A Focused SVM Framework for Offensive Language Detection in Urdu Youtube Comments

Momina Hafeez, Amna Qasim, Gull Mehak, Nisar Hussain*

Instituto Politécnico Nacional, Centro de Investigación en Computación, Mexico City, Mexico

mhafeez2025@cic.ipn.mx, nisar.hussain8400@gmail.com

Abstract. Detecting offensive language in Urdu is challenging due to short and noisy comments, morphological differences, and lack of datasets. We present an empirical study of Support Vector Machine (SVM) based binary offensive language detection using Urdu comments on YouTube videos. The feature sets for unigrams, bigrams, trigrams, and all three are evaluated with traditional classifiers, and for CNN and Bi-LSTM models, the paper's results are used. The best performing ML model is an SVM using unigram features, which achieves a precision, recall, F1-score, and accuracy of 95.50%/95.48%/95.48%/95.48% respectively. If we look at Bigram and Trigram embedding separately, we can see the results of SVM are lowered to 85.82% and 74.75% respectively. However, when we train SVM using both Bigram and Trigram embeddings, we achieved F1-Score of 93.25%. CNN has F1-Score of 95.15% and 95.00% accuracy. Bi-LSTM has F1-Score of 92.22% and 92.00% accuracy. These results show the potential of the standard word-level lexical features on this corpus. Moreover, our work establishes the SVM as a fast, strong baseline for offensive language detection in Urdu. We put our work into context by reviewing recent work on transformer, hybrid and parameter-efficient techniques.

Keywords: Urdu NLP, offensive language detection, support vector machine, SVM, n-grams, TF-IDF, YouTube comments, low-resource languages.

Un marco de trabajo basado en SVM enfocado en la detección de lenguaje ofensivo en comentarios de YouTube en urdu

Resumen. La detección de lenguaje ofensivo en urdu presenta desafíos debido a la brevedad y el ruido de los comentarios, las diferencias morfológicas y la escasez de conjuntos de datos. Presentamos un estudio empírico sobre la detección binaria de lenguaje ofensivo mediante máquinas de vectores de soporte (SVM), utilizando comentarios en urdu de vídeos de YouTube. Se evalúan conjuntos de características basados en unigramas, bigramas, trigramas y la combinación de los tres utilizando clasificadores tradicionales; para los modelos CNN y Bi-LSTM, se utilizan los resultados reportados en la literatura. El modelo de aprendizaje automático con mejor rendimiento es la SVM utilizando

características de unigramas, la cual alcanza una precisión, exhaustividad (recall), puntuación F1 y exactitud del 95.50%, 95.48%, 95.48% y 95.48%, respectivamente. Al analizar por separado las representaciones vectoriales (embeddings) de bigramas y trigramas, observamos que los resultados de la SVM descienden al 85.82% y al 74.75%, respectivamente. Sin embargo, al entrenar la SVM utilizando ambas representaciones (bigramas y trigramas), se obtuvo una puntuación F1 del 93.25%. El modelo CNN alcanzó una puntuación F1 del 95.15% y una exactitud del 95.00%, mientras que el modelo Bi-LSTM obtuvo una puntuación F1 del 92.22% y una exactitud del 92.00%. Estos resultados demuestran el potencial de las características léxicas estándar a nivel de palabra en este corpus. Además, nuestro trabajo establece a la SVM como una línea base rápida y sólida para la detección de lenguaje ofensivo en urdu. Situamos nuestro trabajo en contexto revisando investigaciones recientes sobre técnicas basadas en transformers, enfoques híbridos y métodos de eficiencia paramétrica.

Palabras clave: PNL en urdu, detección de lenguaje ofensivo, máquina de vectores de soporte, SVM, n-gramas, TF-IDF, comentarios de YouTube, idiomas de bajos recursos.

1 Introduction

User-generated content (UGC) on social media platforms creates a large communication space, and the scale of user-generated content on social media platforms makes it impossible to detect the use of abusive language in user-generated content through human observation. This has resulted in a demand for automatic methods for reliably detecting abusive language for moderation.

