Spectrum of Engineering Sciences
ISSN (e) 3007-3138 (p) 3007-312X Volume 4, Issue 1, 2026

MACHINE LEARNING-BASED THREAT DETECTION IN LOW-RESOURCE URDU TWEETS: COMPARATIVE EVALUATION OF CLASSICAL MODELS

Amjad Khan[*1], Saif Ullah Noor[2], Reyan[3], Jawad Ahmad[4], Hilal Khan[5]

[*1, 2, 3, 4,5]*Department of Computer Science, University of Science and Technology, Bannu, Pakistan*

[*1]amjadkhanbscs@gmail.com,[2]saifipathan194@gmail.com
[3]reyankhattak0304@gmail.com,[4]jawadahmadofficial46@gmail.com, [5]hilalpathan01@gmail.com

DOI: https://doi.org/10.5281/zenodo.18439289

Keywords

Article History
Received: 06 December 2025
Accepted: 16 January 2026
Published: 31 January 2026

Corresponding Author: *
Amjad Khan

Abstract

The high rate of social media websites development has brought positive opportunities as well as bad ones, such as threats, harassment, and hate speech. Twitter is a highly volatile source of short messaging or tweets, which makes it very difficult to identify threats in real time because of use of informal language, abbreviations as well as user-defined slang. Machine learning-based automated detection systems will be needed to overcome this challenge. This paper provides a comparative study of four classical machine learning classifiers, namely Logistic Regression, Random Forest, Naive Bayes, and Support Vector machine (SVM), in the threat detection process of the tweets. The authors used 3,170 manually labeled tweets (2,891 non-threat and 279 threat) as the dataset of the study. The preprocessing tools such as tokenization, stop-words elimination and TF-IDF vectorization are used to transform textual data into numbers that one can use to classify. Standard metrics used to measure the model include precision, recall, F1-score, and accuracy. The findings show a balanced performance with a high accuracy of 82.69 which shows that SVM is most accurate in both threat and non-threat classes. Logistic Regression and Naive Bayes give competitive results but slightly with less accuracy whereas Random Forest is robust yet sensitive to class imbalance. This discussion indicates that conventional machine learning classifiers are successful in detecting threats to the social media and serves as a reference point in future studies on the use of deep learning models and real-time use cases.

1 Introduction

Communication has also innovated through the social media platforms as the information can be shared across the world in real-time. Twitter is one of such platforms, and it has become one of the leading microblogging services that contain millions of active users daily, sharing short textual messages, or tweets [1, 2]. Although social media can lead to a sense of communication and sharing of knowledge, it has as well become a platform of malicious acts, which include cyber threats, harassment, misinformation and hate speech [3]. Threats detection in these contents is crucial to the safety of the populace, internet security, and the health of the internet [4]. The shortness of the text, non-professional language, abbreviations, emojis, and slang expressions make it difficult to identify threats in tweets automatically [5]. Conventional systems based on rules are ineffective in the face of semantic ambiguity whereby the false positive and false negative rates are high. By comparison, machine learning-based methods have the ability to extract intricate textual data patterns allowing the detection of threats more accurately and scalably.

Other machine learning algorithms to threat detection and text classification have been investigated recently. Logistic Regression is simple and easy to interpret, and it finds good use on high-dimensional textual data. Naive Bayes

classifiers can be used in probabilistic text classification, but can also perform poorly in cases where the assumptions of the independence of features are not met. Random Forest is an ensemble-based approach that improves classification robustness by aggregating multiple decision trees. Support Vector Machine (SVM) is shown to perform well in high-dimensional and sparse text representations, particularly under class imbalance. Although deep learning models, including LSTM and BERT, are highly popular, classical machine learning classifiers are still useful, particularly when there are limited computational resources or smaller datasets. The systematic comparison of classical algorithms can serve as a baseline and can be used to choose models for realistic threat detection systems.

