

# Hate speech detection in Italian tweets

Group 4

## 1 INTRODUCTION

### 1.1 Motivation

Although hate speech is a phenomenon that existed long before the advent of the Web, communication via the Internet has allowed this phenomenon to spread and develop with peculiar characteristics tied to the nature of online media, such as anonymity on one side, and velocity and breadth of proliferation on the other. Given the relevance and dangerous potential of this social phenomenon, there is widespread interest in recognizing and detecting hate speech online, also on an institutional level.

In May 2016 the European Commission, citing the chilling effect of hate speech on the democratic discourse on online platforms, agreed with Facebook, Microsoft, Twitter and YouTube on a “Code of conduct on countering illegal hate speech online” [3]. In this document the major IT Companies made a public commitment to:

1. Have in place clear and effective processes to review notifications regarding hate speech on their platforms;
2. review the majority of valid notifications of hate speech in less than 24 hours.

The most recent evaluation campaign of the Code of Conduct [7] assessed in general a worse performance of the IT Companies especially in this last respect (only 64.4% of notifications were reviewed in less than 24 hours), confirming a troubling downward trend already observed in 2021.

Accurate, precise and efficient automated systems of hate speech detection are necessary for combating the phenomenon both with proactive action and in the process of reviewing user notification of occurrences of such violations.

Moreover, such systems should prove valuable for providing the data that is still needed to investigate the ecosystem of hate speech online and the magnitude of the phenomenon.

### 1.2 Project goal

The goal of this project is to identify characteristics in the text which allow for the detection of Italian tweets containing hate speech (this category is meant to include expressions of racism, xenophobia and terrorist propaganda). So, we can define this task as a binary classification.

	HS	Not HS	TOTAL
Train	2766	4073	6839
Test Tweets	622	641	1263
Test News	181	319	500

Table 1: Distribution of Hate Speech labels.

A secondary goal was to evaluate how insights obtained by our model can generalize across textual domains. In order to do that, tested our models, trained exclusively on tweets, against Italian newspaper headlines.

The project is based on the main task of the HaSpeeDe 2 shared task presented at EVALITA2020 [8].

## 2 METHODS

### 2.1 Data

We used the datasets made available<sup>1</sup> by the organizers of the Evalita 2020 shared task HaSpeeDe 2 [8].

The train set consists in 6839 Italian Tweets posted between October 2016 and May 2019 annotated for presence of hate speech. The test set includes both a corpus of 1263 Italian Tweets posted between January and May 2019 and a corpus of 500 Italian newspapers’ headlines retrieved between October 2017 and February 2018 from the online editions of *La Stampa*, *La Repubblica*, *Il Giornale* and *Liberquotidiano*. This second corpus allowed us to appraise how well the application generalizes across textual domains.

Table 1 shows the distribution of hate speech labels in the training set and in each of the test sets.

The data sets are available in TSV format and contain three features for each record:

- id: numeric identifier for each document
- text: the body of the document
- hs: boolean value, whether the document contains HS (1) or not (0).
- stereotype: boolean value, whether the document contains a stereotype (1) or not (0).

As part of the preprocessing performed by the organizers of the task, mentions and URLs recurring in the original documents were replaced with @user and URL placeholders.

<sup>1</sup> <https://github.com/msang/haspeede/tree/master/2020>

In order to model the semantic properties of the documents, we used the Italian Twitter embeddings [1] lexicon computed by ItaliaNLP Lab on a corpus of 50,000,000 tweets using the word2vec<sup>2</sup> toolkit. This lexicon consists of embeddings of 128 features for 1,188,949 tokens that were computed with the CBOW model with a symmetric context window of 5 tokens.

The embeddings were made available<sup>3</sup> by the ItaliaNLP Lab as a SQLite database containing a single table store with 130 columns:

- key, representing the token;
- one column for each dimension of the embedding;
- ranking storing the frequency rank of the token.

Finally we used two resources for identifying offensive words: a lexicon of 500 bad words for the Italian language made available on GitHub<sup>4</sup> and the Italian section of the Revised HurtLex lexicon [10] consisting in 7920 words. In the Revised HurtLex lexicon, each headword is annotated with an offensiveness level score derived from ratings provided by a large number of annotators. This resource was made available from the authors in TSV format<sup>5</sup> and was revised by us in order to solve some data quality issues.