It is particularly challenging to detect abusive comments for low-resource languages like Urdu, because Urdu experiences important orthographic and morphological variation, and text on the internet also includes a lot of informal language, misspellings, abbreviations, punctuation noise, and code-mixed words [1]. These features increase sparsity, especially for shorter YouTube comments, which present a smaller context to build a classification upon.

In recent years, multilingual transformers, contextual embeddings, and parameter efficient fine-tuning have been used, but traditional machine-learning methods still remain useful due to their low computational cost and ability to show better performance with smaller datasets. Support Vector Machine (SVM) is the model of choice for high-dimensional sparse representations such as those generated in text classification, and is a competitive model in Urdu and Roman Urdu classification tasks. Newer work has shown an improved performance of SVM when combined with multilingual BERT features in a hybrid approach [2].

This article focuses on the SVM module of an Urdu YouTube offensive-language detection task. It compares word-level unigram, bigram, trigram and combined-feature-based SVM classifier with standard classifiers and the reported neural models[3]. Thus, the aim should be whether SVM still remains a viable computational scheme for the Urdu corpus considered, rather than the state-of-the-art performance.

Our contributions are: (i) an SVM-based analysis on Urdu offensive language, (ii) a systematic comparison of word-level n-gram representations, (iii) a comparison with

conventional and reported neural classifiers, and (iv) a comparison of results with the recent Urdu and Roman Urdu literature on offensive language detection.

Another reason would be the trade-off between classification performance and computational resources. For instance, where the comments would be streamed in, it would be better to use a model that performs decently with fewer computational resources than a model that performs better but is computationally heavier. Other potential advantages of SVM include low cost prediction times, as sparsity allows efficient implementation, and the possibility of inspecting models through discriminative features. This makes SVM attractive both as an end classifier and as a reliable baseline comparator to identify whether more complex classifiers are giving meaningful improvements on the same task.

2 Related Work

Traditional machine learning methods for offensive language and hate speech detection have since been exceeded by deep learning methods, multilingual transformer-based models, and large language models. These methods use human-constructed lexical features, statistical features, and classifiers, including Logistic Regression, Naïve Bayes, and Support Vector Machine (SVM) for the task. These methods are especially relevant to low-resource languages with little annotated data and low computational resources[4]. A recent systematic review on Urdu hate-speech detection claims that even in this specific task, traditional machine-learning methods such as SVM, Logistic Regression, Naïve Bayes and related classifiers continue to be especially important in the field of Urdu NLP. The survey identifies data scarcity, code-switching, and context variety as the key challenges of Urdu hate-speech detection [5].

Other factors include the limited availability of annotated Urdu data sets compared to high-resource languages, spelling variation, informal words, code-switching, and other dimensions of linguistic noise encountered in social media, which may lead to feature sparsity and poor classification performance. Akhter et al. explored these issues in their research on the automatic identification of offensive language in Urdu and Roman Urdu and demonstrated the feasibility of machine learning for low-resourced varieties of the Urdu language [6]. More recent survey research shows that Urdu offensive- and hate-speech detection still requires more effective datasets, standardized evaluation metrics, and methods for modeling linguistic variation and context [7].

However, there are some recent works where SVM is being used on top of stronger (deep) feature representations. For instance, Ashiq et al. used a hybrid framework for detecting Roman Urdu hate speech using multilingual BERT (MBERT) as a feature extraction layer and SVM as a classifier[8]. The fine-tuned model MBERT-SVM achieved 0.95 and 0.88 F1 scores in the Neutral-Hate Speech and Hate Speech-offensive sub-challenges, respectively, in the HS-RU-20 corpus and 0.94 and 0.84 for coarse-grained and fine-grained classification, respectively, in the RUHSOLD corpus[9]. These results were meaningful because they demonstrated that SVM is capable of utilizing the contextual representations of text that are produced by transformer models rather than bag of words features [10].

