

4.3 Seed words

A good set of seed words are crucial for the algorithms to generate a lexicon related to the case study. So the word seeds have to be the best words to which the concept in question is associated, because the algorithm will build on these words.

The seed words employed for the sentiment analysis and hate speech analysis are in the table 4.1 and table 4.2 respectively.

The seed words selection is from the socialsent framework, where it has a module to select different set seeds for various domains: finances, tweets, historical, seeds used in other research and adjectives. For the sentiment analysis, the seed words used are the historical seeds. There are no hard rules to select the seed words, it is a hand-curated selection where the most representative words are looked for. For the positive words of the hate analysis problem, the Twitter set is used.

For the hate analysis, the negative words are obtained by using the hatebase.org database of hate words. It is a hand-curated selection of the more than a thousand words that people collaboratively has annotated as being offensive, and they have also some statistics of the average offensiveness of the word. The selection are words that are commonly used to offend someone, and trying to cover all the collectives or minorities more affected, so negative seed words are words that discriminate blacks, hispanics, women, homosexuals or disabled people.

Positive seeds	Negative seeds
good	bad
lovely	horrible
excellent	poor
fortunate	unfortunate
pleasant	unpleasant
delightful	disgusting
perfect	evil
loved	hated
love	hate
happy	unhappy

Table 4.1: Seed words for sentiment analysis problem

Positive seeds	Negative seeds
love	nigga
loved	bitch
loves	faggot
awesome	nigger
nice	asshole
amazing	motherfucker
best	redneck
fantastic	wetback
correct	retard
happy	gipsy

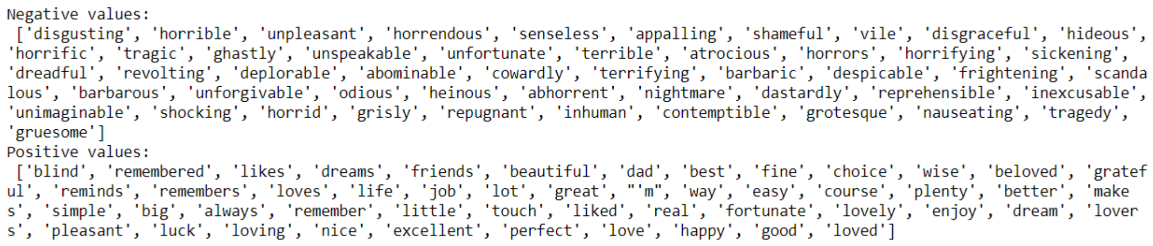
Table 4.2: Seed words for hate speech analysis problem

4.4 Algorithms

Finally, after setting all the different parameters and necessary inputs, the different algorithms can be executed. In this case are executed two different algorithms, sentprop and densifier.

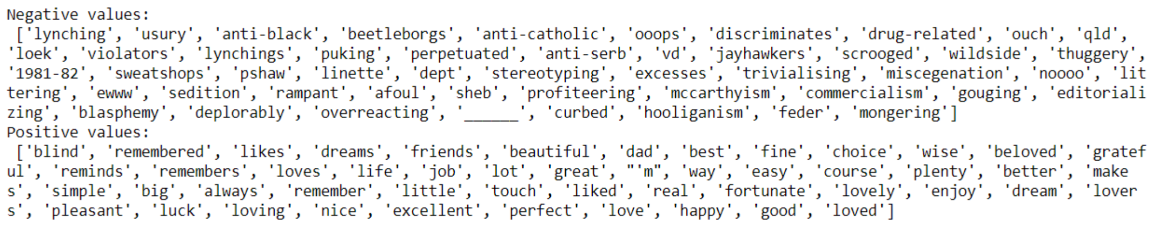
4.4.1 Sentiment analysis

Let's take a look of the lexicons generated by sentprop and densifier algorithms in the sentiment analysis problem. For this purpose, the vocabulary values are sorted in ascending order, so the lower values are at the bottom and the higher at the top. Figure 4.3 and figure 4.4 shows the 50 most positive words and 50 most negative words for the algorithms sentprop and densifier respectively. The results of the sentprop algorithm corresponds to the desired objective, in the extreme values appear the seed words, and the rest of them are clearly positive or negative, like grateful, fine, great or enjoy for positive and tragic, scandalous or repugnant for negative. Densifier results also presents good results, the positive results seems really similar as the sentprop ones, but for the negative words there are more differences between them. For the negative words, sentprop seems to have better results, it captures more negativity in its words, and densifier has some weaker results like ewww, commercialism, littering or ouch.



```
Negative values:
 ['disgusting', 'horrible', 'unpleasant', 'horrendous', 'senseless', 'appalling', 'shameful', 'vile', 'disgraceful', 'hideous',
'horrific', 'tragic', 'ghastly', 'unspeakable', 'unfortunate', 'terrible', 'atrocious', 'horrors', 'horrifying', 'sickening',
'dreadful', 'revolting', 'deplorable', 'abominable', 'cowardly', 'terrifying', 'barbaric', 'despicable', 'frightening', 'scanda
lous', 'barbarous', 'unforgivable', 'odious', 'heinous', 'abhorrent', 'nightmare', 'dastardly', 'reprehensible', 'inexcusable',
'unimaginable', 'shocking', 'horrid', 'grisly', 'repugnant', 'inhuman', 'contemptible', 'grotesque', 'nauseating', 'tragedy',
'gruesome']
Positive values:
 ['blind', 'remembered', 'likes', 'dreams', 'friends', 'beautiful', 'dad', 'best', 'fine', 'choice', 'wise', 'beloved', 'gratef
ul', 'reminds', 'remembers', 'loves', 'life', 'job', 'lot', 'great', "'m", 'way', 'easy', 'course', 'plenty', 'better', 'make
s', 'simple', 'big', 'always', 'remember', 'little', 'touch', 'liked', 'real', 'fortunate', 'lovely', 'enjoy', 'dream', 'lover
s', 'pleasant', 'luck', 'loving', 'nice', 'excellent', 'perfect', 'love', 'happy', 'good', 'loved']
```

