Text Classification Systems
Offensive/Not Offensive Instagram Comments

Pedro Silva, Ramyad Raadi
uc2023235452@student.uc.pt, uc2023205634@student.uc.pt

University of Coimbra, Faculty of Science and Technologies
Bachelor of Artificial Intelligence and Data Science, PLN Course

November 2025

1 Introduction

This study focuses on the automatic classification of Portuguese text, specifically aiming to detect offensive language in Instagram comments. The primary objective is to develop and evaluate systems capable of distinguishing between *Offensive* and *Non-offensive* comments.

To achieve this, we implemented and compared three distinct Natural Language Processing (NLP) approaches: A rule-based system leveraging linguistic pattern; A supervised Machine Learning approach; And a Prompt Engineering strategy using Large Language Models.

This report outlines the implementation of these methods and evaluates their respective performance to determine the most effective approach for this classification task.

2 Materials

2.1 Data Description

The dataset used for this study is publicly available on HuggingFace[1] and contains comments extracted from Brazilian politicians posts on Instagram. The data were manually annotated by three experts, it is already balanced and there are 7000 comments with the following characteristics:

Column	Description
id	Comment Id
comentario	The comment itself
anotator1	The first annotator
anotator2	The second annotator
anotator3	The third Annotator
label_final	classification label: offensive (1)/not offensive (0)
links_post	The link where the comment was retrieved
account_post	Personal account of the linked post

Table 1: Dataset Column Descriptions

[1] **Dataset (corpus):** https://huggingface.co/datasets/franciellevargas/HateBR

3 Exploratory Data Analysis

In this section, we analyzed the dataset and extracted relevant linguistic knowledge. The analysis includes inter-annotator agreement, lexical patterns, and sentiment characteristics of the comments.

3.1 Annotators Concordance

To assess the consistency of manual annotations, the agreement between annotators was calculated using Krippendorff's Alpha. The obtained value of $\alpha = 0.771$ indicates a medium-strong level of agreement, which is reliable for linguistic and classification analysis.

3.2 Lexical Analysis

A lexical analysis was performed on the corpus after removing punctuation, while retaining stop words and emojis. The processed dataset contained 106 015 tokens and 12 093 unique types.

An examination of the most frequent tokens was conducted under three conditions: (1) maintaining stop words and emojis, (2) analyzing stop words within the top 100 tokens, and (3) removing stop words but keeping emojis.

The results showed that emojis appear frequently among the top tokens, which is valuable for sentiment and offensive speech analysis. Removing stop words provided clearer insights into meaningful lexical patterns, allowing the classification to focus more directly on relevant linguistic features.

3.3 Sentiment Analysis

Sentiment analysis was performed using a Portuguese lexicon[2] containing words with one or two polarities per word, where -1 represents negative, 0 neutral, and 1 positive sentiment. Additionally, an annotated emoji dataset from Twitter comments [3] was used to determine the predominant sentiment of each emoji.

A preprocessing pipeline, the same used for lexical analysis and, more ahead, classification rules, was applied. Tokens were filtered to remove stop words, empty spaces, non-representative characters (e.g., the Zero-Width Joiner \u200d), and punctuation, while emojis were separated for individual sentiment assessment.

For each comment, the sentiment score was calculated as the sum of token sentiments divided by the total number of tokens. Sentiment was tested with and without lemmatization, and in some cases, lemmatization slightly altered the sentiment values. The resulting scores ranged from -1 to 1, reflecting the overall sentiment of the comment. Overall, this approach provided a sentiment estimation that correlated well with offensive/not offensive classification, supporting (what we thought) the effectiveness for this task.

4 Methodologies

To address this classification task, we explored three distinct methodologies, ranging from symbolic to statistical and generative approaches:

- **Rule-Based System:** The development of a classifier based on linguistic rules, leveraging data analysis and the extraction of specific features such as grammatical functions and sentiment analysis.
- **Machine Learning:** The implementation of supervised learning models utilizing feature engineering to capture patterns within the textual data.