A similar trend was observed in a research on the detection of offensive language in Roman Urdu YouTube comments, with the SVM model being the best performing

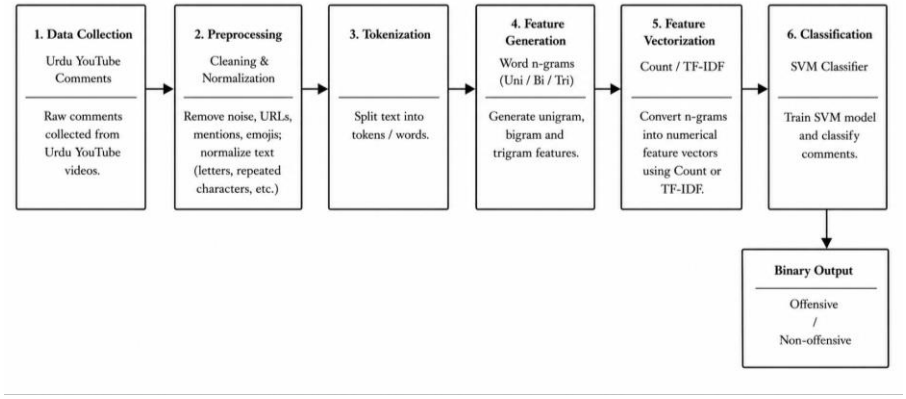

Fig. 1. SVM-centered framework used to structure the Urdu offensive-language detection pipeline.

machine learning model. It also reported the results of CNN and Bi-LSTM architectures, as well as LoRA tuned LLaMA 2 and ModernBERT models. LLaMA 2 models fine-tuned using LoRA exhibited the best performance, achieving an F1-score of 96.58% [11, 12]. The results demonstrated that parameter-efficient fine-tuning with context proves helpful for the detection of offensive-language in Roman Urdu.

Despite these advances, direct comparison between published results should be made cautiously because studies frequently use different datasets, annotation schemes, language varieties, and social-media platforms. Urdu-script and Roman Urdu should not be considered identical evaluation settings, even though they share challenges such as informal writing and linguistic variation [12, 13]. Therefore, a controlled evaluation within the same dataset is necessary to understand the contribution of the classifier and feature representation independently.

Based on the existing literature, an important research direction is to examine whether a conventional classifier such as SVM can still provide strong performance when applied to Urdu social-media text using carefully selected lexical representations. While transformer-based approaches offer richer contextual representations, SVM provides advantages in terms of computational efficiency, relatively simple training, and suitability for sparse high-dimensional text features.

Accordingly, the present study focuses on SVM and systematically evaluates unigram, bigram, trigram, and combined word-level representations for offensive-language detection in Urdu YouTube comments. This focused analysis complements recent transformer-based research by establishing a strong and transparent conventional baseline against which more computationally intensive approaches can be evaluated [15, 16].

3 Dataset and Preprocessing

The evaluated corpus consists of user-generated Urdu comments collected from YouTube news and general-interest channels. Each comment is assigned to one of two classes: offensive or non-offensive. The reported experimental protocol uses an 80% training and 20% test split.

Preprocessing includes removal of URLs, hashtags, punctuation and unwanted symbols, normalization of whitespace and Urdu character variants, and tokenization. The purpose is to reduce orthographic noise while retaining lexical evidence relevant to offensive expressions.

The available source does not provide a sufficiently explicit class-count table in the supplied manuscript; therefore, exact class totals are not reproduced here rather than introducing unsupported numbers. This is a limitation that should be addressed in a final submission by reporting the complete dataset statistics.

4 Proposed SVM-Centered Method

Figure 1 shows the overall pipeline of the proposed offensive-language detection model. Urdu YouTube comments are input, and the pipeline includes preprocessing, tokenization, n-gram generation, numeric feature representation, classification, and lastly, binary classification of comments. SVM is preferred, since the vector spaces defined by the comments are high-dimensional and sparse, and SVM is capable of determining a decision boundary that optimally separates the classes.