## 2.2 Automatic text annotation

We automatically annotated the documents using a Stanza [6] pipeline: this operation consisted of:

- Tokenization
- Part of speech tagging
- Lemma extraction
- Dependency parsing

Before this step we preprocessed the text in order to remove mentions, urls and normalize the punctuation: in particular sequences of multiple full stops were reduced to one full stop, since they were creating issues with the sentence tokenization. As for hashtags, since often single words hashtags are integrated in the syntax of the sentence we decided to remove only the hash symbol and preserve the textual part of the hashtags.

Our choice of using Stanza was predicated on the possibility of using a combined model based on several different treebanks for the Italian language. This allowed us to get a better coverage of the language, which is particularly important when working on a non standard variety like tweets or in general computer mediated writing. In particular, the combined model is based on four treebanks ISDT, VIT, PoSTWITA and TWITTIRO: the latter two were developed specifically for the annotation of tweets.

<sup>2</sup> <http://code.google.com/p/word2vec/>

<sup>3</sup> <http://www.italianlp.it/download-italian-twitter-embeddings/>

<sup>4</sup> <https://github.com/napolux/paroleitaliane/tree/master/paroleitaliane>

<sup>5</sup> [https://github.com/valeriobasile/hurtlex/blob/master/revised\\_hurtlex/IT/revised\\_hurtlex.tsv](https://github.com/valeriobasile/hurtlex/blob/master/revised_hurtlex/IT/revised_hurtlex.tsv)

## 2.3 Linguistic profiling

With the objective of identifying, if possible, stylistic features useful for hate speech detection, we developed 91 different features for the linguistic profiling of the text belonging to five different categories:

- **Raw text properties:** text length in terms of number of tokens and number of sentences. Average length of sentences and words as a really simple proxy of syntactic complexity;
- **Social media language properties:** distribution of URLs, hashtags, mentions and all uppercase words;
- **Lexical variety** expressed in terms of Type/Token Ratio;
- **Morpho-syntactic properties**, namely the distribution of all grammatical categories and lexical density (proportion of content words in the text);
- **Syntactic properties:** distribution of all dependency relationships and relative order of core arguments and verb in sentences. Another aspect that received some attention was the distribution of nominal sentences, since the literature suggested a strong association between Slogan-like nominal utterances and hateful content [2].

In order to get some insights in the challenges associated with the cross-domain task, we scraped headlines from the online editions of *La Stampa*, *La Repubblica*, *Il Giornale* and *Liberquotidiano*. We filtered the headlines for the presence of the same small set of neutral keywords associated with immigrants, muslims and Roma used for the creation of the original dataset of tweets [4].

We thus obtained 748 newspapers headlines, which we passed to the automatic text annotation pipeline and analyzed stylistically. We then trained a Linear Support Vector Machine on the linguistic profiling features in order to determine which features contributed the most in distinguishing between tweets and headlines. The classifier was able to distinguish tweets from headlines with near certainty, obtaining a macro-F1 of 0.949 in 5-fold cross validation.

As expected, social media properties like the number of mentions, hashtags, URLs and the use of uppercase, were mainly responsible for the success of the classifier. In addition to that, the feature importance analysis seems to indicate that tweets tend to be longer and have longer sentences than headlines; on the other hand, headlines seem to use more punctuation and more proper nouns and exhibit higher lexical variety (but since the Type/Token Ratio measure is highly sensitive to text length, this is probably a spurious result). Figure 1 shows the distributions of several linguistic profiling features (document length in tokens, average sentence length, distribution of punctuation, distribution of proper nouns and Type/Token Ratio by form) for tweets and headlines. Although the distributions

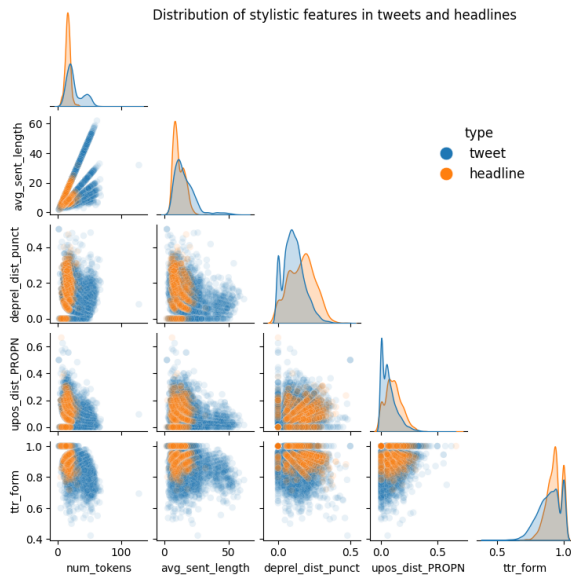


Figure 1: Comparative stylistic analysis of tweets and scraped headlines.

are overlapping, it is possible to discern clear trends that confirm the results of the feature importance analysis.

