Hate Speech Detection in Romanian Facebook Comments: An AI-Based Approach

Madalina Badescu, Liviu Bouruc, Andrei Constantinescu

February 2024

Contents

1 Introduction

Social media platforms like Facebook have become integral parts of our daily lives, serving as platforms for communication, expression, and information sharing. However, with the widespread use of social media comes the challenge of managing and mitigating harmful content, such as hate speech, which can perpetuate discrimination, prejudice, and violence.

Hate speech detection on social media presents a complex and pressing problem, especially in multilingual and multicultural contexts like Romania. The proliferation of hate speech not only threatens individual users' well-being but also undermines the social fabric and cohesion of communities. Therefore, developing effective tools and methods to identify and combat hate speech is crucial for fostering a safe and inclusive online environment.

In this project, we embarked on the task of hate speech detection in Romanian Facebook comments using advanced technologies such as Natural Language Processing (NLP) and machine learning. By leveraging AI models and computational techniques, we aimed to automate the process of identifying hate speech, thereby empowering social media platforms and regulatory bodies to take proactive measures against harmful content.

Through our research, we sought to address several key challenges inherent in hate speech detection, including the nuances of language, cultural sensitivities, and the dynamic nature of online discourse. By collecting and analyzing real-world data from Facebook comments, we aimed to gain insights into the prevalence and patterns of hate speech in the Romanian online sphere.

Our approach involved collaborating with AI models such as ChatGPT, PaLM 1, PaLM 2, and Vertex to annotate and classify comments as hate speech or non-hate speech. We then employed data preprocessing techniques, feature extraction methods, and SVM classification to evaluate the effectiveness of our hate speech detection system.

By documenting our methodology, results, and insights, we hope to contribute to the ongoing efforts to combat hate speech on social media platforms. Ultimately, our goal is to promote digital safety, protect users' rights and dignity, and foster a culture of respect and inclusivity in online spaces.

2 Dataset

In this section, we provide insights into the dataset used for hate speech detection, outlining the process of data collection, and describing the AI models employed for labeling the comments.

2.1 Data Collection Process

We utilized the Facebook API to scrape comments from public posts written in Romanian. The Facebook API enabled us to access and retrieve comments programmatically, ensuring a systematic and efficient data collection process. By focusing on Romanian-language comments, we aimed to capture the nuances and complexities of hate speech in the local online community.

2.2 AI Models for Labeling

Before proceeding with data preprocessing and classification, we employed four AI models to annotate the comments as hate speech or non-hate speech. Each AI model provided its interpretation of the comments, which served as the basis for our labeled dataset. Below are brief descriptions of the AI models used:

- ChatGPT is a state-of-the-art language generation model developed by OpenAI. It excels in understanding and generating human-like text across various contexts, making it suitable for tasks such as sentiment analysis and classification.
- PaLM1 and PaLM2 developed by Google, are large language models designed to perform a wide range of natural language processing tasks. PaLM1 focuses on providing an approachable way to explore and prototype with generative AI applications, while PaLM2 represents a significant improvement in model size, training data, and performance, enabling more accurate and reliable results.
- Vertex AI, offered by Google Cloud, provides access to Gemini, a multimodal model capable of understanding diverse inputs and generating outputs across different modalities, including text, images, video, and code. Gemini's advanced reasoning and generation capabilities make it a powerful tool for tasks such as hate speech detection and content moderation.

2.3 Dataset Overview

After labeling the comments using the aforementioned AI models, we obtained a dataset consisting of hate speech and non-hate speech comments. This labeled dataset served as the foundation for training and evaluating our hate speech detection system.

The ChatGPT model resulted in an almost equal distribution of labeled hate speech and non-hate speech comments. PaLM1 and PaLM2, along with Vertex AI, exhibited an unequal distribution of labels, with more neutral texts labeled than hate speech. We describe how we tackled this issue in the section Balancing the Dataset.

