A Comparative Study of Machine Learning Models for Hate Speech and Stereotype Detection in Italian Texts

Vincenzo Sammartino

Dipartimento di Informatica - Università Di Pisa,
v.sammartino@studenti.unipi.it

Abstract This study presents a comparative analysis of various machine learning models for hate speech and stereotype detection in Italian texts. The research utilizes datasets from the HaSpeeDe task proposed by EVALITA in 2020. Multiple text representation techniques are evaluated, including nonlexical linguistic information, Bag of Words, n-grams (characters, words, and Part-of-Speech tags), Word Embeddings, and a Neural Language Model (BERT). The study compares the performance of these models in different metrics such as accuracy, precision, recall, and F1-score. The results indicate that character n-grams and the Neural Language Model (BERT) generally outperform other techniques, with BERT achieving the highest accuracy (76%) for the detection of hate speech and character n-grams performing the best for the detection of stereotypes (72% accuracy). The research highlights the challenges in detecting stereotypes compared to hate speech and emphasizes the importance of context in classification tasks.

Biographical notes: Vincenzo Sammartino is a PhD in Artificial Intelligence at University of Pisa with a solid background in Digital Humanities from the University of Pisa, where he obtained top marks. His academic career is enriched by cutting-edge research experiences, including the development of distributed and secure systems and the decomposition of databases in compliance with GDPR regulations. Currently, he is engaged in a research contract at the University of Pisa and Siena, focusing on the realization of Digital Twin-related projects in the cybersecurity field and transcription and encoding of ancient texts. His most recent publications include studies on data security and minimizing the impact of computer intrusions. His passion for innovation drives him to continuously explore new technological frontiers while maintaining an ethical and sustainable approach.

1 Introduction

Hate speech has become an increasingly prevalent issue in online platforms, necessitating the development of automated tools for its detection and moderation. Hate speech refers to language that disparages a person or group based on attributes such as race, religion, ethnicity, gender, sexual orientation, or disability. The identification of hate speech is crucial to mitigating its harmful effects on individuals and society. However, the detection of hate speech remains a challenging task in the field of natural language processing (NLP), primarily due to its context-dependent nature, the subtlety of the language used, and the frequent overlap with other forms of abusive language such as offensive speech or cyberbullying [18, 17, 5].

One of the key challenges in hate speech detection is the difficulty in defining and categorizing hate speech, which varies across legal, cultural, and linguistic contexts. As a result, detection models often struggle to maintain high performance across diverse languages, topics, and platforms. Furthermore, hate speech can manifest in implicit or indirect forms, making it difficult for traditional machine learning algorithms to capture its nuances. Despite these challenges, advances in machine learning, particularly deep learning, have shown promising results in improving the accuracy and generalizability of hate speech detection models across various domains [10, 13].

This report presents the results of a comparative analysis of several models applied to the *HaSpeeDe* (Hate Speech Detection Task) [16], an initiative launched as part of the EVALITA competition in 2020. EVALITA is a platform dedicated to promoting the development of tools for the automatic processing of the Italian language, particularly through competitive evaluation and the provision of linguistic resources [15]. The HaSpeeDe dataset, specifically tailored for Italian texts, poses unique challenges due to the language's rich morphology and syntactic variability. Unlike many hate speech datasets that primarily focus on English, this dataset is designed to detect hate and stereotypes in Italian, adding another layer of complexity to the classification task [2, 3, 14].

Common methods for hate speech detection include traditional machine learning techniques such as Support Vector Machines (SVM), Naïve Bayes, and Logistic Regression, which have been extensively used in early studies on text classification. In recent years, deep learning approaches like Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and transformers such as BERT have gained popularity due to their superior performance in capturing complex patterns in text data [7, 1, 13]. These models leverage word embeddings, contextual information, and pre-trained language models to improve the detection of hate speech, particularly in multilingual contexts [8, 9, 11].

In this study, we aim to evaluate and compare the performance of several machine learning models on the HaSpeeDe dataset. This evaluation seeks to identify the most effective approaches for detecting hate speech and stereotypes in Italian, with a focus on balancing accuracy, computational efficiency, and generalizability across different types of text. The following sections will discuss the datasets used, the preprocessing techniques employed, and the machine learning models implemented for text classification [12, 6].

2 Related Work

Hate speech and stereotype detection in Natural Language Processing (NLP) have seen rapid advancements, driven by improvements in both traditional machine learning and deep learning techniques. This section provides an overview of key contributions, highlighting the strengths and weaknesses of various approaches with a particular focus on Italian-language detection tasks.

Early studies, such as [17], laid the groundwork for automated hate speech detection using manual feature extraction and machine learning classifiers like Logistic Regression and Support Vector Machines (SVM). While these models performed reasonably well on domain-specific datasets, their effectiveness was limited by the need for extensive feature engineering and their inability to generalize across different languages and contexts. This reliance on manual feature extraction was a significant limitation as it often failed to capture the complexities of nuanced hate speech.

[5] introduced a more sophisticated approach by incorporating a labeled dataset that distinguished between hate speech, offensive language, and neutral language. Their model, based on n-grams and Logistic Regression, improved classification accuracy by enabling a more granular understanding of online discourse. However, traditional machine learning models like theirs still struggled with context-dependent tasks and often misclassified implicit forms of hate speech.

In the Italian context, the HaSpeeDe 2 task by [15] marked a significant contribution to hate speech detection in non-English languages. The task provided a dedicated dataset for Italian texts, allowing researchers to explore the language-specific challenges of detecting hate and stereotypes. One key challenge identified was the morphological complexity of Italian, which impacts the accuracy of traditional models. [14] further compiled resources for Italian hate speech detection, identifying significant gaps in the coverage and diversity of datasets. While these resources provided valuable starting points, they often lacked the linguistic and contextual depth necessary for detecting more subtle forms of hate speech and stereotypes.

Recent advances have focused on deep learning techniques. [13] provided a thorough review of deep learning methods such as Convolutional Neural Networks (CNNs), Recurrent Neural Networks (RNNs), and transformer-based models like BERT. Pretrained language models, particularly BERT and its multilingual variants, have proven highly effective in context-dependent tasks such as hate speech detection. These models capture complex syntactic and semantic relationships, significantly improving the detection of subtle and implicit hate speech. However, their main drawback lies in the computational resources required for training and inference, making them less feasible for real-time applications on large-scale social media platforms.