## 2.4 Ngrams extraction and modelling

In this section, we describe the process used to extract n-grams from tweets to exploit them as features. The process begins with a pre-processing step applied to the raw training dataset, implemented through the `clean_df` function, which consists of three inner functions.

The first inner function, `count_uppercase`, calculates the number of uppercase characters in the dataset. When we considered the stylistic features, we successfully replaced this simple counter with a function that computes ratio of all uppercase words in a document.

Additionally, we defined a custom punctuation set that extends the standard `string.punctuation` by including special characters encountered in tweets. After applying this function, the dataset includes the original columns, `text` and `hs`, along with a new column, `uppercase_count`.

The second inner function, `split_hashtags`, utilizes the `WordSegment` library to split words within hashtags. This function is stochastic, meaning it may split hashtags differently during each execution. Based on our observations, the function performs well in identifying and splitting longer words within hashtags but struggles with stopwords and shorter words. This second inner function is incorporated into the `process_hashtags` function, which tokenizes hashtags when more than one word is present. An additional condition was added to handle words longer than 10 characters, although this could occasionally split valid long words incorrectly. However, given the rarity of such words in the dataset, this approach is effective for

identifying incorrectly joined words. At this stage, the dataset includes a new column, `text_processed`.

Subsequently, all text is converted to lowercase, mentions, URLs, and numbers are removed, and the tweets are tokenized using *Stanza* [6]. Tokens are then further processed by removing stopwords and punctuation. After tokenization, two additional columns are added: one reporting the final number of tokens per tweet and another counting the occurrence of bad words, if any, from the list of 500 Italian bad words.

Once the tokenized version of the tweets was prepared, we lemmatized the tokens to ensure higher consistency and reduce the number of final n-grams. Given the brevity of tweets, we limited the n-grams to unigrams and bigrams.

The final dataset consisted of 6837 rows and 6366 features, including the original tweet, tokenized version, lemmatized version, the number of uppercase characters, the count of bad words, and extracted unigrams and bigrams. Before extracting additional features and embeddings, we conducted initial experiments by training models directly on this dataset. Several models were tested, including Random Forest, SVM, XGBoost, and Logistic Regression. After applying random search for hyperparameter optimization, the best performance was achieved with XGBoost, yielding a macro average F1 score of 0.759.

The same preprocessing steps were applied to both test sets: one containing tweets and the other consisting of 500 headlines from Italian newspapers. To ensure consistency between the test datasets and the training set, we used the dictionary, the Bigram model, and the TF-IDF model trained on the training set. As a result, the final tweets test dataset has the same number of features as the training set, minus one (the target class). This ensures that the model trained on the training dataset can be correctly applied to the test set. Specifically, if an n-gram is present in both the training and test sets, the corresponding column will report non-zero values; otherwise, the column will consist entirely of zeros. Additionally, by using the same TF-IDF model for both training and testing, the weight of each n-gram remains consistent. The final dataset consisted of 6837 rows and 6366 features, including the original tweet, tokenized version, lemmatized version, the number of uppercase characters, the count of bad words, and extracted unigrams and bigrams. Before extracting additional features and embeddings, we conducted initial experiments by training models directly on this dataset. Several models were tested, including Random Forest, SVM, XGBoost, and Logistic Regression. After applying random search for hyperparameter optimization, the best performance was achieved with XGBoost, yielding a macro average F1 score of 0.759.

## 2.5 Embeddings

Among the approaches we decided to implement, we opted for an SVM classifier trained using embeddings-based textual representations.

	FEATURES	MACRO F1
1	Avg over all tokens	0.744
2	Avg over all N, V and Adj	0.726
3	Concat. of mean vects (N, V, Adj)	0.717

Table 2: SVM classifiers based on different word embeddings aggregations (5-fold cross-validation).

We first tested three different representations based on the same lexicon of precomputed static word embeddings, the Italian Twitter Embeddings [1] by the ItaliaNLP Lab, differing by method of aggregation.

