Detecting Offensiveness in Social Network Comments

Marta Navarrón García and Isabel Segura Bedmar

Computer Science Department
University Carlos III of Madrid
Avenida de la Universidad 30, 28911, Leganés, Madrid, Spain
martanavarron@gmail.com
isegura@inf.uc3m.es

Abstract. Social media undoubtedly has a significant influence on our lives. Although there are exists many advantages, there are also some disadvantages of social media on society, particularly youth. A very large number of social media users are subjected to different types of abuse (such as harassment, racism, personal attacks) everyday. The main goal of MeOffendEs@IberLEF 2021 is to promote research on the analysis of offensive language in social networks for Spanish. This paper describes our participation in the shared task of MeOffendEs@IberLEF 2021 [40]. We have explored different deep learning models such as Long-Short Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT) and also traditional Machine Learning models as Logistic Regression or Support Vector Machine (SVM), among others, to classify the comments (written in Spanish) into the four classes defined in the OffendEs corpus. The results of our experiments show that BERT obtains the best results among all of our models.

Keywords: Multi-Class Text Classification · Machine Learning · Deep Learning · Sentiment Analysis · Long-Short Term Memory · Bidirectional Encoder Representations from Transformers.

1 Introduction

In the last few years, social networks has become a way of life for many people. The people use them to express themselves, make themselves known, advertise, or simply socialise with other people, becoming a tool where there are always opinions and comments to the publications that are made on these platforms. But although it is a way of expression, there are always comments that can become offensive to a group of people or to a specific person or user, becoming a tool of threat which can produce long-term harm to victims.

IberLEF 2021, September 2021, Málaga, Spain.

Among them, YouTube, Instagram and Twitter are ones of the most famous social network having millions of active users around the world. In the case of Twitter, it allows the user to send or receive small posts called tweets. Tweets are comments, mostly sentences which are not more than 280 characters in which a user posts his opinion or comments on a particular topic. Moreover, the post can include images, videos, links or references to other users. In the case of YouTube, it is a website dedicated to sharing videos, where users can comment and share different opinions. Instagram is also a social network whose main function is to share photos and short videos with other users, who can also comment them. Although these social networks already have some measures in place to avoid inappropriate comments or images that may be caused harm to other users, most times they are neither very robust nor very fast to detect all these comments.

There are studies in the field of social networks in which NLP is used to analyse the behaviour of different user profiles or opinions, as well as the detection of user behaviour or trends. For example, there are studies in which it is possible to observe and predict the favorability of users with a political group based on the comments [33], also there are others related to the field of mental health, in which it is possible to detect the level of depression based on comments on Twitter [31], and many others.

Thus, NLP can be used to analyse social media. The goal of this work is to explore different NLP and machine learning techniques to detect and classify the offensiveness that a tweet or a comment could have. This task can be viewed as a task of sentiment analysis, which is the process of detecting polarity, feelings or even intentions in texts.

This work describes our participation in the shared task of MeOffendEs IberLEF 2021 [40], which aims the analysis of offensive language in social networks for Spanish. Although the task has four subtasks, where different scenarios are proposed, we have only participated on the first task, where the goal is to classify the comments (written in Spanish) into the four classes defined in the OffendEs corpus. We explore different deep learning models such as Long-Short Term Memory (LSTM) [23], Bidirectional Encoder Representations from Transformers (BERT) [17], and also traditional Machine Learning models as Super Vector Machines or Logistic Regression, among others. Our approaches only uses the text, without exploiting any contextual information from the users and the related social media.

2 Related work

In the last years, the detection of toxic content in social media has received considerable attention from the NLP community [39], [47], [52]. Most existing approaches have been built on classical machine learning (ML) techniques [19], [48], [43], however recently deep learning methods [23],[17], [23], [15] have been also applied to the task. In this section, we review some of the main studies of toxic language detection in social media.

[13] represented texts with a set of lexical and syntactic features. SVM and Naïve Bayes were used for two different tasks, detect offensive content and identify potential offensive users in social media, being SVM the best classifier with an F1 of 96.2% for the task of detecting offensive texts and an F1 of 77.8% for the task of identifying potential offensive users.

[9] explored several classical machine learning algorithms to detect abusive language (racism, sexism, hate speech, aggression and personal attacks). The authors used the Bag-Of-Words mode [53] to represent the texts. The Naïve Bayes algorithm obtained the top F1-Score (81.85%) on the Wikipedia talk dataset [30].

[12] also used an SVM with linear kernel and FastText [26], a library for text classification based on a neural network, which only has one-hidden layer. The authors only provided recall scores. The experiments showed that SVM outperfomed FastText for the task of abusive language detection.

In [20], the authors created their own dataset of tweets annotated with five categories to classify the level of harassment of each tweet. The categories are: 1) most offensive or violent messages, 2) threats, 3) hate speech, 4) directed Harassment, and 5) potentially offensive.

In [10], several classical machine learning algorithms (such as logistic regression, multinomial Naïve Bayes, and random fores) were applied techniques to detect abusive comments. The authors used TF-IDF to represent texts. They also studied a bidirectional long short-term memory (BiLSTM) [23] to the task.

[54] explored some of the most popular language models based on transformers [54] (such as BERT [17], RoBERTa [32] and XLM [15]) applied to the task of toxic comment classification. Their results shows that BERT and ROBERTa obtained better results than XML.

The majority of previous studies concerning to a behaviour detection in social networks are in English, very few efforts have been made to address this kind of task in Spanish. Below we describe some of the studies about toxic detection from texts written in Spanish.

[41] proposed different approaches to detect the misogyny and xenophobia from Spanish tweet. They applied different classical supervised machine learning techniques such as Naïve Bayes, SVM, logistic regression, decision tree, and an ensemble voting classifier. They also applied LSTM model to deal with the task. Moreover, they develop their own linguistic resource that contains a set of hateful concepts correlated with hateful words toward women or/and immigrants. The authors also employed the iSOL lexicon [18], a dictionary with positive and negative words, and word embeddings from the model [8]. The authors consider their results with the lexicon-based approach "are more than acceptable results", compared to other machine learning approaches. Decision Tree shows the works results with an F1-Score of 0.686, while Multinomial Naïve Bayes and Logistic Regressions obtain the top performance with a F1-score of 0.728 and 0.73 respectively. Moreover, the LSTM model obtained a similar performance with an F1-score of 0.704. The authors also developed an ensemble voting classifier that combined bot the multinomial Naïve bayes and logistic regression, achieving the best result with an F1-score of 74.2%. Later, in 2021, the same authors [42] ex-

plored different pre-trained language models based on transfer learning (BERT, XLM and BETO. BETO). BETO was the approach that obtained the best F1 (77.6%).

3 MeOffendEs@IberLEF 2021 Competition

Most previous work have focused on toxic detection from English texts. MeOffendEs@IberLEF 2021 is a competition to boost research on the detection of offensive language in social media, a sensitive topic that has hardly been addressed for the Spanish language. The organizers of the competition have created a dataset in which comments written in Spanish from different social networks are collected.