[2] **PT Lexicon:** https://b2find.eudat.eu/dataset/b6bd16c2-a8ab-598f-be41-1e7aeecd60d3

[3] **Emojis Dataset:** https://www.kaggle.com/datasets/thomasseleck/emoji-sentiment-data

- **Prompt Engineering:** The exploration of Large Language Models (LLMs) through prompt engineering techniques to evaluate the capabilities of generative AI in text classification tasks.

The dataset was split into 80% for training and 20% for testing in all methodologies, using the same `random_state` to compare better the results. We only changed this is meta 2, since we ran the split 10 times to test the correctness of predicted samples for each model.

4.1 Meta 1: Rule-Based Classification

The Rule-Based System relied on a pre-processing pipeline that included text normalization while retaining stop words and emojis, as they were identified as valuable features. Lemmatization was ultimately excluded from the final pipeline due to a slight decrease in performance metrics. The final classification pipeline used a combination of three rules:

- **Rule 1 (Sentiment):** Comments with a strong negative sentiment score (-1) were classified as offensive.
- **Rule 2 (Lexicon):** Comments containing offensive words from a Portuguese lexicon were classified as offensive.
- **Rule 3 (PoS Heuristic):** Comments containing two or more negative adjectives or verbs were classified as offensive.

4.1.1 Results

An additional rule based on Cosine Similarity to negative seed comments (Rule 4) was tested but did not improve performance and was thus excluded. The best configuration, using Rules 1, 2, and 3, achieved an accuracy of 78% with an F1-score of 0.80 for the non-offensive class and 0.76 for the offensive class, as shown in figure 1a and table 6.

4.2 Meta 2: Classic Machine Learning Classification

The second phase explored traditional Machine Learning models, using the same dataset split and the refined pre-processing pipeline from Meta 1. The approach was divided into two phases:

4.2.1 Phase 1: Vectorizer Selection

Four vectorization methods (CountVectorizer, TfidfVectorizer, Word2Vec, and Doc2Vec) were evaluated using Logistic Regression as a control model. Sparse representations (CountVectorizer/TfidfVectorizer) significantly outperformed dense representations (embeddings) and **CountVectorizer with lemmatization** was identified as the single best-performing representation, achieving a Mean F1-score of 0.8415, as shown in the table below.

Metric	Value
Vectorizer	CountVectorizer
Lemmatization	True
Mean F1	**0.841522**
Std F1	0.011703
Mean Precision	0.844723
Mean Recall	0.841857

Table 2: Best Combination Details

4.2.2 Phase 2: Model Optimization

The optimal CountVectorizer representation was used to tune three top classic models - LogisticRegression, LinearSVC, and MultinomialNB - using the Optuna framework. The **LogisticRegression** model emerged as the clear winner, achieving a final F1-score of 0.85 and an accuracy of 85%. The best hyperparameters for the winning `LogisticRegression` model are detailed in Tables 3 and 4 below.

Metric	Value
Best F1	0.8452
Best Precision	0.8488
Best Recall	0.8455

Table 3: Overall Winner: LogisticRegression Metrics

Hyperparameter	Value
vect_ngram_range	(1, 1)
vect_min_df	1
vect_max_df	0.79809088
clf_solver	'saga'
clf_penalty	'l2'
clf_C	10.66314227

Table 4: Overall Winner: Best Hyperparameters

4.2.3 Results

The Logistic Regression model significantly improved upon the Meta 1 baseline, achieving a 7-point increase in F1-score (from 78% to 85%) and an accuracy of 85% as we can see in figure 1b and table 7.

4.3 Meta 3: Prompt Engineering

The final methodology focused on leveraging the reasoning and language generation capabilities of Large Language Models (LLMs) through Prompt Engineering. This approach aims to perform classification without the need for extensive training or fine-tuning, relying instead on strategically crafted input prompts.

4.3.1 Models Used

To comply with the project constraints (maximum of 8 Billion parameters), two distinct open-source, decoder-only transformer and Instruction-trained models were selected for evaluation:

- **Qwen 2.5 7B**: A model from Alibaba Cloud, known for its strong performance across multiple languages [9.1].
- **Llama 3.1 8B**: A highly popular model developed by Meta, serving as a robust benchmark in the open-source LLM space [9.2].