The detection of stereotypes, particularly in non-English languages, remains an underdeveloped area. [7] emphasized the complexity of stereotype detection due to the inherent subtlety and cultural specificity of stereotypical language. While advanced models like BERT show promise in capturing such nuances, they still struggle with highly context-specific references that require deep cultural and social understanding. Moreover, the lack of comprehensive datasets dedicated to stereotype detection limits the ability to fine-tune models for this specific task.

Overall, deep learning techniques, particularly transformer-based models, demonstrate considerable potential for improving hate speech and stereotype detection. However, their primary limitations—high computational cost and the scarcity of high-quality, diverse datasets—remain significant challenges. Additionally, the subtle and context-dependent nature of stereotypes continues to present a major obstacle for automated systems.

3 Datasets

Several datasets were used to perform the tasks in TSV (Tab Separated Value) format [1] whose distributions are shown in the tables 1 and 3.

The data are formatted as follows:

```
id text hs stereotype
```

1. A unique number replacing the original tweet ID
2. The text of the tweet

Dataset	HS	Not HS	Total
Training Set	2766	4073	6839
Test Set News	181	319	500
Test Set Tweet	622	641	1263

Table 1 Distribution of Hate Speech in the Training Set and Test Set

Dataset	Stereotype	Not Stereotype	Total	
Training Set		3042	3797	6839
Test Set News		175	325	500
Test Set Tweet		569	694	1263

Table 2 Distribution of Stereotypes in the Training Set and Test Set

3. The class *Hate Speech*: 1 if the text contains hate, 0 otherwise
4. The class *Stereotype*: 1 if the text contains stereotypes, 0 otherwise

The data sets in TSV format were transformed into a txt file containing tweets / news divided by line.

A small pre-processing was performed by removing the occurrences of '@user' and 'URL' in order to obtain cleaner texts.

To transform the features (X), the `MaxAbsScaler`[2].

In this report, for the sake of brevity, only the results obtained in the Tweet dataset are shown; in fact, the features that emerged from the experiments on this dataset are very similar to those obtained from the experiments carried out on the news dataset (for more details, please refer to the two notebooks of news classification)[3].

4 Models and Experiments

All the results of the experiments are listed in the following subsections grouped by the representation model used.

The *baseline* used as a reference was made with a `Dummy Classifier`[4] In its `prior` strategy, it labels all documents with the most frequent class as "not containing hate-speech" (0) and "not containing stereotypes" (0).

For all models, with the exception of the 4.7 section where a Neural Language Model was used, the `LinearSVC` classification model was used: this is an implementation of the *Support Vector Machine* (SVM) classifier based on the `liblinear` library, which is designed to work with linear kernels and better fit a large number of samples than the `libsvm` library. This model supports both dense input[5] (e.g. Word Embeddings) that sparsity[6] (models such as Bag of Words) and handles multiclass support according to a *"one-vs-rest"* (ovr) scheme.

In these experiments we chose to use two separate `LinearSVC` for the binary classification of hate and stereotype, without doing a multiclass classification of the type `[hs, stereotype]`, given the better performance obtained by the individual binary classifiers.

For the `LinearSVC` model, tuning of the hyperparameters was carried out; in particular, considerable variations in performance are obtained when C[7], as can be seen in figure 1:

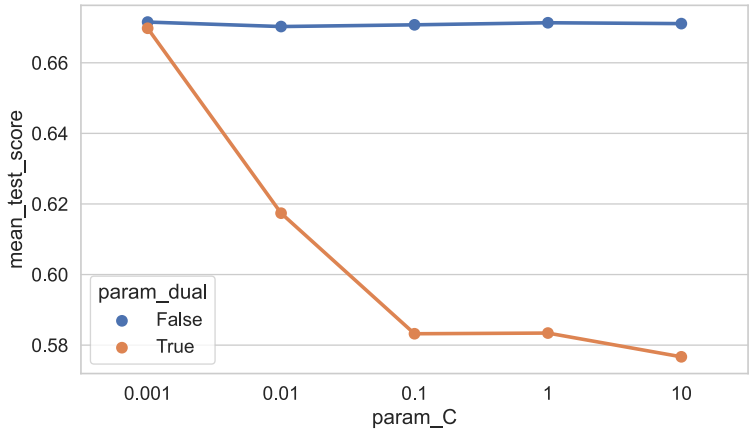

Figure 1 Accuracy when varying the LinearSVC parameter C

After tuning the hyper-parameters on the training set, the following configuration was used for all tests performed:

```
LinearSVC(C=0.001, dual=False)
```

4.1 Not-lexical linguistic information

The first tests were carried out using the non-lexical linguistic information extracted with the ProfilingUD [2] system, a tool for linguistic profiling of texts.

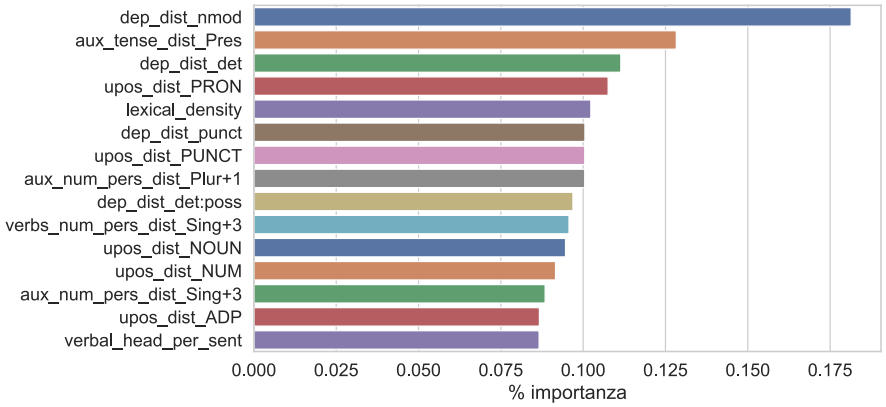

Figure 2 List of the 15 most important features for classification

Figure 2 shows the relative importance of several linguistic and syntactic features analysed in this study. It can be observed that the feature `dep_dist_nmod` (dependency distance between nominal modifiers) has the highest importance (0.181395), followed by

`aux_tense_dist_Pres` (distance between auxiliaries in the present tense) (0.128296) and `dep_dist_det` (dependency distance between determiners) (0.111437).

