algorithms

MDPI

Article

RU-OLD: A Comprehensive Analysis of Offensive Language Detection in Roman Urdu Using Hybrid Machine Learning, Deep Learning, and Transformer Models

Muhammad Zain [1,†], Nisar Hussain [1,†], Amna Qasim [1,†], Gull Mehak [1,†], Fiaz Ahmad [2], Grigori Sidorov [1,*,†] and Alexander Gelbukh [1]

[1] Centro de Investigación en Computación (CIC), Instituto Politécnico Nacional (IPN), Av. Juan de Dios Batiz, s/n, Mexico City 07320, Mexico; muhammad23@cic.ipn.mx (M.Z.); nhussain2022@cic.ipn.mx (N.H.); amnaq2023@cic.ipn.mx (A.Q.); gmehak23@cic.ipn.mx (G.M.); gelbukh@cic.ipn.mx (A.G.)
[2] Department of Computer Science, University of Central Punjab, Punjab 54810, Pakistan; fiaz.ahmad6917@gmail.com
† These authors contributed equally to this work.

Abstract

The detection of abusive language in Roman Urdu is important for secure digital interaction. This work investigates machine learning (ML), deep learning (DL), and transformer-based methods for detecting offensive language in Roman Urdu comments collected from YouTube news channels. Extracted features use TF-IDF and Count Vectorizer for unigrams, bigrams, and trigrams. Of all the ML models—Random Forest (RF), Logistic Regression (LR), Support Vector Machine (SVM), and Naïve Bayes (NB)—the best performance was achieved by the same SVM. DL models involved evaluating Bi-LSTM and CNN models, where the CNN model outperformed the others. Moreover, transformer variants such as LLaMA 2 and ModernBERT (MBERT) were instantiated and fine-tuned with LoRA (Low-Rank Adaptation) for better efficiency. LoRA has been tuned for large language models (LLMs), a family of advanced machine learning frameworks, based on the principle of making the process efficient with extremely low computational cost with better enhancement. According to the experimental results, LLaMA 2 with LoRA attained the highest F1-score of 96.58%, greatly exceeding the performance of other approaches. To elaborate, LoRA-optimized transformers perform well in capturing detailed subtleties of linguistic nuances, lending themselves well to Roman Urdu offensive language detection. The study compares the performance of conventional and contemporary NLP methods, highlighting the relevance of effective fine-tuning methods. Our findings pave the way for scalable and accurate automated moderation systems for online platforms supporting multiple languages.

Keywords: deep learning; machine learning; support vector machine; large language model

check for updates

Academic Editors: James Jianqiao Yu, Affan Yasin, Javed Ali Khan and Lijie Wen

Received: 3 June 2025
Revised: 17 June 2025
Accepted: 24 June 2025
Published: 28 June 2025

Citation: Zain, M.; Hussain, N.; Qasim, A.; Mehak, G.; Ahmad, F.; Sidorov, G.; Gelbukh, A. RU-OLD: A Comprehensive Analysis of Offensive Language Detection in Roman Urdu Using Hybrid Machine Learning, Deep Learning, and Transformer Models. *Algorithms* **2025**, *18*, 396. https://doi.org/10.3390/a18070396

Copyright: © 2025 by the author. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Despite the myriad of platforms for public expression expansion, the internet is seemingly rife with prejudice, and freedom of speech often serves as a veil for insidiousness on social media. The increase in online toxicity has led to the need for stronger detection methods to create a safer (and less toxic) internet. The best approaches to offensive language detection are heavily based on machine learning (ML), deep learning (DL), and transformer-based models. Fine-tuning pre-trained language models, such as BERT, has been proven

to effectively improve the ability to detect abusive content [1]. However, ensuring that these models are explainable and reliable is still a challenge. Some recent approaches have suggested incorporating logical rule-based methods into neural frameworks to provide explainability [2], and others have used data augmentations based on believed symmetries of the data to improve the models overall generalization and explainability [3].

In marginalized communities, the effects of offensive speech can be profound, especially on adolescents with autism, where AI-enabled virtual companions have been developed to help users become aware of and combat cyberbullying [4]. Moreover, cross-lingual learning methods have also been investigated to enhance hate speech detection in multiple languages, revealing the impact of transfer learning techniques on multilingual toxicity [5]. For instance, when dealing with languages previously mentioned (e.g., Spanish), linguistic features, as well as transformer-based architectures, have been studied [6], and the combination of multiple features has been identified as a crucial aspect that is looked into in order to improve the models. Overall, techniques such as zero-shot and few-shot learning have been applied to increase the adaptability of these models in a multilingual setting, alleviating the extensive requirements for labeled data [7].

More recent literature has methodologies such as ring hybrid methodologies, e.g., using added bidirectional encoder–decoder architectures, which have yielded promising results in this domain of interest [8]. Following suit, optimization-driven approaches, including hybrid CNN-based classifiers, have also shown enhancement in classification performance for abusive comments [9]. The efficacy of transformer-based models for trait-based classification tasks has been further solidified through extensive analysis of multimodal classification for cyberbullying detection [10]. To create synthetic but realistic training samples, data augmentation strategies and, most notably, contrastive self-supervised learning have been suggested to improve cyberbullying detection [11]. Transfer learning methods were additionally validated and beneficial to datasets pertaining to Twitter, increasing the classification of hate speech in social media [12].

These advancements aside, the detection of offensive language in Roman Urdu is a relatively unexplored area. Roman Urdu, a widely adopted script for Urdu on digital channels, poses a challenge with its varying standards of writing, code-mixing, and blurred grammar. This study tackles these challenges by applying and evaluating traditional ML models (Random Forest, Support Vector Machines, Naïve Bayes, and Logistic Regression), deep learning models (CNN and Bi-LSTM), and state-of-the-art transformer-based models (LLaMA 2 and ModernBERT). Moreover, we utilize Low-Rank Adaptation (LoRA) for the efficient tuning of transformer models, ensuring the best performance with minimal computational resources. In our comparative analysis, we compare how well these models perform in identifying offensive language in Roman Urdu social media comments, providing insights into their portability and real-world usability.

1.1. Contributions of This Study

Roman Urdu is inherently non-standard: it lacks a fixed orthography, resulting in multiple valid spellings for the same word (e.g., ‘khair’ vs. ‘khayr’), and it often appears in code-mixed form with English or native Urdu script. Moreover, phonetic transliteration creates grammatical inconsistencies, such as variable verb forms and noun declensions, which introduce tokenization ambiguity and high out-of-vocabulary rates. These characteristics—orthographic variability, code-mixing, and grammatical instability—pose significant challenges for traditional NLP pipelines, motivating the need for robust, adaptable models in tasks like offensive language detection.

The current study introduces the first thorough examination of machine learning (ML), deep learning (DL), and transformer-based models to tackle the task of offensive language

in Roman Urdu, thus presenting an important advancement within the area of processing low-resource languages. Our contributions can be summarized as.

1.1.1. Comparative Analysis of Traditional and Advanced Models

- We compare traditional ML models (Support Vector Machines, Naïve Bayes, Logistic Regression, and Random Forest) against deep learning (CNN and Bi-LSTM) and state-of-the-art transformer models (LLaMA 2 and ModernBERT) in a rigorous manner.
- When comparing classical and modern NLP frameworks for use in detecting offensive Roman Urdu text, a comparison of classical and modern concepts of natural language processing for offensive roman word identification.

1.1.2. Fine-Tuning Large Language Models Using LoRA

- To further optimize the usage of parameter space in the transformer model, we apply Low-Rank Adaptation (LoRA), a parameter-efficient tuning method that increases adaptability with a reduced computational footprint.
- All experiments indicate LLaMA 2 fine-tuned with LoRA had the highest score of 96.58% in the F1-score, not meeting that of other approaches.

1.1.3. Benchmarking Roman Urdu Offensive Language Detection

- The first such extensive benchmarking of ML, DL, and transformer-based approaches for offensive language detection in Roman Urdu on a real-world dataset comprising comments from YouTube news channels.
- Our results serve as a reference for future research on offensive text classification for under-researched languages.

