UNIVERSITAT POLITÈCNICA DE CATALUNYA

MASTER THESIS

# Sentiment analysis on short Spanish and Catalan texts using contextual word embeddings

*Author:*
Eudald Cumalat Puig

*Supervisor:*
Marta Ruiz Costa-Jussà
*Co-Supervisor:*
M. Asunción Moreno Bilbao

*A thesis submitted in fulfillment of the requirements*
*for the Master's degree in Telecommunications Engineering*

*in the*

TALP: Centre de Tecnologies i Aplicacions del Llenguatge i la Parla
Departament de Ciències de la Computació

July, 2020

Abstract of thesis entitled

# Sentiment analysis on short Spanish and Catalan texts using contextual word embeddings

Submitted by

**Eudald Cumalat Puig**

for the Master's degree in Telecommunications Engineering

at the Universitat Politècnica de Catalunya

in July, 2020

This project explores several pipelines to accurately perform text classification in a very specific task, which consists on classifying Catalan and Spanish short texts with very informal language by kids and teenagers. The classification categories are the following abusive topics: aggression and violence, substances, sexuality and disorders (anxiety, depression or distress). We count on a small, highly unbalanced, supervised dataset of approximately 200.000 examples. We first added a robust preprocessing to our database and then we built and tested several pipelines, exploring vectorization (TF-IDF, Doc2Vec) and AI classification techniques (RF, SVM, BERT). The best one was the multilingual version of BERT with our proposed preprocessing without stemming. Another pipeline achieved similar results (surpassing BERT in one of the categories) with a much faster computing time, and that is using a BERT model for extracting embeddings of our short texts and classifying them using SVM.

# Sentiment analysis on short Spanish and Catalan texts using contextual word embeddings

by

**Eudald Cumalat Puig**
*UPC*

A Thesis Submitted in Partial Fulfilment
of the Requirements for the Master's
degree in Telecommunications Engineering

at

Universitat Politècnica de Catalunya
July, 2020

To Laia, always caring for me and helping me persevere.

# *Acknowledgements*

First of all, I want to thank my tutors, Marta Ruiz Costa-Jussà and M. Asunción Moreno Bilbao, for guiding, teaching and supporting me through the development of this project. Second, I want to thank Carlos Escolano Peinado for the advice and ideas provided, but also for his patience. Last but not least, I want to thank my family and friends, who were always there supporting me.

Eudald Cumalat Puig
Universitat Politècnica de Catalunya
July, 2020

# Contents

# List of Figures

# List of Tables

# List of Abbreviations

| | |
|---|---|
| **BERT** | **B**idirectional **E**ncoder **R**epresentations from **T**ransformers |
| **SVM** | **S**upport **V**ector **M**achines |
| **RF** | **R**andom **F**orest |
| **TF-IDF** | **T**erm **F**requency - **I**nverse **D**omain **F**requency |
| **ML** | **M**achine **L**earning |
| **DL** | **D**eep **L**earning |
| **MLM** | **M**asked **L**anguage **M**odel |
| **NSP** | **N**ext **S**entence **P**rediction |
| **POS** | **P**art-**O**f-**S**peech |

# Chapter 1

# Introduction

This document contains all the information related to the theoretical and practical development of the project, which consists of developing a topic detector to classify the sentiment from Spanish and Catalan short texts.

## 1.1 Motivation

Abusive behaviour is, unfortunately, a very common conduct in nowadays social networks interactions. Some people think that putting a virtual mask over their identity automatically gives them carte blanche to say mean things to other people through the internet. This behaviour can cause very serious problems to the receiver of the messages, as well as push him/her to do illegal or health-threatening things.

Having all of this in mind, I found this project very interesting, and at the same time necessary. Having a tool that could automatically detect if there is some kind of abuse in a text, and then apply it to the social networks could be an extra security measure to help fight this kind of behaviour. Prematurely detecting abusive behaviour could lead to the ability of helping the victim when no damage has been done yet, or even preventing the interaction to happen by warning the victim before receiving those messages.

Aside from the ethical motivation, the ability to train a program that is able to detect whether a text contains a concrete kind of emotion or not interested me the most. In verbal communication it is fairly easy to detect what emotion the person is feeling, but in written communication there is no body motion, pitch, voice volume or intonation to help us understand what the sender is trying to say. For me, this made the problem even more interesting.

Finally, the fact that we would be working with a small database created in my university, alongside all the other facts mentioned in this section, ended up convincing me to join this project.