The organisation proposes a series of tasks, which mainly consist of classifying the comments into different categories using metadata and additional information. There are a total of four different subtasks:

- Subtask 1: Non-contextual multiclass classification for generic Spanish.
- Subtask 2: Contextual multiclass classification for generic Spanish.
- Subtask 3: Non-contextual binary classification for Mexican Spanish.
- Subtask 4: Contextual binary classification for Mexican Spanish.

The main difference between the tasks are the variant of the language: if it is generic Spanish or Mexican Spanish. Moreover, while the first and third tasks do not provide contextual information, the second and fourth tasks allow to use contextual metadata with information related to the comment such as the user or the related social media.

We have only participated in the subtask 1, whose goal is detect the offensiveness of the comments written in Spanish using only the texts. There are a total of four classes, OFG, OFP, NOM and NO, which will be described in the next sections.

4 Materials Methods

This chapter starts describing in detail the dataset of the MeOffendEs@IberLEF task [40]. Then we present the approaches that we have developed for our participation in the task.

4.1 Dataset

The dataset consist of comments over different social media platforms such as YouTube, Instagram and Twitter. It contains more than 50,000 comments in Spanish, making this corpus the largest and more varied Spanish dataset for offensive language analysis. Each comment in the dataset has a text, a numerical ID and a label that provides the offensive level and its target. The different categories are:

– OFP: the comment is offensive and its target is a person.
– OFG: the comment is offensive and its target is a group of people or collective.
– NOM: the comment is non-offensive, but uses inadequate language.
– NO: the comment is non-offensive.

As an example of our data, the comment "vergüenza ajena like si crees que windy parece retrasada" which means that "like, if you think that Windy looks stupid, cringe," is a clear example of the category OFP, as its content is offensive, it's a clear example where it has used swear words and denigrates a person.

The organisers provided a training set with 16,710 comments. During the evaluation, they also provided a test set with a total of 13,607 comments. These comments are not classified, that is, they do not include their corresponding label.

We randomly split this training dataset into two subsets a ratio 80:20. The first subset is used for training our models and the second one to tune their hyper-parameters.

Fig.1 shows the class distribution, which is very similar on both subsets. There is a strong unbalanced distribution of the classes, being NO the class with more instances. However, there are still a large number of comments using offensive language.

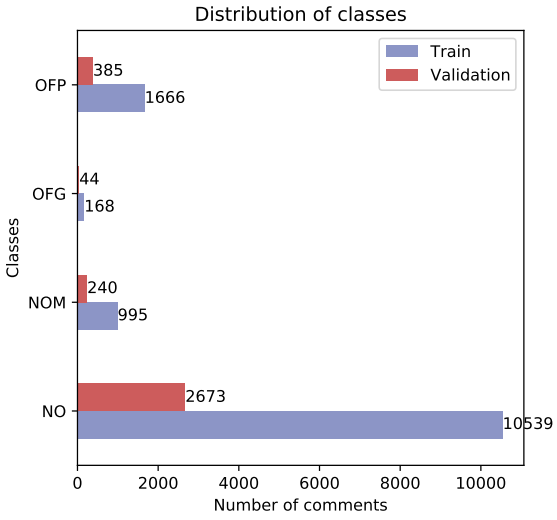


Fig. 1: Class distribution on training and validation datasets

4.2 Traditional Machine Learning approach

Data preprocessing Preprocessing techniques help us clean the texts and reduce the size of the vocabulary to represent the comments. We have applied the following techniques to preprocess the comments of the datasets:

- Convert to lower-case the comments.
- Tokenize the text and remove the stopwords (words without semantic meaning). To do this, we use the `NLTK` library [7].
- Normalise tokens applying the Snowball stemming technique [3].
- Remove different symbols, words with numbers, punctuation, etc.

Another aspect that we have previously analysed is the influence of emoticons. We have carried out an analysis where we converted emoticons and different emojis to text, i.e. each emoticon corresponded to a description such as happy or sad or shy, etc. However, there is not much difference in the results obtained by keeping these emoticons and transforming them than by removing these symbol.

After text processing, we need to transform the representation of the text into a vector, as input of our models. We have applied two different methods. First, we converts each sentence into vectors using the TF-IDF method model [5]. The TF-IDF score is calculated by multiplying the metrics of term frequency (TF) of a word and the inverse document frequency (IDF). To obtain IDF, the total number of documents is divided by the number of documents that contain the word. Then, the logarithm is applied on this result. The higher tf-idf of a word, the more relevant the word is. As result of applying this method, we obtain the processed data and we can start to train the models.

To deal with the problem of unbalanced classes, we have applied different techniques such as undersampling and oversampling [22], and Synthetic Minority Over-sampling Technique (SMOTE) [11]. Undersampling and oversampling techniques handle the imbalance problem by randomly resampling the training dataset. The undersample method deletes instances from the majority class while oversampling duplicates instances from the minority class. SMOTE is an oversampling technique, which focuses on the feature space for each target class and its nearest neighbours, to generate new instances with the help of interpolation between the positive instances that lie together [21]. To apply these techniques we have used the corresponding functions from the package `imblearn` of python using the parameters by default.

Now we briefly explain the different classifiers that we have used.

4.3 Random Forest

Random Forest is a supervised learning algorithm. Random Forest classifier consists on a large number of decision trees that operate as an ensemble. The `RandomForestClassifier` function from the package `sklearn` of python is used to train the model. We use a total of 100 number of trees and the rest of the parameters by default.

4.4 Support Vector Machine (SVM)

SVM [16] is a supervised machine learning algorithm that uses the kernel trick. This technique finds the optimal hyper plane that separate the instances of the classes. The SVM is commonly used for text classification [50], where text are usually represente using the TF-IDF model. The `LinearSVC` function from the package `sklearn` of python is used to train the model. Using the balanced class weight and the rest of the parameters by default.

4.5 Naïve Bayes

Naïve Bayes is a family of probabilistic algorithms that take advantage of probability theory and Bayes' Theorem [6]. It is also used in NLP applying the Bayes' Theorem to predict the "probability for each class such as the probability that given data point belongs to a particular class" [45]. In this study the multinomial Naïve Bayes is applied using the `MultinomialNB` function from the package `sklearn` of python to train the model.

4.6 Logistic Regression

Logistic regression is a statistical method that is used to predict the probability of a binary outcome based on a set of independent variables. In our study, we have used a multinomial logistic regression, as we have a total of four classes. To achieve this the `LogisticRegression` function from the package `sklearn` of python is used to train the model. Just like the rest of the models, using the balanced class weight and the rest of the parameters by default.

4.7 Stochastic Gradient Descent (SGD)

Stochastic Gardient Descent is an optimization technique to fitting linear classifiers and regressors under convex loss functions such as (linear) Support Vector Machines and Logistic Regression [37]. In this study we have applied a linear classifiers with SGD training using the `SGDClassifier` function from the package `sklearn` to train the model. In this case we use the parameters by default, meaning that the loss function gives a linear SVM, also we have used the balanced class weight.

4.8 Gradient Boosting Classifier

Gradient Boosting Classifier [35] is a machine learning technique that is an ensemble of machine learning algorithms and weak prediction models, obtaining as result an outperforming model. Gradient boosting classifier applies boosting as optimization function of aklternative loss functions. For this study, the `GradientBoostingClassifier` function from the package `sklearn` is used to train the model. We have used the default parameters for this task.