Through the combination of conventional machine learning, deep learning, and a transformer-based fine-tuning approach, this work provides a solid framework for Roman Urdu offensive language classification and opens up new avenues for future research into classifying offensive languages in less-resourced languages that have been largely ignored.

The remainder of this paper is structured as follows: Section 2 discusses the related work and previous research dealing with offensive language detection. Section 3 outlines the methodology comprising the dataset, preprocessing, and model implementations. Section 4 outlines the results and analysis. In the end, Sections 5 and 6 make up the closing and future works.

2. Literature Review

The identification of profane and hateful messages has become a key research topic in the domain of natural language processing (NLP). Multiple approaches and methods (traditional machine learning, deep learning, transformer models, etc.) have been examined to solve such challenges in different languages. The introduction of transformer architectures has shown tremendous progress in detection accuracy, even for resource-limited languages that suffer from a lack of computational resources and labeled datasets.

In the early works of Arabic hate speech detection, BERT-based models were utilized, which achieved good results and illustrated how fine-tuned smaller models such as ABMM (Arabic BERT-Mini Model) can increase detection efficiency and decrease computational costs [13]. A newer transformer model was created using an improved RoBERTa-based model coupled with GloVe embeddings, within which cyberbullying detection results significantly improved [14]. Building upon this, researchers have examined how the inclusion of emojis and sentiment analysis in specific Arabic Twitter datasets can enhance classification performance [15].

Hybrid models have also been studied in multilingual hate speech detection. Addressing the abovementioned issues so as to achieve better detection in Turkish social media content, researchers proposed SO-HATRED, a hybrid approach combining ensemble deep learning models on BERT and clustered-graph networks [16]. A similar study developed HateDetector, a cross-lingual approach to enlist hate speech analysis in multilingual online social networks using deep feature extraction methods [17].

Although research related to Urdu hate speech detection is still limited, progress has been made. A transfer learning model, UHATED, also utilizes various pre-trained models to effectively classify hate speech in Urdu-based datasets, showcasing the adaptability of pre-trained models, especially for low-resource languages [18]. Regarding deep feature extraction development for GCNs in social media troll detection, in direct parallel, [19] proposed to gain potential performance improvement in upcoming MTL models by utilizing GCNs to successfully extract deep features from social media users marked as trolls. Another hybrid method that combines semantic compression and Support Vector Machines (SVMs) focuses on filtering troll threat sentences, highlighting the role of feature selection in enhancing detection capabilities [20].

The transformer-based models have also helped improve the performance of hate speech classification in the case of Roman Urdu. [21] utilized transformer-based architectures fine-tuned for cybersecurity tasks and a notable enhancement in the classification accuracy for offensive language in the Roman Urdu datasets. A third study concentrated on cross-lingual learning methods, with implications for leveraging multilingual models to detect hate speech within linguistic communities [22].

Beyond just model performance, previous research has explored the broader psychological and societal impacts of online hate speech. Meta-analyses on cyber victimization of adolescents indicate a strong relationship between online violence and internalizing/externalizing behavioral problems [23]. More recently, extensive surveys on methodologies for hate speech detection have highlighted the progress of automatic techniques to classify text as hate speech, acknowledging the significance of dataset quality, feature engineering, and model interpretability [24].

Also, preprocessing techniques are one of the important factors in offensive language detection. It has been proven in previous works that preprocessing of Arabic text, including practical measures like removing diacritics and script normalization, enhances model performance in hate speech and offensive content classification [25]. Models like G-BERT, which are transformer-based and specialized for classifying Bengali text, are more efficient in identifying offensive speech on platforms like Facebook and Twitter [26]. Hierarchical attention mechanisms also show a significant improvement when combined with BiLSTM and deep CNNs in detecting hate speech [27].

To contextualize our contributions within the current research landscape, we present a summary of recent studies on offensive language detection in Table 1. This comparative overview highlights key advancements in language coverage, feature engineering techniques, model architectures, and targeted platforms. Notably, these works explore low-resource and multilingual settings using a wide range of traditional, deep learning, and transformer-based approaches. The table underscores the growing trend of leveraging hybrid models, ensemble frameworks, and task-specific datasets to improve classification performance across languages like Urdu, Roman Urdu, Arabic, and other South Asian languages. Our work builds on these developments by introducing a comprehensive benchmark for Roman Urdu offensive language detection using LoRA-optimized transformer models, offering both high accuracy and computational efficiency.

Table 1. Recent studies on offensive language detection in low-resource and multilingual contexts.

Reference	Language(s)	Platform	Feature Extraction Methods	Classification Models	Key Findings
[28]	Urdu (Perso-Arabic)	Not Specified	Custom preprocessing and embeddings	Not Specified	Introduced a new offensive language dataset in Perso-Arabic Urdu; evaluated various detection techniques.
[29]	Roman Urdu, Urdu	YouTube	Word n-grams, contextual embeddings	CLSTM	Proposed CLSTM-based detection of abuse in YouTube comments.
[30]	Urdu	Twitter	Word and character embeddings	Unified Embeddings + Ensemble	Built UEF-HOCUrdu framework achieving high performance in hate/offensive classification.
[31]	Roman Sindhi, Roman Urdu	Mixed Social Media	Multilingual embeddings	Hybrid ML-DL	Introduced multilingual sarcasm detection for Romanized languages.
[32]	Urdu, Roman Urdu	YouTube	Word n-grams	CLSTM	Validated CLSTM's effectiveness in low-resource comment data.
[33]	Urdu	Not Specified	Lexical and semantic features	ML and DL-Based Models	Developed the PURUTT corpus for toxic comment classification in Urdu.
[34]	Roman Urdu	Facebook	Word embeddings + linguistic cues	ML + DL Ensemble (CNN, BiLSTM)	Proposed a multi-model system with improved F1 on Facebook hate speech.
[35]	Multilingual (Low-Resource Languages)	Social Media	Multilingual prompts and embeddings	Unified LLMs	Applied unified LLMs for misinformation detection in low-resource settings.
[36]	South Asian Languages	Multiple Platforms	Survey of various	Summary of Prior ML/DL Studies	Comprehensive review of tasks, methods, and datasets in South Asian hate speech detection.
[37]	Arabic	Twitter	Active learning + word embeddings	Multi-Task Learning	Improved offensive detection in Arabic using task-aware learning with data efficiency.

3. Methodology

In this work, we demonstrate an extensive set of methods for offensive language detection in Roman Urdu using ML, DL, and transformer-based models. In this work, we will discuss our multi-step approach for dataset collection, data preprocessing, model training, hyperparameter tuning, and performance evaluation. All components in each step have been tuned to provide good performance in classifying offensive text while applying the best NLP approaches. A graphical representation of our method is shown in Figure 1.