## 1.2 Goals

The aim of this project is to detect if a text contains several types of abusive behaviour. We have worked with four different characteristics, apart from the classification of whether it contains abuse in general or not:

- *Violence*: includes violence and aggression.
- *Substance*: posts that contain use of alcohol, drugs, tobacco, etc.
- *Sex*.
- *Disorders*: includes anxiety, distress or depression.

Taking into account this information, the main goals to achieve the aim of this project are:

- Reproduce the baseline results from [1].
- Incorporate catalan short texts to the experiments.
- Improve the baseline system.
- Study and test several ways to convert our texts to embeddings, in order to further use them for the classification task.
- Study and test different pipelines (combinations of preprocessing and classifiers).

## 1.3 Problems

We have had very few problems during the course of this project. The most impeding one was the elevated training time of the experiments, some of them lasting more than three days to finish. Another problem was the fact that we were working with a small dataset, which left us with very little room to maneuver.

## 1.4 Contribution

The contributions of this project are the following:

- Built a robust preprocessing for our database.
- Tested and ajusted three pipelines for general or multi-label classification.
- Built two new pipelines also for general or multi-label classification.
- Discussed a comparison for the results of all experiments.

Also, this project has been submitted as a paper in the COLING'2020 International Conference on Computational Linguistics[1]. The paper is added as an annex to this thesis.

## 1.5 Project budget

As this projects is about software, there are no material costs aside from the cost of the computer I have been working on or the maintenance costs of the server. Other costs include the salaries of an engineer and two supervisors.

To calculate the salaries we have to first calculate the number of hours of this project. The number of ECTS is 30, and each one requires 30 hours of work, reaching the number of 900 hours. As for the supervisors time, I estimate that they have dedicated approximately 40 hours to the project. The following table contains all the salary details:

| **Role** | **Price** | **Hours** | **Cost** |
|---|---|---|---|
| Software engineer | 15€/h | 900 | 13500€ |
| Supervisor 1 | 50€/h | 40 | 2000€ |
| Supervisor 2 | 50€/h | 40 | 2000€ |
| **Total** | | | **17500€** |

**Table 1.1:** Salaries.

For the material costs I can only count both my computers (one laptop and one desktop computer), which approximately add up to 2000€. Taking into account an amortization of 20%, and also the fact that the project has lasted 6 months, the material cost is **200€**. There are no costs associated to the server because it is located at the university, and students doing projects can use it for free.

To sum up, the final cost of the project is **17700€**.

[1] https://coling2020.org/

## 1.6 Social impact

The social impact of this project could be huge depending on how it is used. If it were applied to the social networks, most of the abusive interactions could be rapidly targeted, reducing them drastically. This would make the social networks a much more safer place.

For other possible abusive material sources, a program or browser plugin could be developed so that parents can discretely protect their children from potentially dangerous sites.

## 1.7 Methodology

The research process has been constant during this project, as I have been experimenting with complex software and also wanted to use the best option for each case. Some ideas for experiments have come to us when investigating how to run another experiment

For the implementation, I have either taken and adapted scripts from the baseline into my casuistry or designed and written my own scripts. After developing them, I have run them in the university server.

This project has been monitored mainly via scheduled online meetings and mailing.

# Chapter 2

# Context and State of the Art

## 2.1 Machine Learning

ML is an application of artificial intelligence that provides algorithms the ability to automatically learn and improve through experience. These algorithms use sample data (training data) to build a mathematical model that will be able to make predictions or decisions without being explicitly programmed to do so.

There are three types of learning:

- *Supervised*: You feed the system with labeled data so that it can train the model, check its accuracy and find an optimal training spot. Some applications are recommendation systems and classification problems.
- *Unsupervised*: The system is fed with unlabeled data, and focus on grouping the unsorted data according to similarities and differences. Some applications are chatbots, facial recognition systems and self-driving cars.
- *Reinforced*: It is the training of a model to make a sequence of decisions, or in other words, the model learns to achieve a goal in an uncertain, complex situation. It does not need labelled input and output pairs because it employs a system of rewards and penalties to compel the computer to solve a problem by itself. Some applications are algorithms that can play a game in a superhuman level, algorithms for autonomous cars or prosthesis, among others.

For the purpose of this thesis, supervised learning is used.