4.9 Deep Learning approach

Data preprocessing and features module First, we clean the texts removing different symbols, words with numbers, punctuation. Then, texts were tokenized by using the keras tokenizer, with 10,000 as maximum number of words. To represent the comments, we use the word embedding technique random initialization [29], that is, for each token of the vocabulary, a vector of numbers is randomly created. The comments are truncated and padded to obtain the same size in all comments (250 was defined as the maximum number of words in a comment). Then, the models are initialized with these vectors.

LSTM for Offensive Classification In this section, we describe the architecture of the LSTM model that we have used for the task of classifying the comments into the four classes defined in the OffendEs corpus.

Long Short Term Memory (LSTM), is a type of recurrent neural network capable of learning order dependence in sequence prediction problems, keeping only relevant information from the past inputs during training.

The architecture of our LSTM model is explained by layers in the next steps:

- The first layer of our LSTM model is the embedded layer. The embedding layer is initialized with the sentence embedding obtained as result of the random initialization process. This layer uses 250 length vectors to represent each word.
- Before the LSTM layer, we add a dropout layer using a dropout rate of 0.2. This will help us to prevent overfitting. To do this, we use the function `SpatialDropout1D` proportioned by `keras` in python [49].
- The last layer is the LSTM layer with a memory dimension of 100 memory units.
- The activation function of the output layer is the softmax function of one single layer, assigning the probabilities of an instance being each class.

The training of the network was performed by the minimization of the categorical cross entropy function, and the learning process was optimized with the Adam [28] algorithm as default.

The next Fig.2 shows the approach based on the LSTM architecture.

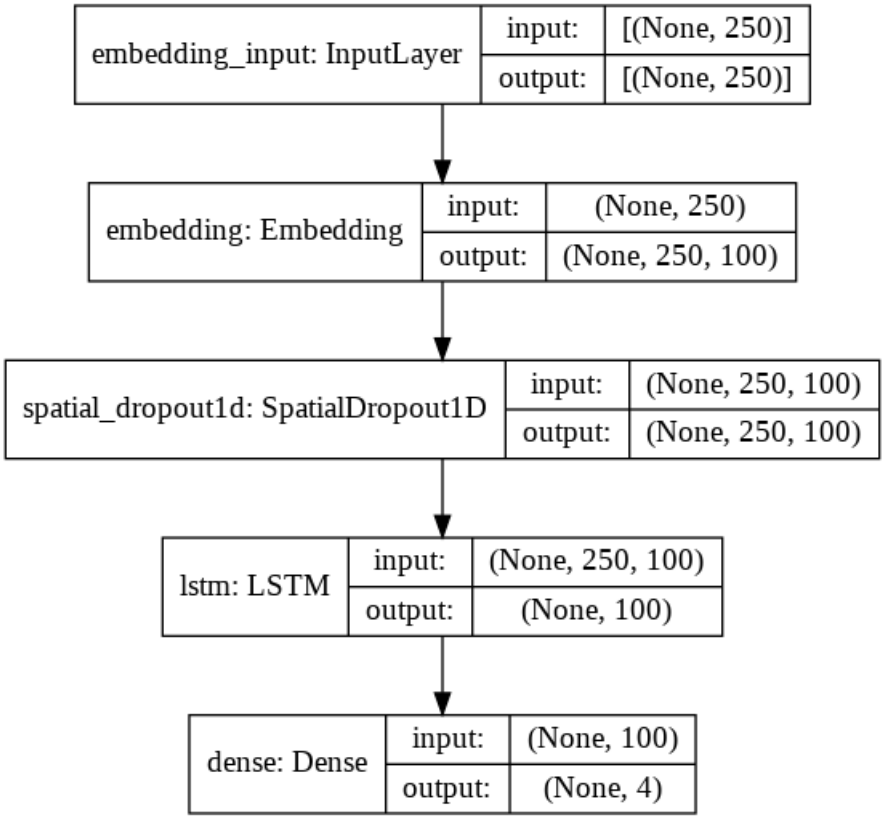


Fig. 2: Overview architecture of our LSTM model

BERT for Offensive Classification The second approach of deep learning architecture is the BERT model. BERT applies a bidirectional training of transformer, which can read entire sequences of tokens at once as opposed to directional models like LSTMs that read sequentially. The transformers are *"an attention mechanism that learns contextual relations between words"* [24], consist of two distinct mechanisms: an encoder and a decoder. The first reads the input, while the latter creates the task prediction (in our case, a class for the input comment). This provides a deeper understanding of language flow and context than one-way language models.

The data preprocess prior to train the model is the same as the one we have applied with the LSTM model. The use the BERT pre-trained model for tokenization, provided by HuggingFace [2], that it was implemented by Google team. After the encoding process, the BERT embedding vector is obtained. This transformations of the data correspond to an input layer of the network.

Again, the activation function of the output layer is the softmax function of one single unit, assigning the probabilities of an instance being each class.

Also as LSTM model, the training of the network was performed by the minimization of the categorical cross entropy function, and the learning process was optimized with the Adam algorithm as default.

4.10 Regularization Details

There are numerous cases when the training performance of a machine learning algorithm is really high, but after all in the test set the performance is poor. This is common and it happens due the overfitting of the model. The overfitting is when the neural network has high data variance and makes it hard for the process when it use new data that it was not in the training.

To solve this, different techniques are applied that help us to handle the overfitting problem, such as the already mentioned dropout or early stopping, both applied in the deep learning models.

Dropout In deep neural networks, dropout refers to the noise or data that is dropped to improve processing and results, it is a regularization technique [46].

Dropout add a penalty to the loss function. At the training stage, the input nodes are randomly selected and ignored with probability $1 - p.$, meaning that the dropout layer randomly sets input units to 0 with a frequency of rate at each step during training [49] There are several studies showing that a dropout rate of 0.5 is effective in most scenarios [27].

Despite of that, we have decided to chose a threshold of 0.2. The decision of choosing a threshold lower to 0.5, is because we have four classes that are very similar to each other and they are unbalanced.

As result of applying dropout we get a much simpler network.

Early Stopping Early stooping is another strategy to prevent the overfitting of the models. The objective of this technique is to train sufficiently with the training data, and stop when the performance on the validation data starts to decline to avoid overfitting. We gave a margin of 2 epochs, that is, the model is allowed to be trained for 2 more epochs to improve the performance of the model. If there is no improvement in the validation loss, the training is stopped.

4.11 Network Training Details

Optimizer The optimizer of our deep learning architectures is Adaptive Moment Estimation (Adam) is a stochastic gradient descent method. According to Diederik P. Kingma et.al [28]. The method is *"computationally efficient, has little memory requirement, invariant to diagonal rescaling of gradients, and is well suited for problems that are large in terms of data/ parameters"* [28]. The default parameters are used for Adam optimizer, with LSTM we use `keras` and with BERT we use the TensorFlow optimizer. The exception is that we have choose a different learning rate for BERT.

- Learning rate LSTM: 0.001
- Learning rate BERT: $2e-5$
- Beta 1: 0.9
- Beta 2: 0.999
- Epsilon: $1e-7$

Loss The selected loss function is the categorical cross entropy, also called Softmax Loss. It is a loss function that is used in multi-class classification tasks.