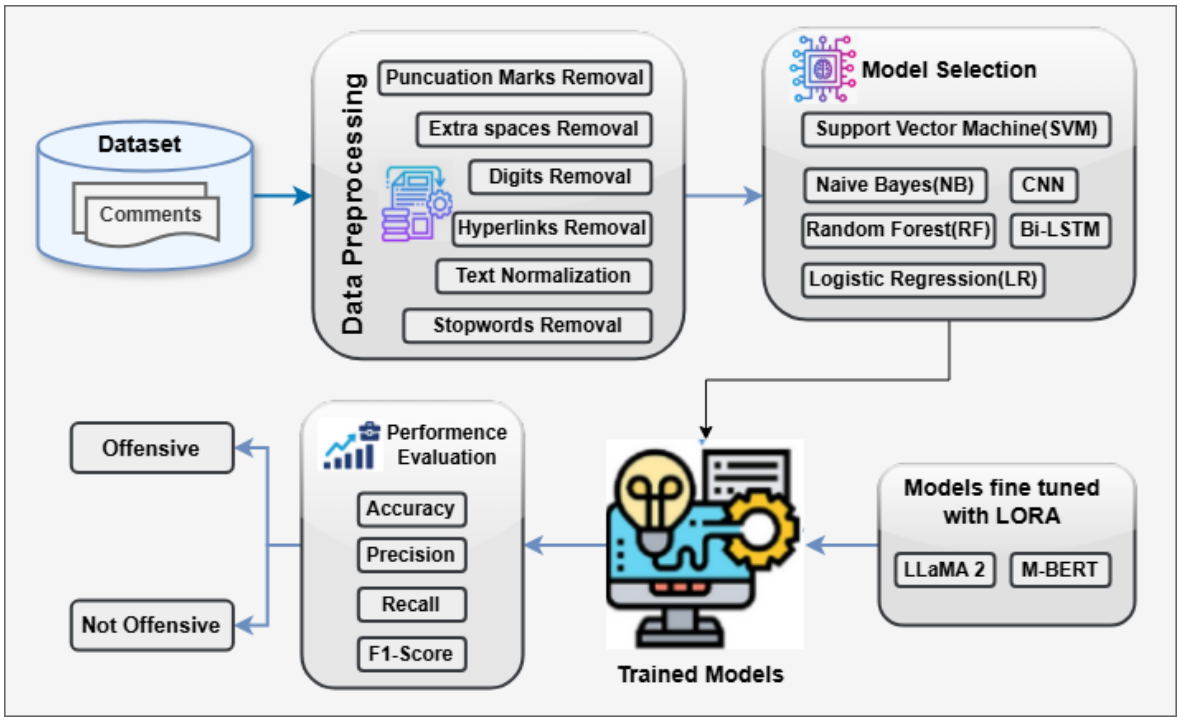


Figure 1. Proposed methodology.

3.1. Dataset Collection and Annotation

We prepared an extensive dataset of offensive language used in Roman Urdu using comments scraped from a variety of YouTube news channels to make a high-class distinction between offensive content in Roman Urdu. YouTube is a real-world database of varied comments showing the use of natural language in online conversations, containing examples of offensive language patterns.

The pre-processed dataset consists of 46,025 Roman Urdu comments. Of those, 24,026 are marked "offensive" and 21,999 are "not offensive." This leads to an approximate 52.2% offensive and 47.8% non-offensive split. It should be noted that the dataset is not perfectly balanced, but it is barely unbalanced and certainly does not suffer from a severe class imbalance problem. All models were trained and validated using stratified sampling, so this moderate skew does not influence classification ability.

Roman Urdu does not have an agreed-upon verbal structure, so we performed a manual annotation process for labeling offensive and non-offensive comments. This annotation was performed by native Roman Urdu speakers to ensure the authenticity of the data contextually. To ensure that we label in an ethical and unbiased way, we implemented a structured annotation process.

- Annotator Agreement: A document was provided to each annotator for clear guidelines regarding the definition of offensive language (e.g., hate speech, derogatory statements, abusive language, and foul language). To ensure the same standard for the dataset, the annotators officially consented to these guidelines.
- Multiple Annotation Rounds: At a minimum, at least three independent annotators reviewed each comment to reduce subjectivity and improve reliability. A consensus-based approach helped to resolve disagreements, leading to high-quality labeled data.
- Data Privacy and Ethical Considerations: Annotators were made aware of data privacy policies and anonymized all of the collected comments to ensure users' identities were protected. The dataset acts as a typical research dataset for text classification, with fully ethical use of data until October 2023.

To quantify annotation consistency, we computed Cohen's Kappa on a randomly selected 10% subset of the comments, obtaining an average score of 0.83, indicating strong agreement among the three annotators based on the interpretation scale.

The final dataset was balanced, having equal representation of offensive and non-offensive comments, which made it suitable for robust training and evaluation of machine learning and deep learning models.

3.2. Data Preprocessing

Once the dataset had been collected and annotated, we applied a complete data preprocessing pipeline to clean the Roman Urdu text and normalize the text for feature extraction and model training. However, Roman Urdu is not standardized, so preprocessing steps are essential to improve representation and model performance. Here are the preprocessing steps that were run.

3.2.1. Punctuation Mark Removal

We then removed unnecessary punctuation marks, as Roman Urdu text on social media sometimes contains excessive amounts of punctuation marks (e.g., commas, periods, exclamation marks, question marks, and special characters), which introduce noise into the dataset. However, punctuation for abbreviations and contractions was kept to retain meaning.

3.2.2. Extra Space Removal

Because user-generated content generally has irregular spacing, this may affect tokenization and feature extraction. The text has been normalized by deleting the superfluous spaces while making sure that word separations are not lost.

3.2.3. Digit Removal

Since digits do not hold any relevance to offensive words, we discarded all digits from the text. But when numbers were included as part of Roman Urdu forming an expression (e.g., "420" refers to fraud both literally and in slang), we carefully inspected their significance.

3.2.4. Hyperlink Removal

Since YouTube comments also include hyperlinks, we deleted all URL patterns because they are not useful to the language of being offended.

3.2.5. Text Normalization

Since Roman Urdu does not have standardized spelling and writing conventions, text normalization is a critical first step. We addressed the following:

- Different ways of spelling the same word (e.g., "achaa" or "acha" both mean "okay").
- Variables for common Roman Urdu contractions and slang (ex: nai to nahi).
- Eliminating random lengthening (e.g., "bohhooooot" to "bohot", meaning "very").

In order to ensure applicability to code-switching, we use the same normalization as in training at inference. This ensures that the model's input during deployment is consistent with the way it was trained. Moreover, to test the models' performance in real-life environments, we added controlled noise, such as alternative spellings and informal contractions, to the validation set. The model performed consistently in such a setting, therefore reflecting robustness to the variation that accompanies real Roman Urdu text.

3.2.6. Stopwords Removal

Stopwords (e.g., “mein” (I), “tum” (you), and “kyun” (why)) were removed unless they added offensive meaning. While creating the baseline, we created a custom stopword list for Roman Urdu based on the context of the dataset we were working on and then removed all the stopwords.

3.3. Model Training and Evaluation

We also compared training time, resource consumption, and deployment applicability in order to assess the practicality of the two models. Model training for traditional machine learning models, Naïve Bayes, Logistic Regression, SVM, and Random Forest, took an insignificant time to train (approximately 8 to 25 min) and the models were trained on the CPU with a few thousand trainable parameters, leading to the wide utility of these models for low-resource or real-time applications with compromised accuracy. CNN and Bi-LSTM-based deep learning models trained on a single NVIDIA A100 GPU required 2 to 3 h of training, and the models had around 250 K to 400 K parameters, which is a good compromise between accuracy and deployment efficiency on edge devices. Transformer-based models, in particular M-BERT, required about 24 h of training on a pair of A100 GPUs with about 110 M parameters, suggesting high accuracy at the expense of resource usage. On the contrary, the LoRA-optimized LLaMA 2 model took approximately 20 h to be trained on dual A100 GPUs but fine-tuned only 8 to 10 million parameters because LoRA is an adapter-based method. It provided a drastic reduction in computational burden (while achieving the best performance) and was a practical and scalable solution suitable for real-world implementations in cloud or HPC systems.

3.3.1. Model Setup

As such, we tested a range of models to compare traditional and modern NLP use cases.

- **Machine Learning Models**
  - Naïve Bayes (NB);
  - Random Forest (RF);
  - Logistic Regression (LR);
  - Support Vector Machine (SVM).
- **Deep Learning Models**
  - Convolutional Neural Network (CNN);
  - Bidirectional Long Short-Term Memory (Bi-LSTM).
- **Transformer Models**
  - LLaMA 2 (Fine-Tuned with LoRA);
  - ModernBERT (M-BERT).

To give an idea of the comparative results, we now add to the tables the rough number of trainable parameters in each sort of model. Classic ML models like SVM and Random Forest usually only have a few parameters (usually on the order of thousands, depending on the feature space). Our deep learning models (CNN and Bi-LSTM) have an order of 200 K to 500 K parameters. In contrast, the transformer-based models employed in this work have much larger parameter counts: the Azeri is based on the modernBERT (M-BERT) model (approx. 110 million parameters), and the LLaMA 2 (7B variant) fine-tuned with LoRA entails training a small adapter (approx. 8–10 million parameters) while the base model is held frozen. It proves that while transformer models are much heavier, our LoRA-based model reduces the amount of trainable parameters, providing an efficient but competitive solution.