## 2.2 Deep Learning

DL is a subfield of ML based on algorithms that try to imitate the structure and functioning of the brain. These algorithms are called artificial neural networks

(ANN), and are based on feature learning, which is a set of techniques that allows a system to automatically learn and distinguish the characteristics needed for making a decision or classification from huge amounts of raw data (voice recordings, photos, texts, etc).

There exist several architectures based on neural networks, such as deep belief networks, convolutional neural networks, recurrent neural networks or deep neural networks. The structure of a very basic deep neural network is shown in the following figure:

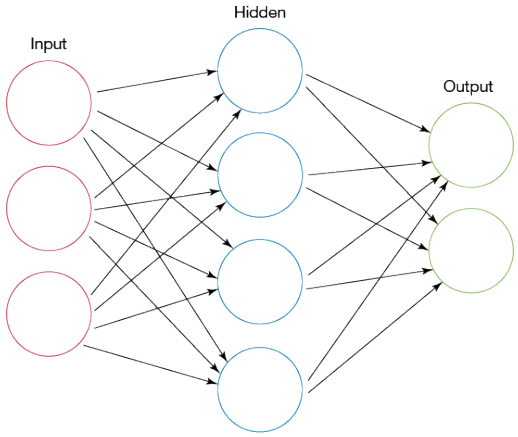


**Figure 2.1:** Simple deep neural network structure.

Every deep neural network has layers, which are basically containers of neurons. All the neurons perform the same function: calculate the weighted sum of inputs and weights, add the bias and execute an activation function. The neurons are grouped in three types of layers:

- *Input layer*: It is the place where you introduce the samples. For that reason, the length of this layer has to match the length of the samples you are going to feed the system. This layer takes in the inputs, performs the calculations via its neurons and then the output is transmitted onto the subsequent layers.
- *Hidden layers*: There can be many of these layers, which make them a fine-tuning parameter. They reside between the input and output layers, and as its name implies, they are not visible to the external systems and are private to the network. Also, the larger the number of hidden layers in a neural network, the longer it will take for the network to produce the output but also the more complex problems the network will be able to solve.

- *Output layer*: It is the layer that gives us the result given the input. Its length is determined by the number of possible results.

## 2.3 Sentiment analysis

Sentiment analysis is the interpretation and classification of emotions within text data using text analysis techniques. In the field of NLP, this is one of the most difficult parts to analyse due to the subjective opinion of each person, which highly depends on culture, education or religion.

Our project falls into the topic analysis category, which is a NLP technique that allows a system to automatically extract meaning from texts by identifying recurrent themes or topics. A good description of the state-of-the-art in this category can be found in [8].

## 2.4 Vectorization techniques

To vectorize is to convert string data into numerical data conserving the features that we are interested in. In our case, we want to be able to analyse the sentimental data of a short text. For this purpose, we use several vectorization methods, described below.

### 2.4.1 TF-IDF

TF-IDF [10] is a numerical statistic that is used to evaluate how important a word is to a text in a dataset. It has two terms:

- *Term Frequency*: How frequently the word appears in the corpus.
- *Inverse Document Frequency*: Used for finding out the importance of the word. It uses the fact that less frequent words can be more informative and important.

The numerical result is computed as the product of the two terms. It increases proportionally to the number of times a word appears in the document and is inversely proportional to the frequency of the word in the corpus, which compensates the fact that some words appear more frequently in general, like the word "*the*", but are not important for this task.

### 2.4.2 Doc2Vec

This concept was presented by Le and Mikolov [5] as an extension to Word2Vec [7]. It is aimed to create a numeric representation of a text document, regardless of its length, that can be used for many purposes, such as document retrieval, web search, topic modeling or spam filtering.

While Word2Vec works on the intuition that the word embedding should be good enough to predict the surrounding words, the intuition of Doc2Vec is that the document representation should be good enough to predict the words in the document. The principle behind Doc2Vec is to train the document vector while also training the word vectors. At the end of training, it holds a numeric representation of the document.

## 2.5 Classifiers

### 2.5.1 Support Vector Machines

An SVM [11] training algorithm builds a model that assigns the training samples to one of two classes. The gap between both categories is called a hyperplane. All the new inputs are then mapped into that same space and predicted to belong to a category based on the side of the gap on which they fall.

As most of the real-life problems can not be classified using a linear classifier, SVMs can efficiently perform a non-linear classification using the so-called kernel trick, which maps the inputs into a high-dimensional feature space, making it able to use a plane to separate the classes.

Using SVM as the classifier has many advantages:

- It works relatively well when the margin of separation between classes is clear.
- It is more effective in high dimensional spaces, even more when the number of dimensions is greater than the number of samples.
- It is memory efficient.