4.3.2 Prompt Engineering Techniques

The classification performance was tested using four widely recognized prompt engineering techniques, designed to influence the model's output and reasoning process:

1. **Zero-Shot (ZS):** The model is provided only with the task description and the input comment. It must rely entirely on its pre-trained knowledge to classify the text.
2. **Few-Shot (FS):** The prompt includes the task description along with a small number of example input/output pairs (in our case were 4 examples), providing the model with in-context learning to guide its response format and classification criteria.
3. **Chain-of-Thought Zero-Shot (CoT-ZS):** The model is instructed to output its reasoning process before giving the final answer. No examples are provided.
4. **Chain-of-Thought Few-Shot (CoT-FS):** This combines the Chain-of-Thought instruction with a few examples (4) that also contain the intermediate reasoning steps.

4.3.3 Results

The main findings, shown in figures 1b and 1c, and tables 5 (below), 8 and 9,include:

- **Few-Shot Efficacy:** The **Qwen** model achieved the highest performance of all models tested in the project using the Few-Shot technique, reaching an F1-score of **0.86** (Accuracy: 0.86), marginally surpassing the best Machine Learning model (Meta 2, F1 0.85).
- **The Impact of CoT:** Incorporating the Chain-of-Thought (CoT) instruction in the Zero-Shot setting dramatically **decreased** performance for both models (Qwen CoT-ZS F1 0.73, Llama CoT-ZS F1 0.63), as shown in table 5. This suggests that forcing the model to articulate its reasoning led to distraction or misinterpretation of the classification rule, rather than providing beneficial structure.
- **Balanced Classification:** The best LLM result (Qwen Few-Shot) demonstrated robust performance across both classes: Precision 0.90 and F1-score 0.85 for the Offensive class, showing strong ability to correctly identify toxic content.
- **Model Preference:** Qwen consistently outperformed Llama 2 in both simple classification (ZS, FS) and reasoning-based tasks (CoT-FS), indicating superior pre-training or alignment for the Portuguese hate speech domain.

Technique	Qwen-7B	Llama 2 7B	Best Meta 2 (ML)
Zero-Shot	0.84	0.84	0.85
Few-Shot	**0.86**	0.84	0.85
CoT Zero-Shot	0.73	0.63	0.85
CoT Few-Shot	0.85	0.79	0.85

Table 5: Performance Comparison of LLM Prompting Techniques (Macro F1-Score)

5 Results and Discussion

The experimental work across all three metas demonstrated a clear and progressive improvement in the text classification system, confirming the benefits of utilizing data-driven and generative models over a purely symbolic, rule-based approach.

5.1 Overall Performance Across Methodologies

The classification performance was evaluated primarily using the Macro F1-score and Accuracy. The results show that both the Machine Learning (Meta 2) and Prompt Engineering (Meta 3) approaches significantly surpassed the baseline established by the Rule-Based system (Meta 1).

- **Meta 1 (Rule-Based):** Achieved a baseline Accuracy of 78% and a Macro F1-score of 78%. While simple and highly interpretable, this approach struggled to capture complex or implicit offensive language.
- **Meta 2 (Machine Learning):** The optimized Logistic Regression model reached an Accuracy of 85% and an F1-score of 85%. This confirmed the effectiveness of classic machine learning models combined with sparse feature representation (CountVectorizer) for this task.
- **Meta 3 (Prompt Engineering):** The best overall performance was achieved by the Qwen-7B model using the Few-Shot prompting technique, resulting in the highest Accuracy of **86**% and a Macro F1-score of **86**%. This demonstrates the superior potential of in-context learning provided by large language models.