4.1 Data Preparation and Text Preprocessing

Data preprocessing can remove the unneeded linguistic and platform noise from the comments, like links, hashtags, punctuation marks and unwanted symbols, as well as normalizing whitespace and variants in Urdu characters that could be used to represent the same letter. The next step is the tokenization. Instead of only removing linguistic information from the data, researchers attempt to reduce noise while leaving words and phrases that can act as markers for the possible offensive content.

After preprocessing, the final sequence of tokens is the input to the model, while the target variable for the task of text classification is a binary label of offensive or non-offensive. Notably, an 80%/20% split of the data is used in training and testing, where 80% is used to derive the feature representation and train classifiers, and the held-out 20% is used for final evaluation.

4.2 Word-Level N-Gram Features

To investigate the importance of the use of local lexical content four different word-level configurations are tested in the study. A unigram denotes a single word, whereas a bigram and a trigram denote a sequence of two or three words, respectively. The fourth configuration is the unigram-bigram-trigram combination. The combined representation captures all three layers.

Unigrams are simple to implement and use little memory, and they capture the direct presence of the most highly offensive words in the corpus. Although bigrams and trigrams allow the model to learn short local patterns, they expand the feature space and add sparsity to smaller collections. Comparing these representations allows us to study whether a local context is helpful or detrimental to classification performance on Urdu comments.

Table 1. Conventional machine-learning performance across word-level n-gram configurations (%).

Features	Model	Precision	Recall	F1 Score	Accuracy
Uni	LR	94.62	94.56	94.56	94.56
Uni	NB	73.16	71.55	70.86	71.55
Uni	SVM	95.50	95.48	95.48	95.48
Uni	RF	94.60	94.56	94.56	94.56
Bi	LR	86.19	86.12	86.12	86.12
Bi	NB	81.34	81.21	81.21	81.21
Bi	SVM	85.85	85.82	85.82	85.82
Bi	RF	78.06	76.77	76.39	76.77
Tri	LR	73.48	69.25	67.42	69.25
Tri	NB	78.24	75.39	74.82	75.39
Tri	SVM	77.68	75.23	74.75	75.23
Tri	RF	70.40	60.20	53.31	60.20
Uni+Bi+Tri	LR	92.26	92.26	92.26	92.26
Uni+Bi+Tri	NB	84.17	83.98	83.93	83.98
Uni+Bi+Tri	SVM	93.32	93.65	93.25	93.40
Uni+Bi+Tri	RF	89.26	89.57	89.45	89.42

4.3 Feature Vectorization

With the n-grams obtained, all the textual features are built with two representations. The first representation regards the n-grams as features and considers their count, while the second representation considers the importance of the n-grams that are used often within a document, but not often within the whole corpus of documents. Since any individual comment only uses a relatively small proportion of the full vocabulary, resulting vectors are sparse.

4.4 Support Vector Machine Classification

For the classification task, Support Vector Machine is used, which seeks to find the hyperplane that separates the offensive class from the non-offensive class, in the feature vector set, with the largest margin. For a linear decision function this can be written as $f(x) = w^T x + b$, where x is the feature representation of the input, w is the weight vector learned by the algorithm, b is a bias term and the class of the input is predicted depending on the sign of f(x).

The nature of this task makes it well-suited for SVMs. Word n-gram representations have thousands of dimensions but only a handful of them are activated for any individual comment. SVM naturally fits this high dimensional sparse space and is

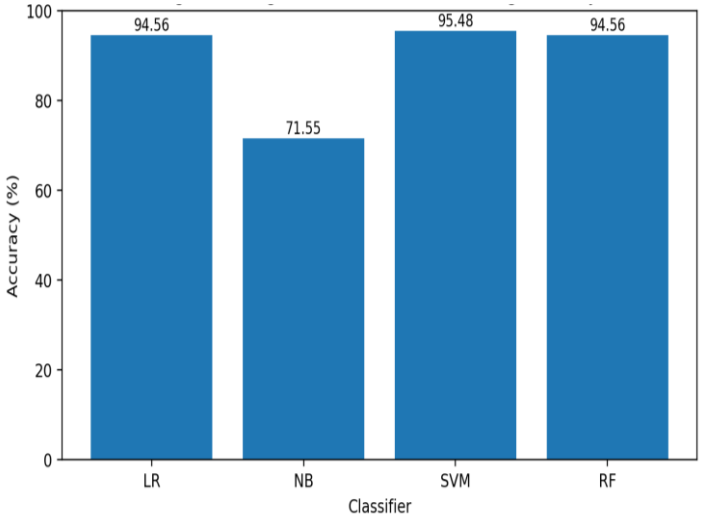

Fig. 2. Unigram-based accuracy comparison among conventional ML models.

computationally inexpensive, making it suitable for high and low resource NLP tasks as well.