But also disadvantages:

- It is not suitable for large datasets.
- It does not perform very well when the dataset has a lot of noise (classes overlapping).

- As the SVM classifier works by putting data points above and below the hyperplane, there is no probabilistic explanation for the classification.

### 2.5.2 Random Forest

RF is a tool that operates by constructing a multitude of decision trees at training time and outputs the class that is the mode of the classes (by majority voting).

The main goal is to split all the information in several subsections in order to analyze them as deep as possible.

RF has many advantages:

- It prevents overfitting because of the information split.
- It reduces the variance, and therefore improves the accuracy.
- It handles non-linear parameters efficiently, as well as missing values.

A drawback of RF is its high complexity due to the number of trees that this method creates. It therefore uses more computational resources and the training time is way longer than with other methods.

## 2.6 BERT

Bidirectional Encoder Representations from Transformers [2] is a model that applies the bidirectional training of Transformer (an attention mechanism that learns contextual relations between words in a text) to language modelling. As opposed to directional models (which read the input sequentially), the encoder reads the entire sequence at once without an specific direction, which allows the model to learn the context of a word based on its entire surroundings. As input it accepts a sequence of tokens, which are first embedded into vectors and then processed in the neural network, and as output it gives a sequence of vectors corresponding to the input tokens.

BERT uses two training strategies: Masked Language Model (MLM) and Next Sentence Prediction (NSP).

**MLM** consists on replacing approximately 15% of the words of each sentence with "[MASK]" before feeding them to BERT. Then, the model tries to predict what those words were based on the context provided by the non-masked words. The process follows the following steps (illustrated in figure 2.2):

1. Adds a classification layer on top of the encoder output.

2. Transforms the output vectors to words multiplying them by the embedding matrix.
3. Calculates the probability of each word inside the vocabulary using the softmax function.
4. The loss function only takes into consideration the prediction of the masked values, which makes the model converge slower than a directional model.

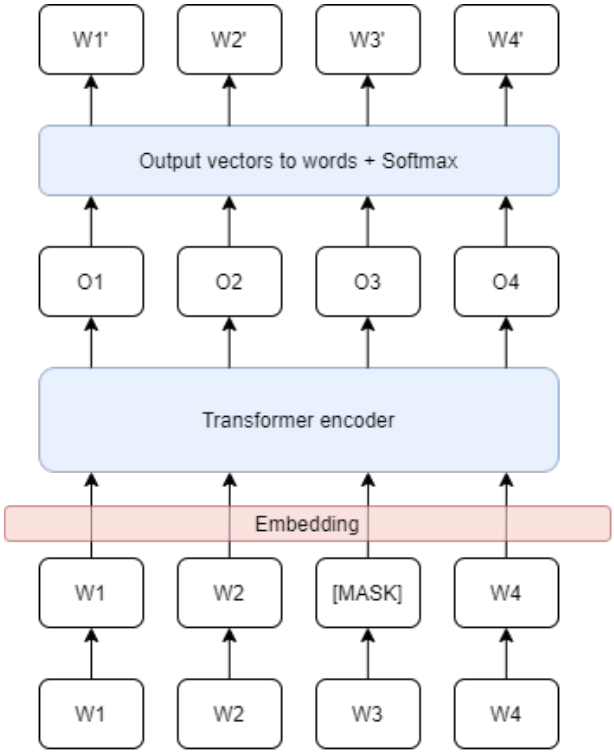


**Figure 2.2:** Diagram explaining the MLM process, inspired in one of the figures from [4].

In **NSP**, during training, the model receives a pair of sentences as input and learns to predict if the second is the one that follows the first one. When pairing, half of the inputs are grouped in pairs where both sentences are consecutive and the other half are grouped randomly, so that both sentences are disconnected. Before entering the model, the pairs are prepared as shown in figure 2.3. The process is the following:

1. A [CLS] token is inserted to indicate the beginning of the first sentence. [SEP] is inserted at the end of each sentence.
2. Another embedding is added to indicate if the token is from the first or the second sentence.