Figure 4.3: Lower and higher values of sentprop lexicon for Imdb reviews



```
Negative values:
 ['lynching', 'usury', 'anti-black', 'beetleborgs', 'anti-catholic', 'ooops', 'discriminates', 'drug-related', 'ouch', 'qld',
'loek', 'violators', 'lynchings', 'puking', 'perpetuated', 'anti-serb', 'vd', 'jayhawkers', 'scrooged', 'wildside', 'thuggery',
'1981-82', 'sweatshops', 'pshaw', 'linette', 'dept', 'stereotyping', 'excesses', 'trivialising', 'miscegenation', 'noooo', 'lit
tering', 'ewww', 'sedition', 'rampant', 'afoul', 'sheb', 'profiteering', 'mccarthyism', 'commercialism', 'gouging', 'editoriali
zing', 'blasphemy', 'deplorably', 'overreacting', '______', 'curbed', 'hooliganism', 'feder', 'mongering']
Positive values:
 ['blind', 'remembered', 'likes', 'dreams', 'friends', 'beautiful', 'dad', 'best', 'fine', 'choice', 'wise', 'beloved', 'gratef
ul', 'reminds', 'remembers', 'loves', 'life', 'job', 'lot', 'great', "'m", 'way', 'easy', 'course', 'plenty', 'better', 'make
s', 'simple', 'big', 'always', 'remember', 'little', 'touch', 'liked', 'real', 'fortunate', 'lovely', 'enjoy', 'dream', 'lover
s', 'pleasant', 'luck', 'loving', 'nice', 'excellent', 'perfect', 'love', 'happy', 'good', 'loved']
```

Figure 4.4: Lower and higher values of densifier lexicon for Imdb reviews

4.4.2 Hate speech analysis

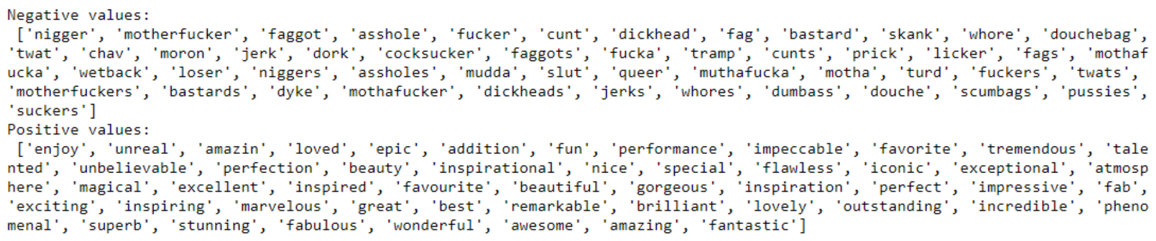
Now the problem it is executed for the hate speech analysis. As it have been said, this problem is separated in two, one to analyze hate tweets and other for offensive tweets.

4.4.2.1 Hate tweets

Figure 4.5 and figure 4.6 shows the 50 lower values and 50 higher lower values of the algorithms sentprop and densifier algorithms respectively for hate tweets. Both algorithms show really good results for the extreme values, the words are really close to the seed words, like moron, cocksucker, scumbag for negative words or gorgeous, brilliant for positive words on the sentprop values. For the densifier results, it is interesting that it captures negative words, but are more slang words or expressionslike gtfoh, mfka, fym and multiple ways of the word bitch. Although both results reflects the hate speech, at a glance with this low-sized sample, the sentprop have more robust results.

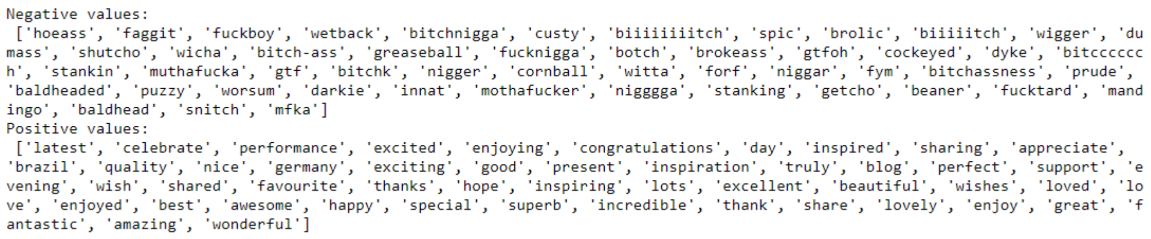
4.4.2.2 Offensive tweets

Now the problem is the same as before but for offensive tweets. Figure 4.7 and figure 4.8 shows the results for sentprop and densifier algorithms respectively. For the sentprop



```
Negative values:
 ['nigger', 'motherfucker', 'faggot', 'asshole', 'fucker', 'cunt', 'dickhead', 'fag', 'bastard', 'skank', 'whore', 'douchebag',
'twat', 'chav', 'moron', 'jerk', 'dork', 'cocksucker', 'faggots', 'fucka', 'tramp', 'cunts', 'prick', 'licker', 'fags', 'mothaf
ucka', 'wetback', 'loser', 'niggers', 'assholes', 'mudda', 'slut', 'queer', 'muthafucka', 'motha', 'turd', 'fuckers', 'twats',
'motherfuckers', 'bastards', 'dyke', 'mothafucker', 'dickheads', 'jerks', 'whores', 'dumbass', 'douche', 'scumbags', 'pussies',
'suckers']
Positive values:
 ['enjoy', 'unreal', 'amazin', 'loved', 'epic', 'addition', 'fun', 'performance', 'impeccable', 'favorite', 'tremendous', 'tale
nted', 'unbelievable', 'perfection', 'beauty', 'inspirational', 'nice', 'special', 'flawless', 'iconic', 'exceptional', 'atmosp
here', 'magical', 'excellent', 'inspired', 'favourite', 'beautiful', 'gorgeous', 'inspiration', 'perfect', 'impressive', 'fab',
'exciting', 'inspiring', 'marvelous', 'great', 'best', 'remarkable', 'brilliant', 'lovely', 'outstanding', 'incredible', 'pheno
menal', 'superb', 'stunning', 'fabulous', 'wonderful', 'awesome', 'amazing', 'fantastic']
```

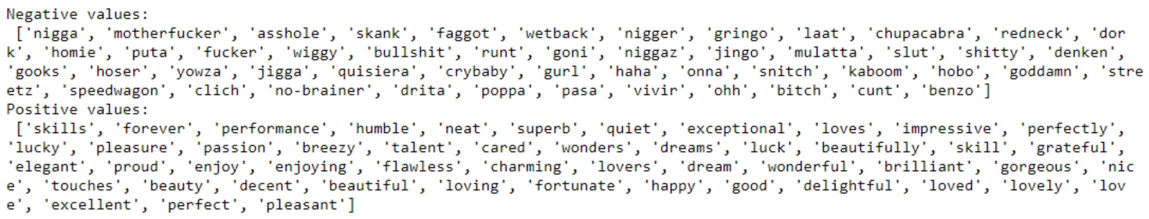
Figure 4.5: Lower and higher values of sentprop lexicon for hate tweets



```
Negative values:
 ['hoeass', 'faggit', 'fuckboy', 'wetback', 'bitchnigga', 'custy', 'biiiiiiiitch', 'spic', 'brolic', 'biiiiitch', 'wigger', 'du
mass', 'shutcho', 'wicha', 'bitch-ass', 'greaseball', 'fucknigga', 'botch', 'brokeass', 'gtfoh', 'cockeyed', 'dyke', 'bitcccccc
h', 'stankin', 'muthafucka', 'gtf', 'bitchk', 'nigger', 'cornball', 'witta', 'forf', 'niggar', 'fym', 'bitchassness', 'prude',
'baldheaded', 'puzzy', 'worsum', 'darkie', 'innat', 'mothafucker', 'nigggga', 'stanking', 'getcho', 'beaner', 'fucktard', 'mand
ingo', 'baldhead', 'snitch', 'mfka']
Positive values:
 ['latest', 'celebrate', 'performance', 'excited', 'enjoying', 'congratulations', 'day', 'inspired', 'sharing', 'appreciate',
'brazil', 'quality', 'nice', 'germany', 'exciting', 'good', 'present', 'inspiration', 'truly', 'blog', 'perfect', 'support', 'e
vening', 'wish', 'shared', 'favourite', 'thanks', 'hope', 'inspiring', 'lots', 'excellent', 'beautiful', 'wishes', 'loved', 'lo
ve', 'enjoyed', 'best', 'awesome', 'happy', 'special', 'superb', 'incredible', 'thank', 'share', 'lovely', 'enjoy', 'great', 'f
antastic', 'amazing', 'wonderful']
```