In order to do so, we loaded the embeddings and our preprocessed and annotated training documents, which we further normalized using the functions provided by the ItaliaNLP lab. We could then build the vocabulary of the training set using the embeddings. Given the characteristics of this type of non-standard language, in particular the high level of both inter and intrapersonal variation, we expected some out-of-vocabulary words, which ended up being 2703 out of 22709, meaning that almost 12% of words didn't have an embedding representation. These words are mostly words that haven't been correctly tokenized (especially multi-words hash-tags), sequences of emojis, and typos. Had we used non-specialized embeddings, we assume that this coverage would have been significantly lower, posing an important issue for the model training.

Using a 5-fold cross-validation process and comparing the resulting macro-averaged F1 score, we determined the best performing model to be the one representing each tweet with the average of all its word embeddings. It performed better than using the aggregation of only lexically-full tokens (nouns, verbs and adjectives), both by mean and by concatenation of the mean vectors for the three aforementioned grammatical categories.

We then followed the same methodology to test representations based on contextual embeddings extracted from two different Italian pre-trained BERT models.

1. **dbmdz/bert-base-italian-cased** or (**Italian BERT**) [9]: trained on a corpus consisting of a Wikipedia dump and texts from the OPUS corpora collection, for a total of 2,050,057,573 tokens.
2. **m\_polignano\_uniba/bert\_uncased\_L-12\_H-768\_A-12\_italian\_alb3rt0** (or **AlBERTo**) [5]: the first Italian BERT model for Twitter language understanding, trained on Italian tweets.

For both models, we tested two methods of *pooling* the tokens' embedding representations generated by the model after processing the input sequence into a single sentence embedding for each tweet (or, more generally, document).

1. **[CLS] pooling**: the model's tokenizer adds the special token [CLS] at the start of every sequence. During the training process of the Large Language Model, its embedding (or hidden state) will capture sequence-level information. By extracting the [CLS] token's embedding of each document from

	[CLS]	MEAN POOLING
Italian BERT	0.689	0.725
AlBERTo	0.741	0.747

Table 3: Mean macro F1 score of SVM classifiers based on different contextual embeddings' poolings (5-fold cross-validation).

the final hidden layer, we obtain a representation that can serve as a sentence-level embedding.

2. **Mean Pooling with Sentence Transformers**: we used a Sentence Transformer, a model that generates a single fixed-length vector representation for an entire sequence. It does so via *mean pooling*, i.e. by computing the mean of all tokens' embeddings in the last hidden layer.

The Italian BERT yielded worse performances, which can be explained by the fact that it is a general model that has not been trained for this specific domain. AlBERTo, on the other hand, provided better results, with the SVM model based on its mean pooled embeddings having the highest mean macro F1 score.

Once we decreed our best performing model, we proceeded to test it on our two test sets, bearing in mind that it is a relatively simple model, using only the text itself as a feature.

	precision	recall	f1-score	support
0	0.741	0.718	0.729	641
1	0.718	0.741	0.729	622
<b>accuracy</b>			0.729	1263
<b>macro avg</b>	0.729	0.729	0.729	1263
<b>weighted avg</b>	0.730	0.729	0.729	1263

Table 4: Classification Report on Tweets test set

	precision	recall	f1-score	support
0	0.730	0.881	0.798	319
1	0.670	0.425	0.520	181
<b>accuracy</b>			0.716	500
<b>macro avg</b>	0.700	0.653	0.659	500
<b>weighted avg</b>	0.708	0.716	0.698	500

Table 5: Classification Report on News headlines test set

The macro F1 drops from 0.747 to 0.729 in the in-domain task, but all metrics are balanced between the two classes. The model seems to be able to generalize well on tweets and has a stable performance.

As expected, the performance drop is more significant for the out-of-domain task, with a macro F1 of 0.659. The model performs worse for class 1, especially with respect to F1 and recall, which means that it struggles with detecting hate speech in news headlines.



## 2.6 Handling Emoji

### 2.6.1 Introduction

Analyzing emoji in Italian tweets is essential for understanding expressiveness, sentiments, and the main topics shared online. Emoji represent a unique form of communication that complements textual language with visual symbols to convey emotions, emphasis, and cultural context. Our objectives are therefore to:

- Identify emoji usage patterns in Italian tweets.
- Analyze the frequency and distribution of emoji.
- Explore the semantic meaning of emoji in relation to textual content.
- Apply clustering techniques to categorize tweets based on emoji usage.