1.1 Contribution of the Paper

The purpose of this paper is to create and analyze the machine learning-based techniques of threat detection in tweets. In particular, the following contributions have been made:

1. It uses four classic machine learning classifiers to detect threatening content in twitter data; the Logistic Regression, random Forest, Naive Bayes, and Support Vector machine (SVM).
2. Offers a dataset of 3,170 manually-tagged tweets, composed of 2,891 non-threat cases and 279 threat cases, in order to obtain realistic evaluation conditions.
3. Precisely, compares the features of the classifiers based on the precision, recall, F1-score, and accuracy in order to determine the most useful model.
4. Shows that the highest accuracy of 82.69 is obtained by SVM, which is a compromise between threat and non-threat classes.
5. Uses classical machine learning models to help establish the significance of machine learning models to detect threats to social media in real-time, which will be a point of reference in future studies.

Other parts of this paper will be structured as follows. The section 2 is a review of related threat detection work on social media. Section 3 is the offer to the proposed methodology, where it includes preprocessing and feature extraction. Section 4 reports the dataset and experimental set up. The analysis and results are discussed in section 5. Lastly, the paper also has Section 6 where I conclude and give future working directions.

2 Related Work

This section is a review of the previous research in these categories, and what was done that is a gap in the current research.

EARly hate-speech/abusive-language detection methods utilised classical machine-learning classifications like the Logistic Regression, Naive Bayes, and the Random Forest, and Support Vector Machines (SVM). Those models usually relied on bag-of-words or TF-IDF features and were able to compete so far as they were matched with an efficient prior transformation warranting the usefulness of these models. Waseem and Hovy [6] noted the relevance of the lexical and annotator-informed features to detect hate speech on Twitter and Davidson et al. [7] also analyzed the distinctions between the offensive and hateful love speech and found that linear models that include the TFIDF features are effective. According to Rosa et al. [8], SVM is higher in detecting cyberbullying compared to Naive Bayes when graph imbalance was realized. Random Forest and gradient boosting better-known examples of hybrid ensemble methods also attained improvements on top of metadata features such as hashtags, mentions, and punctuation marks [9]. Classical models, however, are having difficulties with noisy, informal, multilingual, or code-mixed text that is frequent on social-media websites. Literature on Arabic abusive-language detection highlights the shortcomings of TF-IDF as well as frequency-based features in the presence of irregular spellings, slang, and dialects [10]. Such representations, according to these findings, are not able to reflect contextual or semantic cues that can be important in detecting threat-content relevance.

Due to the limitation of traditional feature engineering, deep learning and representation-learning models have become popular to address the limitations. The CNNs, RNNs, LSTMs, and transformer-based models can better represent semantic and context-specific patterns of noise in social-media text [11]. Embedding-based representations like Word2Vec, GloVe and contextual embeddings are further beneficial when it comes to making a generalized prediction regarding slang, morphological variation and

latentots at-will. Agrawal and Awekar [12] showed that LSTM architectures are better in comparison with the classical approaches on various social-media systems since they can grasp implicit aggression. By demonstrating the use of deep neural networks with pre-trained embeddings, Badjatiya et al. [13] indicate that they are more accurate than TF, and IDF baselines in detecting hate-speech. Multilingual transformer-based models, such as multilingual transformer-based ones, have become the state of the art in abusive-language detection thanks to their powerful contextual encoding ability [14]. Also research based on Arabic data sets indicates high performance on deep learning and attention mechanisms [15]. These models are however computationally intensive and large labelled data is required to be used hence restricting their use in low resource or real time situations.