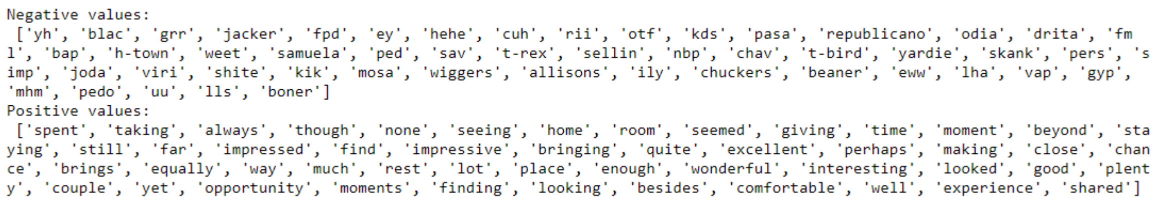
Figure 4.6: Lower and higher values of densifier lexicon for hate tweets

algorithm, the results seems similar to the hate tweets results, but with one degree less of aggresiveness, there are still stong words like cunt, but in general there are softer. In this case, the densifier seems to hae worse results than sentprop, it is curious that the seed words does not appear in the lower values and there are a lot of slang and expressions without offensive meaning like yh, ey or uu. The positive results are much better than the negatives.



```
Negative values:
 ['nigga', 'motherfucker', 'asshole', 'skank', 'faggot', 'wetback', 'nigger', 'gringo', 'laat', 'chupacabra', 'redneck', 'dor
k', 'homie', 'puta', 'fucker', 'wiggy', 'bullshit', 'runt', 'goni', 'niggaz', 'jingo', 'mulatta', 'slut', 'shitty', 'denken',
'gooks', 'hoser', 'yowza', 'jigga', 'quisiera', 'crybaby', 'gurl', 'haha', 'onna', 'snitch', 'kaboom', 'hobo', 'goddamn', 'stre
etz', 'speedwagon', 'clich', 'no-brainer', 'drita', 'poppa', 'pasa', 'vivir', 'ohh', 'bitch', 'cunt', 'benzo']
Positive values:
 ['skills', 'forever', 'performance', 'humble', 'neat', 'superb', 'quiet', 'exceptional', 'loves', 'impressive', 'perfectly',
'lucky', 'pleasure', 'passion', 'breezy', 'talent', 'cared', 'wonders', 'dreams', 'luck', 'beautifully', 'skill', 'grateful',
'elegant', 'proud', 'enjoy', 'enjoying', 'flawless', 'charming', 'lovers', 'dream', 'wonderful', 'brilliant', 'gorgeous', 'nic
e', 'touches', 'beauty', 'decent', 'beautiful', 'loving', 'fortunate', 'happy', 'good', 'delightful', 'loved', 'lovely', 'lov
e', 'excellent', 'perfect', 'pleasant']
```

Figure 4.7: Lower and higher values of sentprop lexicon for offensive tweets



```
Negative values:
 ['yh', 'blac', 'grr', 'jacker', 'fpd', 'ey', 'hehe', 'cuh', 'rii', 'otf', 'kds', 'pasa', 'republicano', 'odia', 'drita', 'fm
l', 'bap', 'h-town', 'weet', 'samuela', 'ped', 'sav', 't-rex', 'sellin', 'nbp', 'chav', 't-bird', 'yardie', 'skank', 'pers', 's
imp', 'joda', 'viri', 'shite', 'kik', 'mosa', 'wiggers', 'allisons', 'ily', 'chuckers', 'beaner', 'eww', 'lha', 'vap', 'gyp',
'mhm', 'pedo', 'uu', 'lls', 'boner']
Positive values:
 ['spent', 'taking', 'always', 'though', 'none', 'seeing', 'home', 'room', 'seemed', 'giving', 'time', 'moment', 'beyond', 'sta
ying', 'still', 'far', 'impressed', 'find', 'impressive', 'bringing', 'quite', 'excellent', 'perhaps', 'making', 'close', 'chan
ce', 'brings', 'equally', 'way', 'much', 'rest', 'lot', 'place', 'enough', 'wonderful', 'interesting', 'looked', 'good', 'plent
y', 'couple', 'yet', 'opportunity', 'moments', 'finding', 'looking', 'besides', 'comfortable', 'well', 'experience', 'shared']
```

Figure 4.8: Lower and higher values of densifier lexicon for offensive tweets

CHAPTER 5

Evaluation

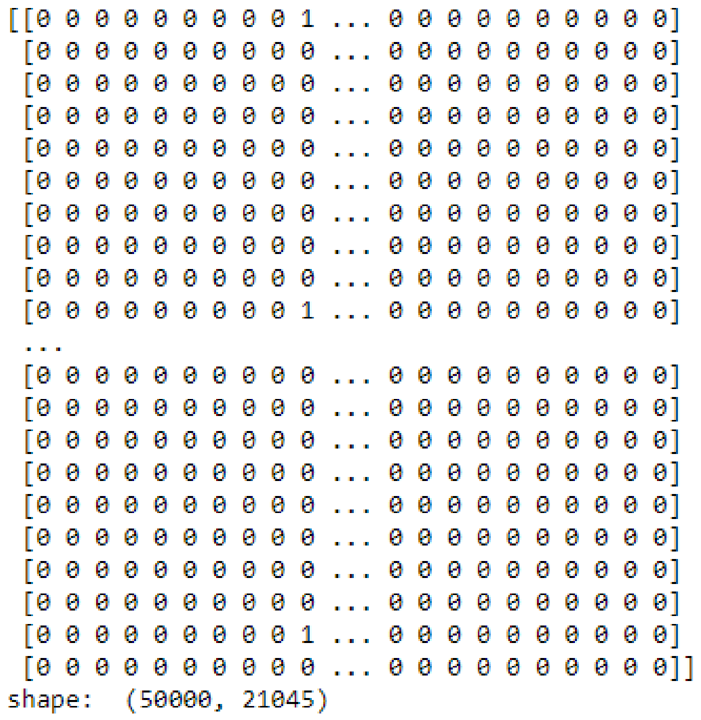
This chapter presents the evaluation methods considered to get metrics and compare different algorithms. After that, the results of the evaluation methods are presented for all the problems and algorithms carried out ovver the course of this project.