### 2.6.2 Dataset and Preprocessing

To ensure the quality of the analysis, the dataset was pre-processed:

1. **Removal of superfluous elements:** Mentions (@user), URLs, hashtags, and non-decodable characters were removed.
2. **Text normalization:** All text was converted to lowercase, and redundant spaces were removed.
3. **Tokenization:** The text was separated into individual words or symbols for detailed analysis.
4. **Emoji extraction:** The emoji library was used to identify and isolate emoji symbols from the text.

### 2.6.3 Extraction and Analysis

We performed two different extractions—one for emoticons and one for emoji. For the former, it was necessary to set a pattern to identify them; for the latter, it was carried out via tokenization and recognition patterns provided by the emoji library. The following analyses were conducted:

- **Emoji counting:** Calculation of the frequency of each emoji in the dataset.
- **Meaning mapping:** Linking emoji to their descriptive names and semantic meanings.
- **Emoji distribution:** Statistical analysis to identify the most common emoji.
- **Relationship with textual content:** Exploration of the correlation between emoji usage and the topics discussed in tweets.

Top 10 Emojis by frequency

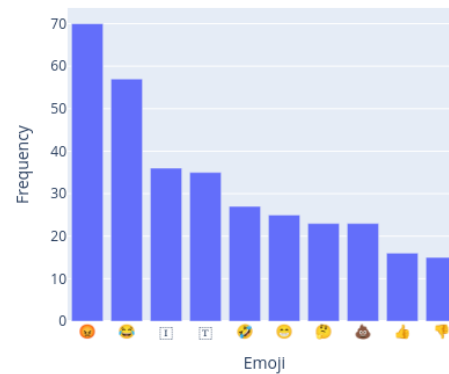


Figure 2: Distribution of the most frequent emoji in tweets.

### 2.6.4 Visualization and Results

The main visualizations developed include:

1. **Emoji frequency:** A bar chart showing the most used emoji in the dataset.
2. **Clustering:** Use of algorithms such as K-Means and PCA to identify groups of tweets based on common emoji usage patterns.

As we can see from the bar chart, the most frequently used emoji is the enraged face. This chart is very helpful in understanding the trend among the tweets under analysis. As expected, the most prominent sentiment is anger. Subsequent results are also significant, showing the laughing face and then the “IT” pair of emoji. When combined, these emoji might suggest the prevalence of sarcasm, anger, and blame directed toward the state.

### 2.6.5 TF-IDF for Emoji and Emoticon Analysis

One of the key techniques applied was the use of TF-IDF (Term Frequency–Inverse Document Frequency) to analyze the meaning and importance of both emoji and emoticons in relation to the textual content of the tweets. The process was structured as follows:

1. **Text tokenization:** Each tweet was divided into tokens, including both emoji and emoticons.
2. **Generation of n-grams:** For each emoji-emoticon, n-grams (from 1-gram to 5-gram) were generated to analyze combined usage patterns.
3. **Frequency calculation:** The relative frequency (TF) of each n-gram was calculated within each tweet.
4. **Inverse weighting calculation:** The relative importance of each n-gram in the corpus was calculated, penalizing those present in all tweets (IDF).
5. **Vectorization:** Each tweet was represented as a TF-IDF vector, where each dimension corresponds to an n-gram.

6. **Parameter optimization:** The optimal value of  $n$  (the length of the n-grams) was chosen based on total frequency.

Implementing TF-IDF made it possible to distinguish generic symbols with common meanings from those that are more specific and characterize particular themes or emotions.

### 2.6.6 Results of the TF-IDF Analysis

Applying TF-IDF highlighted the following main n-grams with their corresponding TF-IDF values:

N-gram	TF-IDF
enraged_face	0.003181
italy	0.002182
face_with_tears_of_joy	0.002172
thinking_face	0.001885
xo	0.001316
pile_of_poo	0.001268
rolling_on_the_floor_laughing	0.001136
beaming_face_with_smiling_eyes	0.001036
d	0.001024
enraged_face_enraged_face	0.000979

Table 6: Top n-grams of emoji names with corresponding TF-IDF values.