Interestingly, some characteristics, such as `verbal_head_per_sent` (average number of verbal heads per sentence) (0.086617) and `upos_dist_ADP` (distribution of parts of speech for prepositions) (0.086711), are relatively less important in this analysis.

Task	**5-fold Cross Validation**				
HS	0.6364	0.5929	0.5455	0.5675	0.6032
	Average: 0.5891				
Stereotype	0.5692	0.5968	0.5573	0.5675	0.5516
	Average: 0.5685				

Table 3 Results of the 5-fold cross validation for the two tasks

Metrics	**Accuracy**	**Precision**	**Recall**	**F1-score**
Not HS	0.60	0.61	0.60	0.60
HS		0.59	0.61	0.60
Not Stereotype	0.57	0.60	0.67	0.63
Stereotype		0.53	0.44	0.48

Table 4 Distribution of Stereotypes in the Training Set and Test Set

The results show that non-lexical linguistic information produces similar performance in the two tasks; in the classification of hatred the accuracy is slightly better. Stereotype detection is more complex, in particular the precision, recall and consequently f1-score values are rather low.

4.2 Bag of Words

This section shows the results obtained with the use of the Bag of Words model, one of the simplest and most intuitive techniques for converting text into numerical format. The basic idea is to create a vector of words present in a corpus of texts and count the frequency of each word in a specific document. To illustrate how the BoW model works, let us consider the following example:

Suppose we have the following documents:

- Document 1: "The cat plays in the garden."
- Document 2: "The dog runs in the park."

To transform the text into vectors, the following steps are performed:

1. Create a vocabulary with all the unique words in the documents:

$$Vocabulary = \{The, cat, play, in, garden, dog, run, park\}$$

1. For each document, create a word frequency vector based on the vocabulary:
   - Document 1: $[1, 1, 1, 1, 1, 0, 0, 0]$

- Document 2: $[1, 0, 0, 1, 0, 1, 1, 1]$

Once the text has been transformed into vectors, these are submitted to the classifier in the form of a sparse matrix (`X_train/X_test`).

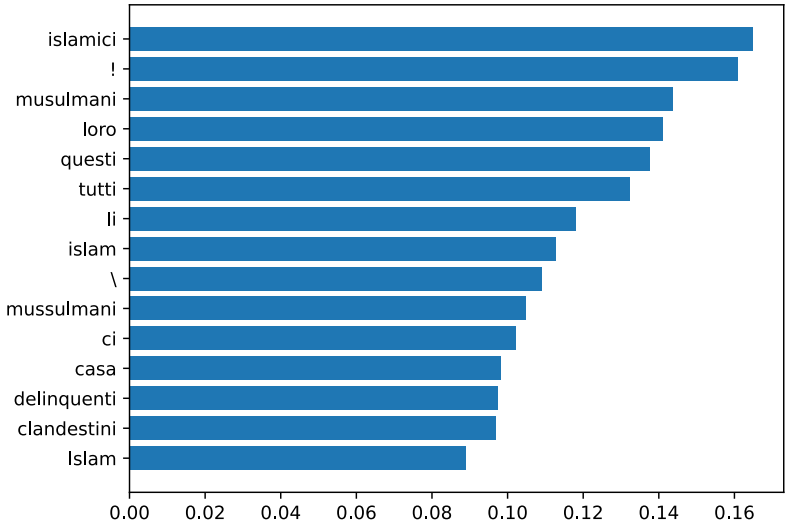

Figure 3 List of the 15 most important features for classification (forms)

Figure 3 shows the 15 features that are the most important for the classifier, ordered by their importance in percentage terms. Among them are many words that could be associated with hate speech and stereotype classification.

In this context, it might be natural to think that it would suffice to create a 'predefined list' of known terms usually associated with hate speech (e.g. Islamists, Muslims, thugs, illegal immigrants, etc.). In reality, it is important to note that the presence of these words alone does not necessarily imply hate speech; in fact, the context in which these words are used is crucial in determining whether it is indeed hate speech or not. This is why it is better to rely on machine learning models such as those reported in this report by also submitting the context of the words (e.g. models using n-gram words).

Task	**5-fold Cross Validation**				
HS	0.7036	0.6838	0.5968	0.5833	0.6270
	Average: 0.6389				
Stereotype	0.6442	0.6245	0.5731	0.6547	0.5873
	Average: 0.6167				

Table 5 Results of the 5-fold cross validation for the two tasks

Metrics	Accuracy	Precision	Recall	F1-score
Not HS	0.69	0.69	0.73	0.71
HS		0.70	0.66	0.68
Not Stereotype	0.65	0.69	0.66	0.68
Stereotype		0.61	0.65	0.63

Table 6 Results of classification with the BoW model on word forms

These tables show that the BoW model on word forms performs significantly better than the model based on non-lexical linguistic information in both tasks.

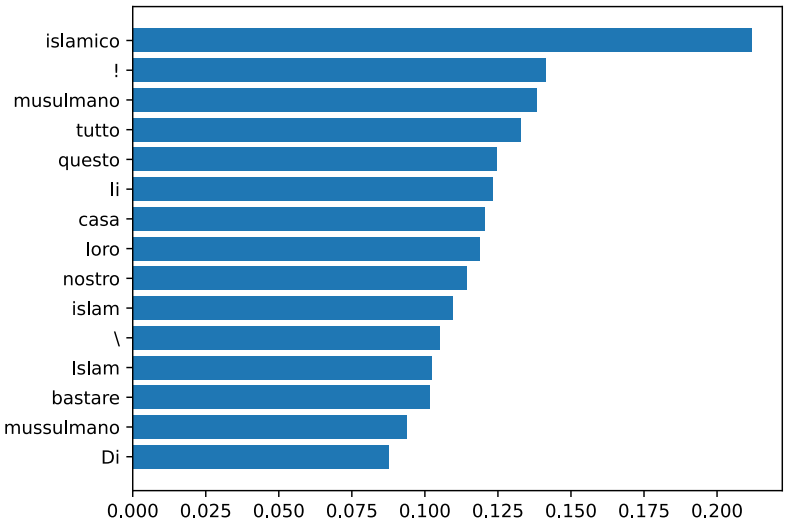

Figure 4 List of the 15 most important features for classification (headwords)

In the figure 3 it is evident that the feature 'Islamic' has a much greater weight than all the others, a phenomenon that is not so evident in the classification using shapes (figure 3); however, it should be emphasised that many features are common to the model using shapes.