5.1 Evaluation methods

In this section are presented the different evaluation methods used to obtain metrics about the performance of the algorithms and be able to compare them. The methods used are logistic regression on unigram sequence, logistic regression on lexicon values and classification with the sum of the values of the lexicon.

5.1.1 Unigram logistic regression

This classification is done with unigrams as the features. A n-gram is a sequence of n contiguous items, so an unigram is a sequence of one item, essentially a word. Therefore, the features are the unique words of the entire corpus, and each entry is a text where its words have a '1' on the corresponding unigram feature column and '0' elsewhere. Figure 5.1 shows an example of the matrix for the sentiment analysis problem, the dimensions of the matrix are huge so it is not easy to show the entire matrix, the shape of the matrix is at the bottom of the figure, where it can be seen that are 50000 entries, one for each test, and it has 21045 columns, each one corresponding to a different word.



```
[[0 0 0 0 0 0 0 0 0 1 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 1 ... 0 0 0 0 0 0 0 0 0 0]
 ...
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 1 ... 0 0 0 0 0 0 0 0 0 0]
 [0 0 0 0 0 0 0 0 0 0 ... 0 0 0 0 0 0 0 0 0 0]]
shape:  (50000, 21045)
```

Figure 5.1: Example of a section of the unigram matrix and the shape of the matrix

The matrix obtained contains data in a structured manner and therefore a machine can understand it and apply machine learning models. It is important to notice that the matrix contains all texts, from both train and test sets, this is because a model is fitted according to the features, so both sets need to have the same features.

Then, with the use of the scikit-learn module of linear models, import a logistic regression model, and make two operations:

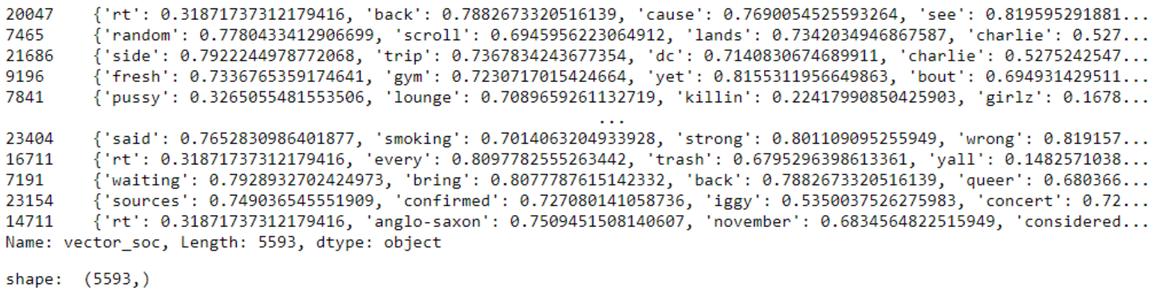
- Fit a logistic regression model with the training data.
- Once the model has been trained, predict the label that the model delivers of the test data.

These two operations are performed with the funcions fit and predict of the scikit-library, they are functions that are part of the linear models, in this case the model is the logistic regression, included in the sklearn library.

5.1.2 Logistic regression on lexicon values

This classification follows the same idea as the one before, it creates a matrix with all the words of the vocabulary as features, but instead that entries mark if a word is present in the text or not with a '1' or a '0' respectively, this classification uses the values of the words from the lexicons created.

To achieve that, first it is created a dictionary for each text, that contains all the words of the text as keys and its respective values are the lexicon scores. Figure 5.2 shows a sample of the dictionaries created for several texts of the Twitter dataset, containing words and its respective lexicon values from the sentprop algorithm. At the bottom of the figure it is shown the shape of the vector of dictionaries, it has as many entries as texts.



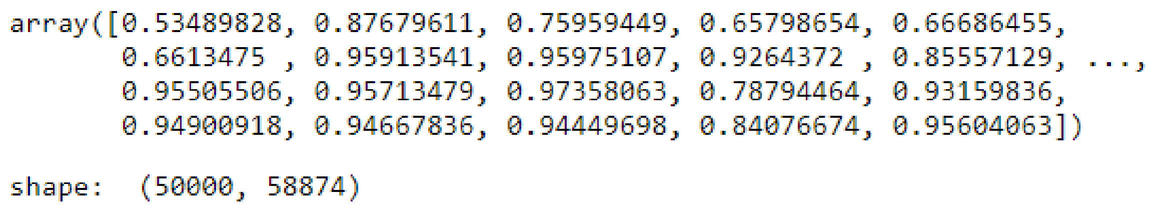
```
20047    {'rt': 0.31871737312179416, 'back': 0.7882673320516139, 'cause': 0.7690054525593264, 'see': 0.819595291881...
7465     {'random': 0.7780433412906699, 'scroll': 0.6945956223064912, 'lands': 0.7342034946867587, 'charlie': 0.527...
21686    {'side': 0.7922244978772068, 'trip': 0.7367834243677354, 'dc': 0.7140830674689911, 'charlie': 0.5275242547...
9196     {'fresh': 0.7336765359174641, 'gym': 0.7230717015424664, 'yet': 0.8155311956649863, 'bout': 0.694931429511...
7841     {'pussy': 0.3265055481553506, 'lounge': 0.7089659261132719, 'killin': 0.22417990850425903, 'girlz': 0.1678...
                                                                                         ...
23404    {'said': 0.7652830986401877, 'smoking': 0.7014063204933928, 'strong': 0.801109095255949, 'wrong': 0.819157...
16711    {'rt': 0.31871737312179416, 'every': 0.8097782555263442, 'trash': 0.6795296398613361, 'yall': 0.1482571038...
7191     {'waiting': 0.7928932702424973, 'bring': 0.8077787615142332, 'back': 0.7882673320516139, 'queer': 0.680366...
23154    {'sources': 0.749036545551909, 'confirmed': 0.727080141058736, 'iggy': 0.5350037526275983, 'concert': 0.72...
14711    {'rt': 0.31871737312179416, 'anglo-saxon': 0.7509451508140607, 'november': 0.6834564822515949, 'considered...
Name: vector_soc, Length: 5593, dtype: object

shape:  (5593,)
```

Figure 5.2: Sample of the dictionary with words as key and sentprop lexicon values of the hate twits data set and the shape of the vector

Then, that list of dictionaries with word:values it is transformed into numpy arrays with feature values using the scikit-learn tool DictVectorizer. The result is a matrix where each word is a column or feature and each row represents a text where it will have the lexicon values on the corresponding columns of the words it contains, and '0' elsewhere.