3 Balancing the Dataset

In hate speech detection tasks, imbalanced datasets where one class (e.g., hate speech) is significantly more prevalent than the other (e.g., non-hate speech) can lead to biased model performance and skewed evaluation metrics. To address this issue and ensure fair and accurate model training, we employed a technique called Random Under-sampling. Even though we employed this technique, we also trained the SVM model on the imbalanced results in order to compare the results.

Random Under-sampling involves randomly selecting a subset of samples from the majority class (in this case, non-hate speech comments) to match the size of the minority class (hate speech comments). By reducing the number of instances in the majority class, we balanced the distribution of hate speech and non-hate speech comments in the dataset, thereby mitigating the impact of class imbalance on model training.

We performed random under-sampling for the PaLM 1, PaLM 2 and Vertex datasets and this is how the distribution of the data looked like:

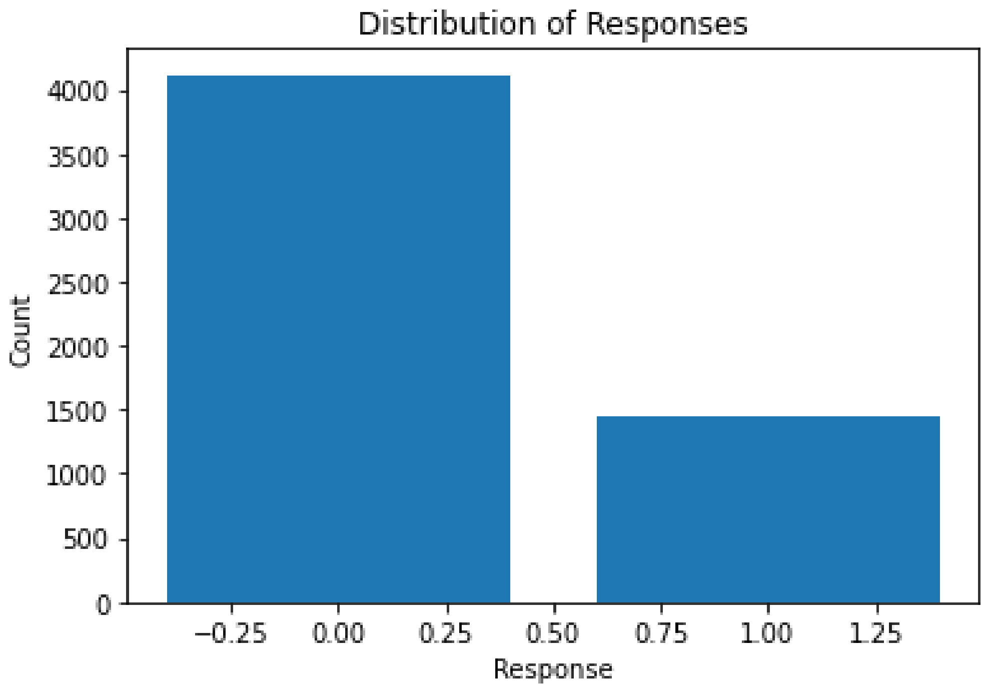

Figure 1: PaLM 1 imbalanced

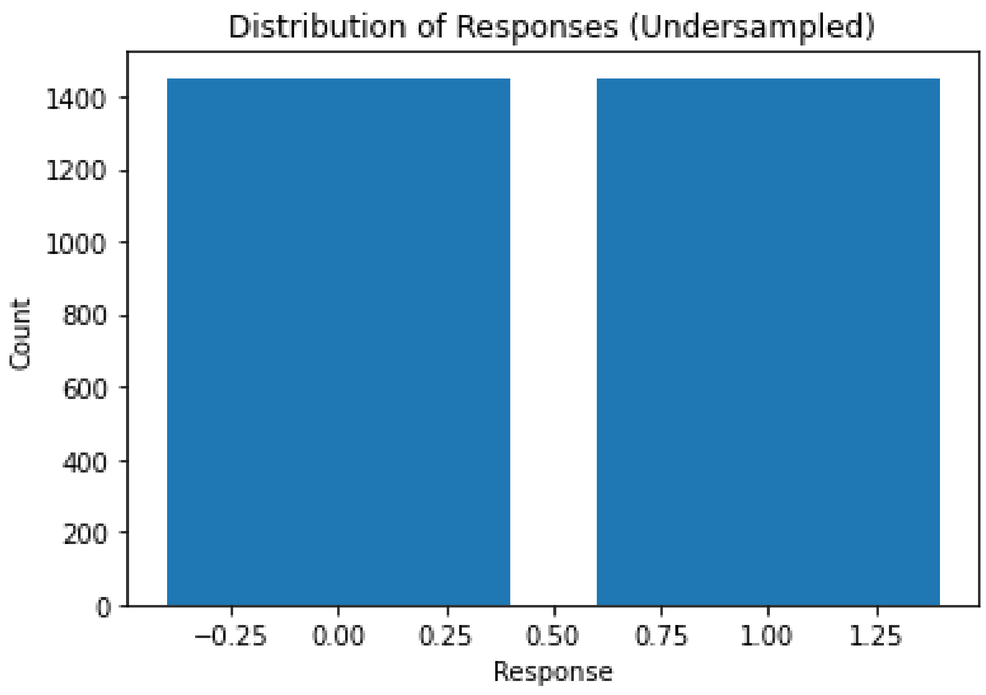

Figure 2: PaLM 1 balanced

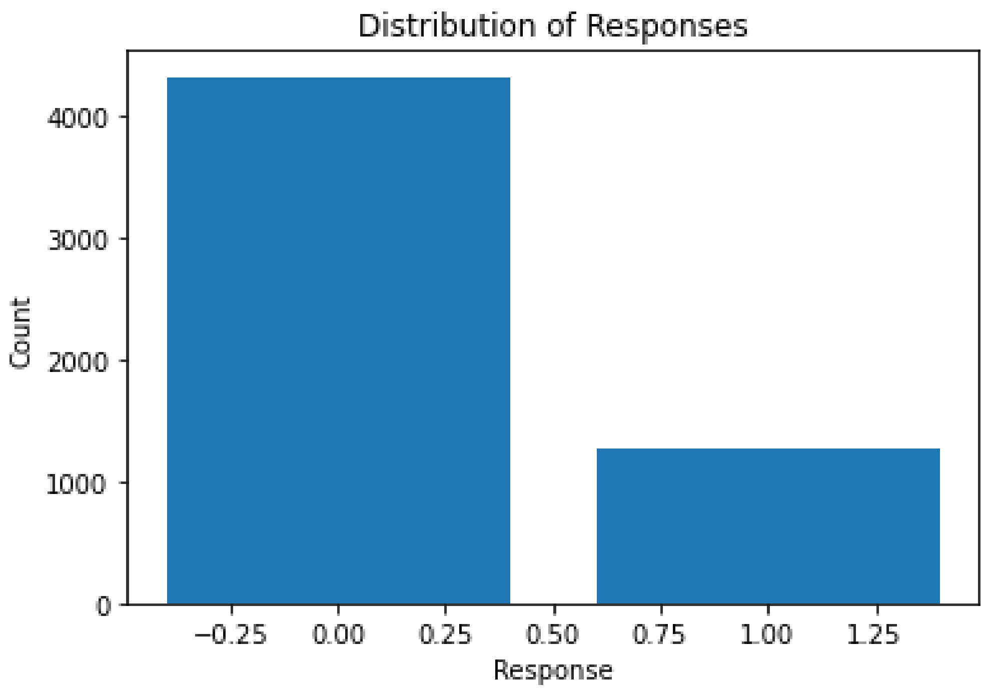

Figure 3: PaLM 2 imbalanced

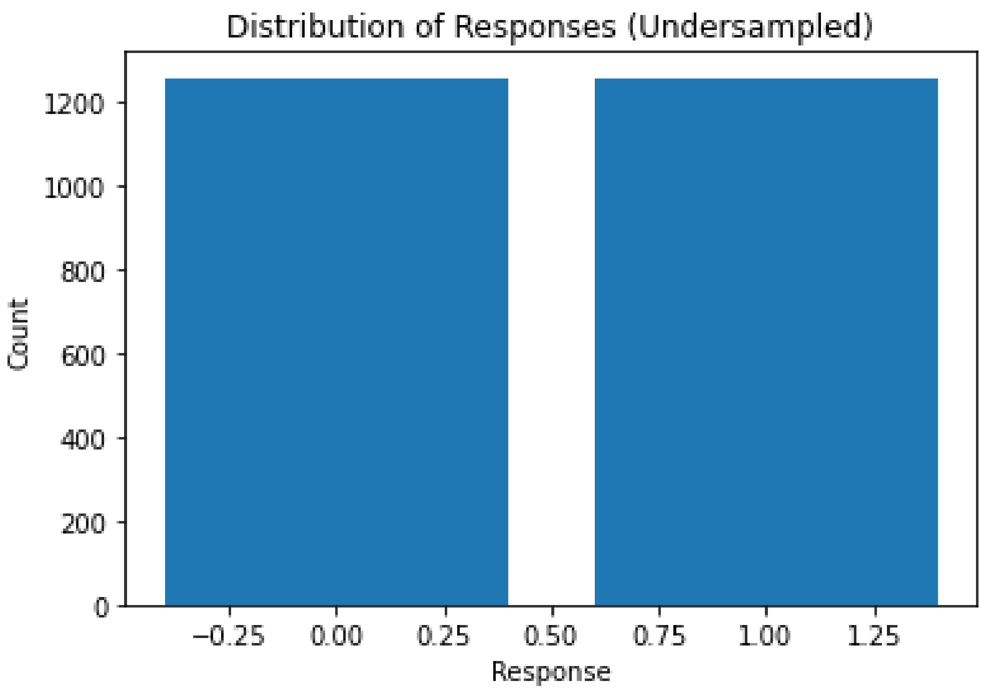

Figure 4: PaLM 2 balanced

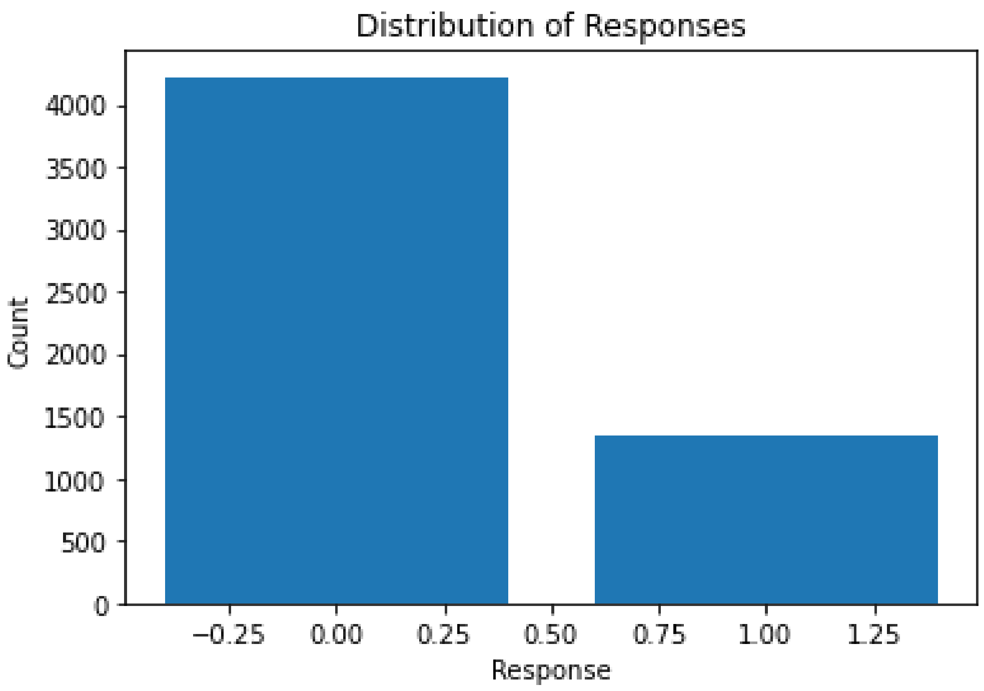