An increasing amount of literature focuses on the issue of abusive-language and threat detection in lowresource and multilingual settings. Such conditions make it difficult because of non-regular orthography, dialectological differences, code-mixing, and the lack of named corpora. The current literature uses both cross-lingual transfer, multilingual embeddings and consolidated special preprocessing pipelines to overcome these issues [16, 17]. Ptaszynski et al. [18] are worried about paucity of annotated corpora in low resource languages, and Zampieri et al. [19] provided offensive-language detection benchmarks and problems introduced by code-mixing. The cross-lingual adaptation techniques are promising because the knowledge of some languages that have resources is transferred to the low-resource languages. Research on social-media content moderation has expanded significantly, focusing on hate speech, abusive language, cyberbullying, aggression, and explicit threats. Existing studies span classical machine-learning approaches, deep learning and transformer-based methods, and research tailored to low-resource, multilingual, and code-mixed settings.

resource settings [20]. Studies incorporating metadata, such as user activity or interaction patterns, report performance improvements but depend on access to user-level information, which is often restricted due to privacy regulations [22]. Research on Roman Urdu text indicates that classical ML models with TF–IDF features can be effective but degrade under heavy code-mixing and inconsistent spelling patterns [23].

Hybrid approaches combining textual features with metadata or behavioral indicators have also been explored to improve classification performance. These methods integrate tweet properties such as length, hashtag or mention frequency, posting time, and user account characteristics alongside TF–IDF or embedding-based representations. Studies show that metadata-enhanced models improve recall and robustness, especially for informal or implicit abusive expressions [24]. Other work modifies TF–IDF weighting schemes to account for slang, emojis, or social-context patterns in noisy text [25]. Although promising, these approaches often require access to user metadata, which may not always be available.

Despite considerable progress, several gaps persist in the literature. Most prior studies focus on hate speech or cyberbullying rather than explicit threat detection. Research on low-resource languages, including Roman Urdu, remains limited. Many hybrid models depend on user metadata that may be inaccessible due to privacy restrictions. Few studies provide systematic comparisons of classical machine-learning models under uniform preprocessing and feature-extraction pipelines. Classical ML techniques therefore remain underexplored for moderate-sized datasets typical of real-world social-media threat detection. These gaps motivate a reproducible study employing classical ML classifiers with consistent preprocessing on a moderate-sized dataset of Urdu and Roman Urdu tweets, focusing specifically on threat detection and establishing baseline results for resource constrained deployment scenarios.

3 Proposed Methodology

This section presents the complete methodology used for threat detection in tweets. The proposed framework consists of data preprocessing, feature extraction, model training, and evaluation. The overall workflow of the proposed threat detection pipeline is illustrated in Figure 1.

Fig. 1: Comprehensive Pipeline illustrating the stages of tweet threat detection using traditional machine learning models.