### 2.6.7 Comparison of Emoji in Tweets With and Without Hate Speech

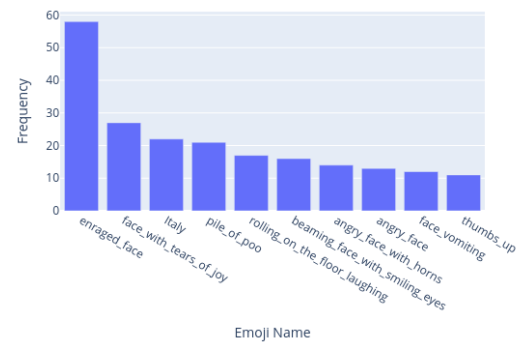
A comparison of the emoji used in tweets with and without hate speech was carried out to analyze differences and specific patterns. The relative frequencies of emoji were calculated in both datasets:

- **Tweets with Hate Speech (HS):** Includes tweets where  $hs = 1$ .
- **Tweets without Hate Speech (NHS):** Includes tweets where  $hs = 0$ .

Emoji frequencies were represented graphically for each group:

As we can see in Figure 3, emoji such as “Italy,” “face\_with\_tears\_of\_joy,” “enraged\_face,” etc., appear in both rankings and also at the top positions. This suggests that the enraged face emoji can sometimes be used to express emotions of disappointment, political fatigue, and distrust regarding the topic of immigration. Conversely, the presence of the laughing emoji in the first ranking is indicative of malevolent sarcasm, mockery, and frustration. A similar observation applies to the Italian flag emoji, which is often used in both types of tweets.

Top 10 emojis in Hate Speech tweets



Top 10 emojis in Non Hate Speech tweets

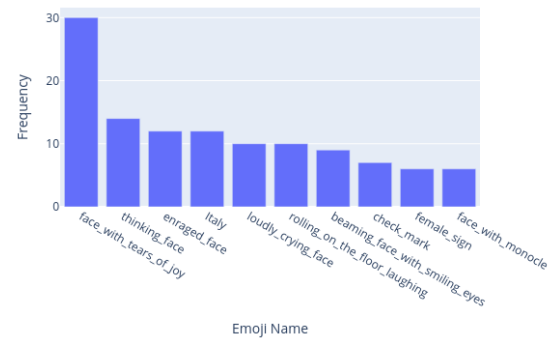


Figure 3: Top 10 Emoji in Tweets With and Without Hate Speech.

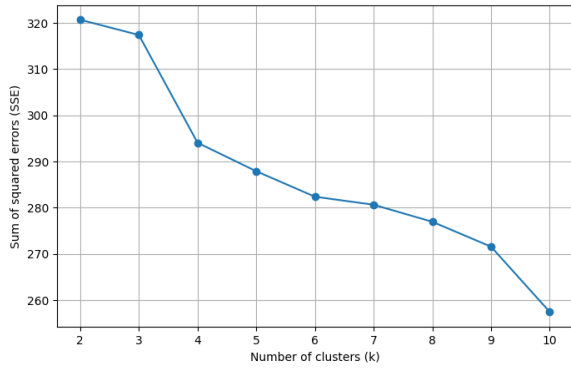


Figure 4: Elbow Method for Determining the Optimal Number of Clusters.

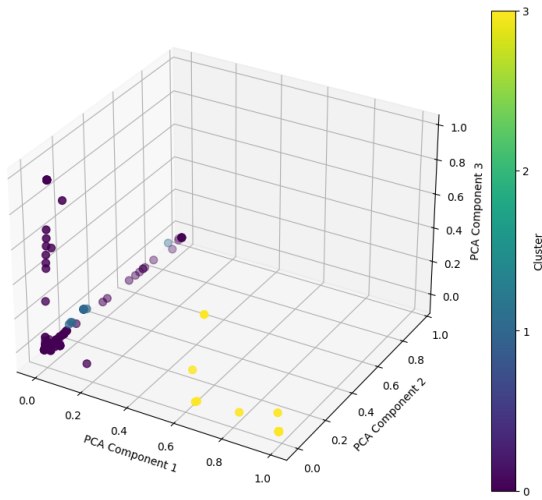


Figure 5: K-Means Emoji Clustering ( $k=4$ ) – 3D Visualization.

### 2.6.8 Tweet Clustering

Clustering was performed to categorize tweets based on emoji and emoticons. The main steps were:

1. **Data transformation:** We used the TF-IDF vectors based on the emoji names created before.
2. **Determining the optimal number of clusters:** The *Elbow* method was used to find the optimal value of  $k$ . Figure 4 shows the Elbow method plot.
3. **Dimensionality reduction:** Principal Component Analysis (PCA) was applied to reduce the data to three dimensions for visualization.
4. **Clustering:** The K-Means algorithm was used to identify four distinct clusters based on emoji usage.