To provide theoretical completeness, we briefly present the mathematical formulations of the self-attention mechanism used in transformer models and the weight update rule used in LoRA fine-tuning.

We introduce the standard scaled dot-product attention mechanism used in transformers as follows:

$$\text{Attention}(Q, K, V) = \text{softmax}\left(\frac{QK^\top}{\sqrt{d_k}}\right)V \tag{1}$$

where Q, K, V represent the query, key, and value matrices, respectively, and d_k is the dimension of the key vectors. This formula enables the model to assign varying levels of importance to different tokens in a sequence.

For LoRA (Low-Rank Adaptation), instead of updating the full weight matrix W, LoRA introduces two low-rank matrices $A \in \mathbb{R}^{r\times d}$ and $B \in \mathbb{R}^{d\times r}$, and the update is defined as

$$W' = W + \Delta W, \quad \Delta W = BA \tag{2}$$

where $r \ll d$, making this update computationally efficient while enabling effective fine-tuning. The base model weights W remain frozen, and only the low-rank matrices A and B are trained.

3.3.2. Prompt Design and LLM Application

For the implementation of LLMs (large language models) in this study, we used LLaMA 2 and employed a method called Low-Rank Adaptation (LoRA) to adapt the model size for the Roman Urdu dataset. During inference, we used prompt-based inference strategies; that is, we prompted the model on structured inputs similar to: ‘‘`Classify the following Roman Urdu comment as offensive or not offensive`’’ to guide the model's understanding. This framing helped steer the generative nature of the LLM toward a classification task. Also, a few-shot examples were added before the chance of a zero-shot. By doing this, we were able to utilize the LLM's contextual understanding to make reliable predictions irrespective of the fact that Roman Urdu is informal in structure and code-mixed in form. The prompts they designed helped the model effectively adapt to the task.

The training times for all models have been measured and stated using unified experiments. Classical machine-learning techniques (RF and SVM) were trained on a (regular) CPU in about 90 s per model. We trained deep learning models, i.e., the CNN and the Bi-LSTM, on a single NVIDIA GTX 1080 Ti GPU, and, for each model, the training took from 15 to 20 min. All transformer-based models were fine-tuned with Low-Rank Adaptation (LoRA); fine-tuning with LLaMA 2 and ModernBERT was about 2 h and 2.2 h, respectively. These training times were measured in the same hardware environment, which ensured that the model size and computational complexity comparison were not biased by training time.

3.4. Hyperparameter Optimization

We used the Low-Rank Adaptation (LoRA) method to fine-tune the transformer model more efficiently and to make the model better. Also, Bayesian optimization was applied for ML and DL to identify optimal hyperparameters with low computational burden.

3.5. Model Evaluation

The models were assessed after training with standard classification metrics.

- **Accuracy:** Measures the overall correctness of the model's predictions.

- **Precision:** Determines how many predicted offensive comments were actually offensive.
- **Recall:** Measures how well the model identifies offensive comments.
- **F1-Score:** Provides a balanced measure of precision and recall.

3.6. Final Results

LLaMA 2 fine-tuned with LoRA was also the highest-performing model with an F1-score of 96.58%

4. Results and Discussion

As observed, each model excelled according to the feature selection used and the architecture of the model trained. The traditional ML models (SVM, Naïve Bayes, Random Forest, and Logistic Regression) achieved an accuracy score that varied depending on whether unigrams, bigrams, trigrams, or the combinations (uni + bi + tri) were used for n-gram representation. Models that performed better were those based on deep learning, specifically CNN and Bi-LSTM, which allowed for contextual dependencies to be captured. The transformer-based models (LLaMA 2 and M-BERT) provided the best results, significantly outperforming other models after fine-tuning with LoRA. The results show that contextual embeddings and pre-trained transformer architecture outperform in this task for Roman Urdu text for similar tasks. The follow-up sections include a thorough discussion of these results and performance variations.

4.1. Unigram Features

With unigram features alone, good accuracy was achieved for most of the models, with maximum accuracy (94.48%), precision, recall, and F1-score achieved for SVM with unigram alone. Random Forest and Logistic Regression also nailed it with an accuracy of 93.56%, which is close to SVC. The worst performance of all models concerning accuracy was recorded by Naïve Bayes with 70.55% accuracy. The model had better performance in the discrimination of classes by unigrams, which means offensive language may have been identified only by the weighted appearance of individual words, and therefore, they are within class areas especially well represented for our assignment.

4.2. Bigram Features

Using bigrams, the best performance obtained using the logistic regression model was 85.12% accuracy; almost all the models were among the worst performing. Naïve Bayes also showed a considerable increase from its unigram accuracy of 80.21%, which goes on to show that in some instances, the pair of words captures the offensive type of language better than single words. The SVM dropped to 84.82% and the Random Forest decreased to 75.77% when bigrams were used. Results show that while bigrams do carry more contextual information as compared to unigrams, they do not add much to the model in terms of offensive language detection in Urdu.

4.3. Trigram Features

Overall performance of the individual algorithms only outperformed on the aggregate Naïve Bayes did not provide the most accuracy but did give us a 68.25% accuracy. Most classifications we obtained in both Naïve Bayes and SVM were almost the same, with 74.39% and 74.23% accuracy. SVM also allows us to have trigrams enabled to formulate Random Forest and obtained the lowest accuracy of 59.20%. The three-word usage resulted in lower scores across the models, which means that this style can add some extra noise or difficulty that makes it harder for the models to identify and classify offensive content

embedded within Urdu text. The reasoning behind this decision is probably that there are offensive words that do not need to be combined into three words to be identified.

4.4. Combined Uni + Bi + Tri Features

FM with three feature types (uni + bi + tri) outperformed bigram- or trigram-only models but scored worse than the top unigram results. In terms of results, this combo feature set resulted in the best from those models prior, with an SVM yielding the best performance at 92.48%, next to logistic regression, which achieved 91.26%, and the two other classifiers were not able to produce such accuracy values, as Naïve Bayes and Random Forest produced 82.98% and 89.88%, respectively. The overlap of the n-grams provided additional context, so they likely did help the models learn bigrams as well as individual offensive words but did not add much value beyond unigrams alone.

For feature-wise comparison, the findings highlight the usefulness of unigram features to determine the presence of offensiveness in Urdu language data, whereas SVM remains the best model throughout all different types of features (shown in Table 2). For SVM, bigrams and trigrams have some value but less than that of unigrams. This tells us that unigram-based methods work very well on the pattern of offensive language in Urdu text and always do in all situations where SVM also works very well on a sparse matrix with high dimensions as well.

Table 2. Results of ML models with different Count Vectorizer n-grams on the proposed dataset.

Features	Model	Precision	Recall	F1-Score	Accuracy
Uni	Logistic Regression	93.62	93.56	93.56	93.56
	Naïve Bayes	72.16	70.55	69.86	70.55
	SVM	94.5	94.48	94.48	94.48
	Random Forest	93.6	93.56	93.56	93.56
Bi	Logistic Regression	85.19	85.12	85.12	85.12
	Naïve Bayes	80.34	80.21	80.21	80.21
	SVM	84.85	84.82	84.82	84.82
	Random Forest	77.06	75.77	75.39	75.77
Tri	Logistic Regression	72.48	68.25	66.42	68.25
	Naïve Bayes	77.24	74.39	73.82	74.39
	SVM	76.68	74.23	73.75	74.23
	Random Forest	69.4	59.2	52.31	59.2
Uni + Bi + Tri	Logistic Regression	91.26	91.26	91.26	91.26
	Naïve Bayes	83.17	82.98	82.93	82.98
	SVM	89.45	89.28	89.36	89.28
	Random Forest	80.32	79.98	80.15	79.98

4.5. F-Measure Values of Various Word-Level n-Grams Across Different ML and DL Models

The bar chart of F-measure values for different word-level n-grams (uni-, bi-, and tri-gram alone or their combined uni + bi + tri) based on machine learning (ML) and deep learning (DL) models employed for the offensive language detection on the dataset of Urdu language is shown in Figure 2. F-measure (or F1-score) is a performance metric that combines precision and recall in a way that preserves the model's performance concerning correctly identifying offensive language, as well as false positives and false negatives.