Task	5-fold Cross Validation				
HS	0.7589	0.6877	0.6245	0.6111	0.6429
	Average: 0.6650				
Stereotype	0.6759	0.6245	0.5692	0.6627	0.5992
	Average: 0.6263				

Table 7 Results of the 5-fold cross validation for the two tasks

Metrics	Accuracy	Precision	Recall	F1-score
Not HS	0.70	0.69	0.75	0.72
HS		0.72	0.65	0.68
Not Stereotype	0.67	0.71	0.69	0.70
Stereotype		0.63	0.65	0.64

Table 8 Results of classification with the BoW model on word lemmas

The use of lemmas instead of words in the Bag Of Words model brings a slight increase in performance for both tasks.

4.3 N-grams of characters

The Bag of n-grams model is an extension of the Bag of Words model and consists of creating frequency vectors based on sequences of characters instead of single words. The n-grams of characters can better capture the context of words and help recognise patterns and similarities between different documents.

In the experiments conducted for this report, character n-grams up to 6 characters were used. In fact, while increasing the value of the characters does not lead to an improvement in performance, it does lead to a considerable increase in the number of elements inserted in the vocabulary, which leads to a not inconsiderable increase in occupied memory.

The steps taken to create the vectors containing the n-grams of characters with $n = 2$ are shown below:

1. Create a set of n-grams of characters with $n = 2$ for both documents:
   - Document 1: {Il, l_, _g, ga, at, tt, to, o_, _g, gi, io, oc, ca}
   - Document 2: {Il, l_, _c, ca, an, ne, e_, _c, co, or, re}
2. For each document, create an n-character-gram frequency vector based on the union of the sets of n-character-grams:
   - Vocabulary: {Il, l_, _g, ga, at, tt, to, o_, _g, gi, io, oc, ca, _c, co, or, re, an, ne, e_}
   - Document 1: [1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0, 0, 0, 0]
   - Document 2: [1, 1, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 1, 1, 1, 1, 1, 1, 1, 1]

As can be seen from this very brief example, the number of elements in the vocabulary starts to become very large. In fact, to reduce the number of elements in the vocabulary, it is necessary to remove elements that occur less than n times (in the specific case of this study, occurrences of less than 5 were removed).

The vectors resulting from this transformation process are given as input to the classifier, and the 15 most important features for classification are shown below:

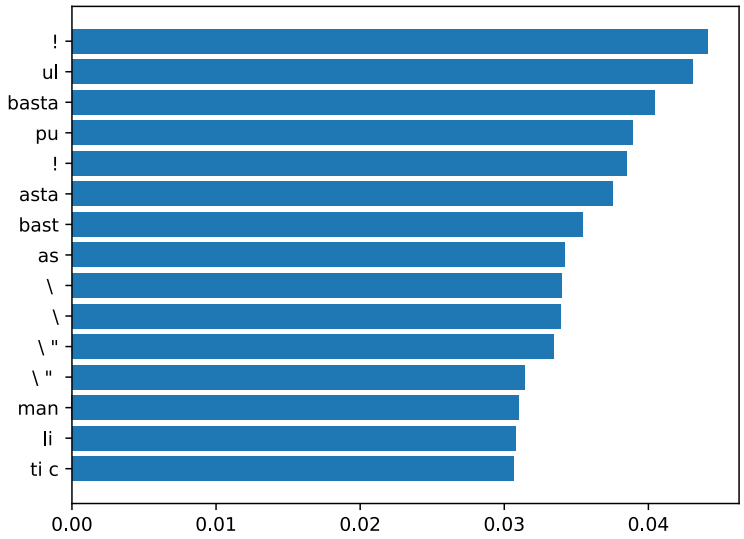

Figure 5 List of the 15 most important features for classification

What can be seen from the figure 5 is that the weight assigned to the most important features is low in relation to the other models seen in this report: this is also due to the size of the vocabulary being significantly larger than in the other models.

Task	**5-fold Cross Validation**				
HS	0.7115	0.7549	0.6403	0.6865	0.7500
	Average: 0.7086				
Stereotype	0.7075	0.7154	0.6482	0.7341	0.7103
	Average: 0.7031				

Table 9 Results of the 5-fold cross validation for the two tasks

Metrics	**Accuracy**	**Precision**	**Recall**	**F1-score**
Not HS	0.74	0.75	0.74	0.75
HS		0.74	0.75	0.74
Not Stereotype	0.72	0.77	0.69	0.73
Stereotype		0.66	0.75	0.71

Table 10 Classification results with the n-gram character model

The tables show that the model using n-word-grams obtains very good results; in fact, for the classification of stereotypes, it is the model with the best results.

4.4 N-word-grams

Models with n-word-grams can come to the aid of capturing the context of words: within the vocabulary, not just individual words are entered, but n-word-grams up to a predefined value of n.

In the specific case of this study, it was decided to use n-grams up to 6, given the low number of items contained in the vocabulary. With larger datasets, it would not be possible to reach this value due to the considerable memory occupation.