Figure 5 shows the cluster distribution in a three-dimensional space based on the computed principal components.

The results show that emoji are not easily clusterable. Even though the optimal  $k$  found is 4, we end up with only 3 actual clusters. This is due to the nature of the available data. The division seems to be on the vertical layer and, based on the trend of the obtained data, we can hypothesize that the blue cluster contains emojis

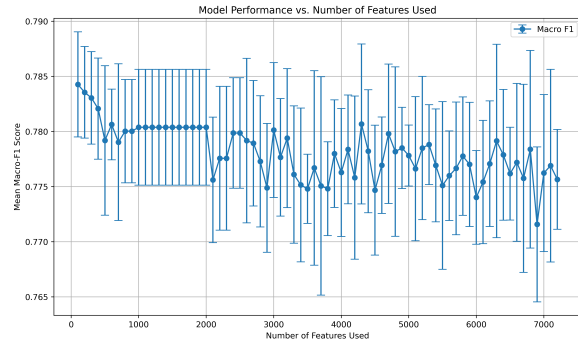


Figure 6: Performance of LGBM classifier vs number of features used by the classifier

with negative sentiment, the yellow cluster contains those with positive sentiment, and the magenta cluster comprises neutral emoji used in tweets featuring hate speech. These are hypotheses since we are not able to derive more relevant information.

### 2.6.9 Conclusions

Analyzing emoji has provided a unique perspective on emotional expressions and immigration-related topics. The most frequent emoji revealed a focus on negative emotions.

## 2.7 Models

The train set used for this phase of exploring various classifiers consists mainly of four datasets: unigrams and bigrams, embeddings, stylistic features, and the offensiveness score. Depending on which embeddings dataset is used the dimensionality may change, but in general it is around 7000 total features.

After tuning the hyperparameters of several classifiers (Logistic Regression, SVM, Random Forest, XG-Boost, Light-GBM) the best one turned out to be Light-GBM (with an average F1-macro of 0.774 on 5 folds of crossvalidation), as well as being clearly the fastest. Thus, we used this first version of the model to carry out several analyses regarding features. For example, with respect to feature importance: although any of the four datasets manages, taken individually for training, to achieve an average F1-macro of at least 0.75, when used all together the model takes into account almost only the embeddings to classify (and a few other interesting features such as the unigrams 'rom' and 'terrorism', the stylistic features 'uppercase\_words\_dist' and 'lexical\_density', and 'offensiveness\_score' and 'badword'). Moreover, thanks to the calculation of feature importance, we were able to observe that the model does not benefit at all from many of our features, as it achieves its highest result (average F1-macro of 0.784) using only the first 100.

Regarding embeddings, since they are apparently the most useful feature set, we trained the model on all the embeddings datasets we calculated (embs\_all\_train, embs\_filt\_train, embs\_filt\_sep\_train, sent\_embs\_cls\_train,

sent\_embs\_pooled\_train, alberto\_sent\_embs\_cls\_train and alberto\_sent\_embs\_pooled\_train) to check if there is any noticeable difference, and we found that the one that performs best is alberto\_sent\_embs\_pooled\_train while the one that performs worst is sent\_embs\_cls\_train.

With regard to the offensiveness score, we found instead that the score normalized by the length of the tweet returned better results than the non-normalized score and than both scores used together.

Finally, after all the information obtained, we performed a more comprehensive randomized search to re-tune the hyperparameters of the Light-GBM in order to have a definitive model to use for testing (the model reached an average F1-macro of 0.778 on the validation set). For the generalisation task, since the stylistic features are very different between tweets and newspaper headlines, we decided to perform a second tuning to generate another model that is not trained on stylistic features.