Figure 5.3 shows a sample of the non-zero elements of a row. It is difficult to illustrate a section of the matrix because it will show only zeros. At the bottom of the image, the shape of the matrix is shown, with 50000 entries or reviews in this case as it is Imdb dataset and 58874 columns representing the unique words of the entire corpus. So each row has dozens of non-zero elements compared to the almost 60k features, the matrix is very sparse.



```
array([0.53489828, 0.87679611, 0.75959449, 0.65798654, 0.66686455,
       0.6613475 , 0.95913541, 0.95975107, 0.9264372 , 0.85557129, ...,
       0.95505506, 0.95713479, 0.97358063, 0.78794464, 0.93159836,
       0.94900918, 0.94667836, 0.94449698, 0.84076674, 0.95604063])

shape:  (50000, 58874)
```

Figure 5.3: Sample of the non-zero elements of a row of a text transformed to a numpy representation

After that process, data has been transformed and it is structured. So as in the previous classification, scikit-learn estimators can use this data representation to fit a machine learning model. The process is the same as for the prior regression, the commands fit and predict are applied to the vectorized data. Training data is used to fit the logistic regression model and predictions are obtained with the test set.

5.1.3 Classification with the sum of lexicon values

The last classification is done directly adding up the lexicon values of the words of a text. The idea is that if each word has a polarity score, then taking into account the score of all the words of a text will give an indicator of the polarity of the whole text.

A function is created where the input is a text and a lexicon, it will go through all the words and getting the corresponding value from the lexicon, adding up all the values, average the sum, and the result is a number indicating the polarity of the text. Then, a classification is done by using a threshold and identifying as positive class all values above that threshold and as negative class the values below the threshold.

As sentprop and densifier algorithms uses different scales to represent the score, sentprop values range from 0 to 1 and densifier values do not have a limit, values can go above +1 and can have negative values that can go below -1. Thresholds for the different algorithms are different, in the case of sentprop is establish at 0.5 and for the densifier at 0 value.

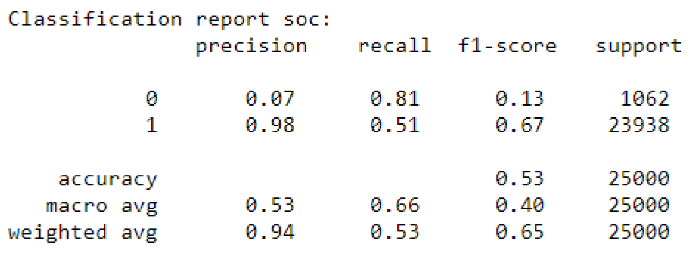
5.2 Results

This section present the results achieved from the different experiments, several statistics like precision, recall,etc are obtained for every problem to be able to compare the performance of the algorithms. Results are presented for both sentiment analysis problem and hate speech analysis problem.

5.2.1 Sentiment analysis

For the sentiment analysis problem, the three evaluation methods are applied to the test data and are obtained the different predicted labels of all the methods. With the predictions and that the true labels are known due to the data set is labelled, several statistics are extracted. They are the precision, recall, f1-score and Pearson coefficient, the latter indicates the linear correlation between two variables or lists.

Table 5.1 shows the results of this problem, where the best statistics are obtained by the unigram logistic regression, but results of both sentprop and densifier logistic regression on lexicon values are almost identical. The classification on sum of values, for both algorithms offer a much worst result compared to the other methods. Although the precision is the highest, it does not mean it is predicting correctly all labels, because the recall metric shows that about 53% of all classified as positive, were really positive, and knowing that data is balanced this shows that almost everything was classified as positive as figure 5.4 shows a clear bias towards the positive class, the classification report statistics shows statistics for each individual class and the support column shows than most of texts are classified as positive class. This is happening because the overall lexicon values are positive, and taking into account all values the average should be close to a neutral value. The word embedding model has a main role and results depend a lot on the model used.



```
Classification report soc:
              precision    recall  f1-score   support

           0       0.07      0.81      0.13      1062
           1       0.98      0.51      0.67     23938

    accuracy                           0.53     25000
   macro avg       0.53      0.66      0.40     25000
weighted avg       0.94      0.53      0.65     25000
```

Figure 5.4: Classification report of the classification on the sum of sentprop lexicon values

Evaluation method	Precision	Recall	F1 score	Pearson coefficient
Unigram logistic regression	0.86	0.86	0.86	0.72
Logistic regression on sentprop lexicon values	0.85	0.85	0.85	0.69
Logistic regression on densifier lexicon values	0.85	0.85	0.85	0.70
Classification on sum of sentprop values	0.94	0.53	0.65	0.13
Classification on sum of densifier values	0.85	0.56	0.64	0.19

Table 5.1: Result for sentiment analysis on Imdb dataset

5.2.2 Hate speech analysis

The hate analysis problem is tested by the same evaluation methods as the sentiment analysis and the same statistics are also extracted. This problem is divided into two subproblems, offensive tweets analysis and hate tweets analysis. Results of both subproblems for offensive and hate tweets are shown in table 5.2 and table 5.3 respectively.

Again, the best results are obtained with the logistic regressions and there is a big drop in the performance of the sum of values. This indicates that the regression problems give importance to the presence of a word or not, and the feature value of that word it is not so relevant.

Analysing the results, again the best result by a significant margin is the unigram logistic regression for both subproblems. On the hate tweets problem, the results for the densifier are clearly better than for sentprop, so for the hate speech tweets the algorithm densifier is the best technique. However, for offensive tweets both algorithms behave similar. Comparing the classification on sum of values, results are better for hate tweets than for offensive tweets, this can happen because hate tweets have a more extreme content, with more words directly related to hate speech and therefore using words with extreme negative values.

In general, for both sentiment and hate analyses, both sentprop and densifier algorithms results are similar, with some improvement on the densifier ones. And the only case where the gap is really significant is in the hate tweets problem.

Although the results of the sum of values classifications are the worst comparing with the other two evaluation methods, this evaluation presents a huge advantage, it has no restrictions on the data set is predicted. For the other evaluation methods, a matrix was constructed with the information of both train and test sets, and a machine learning model was fitted, while this evaluation method proposes predictions without the need for a ML model, being suitable for real-time applications or evaluations on demand.