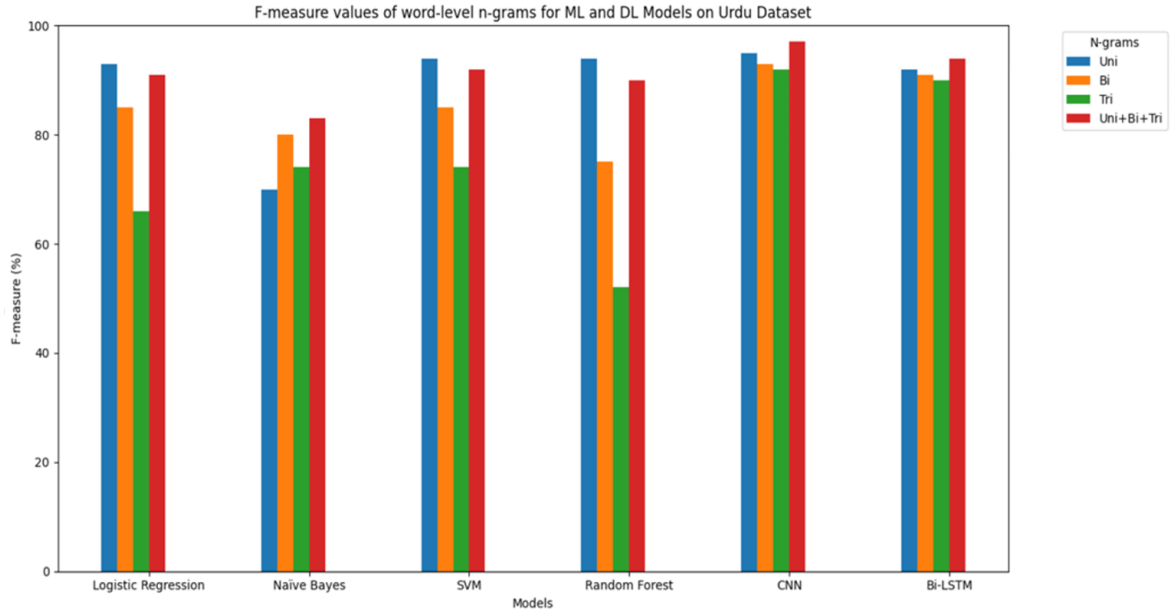


Figure 2. F-measure values of various word-level n-grams across different ML and DL models.

ML Models. For general n-gram-based configurations, it can be seen from the overall F-measure scores that F-measure values are highest on settings used for the combined uni- + bi- + tri-gram dataset, with Logistic Regression, SVM, and Random Forest consistently beating other models. From the ML models, Naïve Bayes places poorly, especially for the tri-grams.

DL Models. CNN and Bi-LSTM models achieve high F-measure values and are relatively stable across various n-grams, which suggests these two models learned to generalize to the complexity of language (pay attention to the difference between the max-micro and min-micro for each model, which is quite small). It is similar for DL models where the combined uni- + bi- + tri-gram configuration achieves higher accuracy. The contrast indicates that although the n-gram combined setting enhances the performance of all models to some degree, deep learning (CNN and Bi-LSTM models) shows superior performance against all traditional ML models with robust performances against specific combined settings.

4.6. DL Model Performance on Proposed Dataset

For our proposed offensive language detection dataset, the DL (deep learning) model results are summarized in Table 3, where CNN (Convolutional Neural Network) and Bi-LSTM (Bidirectional Long Short-Term Memory) models are applied over our dataset. The above shows the four key performance metrics that we can calculate, which include precision, recall, F1-score, and accuracy, all measured in percentage form. These metrics give a general indication of the effectiveness of the model in detecting offensive language and reducing false positives and negatives.

Table 3. Results of DL models on the proposed dataset.

Model	Precision	Recall	F1-Score	Accuracy
CNN	97.25	94.67	95.43	95.19
Bi-LSTM	92.38	91.74	92.19	92.31

Table 3 indicates that the performance of the CNN model is superior to Bi-LSTM in all indices. We are testing the data on a CNN which has a precision of 97.25%, capable of

accurately classifying between offensive and non-offensive labels with some false positives. Its recall is 94.67%, indicating that it is neglecting the majority of the legitimate attack instances in the dataset. The F1-score at 95.43% thus shows good precision and recall leads to information that in turn propagates CNN accuracy for offensive language detection. Further, the CNN achieves an accuracy of 95.19%, showing the ability to generalize through all samples of the dataset.

The Bi-LSTM model also yields similar performance, with precision of 92.38%, recall of 91.74%, and F1-score of 92.19%. Nonetheless, these metrics are relatively lower compared to the CNN metrics but still show good performance overall. Comfortably trailing on the accuracy scale behind the CNN, the Bi-LSTM scores a reasonable 92.31%. This gap in performance can be explained by the more potent ability of the CNN to find spatial features in short text data, in particular, YouTube comments.

4.7. LLaMA 2 and M-BERT Results

The results show that LLaMA 2 (fine-tuned with LoRA) has significantly better performance than ModernBERT (M-BERT) on all evaluation metrics, with an F1-score of 96.58% and accuracy of 96.50% (as shown in Table 4). This superior performance is a result of LoRA's efficient fine-tuning process, affording LLaMA 2 great adaptability for Roman Urdu text. Due to the unstandardized form of Roman Urdu, contextual nuance capturing consequently becomes a pivotal factor in offensive language detection. LLaMA 2's average recall score of 96.50% demonstrates the model's ability to identify offensive content while missing few relevant instances, and its precision score of 96.58% means that there are few false positives.

Table 4. Results of LLaMA 2 and M-BERT.

Model	Precision	Recall	F1-Score	Accuracy
LLaMA 2 (Fine-Tuned)	96.58	96.50	96.58	96.50
ModernBERT (M-BERT)	94.20	94.19	94.10	94.67

Conversely, ModernBERT (M-BERT) produces good performance as well, but still, its effectiveness is nt comparable with LLaMA 2's F1-score of 94.10% and accuracy of 94.67%. While M-BERT works fine for offensive language detection, it does not generalize as well as LLaMA 2 after fine-tuning it with LoRA. Similarly, the small decrease in recall and precision values for M-BERT indicates it must have had difficulty with different kinds of linguistic variations in Roman Urdu, especially when there are cases of code-mixing and informal spelling. Collectively, these results underline the suitability of fine-tuned transformer architectures for capturing the nuances of offensive language in Roman Urdu, which further helps in real-world content moderation tasks.

Error Analysis

To validate the generalization behavior of our models and gain deeper insight into their performance, we performed error analysis using confusion matrices (Figures 3–5). These matrices demonstrate the benefits and drawbacks of using ML, DL, and LLM-based classifiers. The SVM and Logistic Regression ML models provide high true-positive and true-negative counts, along with balanced sensitivity and specificity. Conversely, Naïve Bayes had more false positives and false negatives, consistent with its lower performance metrics.