4.5 Comparative Classifiers and Neural Baselines

While we are interested to find out if the SVM performance is due to classifier-specific behavior or holds true for the general feature representation, we compare SVM results with those of Logistic Regression (LR), Naive Bayes (NB), and Random Forest (RF) using the same feature representations. We also use the best SVM result to compare against the published results from CNN and Bi-LSTM to show a neural-model reference.

While it is true that such comparisons should be taken with caution since the classifiers work on different assumptions (CNNs can capture local features through convolution filters, Bi-LSTM performs computations on the input sequence from left to right and right to left to capture longer dependencies, and SVM is applied directly on sparse representations given by the lexical features), comparing them allows quantifying the effectiveness of the lexical features regardless of the assumptions.

4.6 Evaluation Protocol

Potential performance measures include precision, recall, F1-score, and accuracy. For example, precision is defined as TP/(TP+FP) where TP is the number of offensive comments detected and FP is the number of non-offensive comments detected. Recall is TP/(TP+FN), where FN are the offensive comments the classifier failed to detect.

F1 score is the harmonic mean of precision and recall: F1 = 2PR/(P+R). Accuracy (TP+TN)/(TP+TN+FP+FN) is also useful. Different performance metrics convey different information about the model being used. A model may exhibit high accuracy

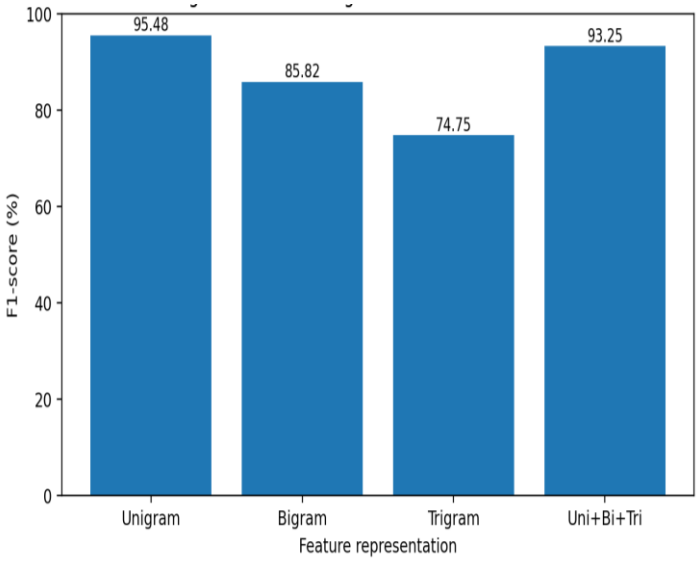

Fig. 3. Effect of n-gram order on SVM F1-score.

while still producing unwanted false positives or false negatives, when identifying offensive language.

Overall, this is a systematic way of expounding on, and observing the relationship between the granularity of the lexical features and classifier performance. Below, in the Results section, the SVM with the highest performance was considered and compared against the normal and reported neural approaches.

5 Results

The results of the Logistic Regression (LR), Naive Bayes (NB), Support Vector Machine (SVM), and Random Forest (RF) with four variations of word-level features are shown in Table 1. It is clear that the SVM performs best when using the unigram word-level features, with the best results for precision, recall, F1 score, and accuracy all equal to 95.50%, 95.48%, 95.48%, and 95.48%, respectively.

This small difference between the precision and recall illustrates that, in this configuration, SVM is balanced, while LR and RF give an accuracy of 94.56% each, and NB only reaches an accuracy of 71.55%.