Evaluation method	Precision	Recall	F1 score	Pearson coefficient
Unigram logistic regression	0.92	0.94	0.93	0.86
Logistic regression on sentprop lexicon values	0.88	0.78	0.82	0.66
Logistic regression on densifier lexicon values	0.86	0.82	0.84	0.68
Classification on sum of sentprop values	0.58	0.58	0.35	0.15
Classification on sum of densifier values	0.56	0.56	0.33	0.126

Table 5.2: Result of offensive twits analysis on Twitter dataset

Evaluation method	Precision	Recall	F1 score	Pearson coefficient
Unigram logistic regression	0.92	0.88	0.90	0.80
Logistic regression on sentprop lexicon values	0.82	0.63	0.65	0.41
Logistic regression on densifier lexicon values	0.90	0.83	0.86	0.73
Classification on sum of sentprop values	0.61	0.62	0.45	0.225
Classification on sum of densifier values	0.59	0.66	0.60	0.24

Table 5.3: Result of hate twits analysis on Twitter dataset

CHAPTER 6

Conclusions

This chapter will describe the achieved goals done by the master thesis following some the key points developed in the project, and the future lines of investigation.

6.1 Project goals

This project has achieved the following goals:

- Generate a lexicon for a sentiment analysis problem, where each word has its corresponding value indicating how positive or negative is.

- Generate a lexicon for a hate speech analysis problem, in a very similar way as the first goal, changing the focus and indicating how related to hate speech a word is. In addition, another lexicon is generated to analyze offensive speech.

- Propose different evaluation methods, and submit each lexicon generated to this methods, obtain several statistics and compare the results.

6.2 Conclusions

The project shows the capabilities of Natural Language Processing and sentiment analysis techniques to generate specific domains lexicons depending on the input of the system, like the dataset or the seed words employed. In particular, this project goes from the sentiment analysis to the hate speech analysis.

The project has fulfilled the proposed objectives, first of all to generate lexicons using the socialsent framework. It has shown the potential of this technologies, with only about ten words per class, it can assign a value to every word and the results for the closest words to the seed words are good. And not only are shown the results of the sentiment and hate analyses and their differences, it is also developed the problem for offensive speech and the differences it has with hate speech.

For the evaluation section, three methods are proposed to get results of each algorithm from each problem and then be able to compare them. Concluding if an algorithm performs better than other, if it always performs better or in just in an specific domain. Results have shown that algorithms used, densifier and algorithm, when aggregating the lexicon values of the words of a text, have to improve a lot to achieve the results of the methods where logistic regression is employed.

The results are promising and indicates that this line of investigation will achieve great performance and could finally be deployed into a real environment, to detect/identify hate speech on social media platforms and protect the affected collectives.

6.3 Future work

This project has demonstrated the potential of this technologies have, but the results can be improved. A number of new tasks or new problems can be proposed for the future, such as:

- Try other embedding models, either importing other known pre-trained models or generating one of its own using the text from the dataset in question.
- Check the importance of the seed words by testing different set ot seeds and comparing results.
- Generate and evaluate lexicons over different sentiment or hate speech datasets and compare them with the ones obtained in this project.
- Generate and evaluate specific lexicons for different domains, analyzing different emotions or aspects.
- Evaluate the lexicon generated with a dataset on a different corpus. For example, evaluate the lexicon generated with movie reviews on Amazon reviews or restaurant reviews.

Bibliography

[1] 8 common examples of natural language processing and their impact on communication. https://www.tableau.com/learn/articles/natural-language-processing-examples.

[2] 'A Tsunami of Hate': The Covid-19 Hate Speech Pandemic. https://www.humanrightspulse.com/mastercontentblog/a-tsunami-of-hate-the-covid-19-hate-speech-pandemic.

[3] Article 10: Freedom of expression — Equality and Human Rights Commission. https://www.equalityhumanrights.com/en/human-rights-act/article-10-freedom-expression.

[4] Clustering Algorithms — Clustering in Machine Learning. https://developers.google.com/machine-learning/clustering/clustering-algorithms.

[5] Conda — Conda documentation. https://docs.conda.io/en/latest/.

[6] European Commission and IT Companies announce Code of Conduct on illegal online hate speech. https://ec.europa.eu/commission/presscorner/detail/en/IP_16_1937.

[7] Gensim: Topic modelling for humans. https://radimrehurek.com/gensim/.

[8] GloVe: Global Vectors for Word Representation. https://nlp.stanford.edu/projects/glove/.

[9] Google Code Archive - Long-term storage for Google Code Project Hosting. https://code.google.com/archive/p/word2vec/.

[10] Hate speech. https://dictionary.cambridge.org/dictionary/english/hate-speech.

[11] https://witanworld.com/article/2018/10/28/naturallanguageprocessing-nlp/. https://witanworld.com/article/2018/10/28/naturallanguageprocessing-nlp/.

[12] In 2016, Microsoft's Racist Chatbot Revealed the Dangers of Online Conversation - IEEE Spectrum. https://spectrum.ieee.org/tech-talk/artificial-intelligence/machine-learning/in-2016-microsofts-racist-chatbot-revealed-the-dangers-of-online-conversation.

[13] Keras: The Python deep learning API. https://keras.io/.

[14] Machine learning. https://dictionary.cambridge.org/dictionary/english/machine-learning.

[15] Natural-language-processing-training. https://mildaintrainings.com/blogs/nlp-natural-language-processing-tutorial-get-started-nlp/attachment/natural-language-processing-training-2/.

[16] Natural Language Toolkit — NLTK 3.6.2 documentation. https://www.nltk.org/.

[17] NumPy. https://numpy.org/.

[18] Origins of Linguistics. https://www.sas.upenn.edu/~haroldfs/sars238/intro/discovery.html.

[19] Pandas - Python Data Analysis Library. https://pandas.pydata.org/.

[20] Permanent suspension of @realDonaldTrump. https://blog.twitter.com/en_us/topics/company/2020/suspension.html.

[21] Project Jupyter. https://www.jupyter.org.

[22] Qué es un delito de odio - Ministerio del Interior. http://www.interior.gob.es/web/servicios-al-ciudadano/delitos-de-odio/que-es-un-delito-de-odio.

[23] Regularized logistic regression tool — SPLab. http://splab.cz/en/download/software/software-pro-regularizovanou-logistickou-regresi.

[24] Sentiment analysis. http://nlpprogress.com/english/sentiment_analysis.html.

[25] Tasks < SemEval-2019. https://alt.qcri.org/semeval2019/index.php?id=tasks.

[26] TensorFlow. https://www.tensorflow.org/.

[27] Twitter's policy on hateful conduct — Twitter Help. https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy.