The loss function is the following:

$$Loss = -\sum_{i=1}^{outputsize} y_i \cdot log(\hat{y}_i) \tag{1}$$

where $\hat{y}_i$ is the i-th scalar value in the model output, y_i is the corresponding target value, and output size is the number of scalar values in the model output [1].

Number of epochs and batch size We have declared a total of 15 number of epochs to fit both models, LSTM model and BERT model.

The batch size is 100 in the case of the LSTM model and 1114 for BERT model.

Monitoring the loss in the validation data, only was necessary 6 epochs for the LSTM model and 4 in the case of the BERT, as result of the early stopping, since after those points the loss validation stopped improving.

Software and Hardware Details The experiments have been developed in Python 3.7.7. Concretely to develop the machine learning algorithms, we have used the Python library `scikit-learn` [38], while the deep learning models were developed making use of the libraries `Keras` [14] on top of Tensorflow [4] and PyTorch [36].

Our experiments were conducted on Google Colab with the GPU activated. Google Colab is a open product from Google Research that allows to execute and create python code through the browser, enabling us to use computational resources, such as GPU or TPU.

There are many other libraries that we have used to plot, visualise the data and evaluate the models, some of them are the libraries `pandas`, `numpy`, `sklearn` and `matplotlib`.

5 Evaluation and Discussion

To evaluate our models, the organiser provided a the test dataset 13,607 comments. These did not include their label.

For the performance of the models, we have used different standard metrics as precision, recall and F1-score. Moreover, using their micro-averaged, macro-averaged and weighted macro-averaged versions, we have obtained the mean Square Error (MSE) from the official results. The micro-average is more suitable for unbalanced datasets. Since we have an unbalanced dataset (see Fig.1), the most appropriate metric for comparing the models is the micro-averaged F1-Score, which we will call micro-F1. Analysing further, we have also been able to obtain results at the class level. With this, we can check the efficiency of our models when classifying and predicting a comment that could be offensive.

Table1 shows the results obtained with the traditional machine learning methods. We can see that all of the models achieve a micro-F1 that ranges from 0.80 to 0.88, although they show certain differences at the class level. The only models that can obtain results for the minority class (OFG) are the the logistic regression and Gradient Boosting models. The rest of them are not able to obtain a result, being 0 for all the metrics. This may be due to the class OFG has only a few instances.

The Table2 shows the results obtained with deep learning architectures: LSTM and BERT. The micro-F1 of this two models is similar obtaining a difference of 0.01. Again at class level, the models obtain an score of 0 for the class OFG while the rest of the classes has an score around 0.9 for NO, and 0.5-0.7 for NOM and OFP. As result, the best model obtained on the validation data set is the Stochastic Gradient Descent achieving a micro-F1 of 0.88, followed by the random forest 0.874 and BERT 0.870. At level class, the best model is logistic regression with a micro-F1 of 0.93 for the class NO, 0.71 for NOM class, 0.19 for OFG class and 0.59 for OFP class.

In addition, the three models that we have presented to the competition and evaluated with the test dataset are the deep learning approach models (LSTM BERT) and logistic regression. We have selected these models because we wanted to focus on this newest algorithms results and also have a reference of a traditional model. Although the logistic regression is not the best model of all the traditional machine learning models, it is the only one that at class level is able to obtain results for all of them, so it has been decided to present this model to the competition.

The results obtained for the test dataset (see Table3) are lower than those obtained with these models in the validation dataset, although this is normal. The best approach is the BERT model. The LSTM model has achieved a micro-F1 of 0.861734 on the validation dataset and 0.80751 on the official results, which are lower than those obtained with the BERT model, micro-F1 of 0.870992 on the validation dataset and 0.84168 on the test dataset. Moreover, we can see with the logistic regression model, it works better than the LTSM model. In particular, the logistic regression has achieved a micro-F1 of 0.860861 on the validation dataset, and 0.816331 on the test dataset. Moreover, the official MSE of LSTM, BERT and Logistic regression models are 0.085417, 0.069783 and 0.075155 respectively.

If we compare the results obtained for both datasets, we can say that even if we have not proposed the best model of the validation dataset, Stochastic Gradient Descent, the results with BERT, LSTM and Logistic regression are the expected.

In the results of this study, there is a pattern that is repeated for all the models and their approaches. In general the results obtained are quite similar, even if we have applied different data process or methods for each approach. Also, the majority class (NO) has a higher score for all of the models, while the classes NOM and OFP also obtain similar results between them two. This happens because these models are trained with unbalanced data. However, the models are able to obtain an score for the rest of the classes, despite of this large imbalance. This fact is also observed in the confusion matrix of these respective models (see Fig3 Fig4). As commented, the majority of the models are not able to obtain a metric other than 0 for the minority class (OFG). This may be due to the dataset is unbalanced and only a 1.27% of the comments corresponds to the OFG class(see Fig. 1). Knowing that, we have explored different methods such as SMOTE, oversampling and undersampling methods to resolve the data imbalance. These techniques were only applied to the traditional machine learning classifiers, because deep learning approach models are robust for imbalanced data [25], [44].

However, the results obtained after applying these methods do not provide any significant difference (see Table 4 and Table 5) on unbalanced data, excepted that the models obtain scores for the minority class (OFG). All the models (except Naive Bayes and Gradient Boosting) use balanced class weight, that is, the training of the models takes into account the weight of each class. Probably due to that fact, the results obtained with the unbalanced data and those applying the unbalancing techniques are similar and no noticeable improvement is obtained.

Even with this fact, we cannot claim that our models find it more difficult to classify comments with offensive language than those that do not contain it. Although the models have been trained with a greater number of non-offensive comments. As we have commented, observing the results obtained in all the rest of the classes, and taking into account this imbalance, most of the models are capable of making a classification and detection of the different classes.

Table 1: Traditional Machine learning model results on the validation dataset.

Model		Precision	Recall	F1-Score
Random Forest:	**Micro-averaged**	0.874625	0.874625	0.874625
	Macro-averaged	0.582415	0.502049	0.533080
	Weighted Macro-averaged	0.852574	0.874625	0.858690
	Class			
	NO	0.90	0.98	0.943
	NOM	0.73	0.57	0.64
	OFG	0.00	0.00	0.00
	OFP	0.70	0.46	0.55
SVM:	**Micro-averaged**	0.868940	0.868940	0.868940
	Macro-averaged	0.550659	0.530712	0.540028
	Weighted Macro-averaged	0.856201	0.868940	0.862165
	Class			
	NO	0.92	0.95	0.94
	NOM	0.67	0.62	0.65
	OFG	0.00	0.00	0.00
	OFP	0.61	0.55	0.58
Naïve Bayes:	**Micro-averaged**	0.809694	0.809694	0.809694
	Macro-averaged	0.602240	0.274338	0.269344
	Weighted Macro-averaged	0.802419	0.809694	0.734240
	Class			
	NO	0.81	1.00	0.89
	NOM	0.67	0.03	0.05
	OFG	0.00	0.00	0.00
	OFP	0.93	0.07	0.13
Logistic Regression:	**Micro-averaged**	0.860861	0.860861	0.860861
	Macro-averaged	0.587131	0.623044	0.603725
	Weighted Macro-averaged	0.870254	0.860861	0.865023
	Class			
	NO	0.94	0.92	0.93
	NOM	0.68	0.75	0.71
	OFG	0.17	0.20	0.19
	OFP	0.56	0.63	0.59
Stochastic Gradient Descent (SGD):	**Micro-averaged**	0.881508	0.881508	0.881508
	Macro-averaged	0.568356	0.553668	0.558338
	Weighted Macro-averaged	0.867754	0.88150	0.873246
	Class			
	NO	0.93	0.96	0.94
	NOM	0.66	0.72	0.69
	OFG	0.00	0.00	0.00
	OFP	0.69	0.54	0.60
Gradient Boosting Classifier:	**Micro-averaged**	0.865050	0.865050	0.865050
	Macro-averaged	0.572678	0.510758	0.534177
	Weighted Macro-averaged	0.848065	0.865050	0.852717
	Class			
	NO	0.90	0.96	0.93
	NOM	0.70	0.65	0.67
	OFG	0.04	0.02	0.03
	OFP	0.64	0.41	0.50

Table 2: LSTM BERT results on the validation dataset.