3. The last embedding is added to indicate the position of the token in the sequence.

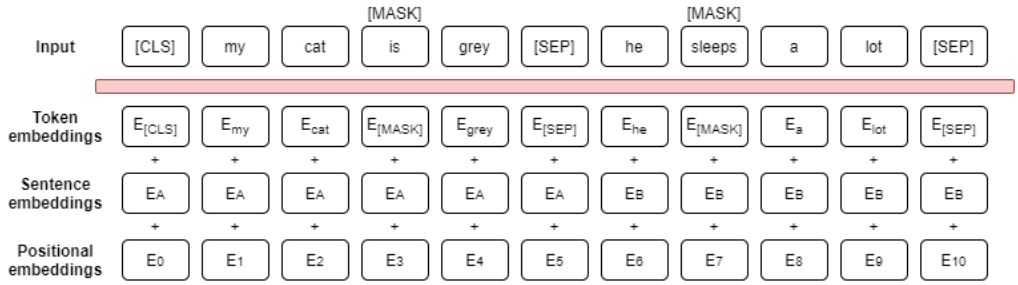


**Figure 2.3:** Diagram explaining the NSP process, inspired in one of the figures from [4].

Now that we know how the training process works, the only thing left to explain is how to use BERT for a task like ours. The process is called fine-tuning, and for tasks such as sentiment analysis it works very similar to NSP. We just add a classification layer (fully connected) on top of the Transformer output for the [CLS] token to perform softmax.

As a final note, BERT's bidirectional training has an advantage but also a disadvantage. It converges slower than left-to-right approaches but outperforms them in terms of accuracy.

# Chapter 3

# Project Development

This chapter is aimed to cover all the technical details of the work done in this project.

## 3.1 Database

The database [9] includes short texts in Catalan and Spanish manually tagged into 7 different categories: aggression, violence, anxiety, depression, distress, sex and substance. Each category is tagged with a value between 1 and 5, 1 being neutral (no abuse) and 5 begin the highest quantity of abuse. In order to simplify the task, we did two arrangements. First, we reduced the number of categories to 4, grouping aggression and violence into a single category, and also anxiety, depression and distress under another category. Second, we only considered whether the text fits one of the categories (value between 2 and 5) or not (the value is 1), basically turning our task into one of topic detection, or in other words, if there is abuse in the text or not. Table 3.1 details the number of posts under each category, and also the number of posts in each language. Note that one post can be tagged within multiple categories.

| **Catalan** | **Spanish** | **Violence** | **Substance** | **Sex** | **Disorders** | **No abuse** |
|---|---|---|---|---|---|---|
| 109141 | 111719 | 42227 | 6209 | 29621 | 43615 | 121747 |

**Table 3.1:** Total number of posts by language and category.

For the experiments, we randomly divided the database into two sets, train (80%) and test (20%), and since this database is highly unbalanced, we evaluated our results using a modified f1 score [3] using true positives, false negatives

and false positives as follows:

$$F1_{tp,fp,fn} = \frac{2TP}{2TP + FP + FN}$$

## 3.2 Baseline

This project is the continuation of [1]. As a baseline, we considered to use the final results of that project but with Catalan short texts added and also using only two sets (train and test). All the steps taken and new experiments performed over this baseline are described in the following sections.

## 3.3 Adaptations

In this section I describe all the adaptations that have been done to the database in order to improve the baseline results.

### 3.3.1 Changing threshold of abuse

The first improvement that we got was when we changed the threshold of abuse. In the baseline project, they considered that 1-2 was not abusive and 3-5 contained abuse, but this left the system with very few abusive samples. We tried changing it so that 1 was considered not abusive and 2-5 was considered abusive. This change improved drastically the results, while it also turned our task into one of topic detection.

### 3.3.2 Adding a robust preprocessing

The next step was to properly prepare the data before using it. Short texts gathered from social networks or other websites contain lots of misspelled words, emoticons and other unimportant stuff. We have built a preprocessing function that solves most of this problems, and consists on the following process:

- *Tokenizer*: we used the *word_tokenize()* function from the *nltk.tokenize* package.
- *Deleting punctuation marks*: using simple python string manipulation.
- *Lower case*: using simple python string manipulation.
- *Unicode to string*: convert all the weird characters or emojis to words.

- *Normalization*: using simple python string manipulation. It eliminates characters that are repeated more than two times, as many of the posts contained words like "wooooooords".
- *Deleting stopwords*: using simple python string manipulation and also using the *stopwords* library from *nltk.corpus*.
- *Stemming*: using the *SnowballStemmer* from *nltk.stem*. It is the process of reducing a word to its root form, and reduces greatly the number of unique words in the dataset vocabulary. This step was avoided for the BERT experiments because most of the times it removes the context that the words offer, and BERT needs it.