[28] Voice Assistant and Smart Speaker Users 2020. https://www.emarketer.com/content/voice-assistant-and-smart-speaker-users-2020.

[29] Welcome to Python.org. https://www.python.org/.

[30] What is Linguistics? — Linguistic Society of America. https://www.linguisticsociety.org/what-linguistics.

[31] Best Definitions from Harvard, Stanford, MIT, CMU AI Research Professors — Mind Data Intelligence Consulting Services. https://minddata.org/what-is-ai-mit-stanford-harvard-cmu-Brian-Ka-Chan-AI, February 2018.

[32] Sensitivity and specificity. *Wikipedia*, May 2021. https://en.wikipedia.org/w/index.php?title=Sensitivity_and_specificity&oldid=1022253723.

[33] Kimmy YamKimmy Yam is a reporter for NBC Asian America. There were 3,800 anti-Asian racist incidents, mostly against women, in past year. https://www.nbcnews.com/news/asian-america/there-were-3-800-anti-asian-racist-incidents-mostly-against-n1261257.

[34] Oscar Araque, Ignacio Corcuera-Platas, J. Fernando Sánchez-Rada, and Carlos A. Iglesias. Enhancing deep learning sentiment analysis with ensemble techniques in social applications. *Expert Systems with Applications*, 77:236–246, 2017.

[35] Oscar Araque, Ganggao Zhu, and Carlos A. Iglesias. A semantic similarity-based perspective of affect lexicons for sentiment analysis. *Knowledge-Based Systems*, 165:346–359, 2019.

[36] Georgios Askalidis and Edward C. Malthouse. The value of online customer reviews. In *RecSys 2016 - Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys 2016 - Proceedings of the 10th ACM Conference on Recommender Systems, pages 155–158. Association for Computing Machinery, Inc, September 2016. Funding Information: We thank the Spiegel Center for Digital and Database Marketing at Northwestern University for support. Publisher Copyright: © 2016 ACM. Copyright: Copyright 2017 Elsevier B.V., All rights reserved.; 10th ACM Conference on Recommender Systems, RecSys 2016 ; Conference date: 15-09-2016 Through 19-09-2016.

[37] Facundo Bre, Juan Gimenez, and Víctor Fachinotti. Prediction of wind pressure coefficients on building surfaces using Artificial Neural Networks. *Energy and Buildings*, 158, November 2017.

[38] Jason Brownlee. What Are Word Embeddings for Text? `https://machinelearningmastery.com/what-are-word-embeddings/`, October 2017.

[39] Ignacio Corcuera-Platas. Development of a Deep Learning Based Sentiment Analysis and Evaluation Service. Master thesis, ETSI Telecomunicación, Madrid, January 2018.

[40] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the 11th International AAAI Conference on Web and Social Media*, ICWSM '17, pages 512–515, 2017.

[41] S. Deerwester, S. Dumais, T. Landauer, G. W. Fumas, and L. L. Beck. Improving information retrieval using latent semantic indexing. 1988.

[42] Jose García. Twitter suspende temporalmente la cuenta de VOX por "incumplir las reglas que prohíben las conductas de incitación al odio". `https://www.xataka.com/legislacion-y-derechos/twitter-suspende-temporalmente-cuenta-vox-incumplir-reglas-que-prohiben-conducta` January 2021.

[43] Yoav Goldberg. Neural network methods for natural language processing. *Synthesis lectures on human language technologies*, 10(1):1–309, 2017.

[44] William L. Hamilton. williamleif/socialsent. original-date: 2016-06-01T23:54:27Z.

[45] William L. Hamilton, Kevin Clark, Jure Leskovec, and Dan Jurafsky. Inducing Domain-Specific Sentiment Lexicons from Unlabeled Corpora. *arXiv:1606.02820 [cs]*, September 2016. `http://arxiv.org/abs/1606.02820`.

[46] W. Hutchins. Warren weaver and the launching of mt: brief biographical note. 2000.

[47] Janna. Major trends in NLP: A review of 20 years of ACL research. `https://towardsdatascience.com/major-trends-in-nlp-a-review-of-20-years-of-acl-research-56f5520d473`, July 2019.

[48] Adnan R. Khan June 26 and 2018. How the internet may be turning us all into radicals. `https://www.macleans.ca/society/technology/how-the-internet-may-be-turning-us-all-into-radicals/`, June 2018.

[49] Andrew L. Maas, Raymond E. Daly, Peter T. Pham, Dan Huang, Andrew Y. Ng, and Christopher Potts. Learning word vectors for sentiment analysis. In *Proceedings of the 49th Annual Meeting of the Association for Computational Linguistics: Human Language Technologies*, pages 142–150, Portland, Oregon, USA, June 2011. Association for Computational Linguistics.

[50] Kosisochukwu Madukwe, Xiaoying Gao, and Bing Xue. In Data We Trust: A Critical Analysis of Hate Speech Detection Datasets. In *Proceedings of the Fourth Workshop on Online Abuse and Harms*, pages 150–161, Online, November 2020. Association for Computational Linguistics.

[51] Mika V. Mäntylä, Daniel Graziotin, and Miikka Kuutila. The evolution of sentiment analysis—A review of research topics, venues, and top cited papers. *Computer Science Review*, 27:16–32, February 2018.

[52] Tomas Mikolov, Ilya Sutskever, Kai Chen, Greg Corrado, and Jeffrey Dean. Distributed Representations of Words and Phrases and their Compositionality. *arXiv:1310.4546 [cs, stat]*, October 2013. `http://arxiv.org/abs/1310.4546`.

[53] Tomas Mikolov, Wen-tau Yih, and Geoffrey Zweig. Linguistic regularities in continuous space word representations. In *Proceedings of the 2013 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, pages 746–751, Atlanta, Georgia, June 2013. Association for Computational Linguistics.

[54] Leo Mirani. Turns out Twitter is even more politically polarized than you thought. `https://qz.com/179158/turns-out-twitter-is-even-more-politically-polarized-than-you-thought/`.

[55] Richard Nagyfi. The differences between Artificial and Biological Neural Networks. `https://towardsdatascience.com/the-differences-between-artificial-and-biological-neural-networks-a8b46db828b7`, September 2018.

[56] 1615 L. St NW, Suite 800Washington, and DC 20036USA202-419-4300 — Main202-857-8562 — Fax202-419-4372 — Media Inquiries. Political Polarization in the American Public. `https://www.pewresearch.org/politics/2014/06/12/political-polarization-in-the-american-public/`, June 2014.

[57] Jeffrey Pennington, Richard Socher, and Christopher Manning. Glove: Global Vectors for Word Representation. In *Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pages 1532–1543, Doha, Qatar, 2014. Association for Computational Linguistics.