To represent documents, we create a set of n-grams of words with n=2 for both documents:

- Document 1: {"Mi piace", "piace ballare", "ballare sotto", "sotto la", "la pioggia"}
- Document 2: {"Mi piace", "piace correre", "correre nel", "nel parco"}

For each document, we create a frequency vector of n-word-grams based on the union of the sets of n-word-grams:

- Vocabulary n-gram words: {"Mi piace", "piace ballare", "ballare sotto", "sotto la", "la pioggia", "piace correre", "correre nel", "nel parco"}
- Document 1: [1, 1, 1, 1, 1, 0, 0, 0]
- Document 2: [1, 0, 0, 0, 0, 1, 1, 1]

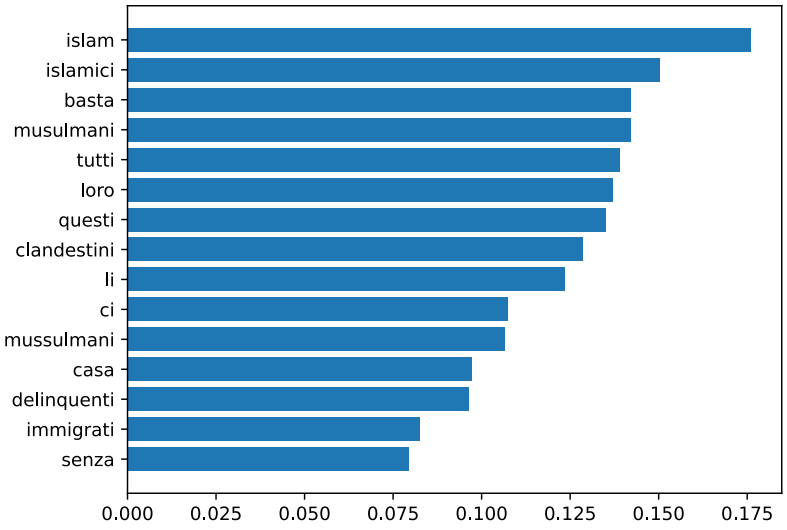

Figure 6 List of the 15 most important features for classification

In this figure, an important feature that can be observed is that the elements that are taken into account the most are, however, the individual words: n-grams of words have a

higher percentage weight than n-grams. However, it must be taken into account that the weight of individual words is significantly lower than that obtained by using only single words (figure 3).

Task	5-fold Cross Validation				
HS	0.6917	0.7115	0.5375	0.6429	0.6786
	Average: 0.6524				
Stereotype	0.6403	0.6680	0.5889	0.6548	0.6706
	Average: 0.6445				

Table 11 Results of the 5-fold cross validation for the two tasks

Metrics	Accuracy	Precision	Recall	F1-score
Not HS	0.70	0.69	0.76	0.72
HS		0.72	0.64	0.68
Not Stereotype	0.69	0.74	0.66	0.70
Stereotype		0.64	0.72	0.68

Table 12 Classification results with the n-gram character model

4.5 N-grams of PoS

In this section, the results of tests using Part-of-Speech (PoS) rather than individual words are shown; vectorisation was carried out following the following steps:

1. Perform PoS tagging for both documents:
   - Document 1: {(Il, DT), (gatto, NN), (gioca, VB), (con, IN), (la, DT), (palla, NN), (nel, IN), (giardino, NN)}
   - Document 2: {(Il, DT), (cane, NN), (corre, VB), (veloce, RB), (nel, IN), (parco, NN), (e, CC), (abbaia, VB)}
2. Create a set of n-grams of POS with $n = 2$ for both documents:
   - Document 1: {(DT, NN), (NN, VB), (VB, IN), (IN, DT), (DT, NN), (NN, IN), (IN, NN)}
   - Document 2: {(DT, NN), (NN, VB), (VB, RB), (RB, IN), (IN, NN), (NN, CC), (CC, VB)}
3. For each document, create a frequency vector of POS n-grams based on the union of the sets of POS n-grams:
   - POS n-gram vocabulary: {(DT, NN), (NN, VB), (VB, IN), (IN, DT), (DT, NN), (NN, IN), (IN, NN), (VB, RB), (RB, IN), (NN, CC), (CC, VB)}
   - Document 1: $[1, 1, 1, 1, 1, 1, 1, 0, 0, 0, 0]$
   - Document 2: $[1, 1, 0, 0, 0, 0, 0, 1, 1, 1, 1]$

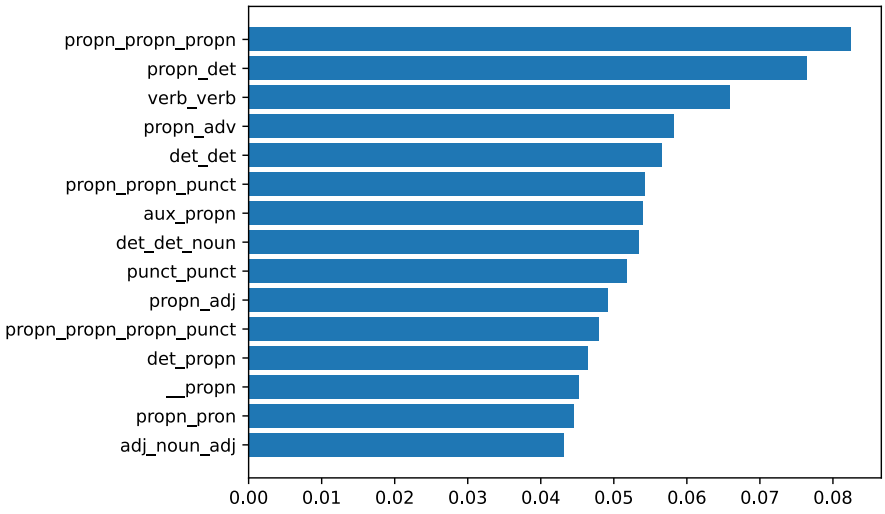

Figure 7 List of the 15 most important features for classification

An interesting thing to note in this analysis is that the weight given to the most important features is significantly lower than in models with many more features (such as Bag of Words of single words).

Task	**5-fold Cross Validation**				
HS	0.6917	0.7115	0.5375	0.6429	0.6786
Average: 0.6524					
Stereotype	0.6403	0.6680	0.5889	0.6548	0.6706
Average: 0.6445					

Table 13 Results of the 5-fold cross validation for the two tasks

Metrics	**Accuracy**	**Precision**	**Recall**	**F1-score**
Not HS	0.60	0.60	0.65	0.62
HS		0.61	0.55	0.58
Not Stereotype	0.59	0.63	0.59	0.61
Stereotype		0.54	0.59	0.56

Table 14 Classification results with the PoS n-gram model

The results show that using PoS does not perform well for both tasks; in fact, for the classification of stereotypes, it is the model that performs the worst (while remaining above baseline).

4.6 Word Embeddings

The Word Embeddings model takes a totally different approach from the models seen above; in fact, it is referred to as an 'implicit language representation': individual words are represented by means of a vector of n elements representing its semantic characteristics.