Model		Precision	Recall	F1-Score
LSTM:	**Micro-averaged**	0.873387	0.850389	0.861734
	Macro-averaged	0.559542	0.479958	0.512573
	Weighted Macro-averaged	0.850232	0.850389	0.847382
	Class			
	NO	0.91	0.95	0.93
	NOM	0.69	0.49	0.57
	OFG	0.00	0.00	0.00
	OFP	0.64	0.48	0.55
BERT:	**Micro-averaged**	0.893610	0.849491	0.870992
	Macro-averaged	0.581578	0.509822	0.539391
	Weighted Macro-averaged	0.878057	0.849491	0.861934
	Class			
	NO	0.94	0.93	0.94
	NOM	0.77	0.52	0.62
	OFG	0.00	0.00	0.00
	OFP	0.62	0.59	0.60

Table 3: Official results of the models.

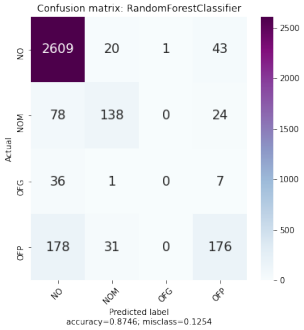
Model		Precision	Recall	F1-Score
LSTM:	**Micro-averaged**	0.807511	0.807511	0.807511
	Macro-averaged	0.622454	0.503332	0.521310
	Weighted Macro-averaged	0.789026	0.807511	0.792949
BERT:	**Micro-averaged**	0.841687	0.841687	0.841687
	Macro-averaged	0.578188	0.545135	0.559502
	Weighted Macro-averaged	0.821612	0.841687	0.829947
Log reg:	**Micro-averaged**	0.816331	0.816331	0.816331
	Macro-averaged	0.602351	0.554992	0.575223
	Weighted Macro-averaged	0.807140	0.816331	0.809762

Table 4: Traditional Machine learning model results on the validation dataset using undersample & oversample methods.

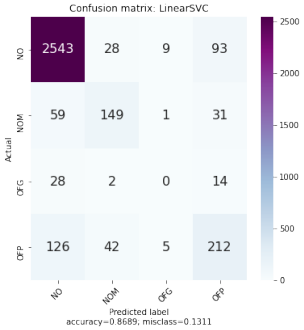
Model		Precision	Recall	F1-Score
Random Forest:	**Micro-averaged**	0.875524	0.875524	0.875524
	Macro-averaged	0.569118	0.524498	0.543076
	Weighted Macro-averaged	0.855881	0.875524	0.863494
	Class			
	NO	0.91	0.97	0.94
	NOM	0.69	0.63	0.66
	OFG	0.00	0.00	0.00
	OFP	0.67	0.50	0.57
SVM:	**Micro-averaged**	0.852783	0.852783	0.852783
	Macro-averaged	0.542529	0.538833	0.539994
	Weighted Macro-averaged	0.850219	0.852783	0.851435
	Class			
	NO	0.93	0.93	0.93
	NOM	0.63	0.61	0.62
	OFG	0.07	0.05	0.05
	OFP	0.55	0.57	0.56
Naïve Bayes:	**Micro-averaged**	0.786056	0.786056	0.786056
	Macro-averaged	0.470048	0.543893	0.494525
	Weighted Macro-averaged	0.837650	0.786056	0.807358
	Class			
	NO	0.94	0.84	0.89
	NOM	0.41	0.62	0.49
	OFG	0.05	0.14	0.07
	OFP	0.48	0.58	0.53
Logistic Regression:	**Micro-averaged**	0.865051	0.865051	0.865051
	Macro-averaged	0.584222	0.596048	0.589732
	Weighted Macro-averaged	0.868461	0.865051	0.866580
	Class			
	NO	0.94	0.93	0.93
	NOM	0.67	0.70	0.69
	OFG	0.15	0.14	0.14
	OFP	0.57	0.62	0.60
Stochastic Gradient Descent (SGD):	**Micro-averaged**	0.858169	0.858169	0.858169
	Macro-averaged	0.551334	0.576870	0.562412
	Weighted Macro-averaged	0.862328	0.858169	0.859759
	Class			
	NO	0.94	0.92	0.93
	NOM	0.61	0.72	0.66
	OFG	0.09	0.07	0.08
	OFP	0.56	0.60	0.58
Gradient Boosting Classifier:	**Micro-averaged**	0.843507	0.843507	0.843507
	Macro-averaged	0.551221	0.591436	0.567099
	Weighted Macro-averaged	0.857574	0.843507	0.849853
	Class			
	NO	0.93	0.90	0.92
	NOM	0.63	0.75	0.68
	OFG	0.08	0.16	0.11
	OFP	0.56	0.56	0.56

Table 5: Traditional Machine learning model results on the validation dataset using SMOTE method.

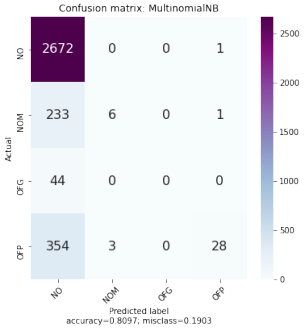
Model		Precision	Recall	F1-Score
Random Forest:	**Micro-averaged**	0.859964	0.859964	0.859964
	Macro-averaged	0.538442	0.509172	0.521811
	Weighted Macro-averaged	0.844352	0.859964	0.851356
	Class			
	NO	0.91	0.95	0.93
	NOM	0.68	0.55	0.61
	OFG	0.00	0.00	0.00
	OFP	0.56	0.54	0.55
SVM:	**Micro-averaged**	0.839318	0.839318	0.839318
	Macro-averaged	0.535233	0.537858	0.534238
	Weighted Macro-averaged	0.844442	0.839318	0.841221
	Class			
	NO	0.93	0.91	0.92
	NOM	0.62	0.59	0.60
	OFG	0.11	0.07	0.08
	OFP 0.49 0.58 0.53			
Naïve Bayes:	**Micro-averaged**	0.806703	0.806703	0.806703
	Macro-averaged	0.499279	0.550670	0.514789
	Weighted Macro-averaged	0.843068	0.806703	0.822894
	Class			
	NO	0.93	0.87	0.90
	NOM	0.49	0.64	0.55
	OFG	0.05	0.16	0.08
	OFP	0.53	0.53	0.53
Logistic Regression:	**Micro-averaged**	0.865051	0.865051	0.865051
	Macro-averaged	0.586328	0.592190	0.588507
	Weighted Macro-averaged	0.869716	0.865051	0.867032
	Class			
	NO	0.94	0.93	0.93
	NOM	0.70	0.67	0.69
	OFG	0.15	0.14	0.14
	OFP	0.56	0.63	0.59
Stochastic Gradient Descent (SGD):	**Micro-averaged**	0.871035	0.871035	0.871035
	Macro-averaged	0.579939	0.583309	0.581045
	Weighted Macro-averaged	0.869333	0.871035	0.870068
	Class			
	NO	0.94	0.94	0.94
	NOM	0.66	0.72	0.69
	OFG	0.11	0.09	0.10
	OFP	0.61	0.59	0.60
Gradient Boosting Classifier:	**Micro-averaged**	0.851287	0.851287	0.851287
	Macro-averaged	0.566121	0.586049	0.573528
	Weighted Macro-averaged	0.856571	0.851287	0.853700
	Class			
	NO	0.93	0.92	0.93
	NOM	0.65	0.69	0.67
	OFG	0.13	0.20	0.16
	OFP	0.56	0.52	0.54