5.2 Error Analysis and False Negatives Reduction

A crucial measure of success for a hate speech detector is the minimization of False Negatives (FNs) - offensive comments incorrectly labeled as harmless. The project showed a major positive trend in FN reduction across the metas:

- **Rule-Based Limitation (Meta 1):** The rigid rules resulted in **230** FNs.
- **Machine Learning Improvement (Meta 2):** The logistic regression model reduced FNs to only **125**, representing a **45.6**% reduction from the initial baseline, proving the superiority of the pattern-learning approach.
- **Prompt Engineering (Meta 3):** The best LLM technique (Qwen Few-Shot) achieved the most balanced result with only **136** FNs and a minimal **60** False Positives (FPs), demonstrating excellent generalization and boundary separation between the offensive and non-offensive classes.

In conclusion, while the Logistic Regression model achieved the most significant single reduction in FNs, the **Qwen-7B Few-Shot model** provided the highest overall performance metrics (Accuracy and F1-score) and reduced 23 FP, making it the definitive best-performing system for this classification task.

6 Acknowledgments

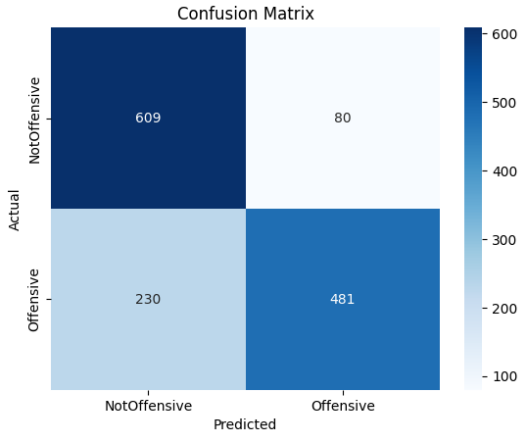
We would like to mention that AI [Gemini] was used to:

- Correct some syntax errors and make the report more cohesive and coherent;
- Provide insights on the methodologies approach;
- Building some components of the code.

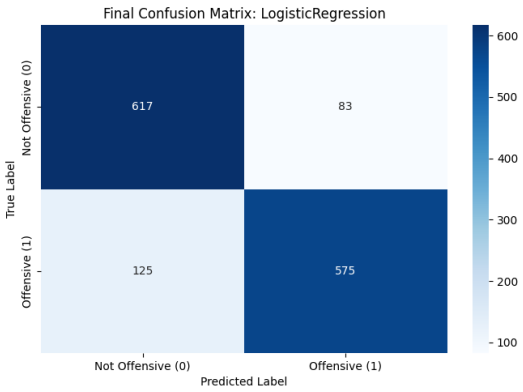
7 References

1. Hugo Oliveira, Isabel Carvalho and Patrícia Ferreira's slides and documents
2. Joachims, T. (1998). *Text categorization with support vector machines: Learning with many relevant features.* In Proceedings of the 10th European Conference on Machine Learning (ECML-98), pp. 137–142.
3. Wang, S., & Manning, C. D. (2012). *Baselines and bigrams: Simple, good sentiment and topic classification.* In Proceedings of the 50th Annual Meeting of the Association for Computational Linguistics (ACL), pp. 90–94.
4. Emojis Token Extraction:
   - `https://www.unicode.org/Public/emoji/1.0//emoji-data.txt`
   - `https://gist.github.com/slowkow/7a7f61f495e3dbb7e3d767f97bd7304b`
   - `https://www.kaggle.com/code/stpeteishii/emoji-with-u200d`
5. Sentiment analysis with Portuguese lexicon and emojis:
   - `https://b2find.eudat.eu/dataset/b6bd16c2-a8ab-598f-be41-1e7aeecd60d3`
   - `https://b2share.eudat.eu/records/93ab120efdaa4662baec6adee8e7585f`
   - `https://doi.org/10.23728/B2SHARE.93AB120EFDAA4662BAEC6ADEE8E7585F`
   - `https://www.kaggle.com/datasets/thomasseleck/emoji-sentiment-data`
   - Yoo, Byungkyu & Rayz, Julia. (2021). Understanding Emojis for Sentiment Analysis. *The International FLAIRS Conference Proceedings, 34.* `https://doi.org/10.32473/flairs.v34i1.128562`
   - `https://www.geeksforgeeks.org/python/reading-writing-text-files-python/`
   - `https://stackoverflow.com/questions/517923/what-is-the-best-way-to-remove-accents-normalize-in-a-python-unicode-string`
6. Portuguese offensive words:
   - `https://pt.scribd.com/document/522716988/palavras-ofensivas` + Pedro Silva Adaptation
7. Optuna Framework: `https://www.kaggle.com/code/stpeteishii/emoji-with-u200d`
8. tqdm Framework for visual completion bars: `https://tqdm.github.io/`
9. LLMs
   - `https://huggingface.co/Qwen/Qwen2.5-7B-Instruct`
   - `https://huggingface.co/meta-llama/Llama-3.1-8B-Instruct`
10. `https://stackoverflow.com/questions/76963311/llama-cpp-python-not-using-nvidia-gpu-cuda`
11. `https://stackoverflow.com/questions/265960/best-way-to-strip-punctuation-from-a-string`