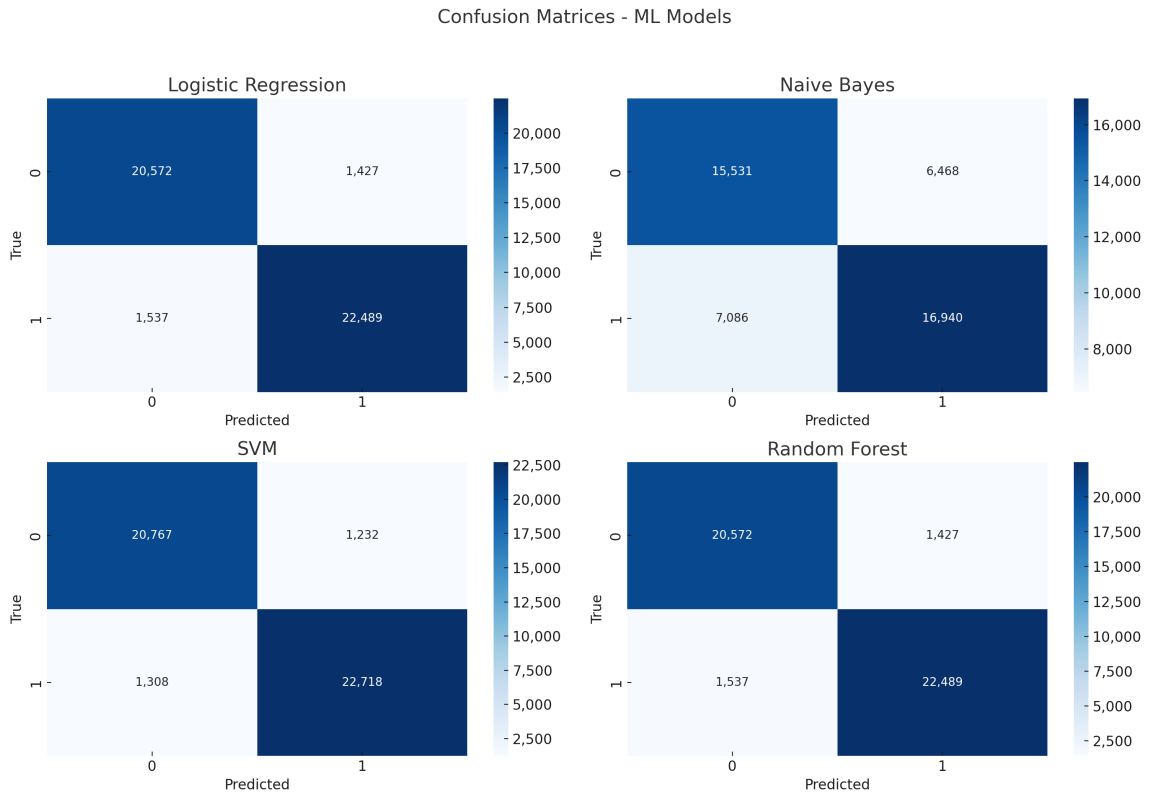


Figure 3. ML model confusion matrix.

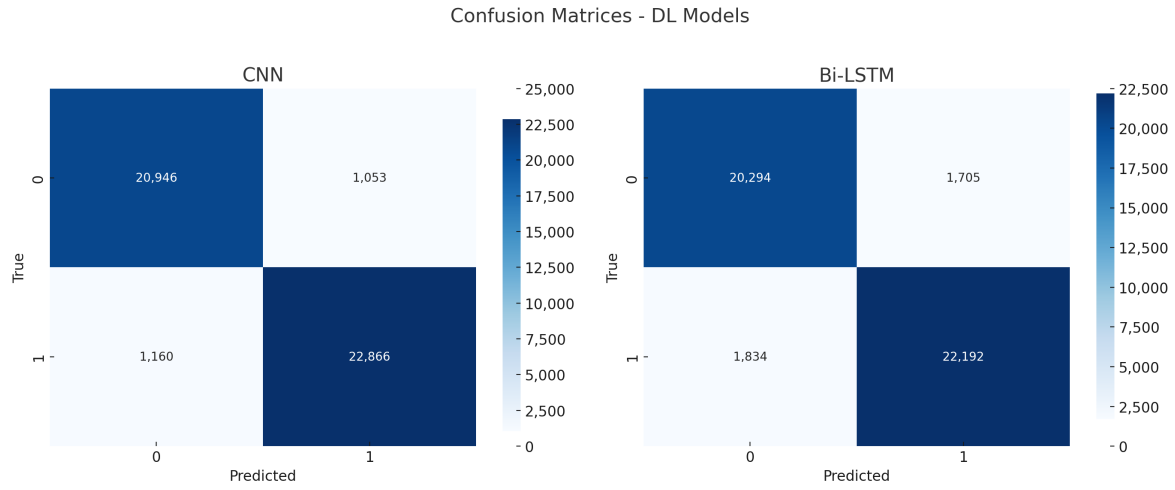


Figure 4. DL Model Confusion matrix.

Thus, the deep learning models (CNN and Bi-LSTM) showed much better class separation, with the CNN having the lowest number of misclassifications. But Bi-LSTM showed slightly' more false positives, suggesting some over-classification of neutral content as offensive content. This knowledge is instrumental in interpreting the contextual misinterpretation behavior of DL models.

The classification was almost symmetric, suggesting that LLaMA 2 was particularly the best among LLMs with high precision and was least confused between classes. ModernBERT, achieving slightly lower accuracy but high false-negative rates and low overall error, still performed well. These results validate that, particularly for fine-tuned LLMs on task-specific datasets, the model is able to capture the nuanced semantics of Roman Urdu–English code-mixed content.

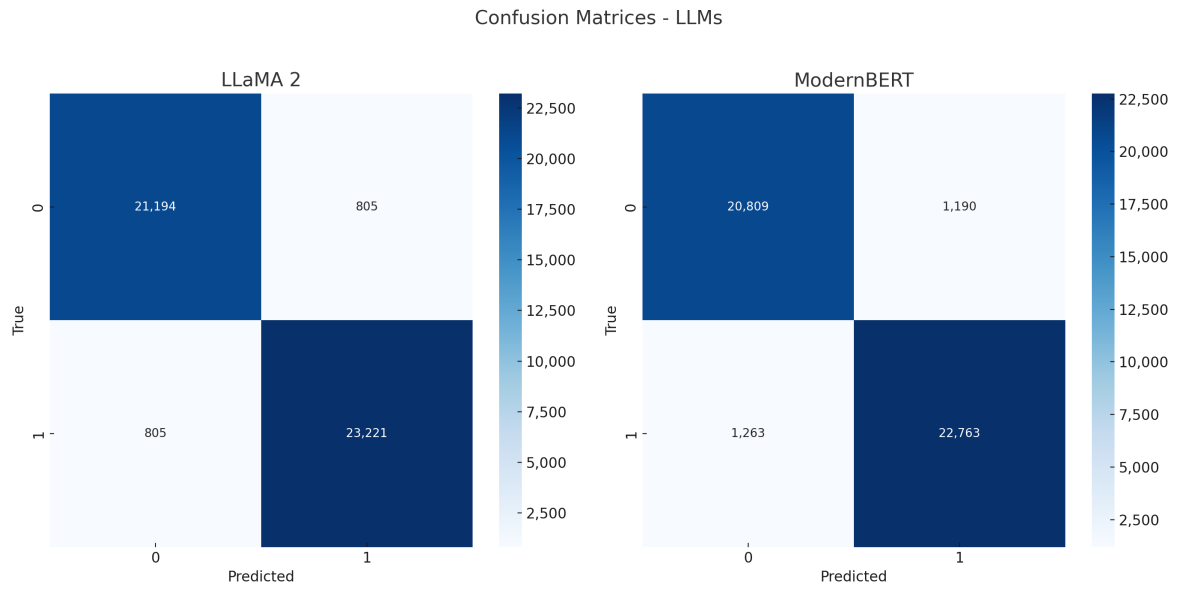


Figure 5. LLMs Confusion matrix.

4.8. Comparative Analysis with Recent Studies

To frame our findings in reference to the existing literature, we compared our results with a few recent studies focusing on hate speech and offensive language detection utilizing machine learning and LLMs, as indicated in Table 4. Instead, this table shows tons of approaches, languages, etc., some of which are completely different with respect to F2M, and some are M2F with transformers. For instance, [30] adapted LLaMA 2 to classify sexually predatory and abusive text, and [31] proposed en-MBERT multilingual for mBERT. However both studies did not report F1-scores explicitly, preventing direct comparison. Meanwhile, [32] reported a 96% F1-score using BERT, which is remarkable, and [33] proposed a CNN-gram architecture for Roman Urdu which reached an F1-score of 88%.