The results in the table also illustrate the effect that the n-gram representation has on the performance of the classifiers under consideration, and thus motivate the examination of SVM in the next subsection.

The close performance of SVM, Logistic Regression, and Random Forest suggests that the unigram representation contains strong lexical signals for the target classes. SVM nevertheless provides the best overall balance across precision, recall, F1-score, and accuracy, making it the most appropriate model for the focused analysis.

Table 2. Comparison of the strongest SVM configuration with reported neural models (%).

Model	Precision	Recall	F1	Accuracy
SVM (Unigram)	95.50	95.48	95.48	95.48
CNN	97.22	94.23	95.15	95.00
Bi-LSTM	92.12	91.56	92.22	92.00

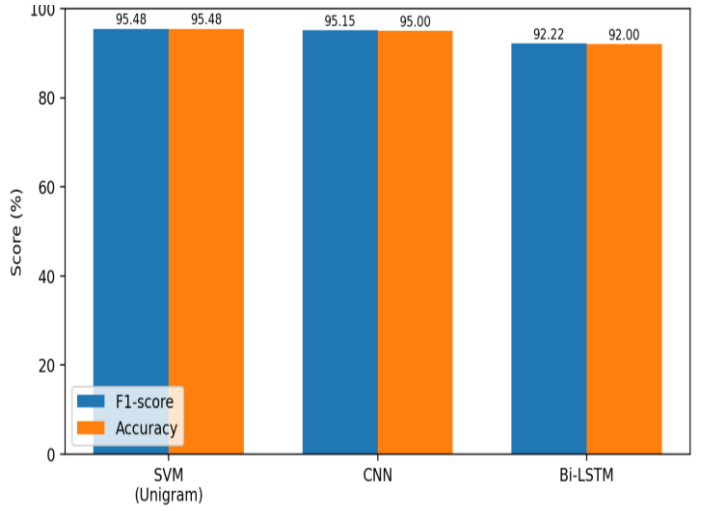

Fig. 4. F1-score and accuracy comparison between SVM, CNN, and Bi-LSTM.

5.1 N-Gram Sensitivity

In fact, the unigram-based accuracy values returned by the four standard classifiers can be seen in Figure 2 below. We see that the best value is returned by SVM with an accuracy value of 95.48%, while both Logistic Regression and Random Forest return an accuracy value of 94.56%. In contrast, the Naive Bayes classifier scores a lower accuracy of 71.55%, which once again reiterates the effectiveness of the SVM classifier choice for the feature-sensitivity analyzes given in Table.

Figure 3 illustrates the effect of increasing n-gram order on SVM F1-score. The unigram configuration clearly outperforms the alternatives at 95.48%. Performance falls to 85.82% with bigrams and 74.75% with trigrams, while the combined representation recovers part of the loss to 93.25%. The trend indicates that, for this short-comment corpus, individual lexical cues are more consistently informative than longer word sequences.

5.2 Comparison with Neural Models

To set the best SVM result in the context of the neural models, Table 2 compares the results of the unigram SVM with the CNN and Bi-LSTM models. Note that neural

Table 3. Positioning against selected recent literature. Metrics are not directly comparable across different datasets and protocols.

Study / Approach	Language / Domain	Best reported metric	Method
Akhter et al. (2020)	Urdu & Roman Urdu	F1 up to 95.9	ML with character/word n-grams
S. Nisar et al. (2022)	Urdu	—	Semantic and embedding models
Ashiq et al. (2024)	Roman Urdu	F1 94.76	Linear SVM
N. Nisar et al. (2025)	Roman Urdu / Facebook	F1 94.76 (ML)	SVM + embeddings / DL
M. Zain et al. (2025)	Roman Urdu / YouTube	F1 96.58	LLaMA 2 + LoRA
Present focused analysis	Urdu / YouTube	F1 95.48	SVM + unigram

approaches compute representations end-to-end with the neural architecture, while SVMs are only able to use lexical representations, so this comparison is fair.