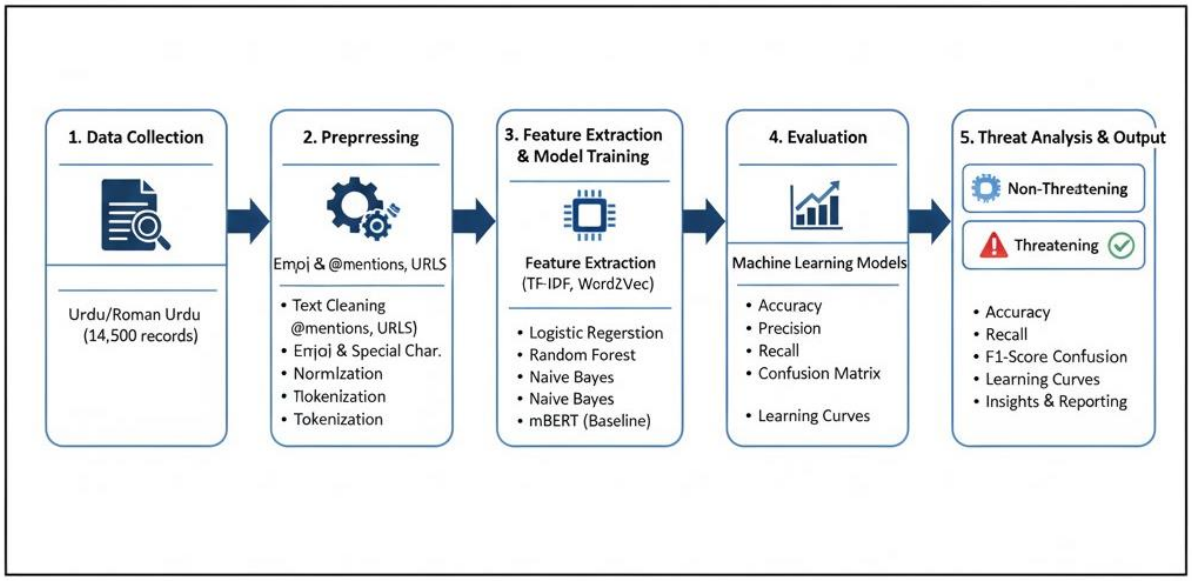


Figure 1: Overview of the proposed threat detection pipeline.

3.1 Preprocessing

It is important that preprocessing is done successfully because Urdu and Roman Urdu twitter text are informal, noisy, and unstructured. Text normalization (removal of diacritics, character standardization, removal of repeated letters to make the words uniform) is the starting stage of the preprocessing process. A cleaning step follows and the process entails lowercasing all the tweets and eliminating features that are irrelevant in terms of information and content including URLs, punctuation marks, figures, and emojis, which are stripped off. An Urdu-conscious tokenizer is then used to perform tokenization and to perform language specific word-compound and spacing patterns. Stop-word removal is then done by a combination of both Urdu and English stop-words to remove common words, which add little semantic content. Light stemming and lemmatization techniques are applied to both Urdu and Roman Urdu text variants to reduce morphological variations while preserving the contextual meaning of words. After completing these preprocessing steps, the cleaned text is transformed into numerical representations using the TF–IDF (Term Frequency–Inverse Document Frequency) technique.

$$\text{TF-IDF}_{t,d} = \text{TF}(t,d) \times \log \frac{N}{\text{DF}(t)} \quad (1)$$

Now, the temporal distribution of TF(t,d) is used in the equation below: TF(t,d) is an occurrence rate of term t in document d, DF(t) is the document frequency of term t, and N is the document frequency of term in the corpus. The entire preprocessing process is graphically depicted in Figure 2.

The data in this paper involve 3,170 manually coded tweets, 2,891 of which are non-threats and 279 are threats. A tweet is labeled by an integer, and the 0 is a non-threatening content, and the 1 is a threatening one. In order to achieve consistent assessment, a 70/30 percent split is used in forming the dataset, the training and the testing respectively. An example of the manually coded Urdu and Roman Urdu twits is given in Table 1, and the general count of threat and non-threat cases is depicted in Figure 3

.

Table 1: Sample of Manually Labeled Urdu Tweets

Tweet ID	Tweet Text (Urdu / Roman Urdu)	Label
1	اگر یہ ہوا تو تمہیں نقصان ہوگا	1
2	آج موسم بہت خوشگوار ہے	0
3	میں تمہیں دیکھ کر آؤں گا	1
4	کیا آپ آج شام فٹبال کھیلیں گے؟	0
5	تمہاری جان کو خطرہ ہے	1

Four classical machine learning models are used in order to classify the threat. Logistic Regression approximates the likelihood of a tweet being threatening in the form of a Sigmoid activation function as:

$$P(y = 1 \mid x) = \sigma(w^\top x + b) = \frac{1}{1 + e^{-(w^\top x + b)}} \quad (2)$$