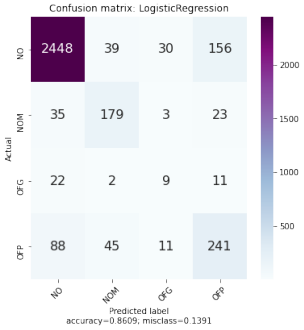
(a) Random Forest Confusion Matrix



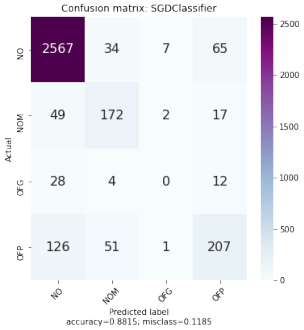
(b) Support Vector Machine (SVM) Confusion Matrix



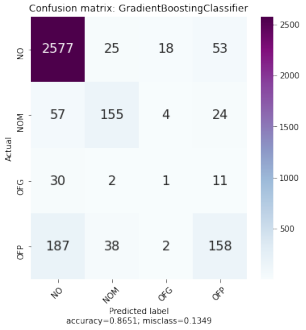
(c) Naïve Bayes Confusion Matrix



(d) Logistic Regression Confusion Matrix



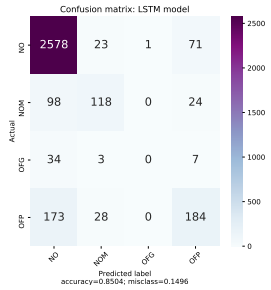
(e) Stochastic Gradient Descent (SGD) Confusion Matrix



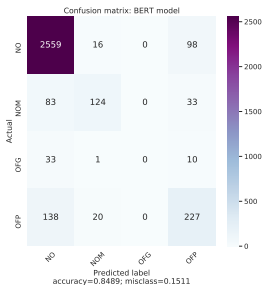
(f) Gradient Boosting Classifier Confusion Matrix

Fig. 3: Confusion matrix on the Validation dataset Traditional Machine Learning approach

The confusion matrix show us our True Negatives on the top left, False Negatives on the top right, True Positives on the bottom right, and False Positives on the bottom left of each class.



(a) LSTM Confusion Matrix



(b) BERT Confusion Matrix

Fig. 4: Confusion matrix on the Validation dataset Deep learning approach

The confusion matrix show us our True Negatives on the top left, False Negatives on the top right, True Positives on the bottom right, and False Positives on the bottom left of each class.

6 Social-Economic Impact

Nowadays most social media, applications and websites, have various tools that prevent different actions that can be dangerous or offensive to users. However, many of these tools are mainly focused on the treatment of images and videos in which different behaviours can be identified, such as violent, unpleasant or illicit behaviour that could be offensive or be sensible. In these cases, a filter is added to these types of applications and usually are identified, blocked or even deleted. While there are not as many tools implemented to deal with text on these platforms. Most of the time, when a person suffers discrimination or cyberbullying on social media [51] is done through text comments or text messages. Although there are identities that are involved to identify and track this kind of behaviours, it would be really efficient if any model dedicated to the identification of offensive comments is incorporated.

7 Law framework

In Spain, the practice of insults, threats and slander are commonplace on social networks, as they are justified by the right to Freedom of Expression, but they are not unpunished.

Freedom of expression is a fundamental right as defined in Article 10 of the European Convention on Human Rights, and Article 20.1.a) of the Spanish Constitution. The counterweight to Freedom of Expression is the Right to Honour. This right is included in Spanish legislation in Article 18 of the Constitution, and is a fundamental right regulated in Organic Law 1/1982, of 5 May, on the civil protection of the right to honour, personal and family privacy and one's own image.

The different offences that can be committed are typified in the Spanish Penal Code (slander in Articles 205 et seq. and libel in Articles 208 et seq.), including harassment or stalking (Article 172 ter CP), sexting (Article 197.7 CP), grooming (Article 183 bis CP), cyberbullying (Article 197 CP), among others.

However, despite the fact that these crimes are commonly committed on social networks, there is no regulatory body that prevents this series of conducts; it simply limits itself to punishing them once they have been committed and reported. It is the companies themselves, such as Facebook or Twitter, that judge which actions damage the rights of other users, all of which is related to the problem posed by the limits of rights.

8 Conclusion and Future Work

One of the main goals of this study is to study different NLP and deep learning models. In particular, this document describes our participation in the shared task of MeOffendEs@IberLEF 2021 [40]. We have explored different deep learning models such as Long-Short Term Memory (LSTM) and Bidirectional Encoder Representations from Transformers (BERT), as well as traditional machine learning models such as Logistic Regression or Support Vector Machines (SVM) among others, to classify the comments (written in Spanish) into the four classes defined in the OffendEs corpus, which allow to label the offensive level and its offensive target described in each comment.

The results of our experiments show that for the test evaluation, BERT obtains the best results obtaining an F1-Score of 84.16% and a MSE of 0.069. Comparing this with the other deep learning approach, LSTM model. We can see that a bidirectional network model works better than a unidirectional model for detection of offensive comments. The bidirectional model, BERT, it is able to obtain the context of a comment giving as result a better performance, also, considering the results of the logistic regression, we can see that for this kind of task it is better to work with a bidirectional network as BERT is better than the logistic regression. Even that, considering the performance of this models in the validation dataset, logistic regression is the only one of the three who is able to get a result for each class (NO 0.93, NOM 0.71, OFG 0.19, OFP 0.59).

We have also studied the influence of emoticons by converting them to text. However, the inclusion of emoticons did not improve the results. In addition, although we have not gone into great depth on this, as we have commented before, several approaches have been used to solve the problem of imbalance data, such as Oversampling and Undersampling or SMOTE methods, however we also have not gone further applying this techniques as the results obtained do not improve.