The data structure created with this model is a dense matrix, which is opposed to the sparse matrices produced by previous models.

Both itWac and Twitter embeddings provided by ItaliaNLP Lab were used in these experiments [4].

The strategies followed to create the vectors were different:

- A vector of 128 elements resulting from the average of all the embeddings of the sentence
- A vector of 384 elements (128 + 128 + 128) resulting from the average of nouns, adjectives, and verbs.

Metrics	**Accuracy**	**Precision**	**Recall**	**F1-score**
Not HS	0.70	0.69	0.73	0.71
HS		0.71	0.67	0.69
Not Stereotype	0.67	0.71	0.67	0.69
Stereotype		0.62	0.67	0.65

Table 15 Classification results with the Word Embedding (itWac) model using a single vector

Task	**5-fold Cross Validation**				
HS	0.7036	0.7233	0.6285	0.6944	0.7540
Average: 0.7007					
Stereotype	0.7115	0.6324	0.5731	0.6905	0.6667
Average: 0.6445					

Table 16 Results of 5-fold cross validation for the two tasks (itWac, single vector)

Following the single vector strategy of 128 elements, good results are obtained for both tasks, slightly better for the hate classification.

Metrics	**Accuracy**	**Precision**	**Recall**	**F1-score**
Not HS	0.63	0.62	0.69	0.65
HS		0.64	0.56	0.60
Not Stereotype	0.64	0.67	0.69	0.68
Stereotype		0.60	0.58	0.59

Table 17 Classification results with the Word Embedding (itWac) model using 3 vectors

Task	**5-fold Cross Validation**				
HS	0.7036	0.7233	0.6285	0.6944	0.7540
	Average: 0.7007				
Stereotype	0.7115	0.6324	0.5731	0.6905	0.6667
	Average: 0.6445				

Table 18 Results of 5-fold cross validation for the two tasks (itWac, 3 vectors)

In the table 17, it can be seen that the strategy using 3 separate vectors containing the average of the 'full words' (nouns, adjectives and verbs) actually does not achieve an improvement in terms of classification performance.

Metrics	**Accuracy**	**Precision**	**Recall**	**F1-score**
Not HS	0.73	0.71	0.77	0.74
HS		0.74	0.68	0.71
Not Stereotype	0.67	0.70	0.71	0.70
Stereotype		0.64	0.63	0.63

Table 19 Classification results with the Word Embedding (Twitter) model using a single vector

Task	**5-fold Cross Validation**				
HS	0.7984	0.6957	0.6166	0.6667	0.6349
	Average: 0.6825				
Stereotype	0.6403	0.6008	0.5850	0.6389	0.5913
	Average: 0.6112				

Table 20 Results of the 5-fold cross validation for the two tasks (Twitter, 1 vector)

Through the use of embeddings created in the same domain, there is a good improvement in performance in the classification of hate, while in the classification of stereotypes, performance remains unchanged.

Metrics	**Accuracy**	**Precision**	**Recall**	**F1-score**
Not HS	0.69	0.68	0.72	0.70
HS		0.70	0.65	0.67
Not Stereotype	0.65	0.67	0.71	0.69
Stereotype		0.62	0.58	0.60

Table 21 Classification results with the Word Embedding (Twitter) model using 3 vectors.

Task	5-fold Cross Validation				
HS	0.7984	0.6957	0.6166	0.6667	0.6349
	Average: 0.6825				
Stereotype	0.6719	0.6008	0.5850	0.6389	0.5913
	Average: 0.612				

Table 22 Risultati della 5-fold cross validation per i due task (Twitter, 3 vettori)

The strategy of the 3 vectors created on Twitter leads to a considerable improvement in the classification of hatred and a slight improvement in the classification of stereotypes, although the strategy using a single vector with 128 elements still achieves better results.

4.7 Neural Language Model

The most advanced experiment for this study is the 'bert-base-italian-cased' model specific to the Italian language.

This model was pre-trained with a 12-layer, 768-dimensional hidden neural network on an Italian corpus of about 30 GB of text, which included different types of text, such as news, books, conversations, scientific texts and more. It can be used for natural language processing purposes, such as text classification, sentiment analysis, question answering and text generation.

In these experiments, a fine-tuning of 5 epochs was performed on the training set with a *batch size* of 80 (such a large size requires about 12 GB of GPU RAM).

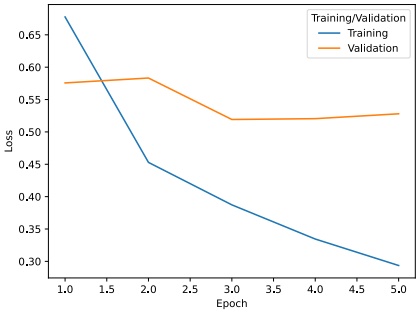

Figure 8 Trend of loss function in the 5 epochs for HS Classification

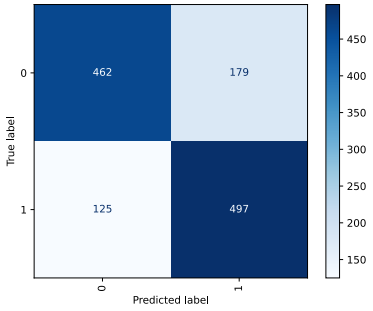

Figure 9 Confusion Matrix for the two classes of hate speech

Figure 8 shows a loss *lineplot*, a graph representing the loss function of a machine learning algorithm during training. This graph helps to visualise the optimisation process of the algorithm and to understand whether the algorithm is learning correctly from the training data. In other words, it shows how the loss decreases as the algorithm improves over iterations or epochs.

In the case of the hate classification, the trend is decreasing for the training set, while in the case of the validation set, the loss remains constant.