Here, the SVM model again performs best, with an F1-score of 95.48% and an accuracy of 95.48%. However, it is marginally better than the CNN model's 95.15% F1-score and 95.00% accuracy. The precision of CNN is also the highest with 97.22%. Finally, Bi-LSTM comes in last with the F1 score and accuracy of 92.22% and 92%, respectively. This shows that a relatively simple sparse-feature classifier is highly competitive when the corpus is short-text and lexically informative.

Figure 4 makes the comparison easier to interpret by showing F1-score and accuracy together. The two SVM values are closely matched, while CNN shows a similarly strong overall level but a higher precision. The Bi-LSTM results are lower on both aggregate measures in the reported experiment.

5.3 Comparison with Literature Review

Table 3 positions the present result against selected recent studies covering Urdu and Roman Urdu offensive- and hate-speech detection. The comparison should be interpreted as contextual rather than as a strict leaderboard because the studies use different datasets, domains, annotation schemes, and evaluation protocols. Nevertheless, the reported results show that SVM remains competitive with recent conventional approaches, including hybrid systems using contextual embeddings.

At the same time, the recent RU-OLD study demonstrates that parameter-efficient LLM adaptation can achieve higher performance on a related Roman Urdu YouTube task. This contrast motivates future work that combines the efficiency of SVM with contextual representations rather than treating classical and neural approaches as mutually exclusive.

6 Discussion and Future Work

The results collectively indicate that feature representation is a major determinant of offensive-language detection performance in the evaluated Urdu YouTube corpus. SVM performs best with unigram features, suggesting that individual lexical items provide strong discriminative information. The substantial decline observed for bigrams and trigrams indicates that increasing feature order also increases sparsity and may introduce rare sequences that do not generalize well. The combined n-gram representation improves over trigrams but does not recover the unigram result. From a practical perspective, the strongest SVM configuration offers an attractive balance between accuracy and computational simplicity.

Future work should extend this analysis in two directions. First, character n-grams, subword features, and optimized SVM hyper parameters can be evaluated to improve robustness to Urdu spelling variation and informal online writing. Second, SVM can be combined with contextual representations from multilingual BERT or XLM-R, allowing the classifier to retain its efficient decision mechanism while receiving richer semantic features. Parameter-efficient approaches such as LoRA can also be evaluated on the same dataset. Cross-platform validation, class-wise confusion matrices, statistical significance testing, and detailed error analysis would further strengthen the evaluation.

7 Conclusion

For offensive-language detection in Urdu YouTube comments, the strongest performance, with 95.48% accuracy and F1-score for traditional classifiers and feature sets, was achieved by the unigram SVM classifier. The performance of the SVM was competitive with the neural models despite requiring a much simpler pipeline of features and classifiers, showing the continued need for strong lexical representations in the low-resource short-text domain. Despite this, transformer and parameter-efficient models have more recently been proposed to better capture contextual information, which cannot be modeled by lexical representations alone. Future work could explore the performance of contextual embeddings, character/subword representations, support vector machine classifiers with optimal hyperparameter configuration, and efficient hybrid models in the same evaluation setup.

This paper presents the first complete empirical work on offensive language classification in YouTube comments (in Urdu) using SVM. The Unigram SVM achieved the best reported conventional ML results, with an accuracy of 95.48% and an F1-score of 95.48%. It performs worst on bigram and trigram representations, suggesting that sparse lexical information is especially important in this corpus. When comparing SVM and CNN classifiers, the SVM slightly outperformed the CNN in both F1 and accuracy, while the CNN outperformed the SVM in precision. Recent literature shows the SVM remains a strong conventional baseline, and transformer and LoRA-based approaches considerably improve performance in similar Roman Urdu contexts. The framing proposed in this paper adds a simple, efficient and reproducible SVM baseline to inform future work in context-based modeling.