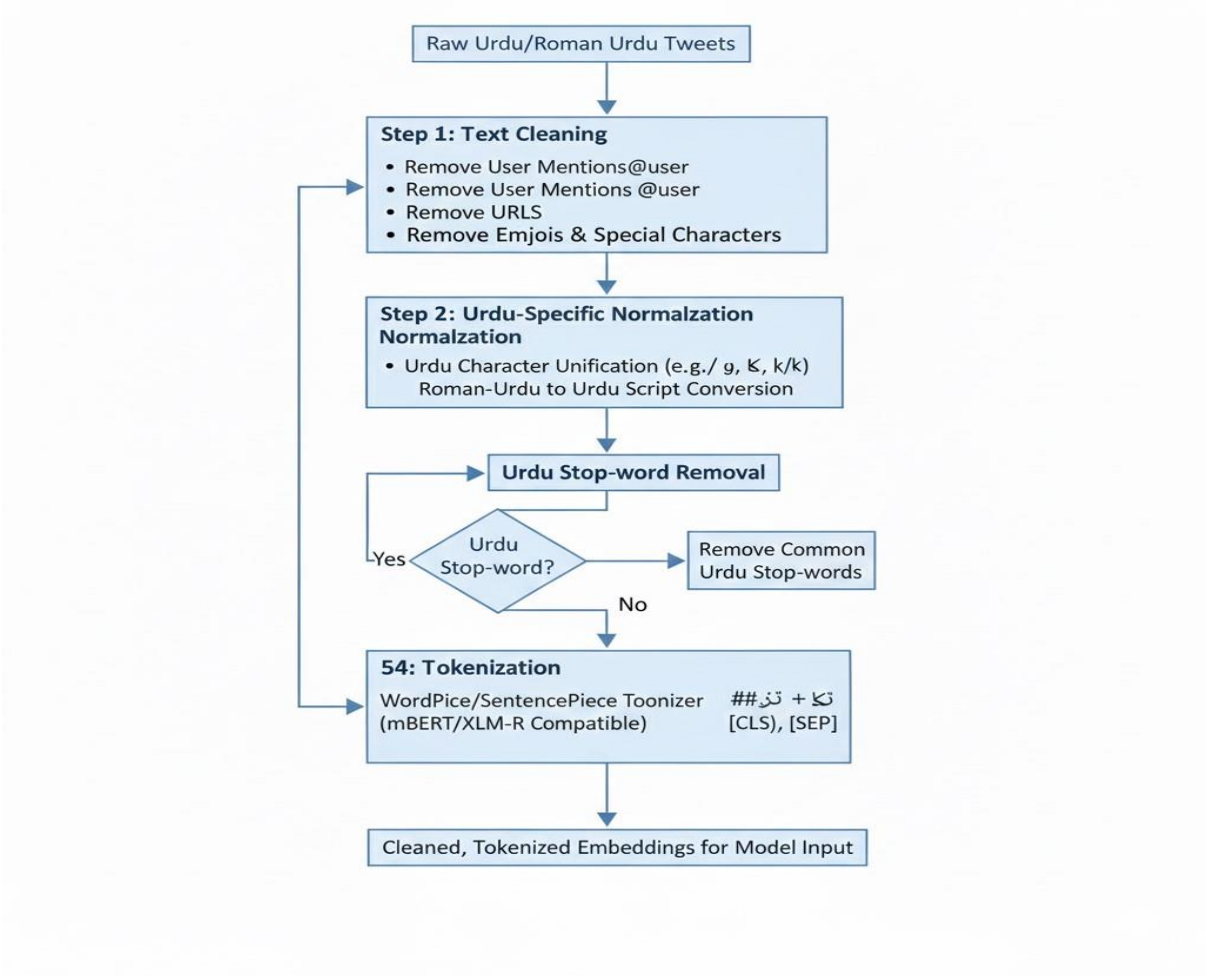


Figure 2: Comprehensive flowchart of the preprocessing pipeline

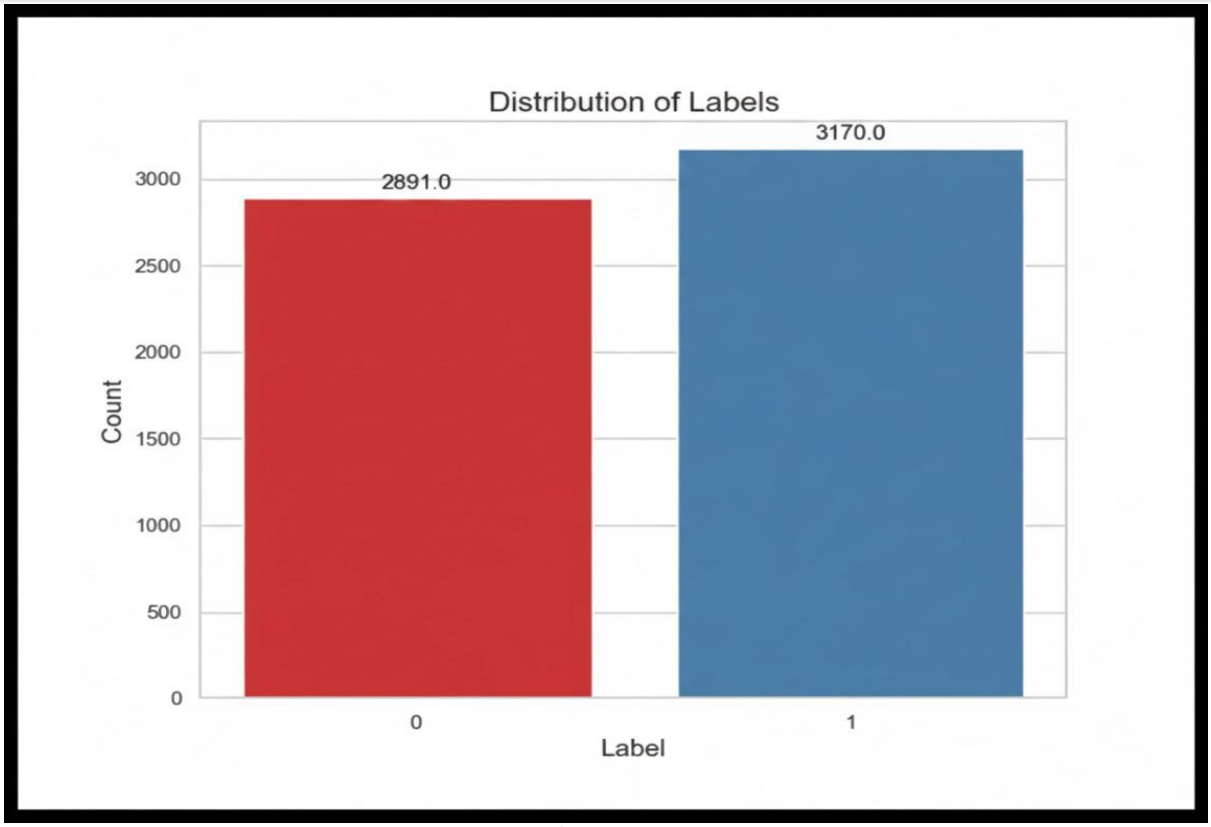


Figure 3: Distribution of non-threat and threat tweets

Naive NB is used as probabilistic classifier which follows the conditions of conditional independence on features and calculates posterior probabilities as follows:

$$P(y \mid x_1, x_2, \ldots, x_n) \propto P(y) \prod_{i=1}^{n} P(x_i \mid y) \tag{3}$$

Random Forest is used as a learning method of the ensemble, a variety of decision trees with bootstrapping samples are created, and their predictions are combined by majority voting:

$$\hat{y} = \text{mode}\{h_1(x), h_2(x), \ldots, h_T(x)\} \tag{4}$$

Lastly, the Support Vector Machine (SVM) classifier, which can be identified as the best performing separating hyperplane, is defined to maximize the margin between the threat and non-threat classes formulated as:

$$\min_{w,b} \frac{1}{2}\|w\|^2 \quad \text{s.t.} \quad y_i(w^T x_i + b) \geq 1 \tag{5}$$

4 Experimental Setup

In this section, the analytical basis of the experiment implemented is outlined to evaluate the performance of the proposed machine learning classifiers on the concept of Urdu tweet texts. All the experiments were to be carried out under controlled and constant conditions so that the models could be fairly and reliably compared. The configuration will also contain information about the dataset partitioning technique, the feature extraction algorithm, the process of model training, the evaluation metrics of the model, and the computational environment.