Table 3.2 contains an example of the preprocessing function applied to a short Spanish text.

| **Action** | **Sentence** |
|---|---|
| Original | El chico me manda cositas muy lindassss y tiernas y yo menos inspiracion que una piedra. |
| Tok. | El chico me manda cositas muy lindassss y tiernas y yo menos inspiracion que una piedra . |
| No-punct | El chico me manda cositas muy lindassss y tiernas y yo menos inspiracion que una piedra |
| Lowercase | el chico me manda cositas muy lindassss y tiernas y yo menos inspiracion que una piedra |
| Normaliz. | el chico me manda cositas muy lindas y tiernas y yo menos inspiracion que una piedra |
| No-stopwrd | chico manda cositas lindas tiernas menos inspiracion piedra |
| Stemming | chic mand cosit lind tiern men inspiracion piedr |

**Table 3.2:** Preprocessing example.

### 3.3.3 Tagging

We tried to improve more the results by manually POS tagging the less frequent words. As many important words are also a bit infrequent, we used a list of highly abusive words to prevent this tagging from deleting them. Table 3.3 shows an example where two words are tagged.

| **Before tagging** | El músculo esternocleidomastoideo está situado en el cuello. |
|---|---|
| **After tagging** | El <unk> <unk> está situado en el cuello. |

**Table 3.3:** POS tagging example.

## 3.4 Pipelines

This section contains all the technical details, as well as model parameters, for the experiments in this project. Figure 3.1 sums up all of them.

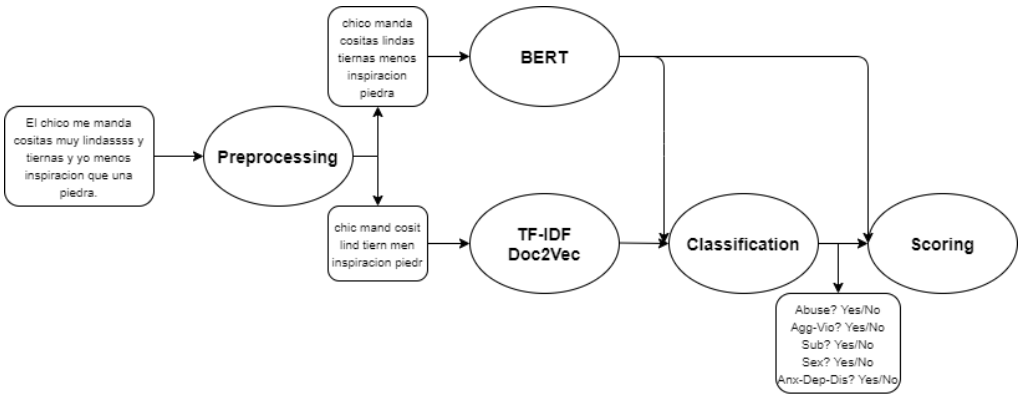


**Figure 3.1:** Block diagram of the experiments.

### 3.4.1 TF-IDF - SVM

For the TF-IDF vectorization, we used the scikit-learn toolkit[1], in particular the *TfidfVectorizer()* function with its default parameters. For the classification, we used the *LinearSVC()* function, also from the scikit-learn toolkit.

### 3.4.2 TF-IDF - RF

We have applied the same TF-IDF vectorization as in 3.4.1, and for the classification, we worked with the *RandomForestClassifier()* function from scikit-learn with 10 estimators.

### 3.4.3 Doc2Vec - SVM

To extract the Doc2Vec embeddings we used the gensim library[2]. During a tuning process where we tried with lots of different values, the parameters that gave the best results were a vector size of 200 and 70 epochs. Also, POS tagging

[1]https://scikit-learn.org/stable/
[2]https://radimrehurek.com/gensim/models/doc2vec.html

was used before extracting the embeddings. For the classification, we used the SVM explained in 3.4.1.

### 3.4.4 BERT

We used one of the models that can be found in their github page[3] as the pre-trained model, and fine-tuned it using our dataset. The model is "BERT-Base, Multilingual Uncased" with 102 languages, 12-layer, 768-hidden, 12-head and 110M parameters, which is the default configuration.

### 3.4.5 BERT - SVM

Using the best model that we fine-tuned, and also using the huggingface sentence_transformers library[4], we extracted the BERT embeddings of our dataset and then used the SVM explained in 3.4.1 to do the classification.