References

1. Akhter, M. P., Jiangbin, Z., Naqvi, I. R., Abdelmajeed, M., Sadiq, M. T.: Automatic detection of offensive language for Urdu and Roman Urdu. IEEE Access, 8, 91213–91226 (2020). https://doi.org/10.1109/ACCESS.2020.2994950
2. Hussain, S., Malik, M. S. I., Masood, N.: Identification of offensive language in Urdu using semantic and embedding models. PeerJ Computer Science, 8, e1169 (2022). https://doi.org/10.7717/peerj-cs.1169
3. Ashiq, W., Kanwal, S., Rafique, A.: Roman Urdu Hate Speech Detection Using Hybrid Machine Learning Models and Hyperparameter Optimization. Scientific Reports, 14, 28590 (2024). Doi: 10.1038/s41598-024-79106-7.
4. 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, 16(2) (2025). Doi: 10.3390/info16020139.
5. 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, 18(7):396 (2025)
6. Hussain, I., Farooq, U., Cheema, A.N.: A Comprehensive Survey on Urdu Hate Speech Detection: Methods, Evaluation, and Challenges. IEEE Access, 13 (2025). Doi: 10.1109/ACCESS.2025.3591143.
7. Aklouche, A., Bazine, Y., Ghalia-Bououchma, Z.: Offensive Language and Hate Speech Detection Using Transformers and Ensemble Learning Approaches. Computación y Sistemas, 28(3), pp. 1031–1039 (2024)
8. Daouadi, K. E., Boualleg, Y., Guehairia, O.: Comparing pre-trained language model for Arabic hate speech detection. Computación y Sistemas, 28(2), 681–693 (2024). https://doi.org/10.13053/CyS-28-2-4130
9. Rahman-Laskar, S., Gupta, G., Badhani, R., Pinto-Avendaño, D. E.: Cyberbullying detection in a multi-classification codemixed dataset. Computación y Sistemas, 28(3), 1091–1113 (2024). https://doi.org/10.13053/CyS-28-3-4989
10. Hafeez, F., Hafeez, M., Shaff Bin Imran, M., Qasim, A., Hussain, N., Ahmad, F., Sidorov, G.: A comparative study of deep learning and transformer models for Twitter sentiment analysis. International Journal of Combinatorial Optimization Problems and Informatics, 17(1), 85–89 (2026). https://doi.org/10.61467/2007.1558.2026.v17i1.1241
11. Kaur, M., Saini, M.: Artificial intelligence inspired method for cross-lingual cyberhate detection from low-resource languages. ACM Transactions on Asian and Low-Resource Language Information Processing, 23(9), Article 134, 1–23 (2024). https://doi.org/10.1145/3677176
12. Ahmad, F., Hussain, N., Qasim, A., Usman, M., Zain, M., Hafeez, M., Hafeez, F., Sidorov, G.: Toward robust offensive language detection in Urdu YouTube discourse. International Journal of Combinatorial Optimization Problems and Informatics, 17(1), 76–84 (2026). https://doi.org/10.61467/2007.1558.2026.v17i1.1240
13. Hafeez, M., Shah, M. Q., Zain, M., Qasim, A., Hussain, N., Sidorov, G.: Beyond transformers: Lightweight multilingual hate speech detection using MLP and TF-IDF. Polibits, 67(1) (2025)
14. Hafeez, M., Hussain, N., Qasim, A., Zain, M., Mehak, G., Kolesnikova, O., Sidorov, G., Gelbukh, A.: Sarcasm detection in Roman Urdu text: A comprehensive study using machine learning and large language model. In: Advances in Soft Computing, pp. 245–254 (2025). https://doi.org/10.1007/978-3-032-09037-9_20
15. Abro, S., Shaikh, S., Khand, Z. H., Ali, Z., Khan, S., Mujtaba, G.: Automatic hate speech detection using machine learning: A comparative study. International Journal of Advanced

Computer Science and Applications, 11(8) (2020). https://doi.org/10.14569/IJACSA.2020.0110861

16. Ojo, O.E., Ta, H.T., Gelbukh, A., Calvo, H., Sidorov, G., Adebanji, O.O.: Automatic hate speech detection using CNN model and word embedding. Computación y Sistemas, 26(2), 1007–1013 (2022). https://doi.org/10.13053/CyS-26-2-4107