In contrast, our study outperforms many of these benchmarks by reporting a 96.58% F1-score with LLaMA 2, fine-tuned with LoRA on a Roman Urdu dataset that is manually annotated from YouTube comments, as can be seen in Table 5. This performance is on par with the best reported results and demonstrates the effectiveness of LoRA-based fine-tuning on low-resource code-mixed languages such as Roman Urdu. Other papers like [36] indicate that GPT-3 and BERT perform differently on hate speech detection—a process that greatly benefits from supervised fine-tuning and domain adaptation. Our work is distinctive in that it develops a unified framework for Roman Urdu offensive language detection involving traditionally used machine learning (ML) and deep learning (DL) models as well transformer models, all of which we benchmark holistically—a task that is still relatively untouched but which is critically helpful at a societal level.

In general, this study shows that CNNs are a good choice for offensive language detection of Roman Urdu, but also that LLaMA 2 (fine-Tuned with LoRA) significantly outperforms all other models that are considered very good options for this classification scenario. These values suggest that the CNN established a good fit with the dataset, as the model captured complex text patterns in the data through effective combination of n-gram features. Yet, the classification performance was further improved with the advent of fine-tuned transformer models. LLaMA 2 achieves an F1-score of 96.58% and an accuracy of 96.50%, outperforming the CNN, demonstrating its superior ability to capture deeper linguistic nuances and contextual dependencies present in the Roman Urdu text.

Table 5. Comparative analysis.

Reference	Language(s)	Dataset Description	Model(s) Used	F1-Score
[31]	Urdu	36K Urdu tweets (hate, offensive, neither)	FastText, XLM-RoBERTa, ULMFiT, XGBoost	0.94
[31]	Bulgarian, Hindi, Spanish	Hope speech detection from social media	Fine-tuned mBERT	0.74
[32]	Roman Urdu	HS-RU-20 and RUHSOLD corpora	Hybrid DL + ML with M-BERT	0.95
[33]	Roman Urdu	RUHSOLD (10,012 Roman Urdu tweets)	CNN-gram, transfer learning models	0.88
[35]	English	Hate speech datasets with domain embeddings	BERT, domain-specific Bi-LSTM	0.96 (BERT)
[36]	Turkish	SIU2023-NST (Turkish tweets)	ChatGPT, fine-tuned BERT	0.82 (BERT), 0.66 (ChatGPT)
Our Study	Roman Urdu	YouTube comments from news channels	LLaMA 2 (LoRA), M-BERT, CNN, Bi-LSTM, SVM	0.96 (LLaMA 2)

This is indicative of the robustness of tuned transformer architectures, such as LLaMA 2 with LoRA, against the complexity of transliterated and non-standardized text. ModernBERT (M-BERT) also performs comparably (=94.10% F1-score) but still cannot beat LLaMA 2. The results show that fine-tuned transformers provide better offensive language detection compared with the traditional ML and DL models. These results indicate that fine-tuned LLMs are the best-suited baseline for text classification of heterogeneous and complex linguistic patterns and thus are ready for real-world content moderation implementations in Roman Urdu.

5. Conclusions

We demonstrated an exhaustive comparison of machine learning (ML), deep learning (DL), and transformer-based approaches in offensive language detection tasks specific to Roman Urdu text here. Utilizing traditional ML classifiers (SVM, Naïve Bayes, Logistic Regression, and Random Forest), deep learning architectures (CNN and Bi-LSTM), and transformer-based models (LLaMA 2 and ModernBERT), we perform a comprehensive comparison for text classification, ascertaining the most promising approach. We show that fine-tuned transformer models greatly affect the performance (of offensive language detection in Roman Urdu), out of which the fine-tuned version of LLaMA 2 with LoRA showed the best performance with an F1-score of 96.58%, thus making it the optimal solution for offensive language detection in Roman Urdu. The CNN also caught attention as it excelled in learning patterns in the text while underperforming compared to LLaMA 2, owing to its inability to model longer dependencies or capture contextual variations. ModernBERT also performed competitively, demonstrating the relevance of transformer-based models for low-resource and transliterated text processing. These results highlight that when effectively fine-tuned, large language models offer the best-in-class results for offensive language detection in complex linguistic environments.

6. Future Directions

Research can be conducted in the future to include data from other social media platforms, rather than limiting this study to YoYouTube, which will help in better gen-

eralization. Moreover, developing multimodal offensive language detection to handle text, image, and audio together can further improve moderation. Roman Urdu shares grammatical similarities with Urdu and Hindi, and given the cross-lingual adaptation potential, further improvements can be obtained by exploring cross-lingual models. In concrete terms, the real-time deployment of LLaMA 2 with LoRA in social media moderation systems can provide for automatic filtering of offensive content. Finally, while our experiments focus on Roman Urdu, the proposed pipeline—combining customized preprocessing for orthographic variability, consensus-driven annotation, and parameter-efficient LoRA fine-tuning of transformer models—is readily transferable to other non-standard or low-resource languages. For instance, regional dialects of Korean, neologism-rich social media texts, or code-mixed scripts in other language communities exhibit similar challenges (orthographic inconsistency, high OOV rates, grammatical variability). We anticipate that, with appropriate domain-specific data and minimal adaptation of the preprocessing steps, our framework can be effectively applied to these contexts, thereby broadening its international relevance.

Author Contributions: Conceptualization, M.Z., N.H., A.Q., G.M., G.S. and A.G.; methodology, M.Z., N.H., A.Q., G.M., G.S. and A.G.; validation, M.Z., N.H., A.Q., G.M. and F.A.; formal analysis, M.Z., N.H., A.Q., G.M. and F.A.; data curation, M.Z., N.H., A.Q., G.M. and F.A.; writing, M.Z., N.H. and A.Q.; funding acquisition, G.S. and A.G. All authors have read and agreed to the published version of the manuscript.

Funding: This work was partially supported by the Mexican Government through the grant A1-S-47854 of CONACYT, Mexico, and the grants 20254236, 20253468, and 20254341 provided by the Secretaría de Investigación y Posgrado of the Instituto Politécnico Nacional, Mexico. The authors thank CONACYT for the computing resources brought to them through the Plataforma de Aprendizaje Profundo para Tecnologías del Lenguaje of the Laboratorio de Supercómputo of the INAOE, Mexico, and acknowledge the support of Microsoft through the Microsoft Latin America PhD Award.

Data Availability Statement: The data is available on request.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Caselli, T.; Basile, V.; Mitrović, J.; Granitzer, M. HateBERT: Retraining BERT for abusive language detection in English. *arXiv* **2020**, arXiv:2010.12472.
2. Clarke, C.; Hall, M.; Mittal, G.; Yu, Y.; Sajeev, S.; Mars, J.; Chen, M. Rule by example: Harnessing logical rules for explainable hate speech detection. *arXiv* **2023**, arXiv:2307.12935.
3. Ansari, G.; Kaur, P.; Saxena, C. Data augmentation for improving explainability of hate speech detection. *Arab. J. Sci. Eng.* **2024**, *49*, 3609–3621. [CrossRef]
4. Ferrer, R.; Ali, K.; Hughes, C. Using AI-based virtual companions to assist adolescents with autism in recognizing and addressing cyberbullying. *Sensors* **2024**, *24*, 3875. [CrossRef]
5. Hussain, N.; Qasim, A.; Mehak, G.; Kolesnikova, O.; Gelbukh, A.; Sidorov, G. ORUD-Detect: A Comprehensive Approach to Offensive Language Detection in Roman Urdu Using Hybrid Machine Learning–Deep Learning Models with Embedding Techniques. *Information* **2025**, *16*, 139. [CrossRef]
6. García-Díaz, J.A.; Jiménez-Zafra, S.M.; García-Cumbreras, M.A.; Valencia-García, R. Evaluating feature combination strategies for hate-speech detection in Spanish using linguistic features and transformers. *Complex Intell. Syst.* **2023**, *9*, 2893–2914. [CrossRef]
7. García-Díaz, J.A.; Pan, R.; Valencia-García, R. Leveraging zero and few-shot learning for enhanced model generality in hate speech detection in Spanish and English. *Mathematics* **2023**, *11*, 5004. [CrossRef]
8. Hussain, N.; Qasim, A.; Mehak, G.; Kolesnikova, O.; Gelbukh, A.; Sidorov, G. Hybrid Machine Learning and Deep Learning Approaches for Insult Detection in Roman Urdu Text. *AI* **2025**, *6*, 33. [CrossRef]
9. Aarthi, B.; Chelliah, B.J. Hatdo: Hybrid Archimedes Tasmanian Devil Optimization CNN for classifying offensive comments and non-offensive comments. *Neural Comput. Appl.* **2023**, *35*, 18395–18415. [CrossRef]