[3]https://github.com/google-research/BERT/blob/master/multilingual.md
[4]https://github.com/huggingface/transformers

# Chapter 4

# Results

This chapter contains all the results of this project. It is divided in two sections, the first one to see the improvements of each step taken over the baseline (section 4.1), and the second one to compare the final and best results of each experiment (section 4.2). The categories of the tables are explained below:

- *Violence*: includes the posts tagged with violence or aggression.
- *Substance*: includes the posts tagged with substance.
- *Sex*: includes the posts tagged with sex.
- *Disorders*: includes the posts tagged with anxiety, distress or depression.
- *General*: also called binary classification. Only takes into account whether the post fits in any of the categories or not.

## 4.1 Data preprocessing

This section presents the results of the process explained in section 3.3, both with the setups of TF-IDF + SVM (table 4.1) and TF-IDF + RF (table 4.2).

| Step | violence | substance | sex | disorders | general |
|---|---|---|---|---|---|
| Baseline | 11.11 | 3.00 | 2.65 | 36.30 | 49.72 |
| Turn to topic detection | **20.21** | 3.40 | **9.16** | **46.82** | 57.14 |
| Robust preprocessing | 19.64 | **3.46** | 8.52 | 46.00 | **70.91** |
| POS tagging | 19.18 | 3.44 | 8.17 | 45.55 | 70.72 |

**Table 4.1:** Comparison between all the steps taken over the baseline, using the **SVM** classifier.

| Step | violence | substance | sex | disorders | general |
|---|---|---|---|---|---|
| Baseline | 1.43 | 1.84 | 1.00 | 3.34 | 15.23 |
| Turn to topic detection | 6.84 | 2.82 | **8.76** | 10.08 | 46.07 |
| Robust preprocessing | **8.98** | 3.51 | 8.20 | **11.70** | 46.96 |
| POS tagging | 6.85 | **4.30** | 7.46 | 8.47 | **47.27** |

**Table 4.2:** Comparison between all the steps taken over the baseline, using the **RF** classifier.

As it can be appreciated in both tables above, when we changed from abuse level detection to topic detection, the results substantially increased. Adding a robust preprocessing before vectorization and classification did not make a very big difference in the individual categories (violence, substance, sex or disorders), but in the case of the binary classification the improvement was significant.

For the final experiments, we only used the POS tagging technique for the Doc2Vec pipeline, because it is very time consuming to process all the texts from the database and tag the targeted words. Doc2Vec was the only experiment that benefited from it and actually improved its results.

## 4.2 Pipelines

This section includes the results of the experiments explained in section 3.4, compared also with the Majority Voting (MV) system.

| Experiment | violence | substance | sex | disorders | general |
|---|---|---|---|---|---|
| MV | 32.23 | 5.16 | 23.49 | 33.19 | 61.95 |
| TFIDF-RF | 8.98 | 3.51 | 8.20 | 11.70 | 46.96 |
| TFIDF-SVM | 19.64 | 3.46 | 8.52 | 46.00 | 70.91 |
| Doc2Vec-SVM | 37.31 | 50.40 | 37.32 | 31.36 | 68.86 |
| BERT | **65.20** | 73.45 | **66.02** | **50.74** | **74.93** |
| BERT-SVM | 64.38 | **73.83** | 64.93 | 47.20 | 74.82 |

**Table 4.3:** Classification accuracy measured by the modified f1 score for unbalanced datasets.

Improvements vary depending on the techniques used, and some of them are consistent in all classification tasks, while others are not. Using doc2vec as opposed to tf-idf as vectorization improves on 3 out of 5 tasks, the ones that

have the fewer samples to work with. On the other hand, using BERT increases consistently in all tasks compared to using tf-idf or doc2vec with SVM or RF. Finally, globally, it is better to use a default BERT classifier token with a fully connected layer to perform softmax than using this token as input to a SVM classifier. However, using the BERT tokens as input to a SVM classifier could also be an interesting option if you need fast results, as you only need to load any pretrained model that is suitable for your task, extract the embeddings and do the classification, which is a much faster process than fine-tuning your own BERT model.

# Chapter 5

# Conclusions and future work

During the course of the project, all the goals detailed in section 1.2 have been achieved. In addition to building a solid preprocessing for the database (which made the baseline results improve substantially), I tested three other systems: Doc2Vec+SVM, BERT fine-tuning and BERT embeddings+SVM.