[58] La Redacción. El Corte Inglés rectifica y retira su polémica imagen de la campaña 'Vuelta al Cole'. https://lapublicidad.net/el-corte-ingles-rectifica-y-retira-su-polemica-imagen-de-la-campana-la-vuelta-al August 2020.

[59] Sascha Rothe, Sebastian Ebert, and Hinrich Schütze. Ultradense word embeddings by orthogonal transformation. *arXiv preprint arXiv:1602.07572*, 2016.

[60] Elizabeth Schulze. EU says Facebook, Google and Twitter are getting faster at removing hate speech online. https://www.cnbc.com/2019/02/04/facebook-google-and-twitter-are-getting-faster-at-removing-hate-speech-online-eu html, February 2019.

[61] Anshul Singh. A Beginner's Guide to Machine Learning. https://medium.com/@anshulraghav2222/a-beginners-guide-to-machine-learning-14753f433085, August 2019.

[62] Himanshu Singh. Supervised Learning Methods using Python. https://medium.com/@himanshuit3036/supervised-learning-methods-using-python-bb85b8c4e0b7, June 2018.

[63] Saishruthi Swaminathan. Logistic Regression — Detailed Overview. https://towardsdatascience.com/logistic-regression-detailed-overview-46c4da4303bc, January 2019.

[64] Andranik Tumasjan, Timm O Sprenger, Philipp G Sandner, and Isabell M Welpe. Predicting Elections with Twitter: What 140 Characters Reveal about Political Sentiment. page 8.

[65] James Vincent. Reddit reports 18 percent reduction in hateful content after banning nearly 7,000 subreddits. https://www.theverge.com/2020/8/20/21376957/reddit-hate-speech-content-policies-subreddit-bans-reduction, August 2020.

[66] Dengyong Zhou, Olivier Bousquet, Thomas N Lal, Jason Weston, and Bernhard Schölkopf. Learning with local and global consistency. In *Advances in neural information processing systems*, pages 321–328, 2004.

[67] Victor Zhou. A Simple Explanation of the Bag-of-Words Model. https://towardsdatascience.com/a-simple-explanation-of-the-bag-of-words-model-b88fc4f497 December 2019.

APPENDIX A

Software requirements

Programs have been coded with the programming language Python, in both Integrated Development Environment and jupyter notebooks. Two different Python environments have been created, one with Python 2 to run the code from the socialsent framework and other one with Python 3 for the rest of the tasks. The libraries used for each of the environments are in table A.1 and table A.2 for the Python 2 and Python 3 environment respectively.

Package	Version
python	2.7.13
numpy	1.12.1
pip	9.0.1
keras	0.3.3
nltk	3.2.4
gensim	3.8.0
tqdm	4.59.0
theano	1.0.5
matplotlib	2.0.2
scipy	0.19.1

Table A.1: Libraries and corresponding versions used in the Python 2 environment

Package	Version
python	3.8.6
numpy	1.20.1
pip	21.0.1
nltk	3.6.2
pandas	1.2.2
gsitk	0.2.3
scikit-learn	0.24.1
scipy	1.6.0

Table A.2: Libraries and corresponding versions used in the Python 3 environment

APPENDIX B

Ethical, economic, social and environmental aspects

B.1 Introduction

This appendix discusses the impact this project may have in ethical, economic, social and environmental aspects. This project is born out of the phenomenon in today's society of the rise of hate speech, so it is intended to have impact on various aspects and tries to help to identify some crimes and injustices.

B.2 Description of the relevant impacts related to the project

B.2.1 Ethical impact

The final goal to which the development of these researches aspires is to put an end to hate speech on social media platforms and on Internet in general. The decision of what is hate speech or what is not hate speech is subjective, there will be cases where someone could argument there is hate speech and some other person argue that it is not. So the ethical and legal impacts on the decision of what is hate speech and remove or restrict some messages has broader implications. Article 10 of the Human Rights Act [3], declares that everyone has the right to freedom of expression, by any means either books, articles, television and on social media. So the actions to restrict some information on social media platforms have to be sufficiently justified because applying the sanction unfairly goes against a foundation of human rights. Article 10 states that the freedom of speech is a right but it also entails responsibilities and rights of other people has to be respected, therefore a public authority can restrict this right or apply penalties to protect health, morals or rights of others.

This responsibility is carried out by the governments of each state, and each one can have different interpretations. For example in Spain exists a hate crime [22], while in United States hate speech is protected by the First Amendment and it is only criminalized when it incites to criminal activity.

In addition to that, social media platforms like Twitter has a code of conduct [27] which does not allow violence or attacks to some collectives based on gender, race, sexual orientation, etc. So agreeing to the terms of the platform, hate speech messages can not be spread.

In this aspect, although it is a difficult task to have a common definition of hate speech, the removal or restrictions of certain hate speech messages if they are justified, are subject to legal and ethical grounds.

B.2.2 Economic aspect

The motivation of the identification of hate speech messages primarily seeks the well-being of people and society, so there are no clear economic aspects. It is expected that if social media platforms and Internet web sites are free of hate speech content, it could make companies invest more on this platforms, like on advertisement, because they are a safe place, and they are not going to be related with controversial content.

B.2.3 Social aspect

The impact of hate speech on social media platforms is clear, with the spread of hate messages without control the last years, some comments like the mentioned in chapter 1 by important political figures, can have an impact on people's real lives. There is a need to battle racism, sexism, discrimination by sexual orientation, ethnicity, etc., they are present in many ways in some people's daily lives, whether at home, on television, in politics or where this project is about, in social media specially. Identifying and removing some hate content from Internet is a step forward for the integration of discriminated groups into society.

B.2.4 Environmental aspect

The analysis of hate speech or sentiment analysis is unclear that could have a direct impact on an environmental aspect. It is centered on the ethical and social aspect because it is more about language and the consequences on collectives.

B.3 Conclusions

As it has been described, the efforts made on identifying hate speech are to solve some social and ethical problems that have not gone unnoticed in recent years, and have exposed a problem that existed before but has been reinforced with the rise of social media platforms and the change in how people are informed. With the correct use of this technologies there are no negative impacts but it has to ensure to comply with the human right, freedom of expression.

APPENDIX C

Economic budget

Table C.1 shows the economic budget of the project.

Cost of labor (Direct cost)	Hours	Price/Hour	Total
	300	30 €	9000 €

Material costs (Direct cost)

	Price	Use in months	Amortization(years)	Total
Personal computer	1200 €	5	4	126 €
Remote server service	9000 €	3	1	3600 €

Total costs

Cost of labor			9000 €
Material costs			3726 €
General expenses (Indirect cost)	15%	Over Direct cost	1908.9 €
Industrial benefits	6%	Over direct and indirect costs	878 €
VAT	21%		3257.7 €
Total budget			18770.6 €

Table C.1: Economic budget of the project