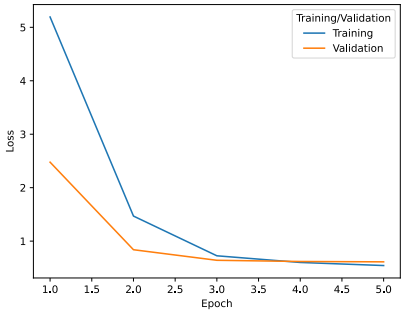

Figure 10 Trend of loss function in the 5 epochs in stereotype classification

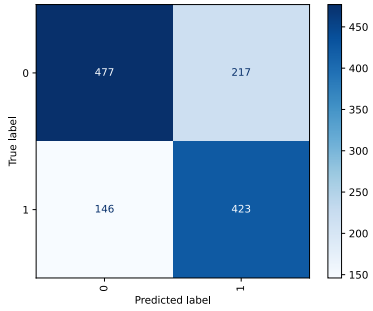

Figure 11 Confusion Matrix for the two classes of stereotypes

In the case of the classification of stereotypes, one sees a clear decrease in the loss for both the training and the validation set in the first three epochs, before remaining fairly constant in the next two.

Metrics	Accuracy	Precision	Recall	F1-score
Not HS	0.76	0.79	0.72	0.75
HS		0.74	0.80	0.77
Not Stereotype	0.71	0.77	0.69	0.72
Stereotype		0.66	0.74	0.70

Table 23 Classification results with the Neural Language Model

The table 23 shows that the model performed slightly better in the classification of Hate Speech than in the classification of stereotypes.

For the classification of Hate Speech, the model correctly classified 76% of the examples. The Precision is 0.79 for the Not HS class and 0.74 for the HS class, meaning that the model has a higher ability to correctly identify Not HS examples than the HS class. Recall, on the other hand, is higher for the HS class (0.80) than the Not HS class (0.72), implying that the model is more sensitive in detecting Hate Speech examples than Not HS examples. Finally, the F1-score, combining Precision and Recall, is 0.75 for the Not HS class and 0.77 for the HS class, indicating a general balance between Precision and Recall.

For the classification of stereotypes, the Accuracy is 0.71, showing that the model correctly classified 71% of the examples. Precision is 0.77 for the Not Stereotype class and 0.66 for the Stereotype class: the model has a greater ability to correctly identify Not Stereotype examples than the Stereotype class. Recall, on the other hand, is higher for the Stereotype class (0.74) than for the Not Stereotype class (0.69), indicating that the model is more sensitive in detecting Stereotype examples than Not Stereotype examples. Finally, the F1-score is 0.72 for the Not Stereotype class and 0.70 for the Stereotype class, showing a general balance between Precision and Recall.

In the table 24, it can be seen that the Training Loss decreases steadily in each epoch, showing that the model is learning from the training data.

Epoch	Training Loss	Validation Loss	F1 Score
1	0.677600	0.575667	0.730411
2	0.452900	0.583253	0.716739
3	0.387300	0.519184	0.759018
4	0.334500	0.520468	0.753589
5	0.293600	0.527987	0.763272

Table 24 Loss values and F1 scores during the 5 fine-tuning epochs for HS Classification

The Validation Loss does not show a similar decreasing trend as the Training Loss and seems to fluctuate throughout the epochs. This could suggest that the model does not significantly improve its ability to generalise to new data.

The F1 Score also tends to fluctuate, but still increases from 0.730411 to 0.763272. This means that the model is improving its ability to correctly classify examples of Hate Speech, but there is still room for improvement.

Epoch	Training Loss	Validation Loss	F1 Score
1	5.193100	2.476944	0.389722
2	1.468800	0.839115	0.649148
3	0.725700	0.641396	0.698239
4	0.600900	0.619095	0.698847
5	0.543300	0.612417	0.713281

Table 25 Loss values and F1 scores during the 5 fine-tuning epochs for Stereotype Classification

Table 25 shows that the Training Loss decreases steadily in each epoch, which indicates that the model is learning from the training data. The Validation Loss does not show the same decreasing trend, this could suggest a possible overfitting of the model to the training data.

The F1 Score, which takes into account both accuracy and recall, generally increases over the epochs, from 0.389722 to 0.713281. This denotes that the model is improving its ability to correctly classify stereotypes, but there is still room for improvement.

The model is learning from the training data, but may be subject to overfitting. To further improve the model, one could consider regularisation techniques, such as using dropout[8] or adjusting optimisation parameters such as the learning rate[9].

5 Results

This section presents the results obtained from the experiments performed on hate speech and stereotype classification tasks. The performance of each model is evaluated using metrics such as Accuracy, Precision, Recall, and F1-score, allowing for a comprehensive comparison of model effectiveness.

5.1 Hate Speech Classification

Table 26 summarizes the evaluation metrics for the different models used in hate speech classification. The Neural Language Model achieves the best performance, with the highest scores across all metrics.

Model	Accuracy	Precision	Recall	F1-score
Non-lexical linguistic information	0.60	0.60	0.60	0.60
Bag of Words (forms)	0.69	0.69	0.69	0.69
Bag of Words (lemmas)	0.70	0.70	0.70	0.70
N-grams of characters	0.74	0.74	0.74	0.74
N-grams of words	0.70	0.70	0.70	0.70
N-grams of POS	0.60	0.60	0.60	0.60
Word Embeddings itWac (1 vector)	0.70	0.70	0.70	0.70
Word Embeddings itWac (3 vectors)	0.63	0.63	0.63	0.63
Word Embeddings Twitter (1 vector)	0.73	0.73	0.72	0.72
Word Embeddings Twitter (3 vectors)	0.69	0.69	0.69	0.69
Word Embeddings (Word2Vec)	0.63	0.65	0.62	0.61
Neural Language Model	**0.76**	**0.76**	**0.76**	**0.76**
Baseline	0.51	0.25	0.50	0.34

Table 26 Evaluation metrics for different models in the classification of hate

5.2 Stereotype Classification

Table 27 presents the results for stereotype classification. Interestingly, the model based on character n-grams outperforms more complex models, such as the Neural Language Model, in terms of accuracy and F1-score, suggesting that simpler models may be better suited for this specific task.