The Doc2Vec+SVM option presented huge improvements in the categories with less samples. For example, in the substance category, the result improved a 1357% compared to TFIDF+SVM. However, both in the disorders and the general categories, the results worsened a bit.

The BERT fine-tuning option was very time consuming, but got the best overall results of all the experiments, reaching the highest f1 score in the general category (74.93).

The last experiment, extracting the BERT embeddings of our dataset and then using an SVM for classification surpassed the BERT fine-tuning experiment in the substance category, reaching an f1 score of 73.83. The results of the other categories are highly similar to the BERT fine-tuning ones, and this makes this experiment a very promising one because it was not nearly as time consuming as the other BERT one.

Talking about classification methods, I have mostly tested SVMs as opposed to other fancy Neural Networks classifiers such as non-linear layers, recurrent or convolutional, because these ones were proven without success in previous works [6].

To sum up, this project provides several different options that can perform a general classification even when working with a limited database. For specific categories we only managed to obtain similar results in the two BERT experiments, but that could be because of the limited samples that we had. The main

contribution of this project is the last experiment, where I propose a system that works very well for our task.

For future work, one possible option is to try to improve the specific category results. Other possible tasks for future projects could be to quantitatively evaluate the abuse detected, try to find another classification method that makes the BERT embeddings experiment even better, or even create a program that automatically gathers random texts from social networks and labels them, which would also be useful to enlarge the existing database.

# Bibliography


[1] M. R. Costa-jussà, E. González, A. Moreno, and E. Cumalat. "Abusive language in Spanish children and young teenager's conversations: data preparation and short text classification with contextual word embeddings". In: *Proceedings of The 12th Language Resources and Evaluation Conference*. Marseille, France: European Language Resources Association, May 2020, pp. 1533–1537. URL: https://www.aclweb.org/anthology/2020.lrec-1.191.

[2] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. "Bert: Pre-training of deep bidirectional transformers for language understanding". In: *arXiv preprint arXiv:1810.04805* (2018).

[3] G. Forman and M. Scholz. "Apples-to-Apples in Cross-Validation Studies: Pitfalls in Classifier Performance Measurement". In: *SIGKDD Explor. Newsl.* 12.1 (Nov. 2010), pp. 49–57. ISSN: 1931-0145. DOI: 10.1145/1882471.1882479. URL: https://doi.org/10.1145/1882471.1882479.

[4] R. Horev. *BERT Explained: State of the art language model for NLP*. https://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270.

[5] Q. Le and T. Mikolov. "Distributed Representations of Sentences and Documents". In: *Proceedings of the 31st International Conference on International Conference on Machine Learning - Volume 32*. ICML'14. Beijing, China: JMLR.org, 2014, II–1188–II–1196.

[6] G. Ma. "Tweets Classification with BERT in the Field of Disaster Management". In: *Stanford Report*. 2019.

[7] T. Mikolov, I. Sutskever, K. Chen, G. Corrado, and J. Dean. "Distributed Representations of Words and Phrases and Their Compositionality". In: *Proceedings of the 26th International Conference on Neural Information Processing Systems - Volume 2*. NIPS'13. Lake Tahoe, Nevada: Curran Associates Inc., 2013, pp. 3111–3119.

[8] S. M. Mohammad and S. Kiritchenko. "Using Hashtags to Capture Fine Emotion Categories from Tweets". In: *Computational Intelligence* 31.2 (2015), pp. 301–326. DOI: 10.1111/coin.12024. eprint: https://onlinelibrary.

wiley.com/doi/pdf/10.1111/coin.12024. URL: https://onlinelibrary.wiley.com/doi/abs/10.1111/coin.12024.

[9] A. Moreno, A. Bonafonte, I. Jauk, L. Tarrés, and V. Pereira. "Corpus for Cyberbullying Prevention". In: *Proc. IberSPEECH*. 2018, pp. 170–171.

[10] "TF–IDF". In: *Encyclopedia of Machine Learning*. Ed. by C. Sammut and G. I. Webb. Boston, MA: Springer US, 2010, pp. 986–987. ISBN: 978-0-387-30164-8. DOI: 10.1007/978-0-387-30164-8_832. URL: https://doi.org/10.1007/978-0-387-30164-8_832.

[11] C. Yin, J. Xiang, H. Zhang, J. Wang, Z. Yin, and J. Kim. "A New SVM Method for Short Text Classification Based on Semi-Supervised Learning". In: *2015 4th International Conference on Advanced Information Technology and Sensor Application (AITS)*. 2015, pp. 100–103.