Figure 5: Vertex imbalanced

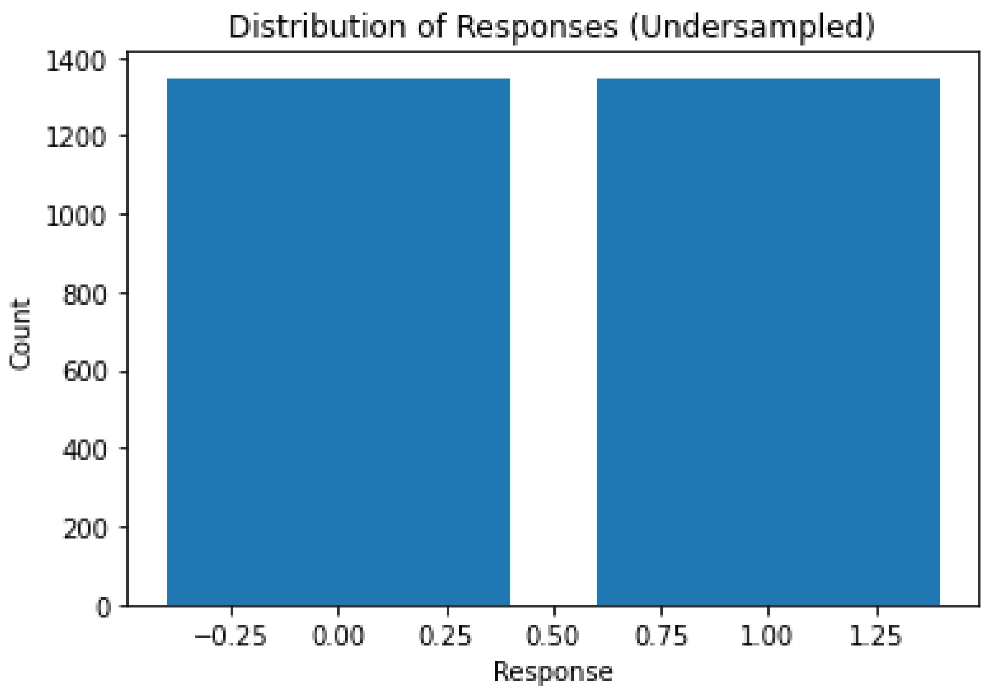

Figure 6: Vertex balanced

Balancing the dataset through Random Under-sampling helps prevent the model from being biased towards the majority class during training. By presenting the model with an equal number of instances from both classes, we ensure that it learns to distinguish between hate speech and non-hate speech comments effectively, leading to more reliable and generalizable results.

While Random Under-sampling effectively addresses class imbalance, it also reduces the amount of training data available for the majority class. This reduction in data can potentially lead to loss of information and decreased model performance, especially if the original dataset is small. Therefore, it's essential to strike a balance between addressing class imbalance and preserving sufficient data diversity for robust model training. This is the main reason why we trained our model both on the balanced and imbalanced data.

4 Data Preprocessing

Data preprocessing is a crucial step in natural language processing (NLP) tasks, aimed at transforming raw text data into a format suitable for machine learning models. It involves cleaning, normalizing, and transforming textual data to enhance its quality and remove noise, thereby improving the effectiveness of subsequent analysis and modeling.