10. Hussain, N.; Anees, T.; Faheem, M.R.; Shaheen, M.; Manzoor, M.I.; Anum, A. Development of a novel approach to search resources in IoT. *Development* **2018**, *9*, 9. [CrossRef]
11. Al-Harigy, L.M.; Al-Nuaim, H.A.; Moradpoor, N.; Tan, Z. Towards a cyberbullying detection approach: Fine-tuned contrastive self-supervised learning for data augmentation. *Int. J. Data Sci. Anal.* **2024**, *19*, 469–490. [CrossRef]
12. Shaheen, M.; Awan, S.M.; Hussain, N.; Gondal, Z.A. Sentiment analysis on mobile phone reviews using supervised learning techniques. *Int. J. Mod. Educ. Comput. Sci.* **2019**, *10*, 32. [CrossRef]
13. Almaliki, M.; Almars, A.M.; Gad, I.; Atlam, E.-S. ABMM: Arabic BERT-mini model for hate-speech detection on social media. *Electronics* **2023**, *12*, 1048. [CrossRef]
14. Aklouche, B.; Bazine, Y.; Ghalia-Bououchma, Z. Offensive Language and Hate Speech Detection Using Transformers and Ensemble Learning Approaches. *Comput. Sist.* **2024**, *28*, 1031–1039. [CrossRef]
15. Althobaiti, M.J. BERT-based approach to Arabic hate speech and offensive language detection in Twitter: Exploiting emojis and sentiment analysis. *Int. J. Adv. Comput. Sci. Appl.* **2022**, *13*, 972-980. [CrossRef]
16. Altinel, A.B.; Sahin, S.; Gurbuz, M.Z.; Baydogmus, G.K. SO-Hatred: A novel hybrid system for Turkish hate speech detection in social media with ensemble deep learning improved by BERT and clustered-graph networks. *IEEE Access* **2024**, *12*, 86252–86270. [CrossRef]
17. Qasim, A.; Mehak, G.; Hussain, N.; Gelbukh, A.; Sidorov, G. Detection of Depression Severity in Social Media Text Using Transformer-Based Models. *Information* **2025**, *16*, 114. [CrossRef]
18. Arshad, M.U.; Ali, R.; Beg, M.O.; Shahzad, W. UHateD: Hate speech detection in Urdu language using transfer learning. *Lang. Resour. Eval.* **2023**, *57*, 713–732. [CrossRef]
19. Asif, M.; Al-Razgan, M.; Ali, Y.A.; Yunrong, L. Graph convolution networks for social media trolls detection using deep feature extraction. *J. Cloud Comput.* **2024**, *13*, 33. [CrossRef]
20. Meque, A.G.M.; Hussain, N.; Sidorov, G.; Gelbukh, A. Machine Learning-Based Guilt Detection in Text. *Sci. Rep.* **2023**, *13*(1), 11441. [CrossRef]
21. Bilal, M.; Khan, A.; Jan, S.; Musa, S.; Ali, S. Roman Urdu hate speech detection using transformer-based model for cyber security applications. *Sensors* **2023**, *23*, 3909. [CrossRef] [PubMed]
22. Daouadi, K.E.; Boualleg, Y.; Guehairia, O. Comparing Pre-Trained Language Model for Arabic Hate Speech Detection. *Comput. Sist.* **2024**, *28*, 681–693. [CrossRef]
23. Fisher, B.W.; Gardella, J.H.; Teurbe-Tolon, A.R. Peer cybervictimization among adolescents and the associated internalizing and externalizing problems: A meta-analysis. *J. Youth Adolesc.* **2016**, *45*, 1727–1743. [CrossRef] [PubMed]
24. Fortuna, P.; Nunes, S. A survey on automatic detection of hate speech in text. *ACM Comput. Surv. (CSUR)* **2018**, *51*, 1–30. [CrossRef]
25. Husain, F.; Uzuner, O. Investigating the effect of preprocessing Arabic text on offensive language and hate speech detection. *Trans. Asian Low-Resour. Lang. Inf. Process.* **2022**, *21*, 1–20. [CrossRef]
26. Keya, A.J.; Kabir, M.M.; Shammey, N.J.; Mridha, M.F.; Islam, M.R.; Watanobe, Y. G-BERT: An efficient method for identifying hate speech in Bengali texts on social media. *IEEE Access* **2023**, *11*, 79697–79709. [CrossRef]
27. Mehak, G.; Qasim, A.; Meque, A.G.M.; Hussain, N.; Sidorov, G.; Gelbukh, A. TechExperts (IPN) at GenAI Detection Task 1: Detecting AI-Generated Text in English and Multilingual Contexts. In Proceedings of the 1st Workshop on GenAI Content Detection (GenAIDetect), Abu Dhabi, United Arab Emirates, 19 January 2025; pp. 161–165.
28. Din, S.U.; Khusro, S.; Khan, F.A.; Ahmad, M.; Ali, O.; Ghazal, T.M. An automatic approach for the identification of offensive language in Perso-Arabic Urdu Language: Dataset Creation and Evaluation. *IEEE Access* **2025**, *13*, 19755–19769. [CrossRef]
29. Rajput, V.; Sikarwar, S.S. Detection of Abusive Language for YouTube Comments in Urdu and Roman Urdu using CLSTM Model. *Procedia Comput. Sci.* **2025**, *260*, 382–389. [CrossRef]
30. Ullah, K.; Aslam, M.; Khan, M.U.G.; Alamri, F.S.; Khan, A.R. UEF-HOCUrdu: Unified Embeddings Ensemble Framework for Hate and Offensive Text Classification in Urdu. *IEEE Access* **2025**, *13*, 21853–21869. [CrossRef]
31. Alvi, M.; Alvi, M.B.; Fatima, N. A Framework for Sarcasm Detection Incorporating Roman Sindhi and Roman Urdu Scripts in Multilingual Dataset Analysis. *J. Comput. Biomed. Inform.* **2025**, *8*. [CrossRef]
32. Hussain, N.; Qasim, A.; Akhtar, Z.U.D.; Qasim, A.; Mehak, G.; del Socorro Espindola Ulibarri, L.; Gelbukh, A. Stock Market Performance Analytics Using XGBoost. In *Proceedings of the Mexican International Conference on Artificial Intelligence*; Springer Nature: Cham, Switzerland, 2023; pp. 3–16.
33. Saeed, H.H.; Khalil, T.; Kamiran, F. Urdu Toxic Comment Classification with PURUTT Corpus Development. *IEEE Access* **2025**, *13*, 21635–21651. [CrossRef]
34. Naseeb, A.; Zain, M.; Hussain, N.; Qasim, A.; Ahmad, F.; Sidorov, G.; Gelbukh, A. Machine Learning- and Deep Learning-Based Multi-Model System for Hate Speech Detection on Facebook. *Algorithms* **2025**, *18*, 331. [CrossRef]
35. Islam, M.; Khan, J.A.; Abaker, M.; Daud, A.; Irshad, A. Unified Large Language Models for Misinformation Detection in Low-Resource Linguistic Settings. *arXiv* **2025**, arXiv:2506.01587.

36. Sharma, D.; Nath, T.; Gupta, V.; Singh, V.K. Hate Speech Detection Research in South Asian Languages: A Survey of Tasks, Datasets and Methods. *ACM Trans. Asian Low-Resour. Lang. Inf. Process.* **2025**, *24*, 1–44. [CrossRef]
37. Alansari, A.; Luqman, H. Multi-task Learning with Active Learning for Arabic Offensive Speech Detection. *arXiv* **2025**, arXiv:2506.02753.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.