As future work, we plan to address the other subtasks proposed in MeOffendEs@IberLEF 2021 as the comparison of using Mexican or general Spanish language with our models. We will explore other pre-trained models trained on tweets and comments of other Social networks as XLM that it use in [42] model

or RoBERTa also applied in [54]. We will use the contextual information about the user and the social media. In addition, we plan to develop a multimodal system that also exploits the information from images or videos to identify offensive content in social media.

Also it could be interesting to relate this task with another different task that is provided by IberLeF [34]. They propose numerous task as the identification or classification of emotions, Stance and Opinions, harmful information, health related information extraction and knowledge discovery, humour and irony or lexical acquisition. Saying this and as future work it could be interesting to merge the emotion classification with the offensive detection as we could find different behaviours from the way the users react to a different type of comment.

Acknowledgements

This work was supported by the NLP4RARE-CM-UC3M, which was developed under the Interdisciplinary Projects Program for Young Researchers at University Carlos III of Madrid. The work was also supported by the Multiannual Agreement with UC3M in the line of Excellence of University Professors (EPUC3M17), and in the context of the V PRICIT (Regional Programme of Research and Technological Innovation).

References

1. Categorical crossentropy loss function: Peltarion platform, bluehttps://peltarion.com/knowledge-center/documentation/modeling-view/build-an-ai-model/loss-functions/categorical-crossentropy
2. Hugging face – the ai community building the future., bluehttps://huggingface.co/
3. snowballstemmer, bluehttps://pypi.org/project/snowballstemmer/
4. Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X.: TensorFlow: Large-scale machine learning on heterogeneous systems (2015), bluehttps://www.tensorflow.org/, software available from tensorflow.org
5. Baeza-Yates, R., Ribeiro-Neto, B., et al.: Modern information retrieval, vol. 463. ACM press New York (1999)
6. Berrar, D.: Bayes' theorem and naive bayes classifier. Encyclopedia of Bioinformatics and Computational Biology: ABC of Bioinformatics; Elsevier Science Publisher: Amsterdam, The Netherlands pp. 403–412 (2018)
7. Bird, S., Klein, E., Loper, E.: Natural language processing with Python: analyzing text with the natural language toolkit. " O'Reilly Media, Inc." (2009)
8. Bojanowski, P., Grave, E., Joulin, A., Mikolov, T.: Enriching word vectors with subword information. Transactions of the Association for Computational Linguistics **5**, 135–146 (2017)

9. Bourgonje, P., Moreno-Schneider, J., Srivastava, A., Rehm, G.: Automatic classification of abusive language and personal attacks in various forms of online communication. In: International Conference of the German Society for Computational Linguistics and Language Technology. pp. 180–191. Springer, Cham (2017)
10. Chandrika, C., Kallimani, J.S.: Classification of abusive comments using various machine learning algorithms. In: Cognitive Informatics and Soft Computing, pp. 255–262. Springer (2020)
11. Chawla, N.V., Bowyer, K.W., Hall, L.O., Kegelmeyer, W.P.: Smote: Synthetic minority over-sampling technique. Journal of Artificial Intelligence Research **16**, 321–357 (Jun 2002). https://doi.org/10.1613/jair.953, bluehttp://dx.doi.org/10.1613/jair.953
12. Chen, H., McKeever, S., Delany, S.J.: Abusive text detection using neural networks. In: McAuley, J., McKeever, S. (eds.) Proceedings of the 25th Irish Conference on Artificial Intelligence and Cognitive Science, Dublin, Ireland, December 7 - 8, 2017. CEUR Workshop Proceedings, vol. 2086, pp. 258–260. CEUR-WS.org (2017), bluehttp://ceur-ws.org/Vol-2086/AICS2017_paper_44.pdf
13. Chen, Y., Zhou, Y., Zhu, S., Xu, H.: Detecting offensive language in social media to protect adolescent online safety. In: 2012 International Conference on Privacy, Security, Risk and Trust and 2012 International Confernece on Social Computing. pp. 71–80. IEEE (2012)
14. Chollet, F., et al.: Keras (2015), bluehttps://github.com/fchollet/keras
15. Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., Stoyanov, V.: Unsupervised cross-lingual representation learning at scale. In: Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics. pp. 8440–8451. Association for Computational Linguistics, Online (Jul 2020). https://doi.org/10.18653/v1/2020.acl-main.747, bluehttps://www.aclweb.org/anthology/2020.acl-main.747
16. Cortes, C., Vapnik, V.: Support-vector networks. Machine Learning **20**(3), 273–297 (1995). https://doi.org/10.1007/bf00994018
17. Devlin, J., Chang, M.W., Lee, K., Toutanova, K.: BERT: Pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers). pp. 4171–4186. Association for Computational Linguistics, Minneapolis, Minnesota (Jun 2019). https://doi.org/10.18653/v1/N19-1423, bluehttps://www.aclweb.org/anthology/N19-1423
18. Dolores Molina-González, M., Martínez-Cámara, E., Teresa Martín-Valdivia, M., Alfonso Ureña-López, L.: A spanish semantic orientation approach to domain adaptation for polarity classification. Information Processing Management **51**(4), 520–531 (2015). https://doi.org/https://doi.org/10.1016/j.ipm.2014.10.002, bluehttps://www.sciencedirect.com/science/article/pii/S0306457314000910
19. Domínguez-Almendros, S., Benítez-Parejo, N., Gonzalez-Ramirez, A.: Logistic regression models. Allergologia et immunopathologia **39**(5), 295–305 (2011)
20. Golbeck, J., Ashktorab, Z., Banjo, R.O., Berlinger, A., Bhagwan, S., Buntain, C., Cheakalos, P., Geller, A.A., Gergory, Q., Gnanasekaran, R.K., Gunasekaran, R.R., Hoffman, K.M., Hottle, J., Jienjitlert, V., Khare, S., Lau, R., Martindale, M.J., Naik, S., Nixon, H.L., Ramachandran, P., Rogers, K.M., Rogers, L., Sarin, M.S., Shahane, G., Thanki, J., Vengataraman, P., Wan, Z., Wu, D.M.: A large labeled corpus for online harassment research. In: Fox, P., McGuinness, D.L., Poirier, L., Boldi, P., Kinder-Kurlanda, K. (eds.) Proceedings of the