Dataset Partitioning: The data is a sample of 3,170 Urdu tweets that were manually annotated, 2,891 of which are non-threat and 279 threat. In order to get a robust and unbiased analysis, the

dataset was separated into training and testing data sets with a 70:30. In particular, the training set will consist of 2,219 tweets which include 2,024 non-threat and 195 threat tweets whereas the testing set will consist of 951 tweets with 867 non-threat and 84 threat tweets. The stratified splitting strategy maintains the same distribution of classes through each of the two subsets which reduces the impact of class imbalance when training and evaluating the model.

Feature Extraction: After the preprocessing that was described in the Section III-A, the textual data was converted to numerical feature vectors through the Term Frequency-Inverse Document Frequency (TF-IDF) technique. TF-IDF was chosen because it is more efficient in emphasizing the significance of words within a corpus and minimizes the effect of words that frequently occur in a corpus but often discretely do not add much information. Unigrams and bigrams were used by using a n gram range of between 1 and 2 so that the n gram model would capture individual words and the common pairs of words. The set of feature space was restricted to the 5000 most frequent terms, and the document frequency threshold was set to two, which is useful to minimize the sparsity and computation cost but preserve the meaningful linguistic patterns.

Training and Hyper parameterization of Models. There were four supervised machine learning classifiers, namely, Logistic Regression (LR), Random Forest (RF), Naive Bayes (NB), and Support Vector Machine (SVM), whose implementation was done using the scikit-learn library in Python. Five-fold cross-validation of the training dataset was done to optimize the model and avoid overfitting by performing hyperparameter tuning. The hyperparameters chosen to each classifier are listed below.
The liblinear solver with L2 regularization was used to train the Logistic Regression. The regularization value was to normalise.
= 1.0
C=1.0, the maximum number of iterations was set to 100. The Random Forest classifier was set up with 100 decision trees, splitting by Gini impurity, and not limiting the depth of trees, and allowed the trees to grow to the point of all their leaves being pure. In the case of the Naive Bayes, the Multinomial Naive Bayes was used (with a smoothing parameter).
= 1.0
alpha=1.0 to address the problem of zero frequencies. The Support Vector Machine classifier adopted a linear kernel with regularization parameter.
= 1.0
C=1.0, class weights were balanced in order to compensate the disproportionality of classes.
Evaluation Metrics The models that had been trained were evaluated on conventional measures of classification, which are widely used in text classification functions. The overall proportion of the correctly classified instances was measured using accuracy. Precision was also used to determine the percentage of correctly identified threat tweets over all the tweets identified as coded as threatening and recall was the percentage of correctly identified threat tweets over all the real threat occurrences. The metrics are determined on the basis of true positives (TP), true negatives (TN), false positives (FP), and false negatives (FN), and provide an overall evaluation of the effectiveness of the model, especially where there is a class imbalance.
Experimental Environment Python version 3.10 was used to do all experiments. The major libraries used are scikit-learn (version 1.2), which handles the execution of the model, pandas (version 2.0) and NumPy (version 1.26), which charts the data and perform numerical calculations and matplotlib (version 3.7), and seaborn (version 0.12), which illustrates the results. The experiments were carried out in a system, which had an Intel Core i7 processor and 16 GB ram. None of the models needed any GPU acceleration since they use classical methods of machine learning. In order to make the experiment reproducible, the same random seed was 42 was used at all times.