### 3 RESULTS

### 4 DISCUSSION

### 5 CONCLUSION

### REFERENCES

- [1] Andrea Cimino, Lorenzo De Mattei, and Felice Dell’Orletta. “Multi-task Learning in Deep Neural Networks at EVALITA 2018”. In: *EVALITA Evaluation of NLP and Speech Tools for Italian*. Ed. by Tommaso Caselli et al. Torino: Accademia University Press, 2018, pp. 86–95. ISBN: 978-88-319-7842-2 978-88-319-7869-9. DOI: [10 . 4000 / books . aaccademia . 4527](https://books.aaccademia.4527). URL: <https://books.openedition.org/aaccademia/4527> (visited on 06/04/2024).
- [2] Gloria Comandini and Viviana Patti. “An Impossible Dialogue! Nominal Utterances and Populist Rhetoric in an Italian Twitter Corpus of Hate Speech against Immigrants”. In: *Proceedings of the Third Workshop on Abusive Language Online*. Proceedings of the Third Workshop on Abusive Language Online. Florence, Italy: Association for Computational Linguistics, 2019, pp. 163–171. DOI: [10 . 18653 / v1 / W19 - 3518](https://www.aclweb.org/anthology/W19-3518). URL: <https://www.aclweb.org/anthology/W19-3518> (visited on 12/04/2024).
- [3] European Commission. *The EU Code of conduct on countering illegal hate speech online*. URL: [https://commission.europa.eu/strategy-and-policy/policies/justice-and-fundamental-rights/combating-discrimination-racism-and-xenophobia/eu-code-conduct-countering-illegal-hate-speech-online\\_en](https://commission.europa.eu/strategy-and-policy/policies/justice-and-fundamental-rights/combating-discrimination-racism-and-xenophobia/eu-code-conduct-countering-illegal-hate-speech-online_en) (visited on 10/11/2024).
- [4] Fabio Poletto et al. “Hate Speech Annotation: Analysis of an Italian Twitter Corpus”. In: *Proceedings of the Fourth Italian Conference on Computational Linguistics CLiC-it 2017*. Ed. by Roberto Basili, Malvina Nissim, and Giorgio Satta. Torino: Accademia University Press, 2017, pp. 263–268. ISBN: 978-88-99982-94-2. DOI: [10 . 4000 / books . aaccademia . 2448](https://books.aaccademia.2448). URL: <https://books.openedition.org/aaccademia/2448> (visited on 12/12/2024).
- [5] Marco Polignano et al. “AIBERTo: Italian BERT Language Understanding Model for NLP Challenging Tasks Based on Tweets”. In: *Proceedings of the Sixth Italian Conference on Computational Linguistics (CLiC-it 2019)*. Vol. 2481. CEUR, 2019. URL: <https://www.scopus.com/inward/record.uri?eid=2-s2.0-85074851349&partnerID=40&md5=7abed946e06f76b3825ae5e294ffac14>.
- [6] Peng Qi et al. “Stanza: A Python Natural Language Processing Toolkit for Many Human Languages”. In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics: System Demonstrations*. 2020. URL: <https://nlp.stanford.edu/pubs/qi2020stanza.pdf>.
- [7] Didier Reynders. *7th evaluation of the Code of Conduct*. Fact-sheet. European Commission, Nov. 2022. URL: [https://commission.europa.eu/document/download/5dcc2a40-785d-43f0-b806-f065386395de\\_en?filename=Factsheet%20-%207th%20monitoring%20round%20of%20the%20Code%20of%20Conduct.pdf](https://commission.europa.eu/document/download/5dcc2a40-785d-43f0-b806-f065386395de_en?filename=Factsheet%20-%207th%20monitoring%20round%20of%20the%20Code%20of%20Conduct.pdf).
- [8] Manuela Sanguinetti et al. “HaSpeeDe 2@ EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task”. In: *Proceedings of the 7th evaluation campaign of Natural Language Processing and Speech tools for Italian (EVALITA 2020)*. Ed. by Valerio Basile et al. Online: CEUR.org, 2020.
- [9] Stefan Schweter. *Italian BERT and ELECTRA models*. Version 1.0.1. Nov. 2020. DOI: [10 . 5281 / zenodo . 4263142](https://doi.org/10.5281/zenodo.4263142). URL: <https://doi.org/10.5281/zenodo.4263142>.
- [10] Alice Tontodimamma et al. “An Italian lexical resource for incivility detection in online discourses”. In: *Quality & Quantity* 57.4 (Aug. 2023), pp. 3019–3037. ISSN: 0033-5177, 1573-7845. DOI: [10 . 1007 / s11135 - 022 - 01494 - 7](https://link.springer.com/10.1007/s11135-022-01494-7). URL: <https://link.springer.com/10.1007/s11135-022-01494-7> (visited on 11/29/2024).