2017 ACM on Web Science Conference, WebSci 2017, Troy, NY, USA, June 25 - 28, 2017. pp. 229–233. ACM (2017). https://doi.org/10.1145/3091478.3091509, bluehttps://doi.org/10.1145/3091478.3091509
21. Happy95: Smote: Overcoming class imbalance problem using smote (Jan 2021), bluehttps://www.analyticsvidhya.com/blog/2020/10/overcoming-class-imbalance-using-smote-techniques/
22. Hernandez, J., Carrasco-Ochoa, J.A., Martínez-Trinidad, J.F.: An empirical study of oversampling and undersampling for instance selection methods on imbalance datasets. Progress in Pattern Recognition, Image Analysis, Computer Vision, and Applications Lecture Notes in Computer Science p. 262–269 (2013). https://doi.org/10.1007/978-3-642-41822-8$_3$3
23. Hochreiter, S., Schmidhuber, J.: Long short-term memory. Neural computation **9**(8), 1735–1780 (1997)
24. Horev, R.: Bert explained: State of the art language model for nlp (Nov 2018), bluehttps://towardsdatascience.com/bert-explained-state-of-the-art-language-model-for-nlp-f8b21a9b6270
25. Jin, D., Jin, Z., Zhou, J.T., Szolovits, P.: Is BERT really robust? natural language attack on text classification and entailment. CoRR **abs/1907.11932** (2019), bluehttp://arxiv.org/abs/1907.11932
26. Joulin, A., Grave, E., Bojanowski, P., Douze, M., Jégou, H., Mikolov, T.: Fasttext. zip: Compressing text classification models. arXiv preprint arXiv:1612.03651 (2016)
27. Kim, Y.: Convolutional neural networks for sentence classification. Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP) (2014). https://doi.org/10.3115/v1/d14-1181
28. Kingma, D.P., Ba, J.: Adam: A method for stochastic optimization (2017)
29. Kocmi, T., Bojar, O.: An exploration of word embedding initialization in deep-learning tasks (2017)
30. Leskovec, J., Huttenlocher, D., Kleinberg, J.: Predicting positive and negative links in online social networks. In: Proceedings of the 19th international conference on World wide web. pp. 641–650 (2010)
31. Li, I., Li, Y., Li, T., Alvarez-Napagao, S., Garcia-Gasulla, D., Suzumura, T.: What are we depressed about when we talk about covid-19: Mental health analysis on tweets using natural language processing. Lecture Notes in Computer Science Artificial Intelligence XXXVII p. 358–370 (2020). https://doi.org/10.1007/978-3-030-63799-6$_2$7
32. Liu, Y., Ott, M., Goyal, N., Du, J., Joshi, M., Chen, D., Levy, O., Lewis, M., Zettlemoyer, L., Stoyanov, V.: Roberta: A robustly optimized BERT pretraining approach. CoRR **abs/1907.11692** (2019), bluehttp://arxiv.org/abs/1907.11692
33. Maynard, D., Funk, A.: Automatic detection of political opinions in tweets. Lecture Notes in Computer Science The Semantic Web: ESWC 2011 Workshops p. 88–99 (2012). https://doi.org/10.1007/978-3-642-25953-1$_8$
34. Montes, M., Rosso, P., Gonzalo, J., Aragón, E., Agerri, R., Álvarez-Carmona, M.Á., Álvarez Mellado, E., Carrillo-de Albornoz, J., Chiruzzo, L., Freitas, L., Gómez Adorno, H., Gutiérrez, Y., Jiménez-Zafra, S.M., Lima, S., Plaza-de Arco, F.M., Taulé, M. (eds.): Proceedings of the Iberian Languages Evaluation Forum (IberLEF 2021) (2021)
35. Natekin, A., Knoll, A.: Gradient boosting machines, a tutorial. Frontiers in neurorobotics **7**, 21 (2013)

36. Paszke, A., Gross, S., Massa, F., Lerer, A., Bradbury, J., Chanan, G., Killeen, T., Lin, Z., Gimelshein, N., Antiga, L., Desmaison, A., Kopf, A., Yang, E., DeVito, Z., Raison, M., Tejani, A., Chilamkurthy, S., Steiner, B., Fang, L., Bai, J., Chintala, S.: Pytorch: An imperative style, high-performance deep learning library. In: Advances in Neural Information Processing Systems 32, pp. 8024–8035. Curran Associates, Inc. (2019), bluehttp://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf
37. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Duchesnay, E.: Scikit-learn: Machine learning in Python. Journal of Machine Learning Research **12**, 2825–2830 (2011)
38. Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Müller, A., Nothman, J., Louppe, G., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., Édouard Duchesnay: Scikit-learn: Machine learning in python (2018)
39. Phd, P., Adigun, A., O, O.: Identification and classification of toxic comments on social media using machine learning techniques pp. 2454–6194 (11 2019)
40. Plaza-del-Arco, F.M., Casavantes, M., Escalante, H., Martin-Valdivia, M.T., Montejo-Ráez, A., Montes-y-Gómez, M., Jarquín-Vásquez, H., Villaseñor-Pineda, L.: Overview of the MeOffendEs task on offensive text detection at IberLEF 2021. Procesamiento del Lenguaje Natural **67**(0) (2021)
41. Plaza-Del-Arco, F.M., Molina-González, M.D., Ureña-López, L.A., Martín-Valdivia, M.T.: Detecting misogyny and xenophobia in spanish tweets using language technologies. ACM Transactions on Internet Technology **20**(2), 1–19 (2020). https://doi.org/10.1145/3369869
42. Plaza-Del-Arco, F.M., Molina-González, M.D., Ureña-López, L.A., Martín-Valdivia, M.T.: Comparing pre-trained language models for spanish hate speech detection. Expert Systems with Applications **166**, 114120 (2021). https://doi.org/10.1016/j.eswa.2020.114120
43. Rish, I., et al.: An empirical study of the naive bayes classifier. In: IJCAI 2001 workshop on empirical methods in artificial intelligence. vol. 3, pp. 41–46 (2001)
44. Sangiorgio, M., Dercole, F.: Robustness of lstm neural networks for multi-step forecasting of chaotic time series. Chaos, Solitons Fractals **139**, 110045 (2020). https://doi.org/https://doi.org/10.1016/j.chaos.2020.110045, bluehttps://www.sciencedirect.com/science/article/pii/S0960077920304422
45. Saxena, R.: How the naive bayes classifier works in machine learning. Data Science, Machine Learning (2017)
46. Shacklett, M.E.: What is dropout? understanding dropout in neural networks (Mar 2021), bluehttps://searchenterpriseai.techtarget.com/definition/dropout
47. Singh, S., Sachan, M.: Importance and challenges of social media text. International Journal of Advanced Computer Research **8**, 831–834 (04 2017). https://doi.org/10.26483/ijarcs.v8i3.3108
48. Suthaharan, S.: Support vector machine. In: Machine learning models and algorithms for big data classification, pp. 207–235. Springer (2016)
49. Team, K.: Keras documentation: Spatialdropout1d layer, bluehttps://keras.io/api/layers/regularization_layers/spatial_dropout1d/
50. Tong, S., Koller, D.: Support vector machine active learning with applications to text classification. Journal of machine learning research **2**(Nov), 45–66 (2001)
51. Whittaker, E., Kowalski, R.M.: Cyberbullying via social media. Journal of School Violence **14**(1), 11–29 (2015). https://doi.org/10.1080/15388220.2014.949377, bluehttps://doi.org/10.1080/15388220.2014.949377

52. Xu, J.M., Jun, K.S., Zhu, X., Bellmore, A.: Learning from bullying traces in social media. In: Proceedings of the 2012 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. pp. 656–666. Association for Computational Linguistics, Montréal, Canada (Jun 2012), bluehttps://www.aclweb.org/anthology/N12-1084
53. Zhang, Y., Jin, R., Zhou, Z.H.: Understanding bag-of-words model: a statistical framework. International Journal of Machine Learning and Cybernetics **1**(1-4), 43–52 (2010)
54. Zhao, Z., Zhang, Z., Hopfgartner, F.: A comparative study of using pre-trained language models for toxic comment classification. In: Companion Proceedings of the Web Conference 2021. pp. 500–507 (2021)