The general steps that we took for preprocessing:

- Lowercasing: Convert all text to lowercase to ensure uniformity in text representation and reduce the complexity of text processing.
- Unicode Normalization: Normalize text using the unidecode library to convert accented characters and Unicode characters into their closest ASCII equivalents, simplifying text representation.
- Tokenization: Tokenize the text into individual words or tokens using spaCy's language processing pipeline (nlp), enabling granular analysis and manipulation of text at the token level.
- Stopword Removal: Remove stopwords, such as common words like ”the,” ”and,” ”is,” etc., using a predefined list of stopwords for the Romanian language. This step helps eliminate noise and irrelevant information from the text.
- Punctuation removal: Removes punctuation, such as ”,”, ”!”, ”.”, since it is not important to the overall model.
- Spell check: The spelling check aims at improving the quality of the text data before it is used for classification or analysis. Specifically, it attempts to correct misspelled words using a spell-checking library (pspell).
- Stemming: Apply stemming using the SnowballStemmer for Romanian (”romanian”) to reduce words to their root or base form. Stemming helps standardize word variations and improve the model's ability to generalize across different forms of words.

5 Feature representation

Feature representation is a critical aspect of natural language processing (NLP) and machine learning tasks, as it involves transforming raw text data into a format that can be understood and processed by algorithms. The goal of feature representation is to capture the essential information from the text while minimizing noise and irrelevant details. Effective feature representation plays a crucial role in the performance of NLP models, as it directly impacts their ability to understand and extract meaningful patterns from textual data.

For feature representation, we used TF-IDF and BoW. TF-IDF is a popular feature representation technique that assigns weights to terms based on their frequency in a document relative to their frequency across all documents in the corpus. It measures the importance of a term within a document while considering its prevalence in the entire corpus. Terms that are frequent in a document but rare in the corpus are assigned higher weights, indicating their significance. BoW is a simple yet effective feature representation technique that represents text data as a vector of word counts or frequencies. It disregards the order and structure of the words in the text, treating each document as an unordered collection of words. BoW representation is based on the vocabulary of the corpus, where each unique word corresponds to a dimension in the feature space. The value of each dimension represents the frequency of the corresponding word in the document.

6 SVM, RNN and results

Support Vector Machine (SVM) is a supervised machine learning algorithm used for classification and regression tasks. It works by finding the hyperplane that best separates the data points into different classes. In the context of hate speech detection, SVM aims to classify comments into hate speech and non-hate speech categories based on the features extracted from the text data.

The SVM classification results for hate speech detection using TF-IDF and Bag-of-Words (BoW) features are presented below, along with specific details for each run:

- TF-IDF SVM: Achieved an F1 score ranging from 0.63 to 0.84 across different iterations. Notably, the highest F1 score was observed in the Vertex dataset, reaching 0.84.
- BoW SVM: Obtained F1 scores ranging from 0.62 to 0.81. Similar to TF-IDF, the highest F1 score was achieved in the Vertex dataset, with a value of 0.81.

We tried more runs on the ChatGPT generated dataset, since it was the one with the balanced data. These are the results that we got from it (the percentage mentioned first in the results is the F1 score for accuracy and the one mentioned second is the F1 score for weighted average):

- Run 1: Splits text into tokens based on whitespace, removes stopwords from the tokens, applies stemming to the tokens with results for TF-IDF 0.63, 0.63 and for BoW 0.63, 0.62.
- Run 2: Splits text into tokens using word tokenization, removes stopwords from the tokens, pplies stemming to the tokens with results for TF-IDF 0.65, 0.65 and for BoW 0.64, 0.63.
- Run 3: Splits text into tokens using word tokenization, removes stopwords from the tokens, no Stemming applied with results for TF-IDF 0.63, 0.63 and for BoW 0.64, 0.62.
- Run 4: Splits text into tokens using SpaCy tokenization, removes stopwords from the tokens, no Stemming applied with results for TF-IDF 0.65, 0.64 and for BoW 0.65, 0.63.
- Run 5: Splits text into tokens using SpaCy tokenization, removes stopwords from the tokens, applies stemming to the tokens with results for TF-IDF 0.66, 0.66 and for BoW 0.66, 0.65.
- Run 6: Splits text into tokens using SpaCy tokenization, removes stopwords from the tokens, applies stemming to the tokens, removes punctuation marks from the tokens with results for TF-IDF 0.65, 0.65 and for BoW 0.65, 0.65.
- Run 7: Splits text into tokens using SpaCy tokenization, removes stopwords from the tokens, removes punctuation marks from the tokens, corrects misspelled words using a spell checker and applies stemming to the corrected words with results for TF-IDF 0.63, 0.63 and for BoW 0.63, 0.63