Model	Accuracy	Precision	Recall	F1-score
Non-lexical linguistic information	0.57	0.56	0.56	0.56
Bag of Words (forms)	0.65	0.65	0.65	0.65
Bag of Words (lemmas)	0.67	0.67	0.67	0.67
N-grams of characters	**0.72**	**0.72**	**0.72**	**0.72**
N-grams of words	0.69	0.69	0.69	0.69
N-grams of POS	0.59	0.59	0.59	0.59
Word Embeddings itWac (1 vector)	0.67	0.67	0.67	0.67
Word Embeddings itWac (3 vectors)	0.64	0.64	0.63	0.64
Word Embeddings Twitter (1 vector)	0.67	0.67	0.67	0.67
Word Embeddings Twitter (3 vectors)	0.65	0.65	0.64	0.64
Word Embeddings (Word2Vec)	0.62	0.62	0.61	0.60
Neural Language Model	0.71	0.71	**0.72**	0.71
Baseline	0.55	0.27	0.50	0.35

Table 27 Evaluation metrics for different models in stereotype classification

5.3 Summary of Results

Across both tasks, all models performed better than the baseline. The Neural Language Model emerged as the best model for hate speech classification, achieving 76% accuracy.

However, in stereotype classification, the character n-grams model outperformed others, suggesting that different tasks may require distinct modeling approaches. The use of 3 combined vectors did not consistently improve performance, and the PoS n-grams model performed the worst in both tasks.

6 Conclusions

This study contributes to the ongoing effort to improve hate speech and stereotype detection in natural language processing by evaluating a range of machine learning and deep learning models on the HaSpeeDe dataset for Italian texts. The comparative analysis of various models provides insights into the strengths and weaknesses of each approach in detecting hate speech and stereotypes. One of the key contributions of this work is the exploration of both traditional machine learning methods, such as bag-of-words and n-grams, and more advanced techniques, including neural language models and word embeddings, in the context of Italian-language classification tasks.

Among the contributions, this research underscores the importance of model selection based on the nature of the classification task. Models that perform well for hate speech detection may not necessarily excel in stereotype classification, demonstrating the challenge of creating a one-size-fits-all solution. This work also highlights the complexity of detecting subtle linguistic nuances, particularly in stereotype detection, where certain models struggle to generalize effectively.

However, the study also has several limitations. First, the lack of domain-specific feature engineering in some of the models may have hindered their performance. Additionally, the reliance on pre-trained word embeddings for Italian, which are less developed than their English counterparts, likely impacted the results. Another shortcoming is the limited generalizability of the results beyond the Italian language and the HaSpeeDe dataset, as the unique linguistic and cultural features of Italian texts may not translate well to other languages or contexts.

Future work should focus on improving the robustness and generalization capabilities of hate speech and stereotype detection models by leveraging more sophisticated pre-trained language models and exploring cross-linguistic approaches. Expanding the diversity and size of training datasets and incorporating more nuanced linguistic and contextual features are critical steps toward achieving better model performance.

References

[1] Valerio Basile et al. “SemEval-2019 task 5: Multilingual detection of hate speech against immigrants and women in Twitter”. In: *Proceedings of the 13th International Workshop on Semantic Evaluation (SemEval-2019)* (2019), pp. 54–63.

[2] Dominique Brunato et al. “Profiling-UD: a Tool for Linguistic Profiling of Texts”. In: *Proceedings of the Twelfth Language Resources and Evaluation Conference*. European Language Resources Association. Marseille, France, May 2020, pp. 7145–7151.

[3] Andrea Cimino, Lorenzo De Mattei, and Felice Dell’Orletta. “Multi-task Learning in Deep Neural Networks at EVALITA 2018”. In: *Proceedings of EVALITA 2018, Evaluation of NLP and Speech Tools for Italian, 12-13 December, Turin, Italy*. 2018.

[4] Andrea Cimino, Lorenzo De Mattei, and Felice Dell'Orletta. "Multi-task Learning in Deep Neural Networks at EVALITA 2018". In: *Proceedings of EVALITA '18, Evaluation of NLP and Speech Tools for Italian, 12-13 December, Turin, Italy*. 2018.

[5] Thomas Davidson et al. "Automated hate speech detection and the problem of offensive language". In: *Proceedings of the International AAAI Conference on Web and Social Media*. Vol. 11. 1. 2017, pp. 512–515.

[6] Fabio Del Vigna et al. "Abusive language detection in online communities using natural language processing: A survey". In: *Proceedings of the 30th ACM International Conference on Information and Knowledge Management (CIKM 2021)*. 2021.

[7] Paula Fortuna and Sérgio Nunes. "A survey on the automatic detection of hate speech in text". In: *ACM Computing Surveys (CSUR)* 51.4 (2018), pp. 1–30.

[8] Antoine Gambs et al. "Representation learning for hate speech detection using deep neural networks". In: *Proceedings of the 2017 ACM on Conference on Information and Knowledge Management (CIKM 2017)*. 2017, pp. 2233–2236.

[9] Hyun Ko et al. "Modular Approach to Hate Speech Detection with Domain Adaptation and Transfer Learning". In: *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics*. 2020.

[10] Sean MacAvaney et al. "Hate speech detection: Challenges and solutions". In: *Proceedings of the 28th ACM International Conference on Information and Knowledge Management*. 2019, pp. 2543–2546.

[11] Subha Malik et al. "Classification of Offensive Language in Social Media using Deep Learning and Transfer Learning". In: *Proceedings of the 12th Language Resources and Evaluation Conference*. 2020, pp. 5079–5083.

[12] Alexander Mosca et al. "Modular Neural Network Architectures for Hate Speech Detection". In: *Artificial Intelligence Research* 11.1 (2022), pp. 46–61.

[13] Umar Naseem et al. "A survey of machine learning for big data processing". In: *Neural Computing and Applications* 33.4 (2021), pp. 1183–1203.

[14] Fabio Poletto et al. "Resources and benchmark corpora for hate speech detection: A systematic review". In: *Language Resources and Evaluation Conference (LREC)*. 2020.

[15] Manuela Sanguinetti et al. "HaSpeeDe 2@ EVALITA2020: Overview of the EVALITA 2020 Hate Speech Detection Task". In: *CEUR Workshop Proceedings* 2020 (2020), pp. 20–24.

[16] 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.

[17] Zeerak Waseem and Dirk Hovy. "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter". In: *Proceedings of the NAACL Student Research Workshop*. 2016, pp. 88–93.

[18] Zhuo Zhang et al. "Detecting hate speech on Twitter using a convolution-GRU based deep neural network". In: *Neural Computing and Applications* 29.10 (2018), pp. 1–10.