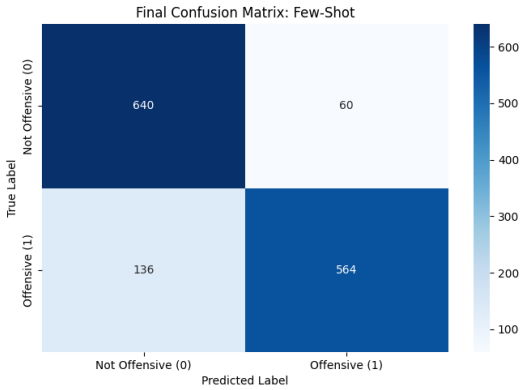
8 Figures and Tables



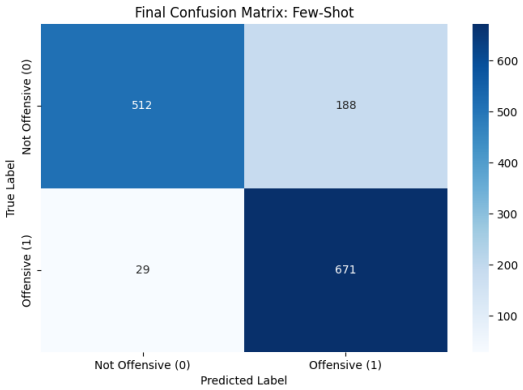
(a) Meta 1 Best Experiment



(b) Meta 2 Best Experiment



(c) Meta 3 Best Experiment (QWEN LLM)



(d) Meta 3 Best Experiment (LLAMA LLM)

Figure 1: Confusion Matrices of all study phases

Classification	Precision	Recall	F1-Score	Support
Not Offensive	0.73	0.88	0.80	700
Offensive	0.86	0.68	0.76	700
Accuracy			**0.78**	1400
Macro Avg	0.79	0.78	0.78	1400
Weighted Avg	0.79	0.78	0.78	1400

Table 6: Classification Report of meta 1 best experiment

Classification	Precision	Recall	F1-Score	Support
Not Offensive	0.83	0.88	0.86	700
Offensive	0.87	0.82	0.85	700
Accuracy			**0.85**	1400
Macro Avg	0.85	0.85	0.85	1400
Weighted Avg	0.85	0.85	0.85	1400

Table 7: Classification Report of meta 2 best experiment

Classification	Precision	Recall	F1-Score	Support
Not Offensive	0.82	0.91	0.87	700
Offensive	0.90	0.81	0.85	700
Accuracy			**0.86**	1400
Macro Avg	0.86	0.86	0.86	1400
Weighted Avg	0.86	0.86	0.86	1400

Table 8: Classification Report for QWEN: Few-Shot Technique (Accuracy: 0.860)

Classification	Precision	Recall	F1-Score	Support
Not Offensive	0.95	0.73	0.83	700
Offensive	0.78	0.96	0.86	700
Accuracy			**0.84**	1400
Macro Avg	0.86	0.84	0.84	1400
Weighted Avg	0.86	0.84	0.84	1400

Table 9: Classification Report for LLAMA: Few-Shot Technique (Accuracy: 0.845)