- Run 8: Splits text into tokens using SpaCy tokenization, removes stopwords from the tokens, removes punctuation marks from the tokens using regex substitution, and applies stemming to the corrected words with results for TF-IDF 0.68, 0.68 and for BoW 0.68, 0.67

We can observe that run 5 is the most successful one and in the preprocessing of the datasets labeled by the other AI models we use only those specific steps. Results on the other models:

- PaLM 1 Dataset: TF-IDF SVM achieved F1 scores of 0.82, 0.80 without undersampling and 0.72, 0.74 with undersampling. BoW SVM obtained F1 scores of 0.81, 0.78 without undersampling and 0.73, 0.74 with undersampling.

- PaLM 2 Dataset: TF-IDF SVM achieved F1 scores of 0.83, 0.80 without undersampling and 0.72, 0.74 with undersampling. BoW SVM obtained F1 scores of 0.81, 0.78 without undersampling and 0.75, 0.76 with undersampling.

- Vertex Dataset: TF-IDF SVM achieved F1 scores of 0.84, 0.81 without undersampling and 0.71, 0.73 with undersampling. BoW SVM obtained F1 scores of 0.81, 0.78 without undersampling and 0.73, 0.75 with undersampling.

Besides the SVM model, we implemented a RNN model (BiLSTM) and a Word2Vec feature extractor with F1 score accuracy 0.69. The word embeddings used were from Sergiu Nisioi's github: `https://github.com/senisioi/ro_resources`. Furthermore, we implemented a BERT model with an accuracy of 0.70.

These results demonstrate the effectiveness of SVM in hate speech detection and highlight the impact of different preprocessing techniques on classification performance across various datasets.

7 Conclusion

Hate speech detection on social media platforms is a critical endeavor in today's digital age, especially considering the harmful consequences it can have on individuals and communities. In this project, we undertook the task of hate speech detection in Romanian Facebook comments using advanced technologies such as Natural Language Processing (NLP) and machine learning.

Through the collaborative efforts of AI models such as ChatGPT, PaLM 1, PaLM 2, and Vertex, we collected and labeled a dataset of Facebook comments, enabling us to train and evaluate hate speech detection systems. Our approach involved leveraging sophisticated NLP techniques, including data preprocessing, feature representation, and Support Vector Machine (SVM) classification, to analyze and classify comments as hate speech or non-hate speech.

One of the key highlights of our project is that we utilized the Facebook API and AI models to collect and annotate Romanian Facebook comments, creating a labeled dataset for hate speech detection. This dataset served as the foundation for our machine learning experiments. Next, we employed Random Under-sampling to balance the distribution of hate speech and non-hate speech comments in the dataset, ensuring fair and accurate model training. Then, we applied various preprocessing techniques, including lowercasing, Unicode normalization, tokenization, stopword removal, punctuation removal, spell check, and stemming, to clean and standardize the text data before feature extraction and classification. After, we explored two fundamental feature representation techniques, TF-IDF and Bag-of-Words (BoW), to transform text data into numerical representations suitable for machine learning algorithms. These techniques capture the essential information from the text while preserving semantic and syntactic properties. In the end, SVM effectively learned to distinguish between hate speech and non-hate speech comments based on the features extracted from the text data.

Our experiments yielded valuable insights into the effectiveness of hate speech detection systems in the Romanian online context. By documenting our methodology, results, and observations, we contribute to the ongoing efforts to combat hate speech on social media platforms and promote digital safety and inclusivity.

Moving forward, further research and development in NLP and machine learning are essential to enhance the accuracy, efficiency, and scalability of hate speech detection systems. By leveraging cutting-edge technologies and interdisciplinary collaborations, we can continue to advance the field and create safer and more inclusive online environments for all users.