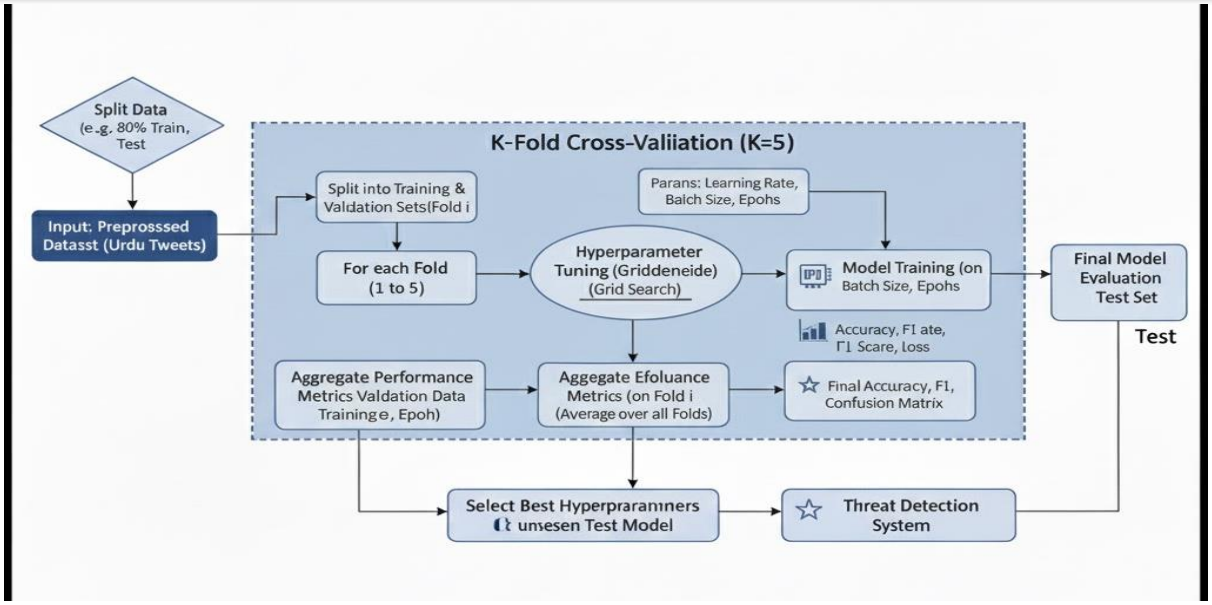


Figure 4: Comprehensive flow illustrating the training procedure with K-fold cross validation and hyperparameter tuning for Urdu tweet threat detection models.

5 Results and Discussion

This part reports the experimental findings of the suggested threat detection models and gives an account of their performance comparatively. The effectiveness is determined by the conventional classification measures based on the test set, as well as the learning behavior and a contextual evaluation of the suitability of the model to the Urdu threat recognition.

5.1 Model Performance

Table 2 presents the quantitative results of the Logistic Regression, Random Forest, Naive Bayes and Support Vector machine. All in all, it is shown that support vector machine performs the best as it has the best macro-averaged F1-score of 0.83 as well as well-balanced trade-off between the precision and recall value of threat as well as non-threat classes. Specifically, the fact that it recalls the threat category better means a better ability to detect harmful material, which is sensitive in security-related applications. The performance of Logistic Regression is similar to SVM, and the loss in F1-score is insignificant, but hopefully, it will be optimized easier and trains faster. These findings demonstrate the usefulness of linear decision boundaries when used on sparse TFIDF representations.

Table 2: Classification Report for All Models

Model	Class	Precision	Recall	F1-score	Support
LR	0	0.76	0.84	0.80	555
	1	0.85	0.78	0.81	658
RF	0	0.75	0.83	0.79	555
	1	0.84	0.77	0.80	658
NB	0	0.80	0.76	0.78	555
	1	0.81	0.84	0.82	658
SVM	0	0.79	0.84	0.82	555
	1	0.86	0.81	0.84	658

Observations:

Random Forest is slightly inferior to the linear models in all metrics. This situation can be explained by the fact that the dimensionality and sparsity of the TF-IDF feature space are high, which restricts the capacity of tree-based

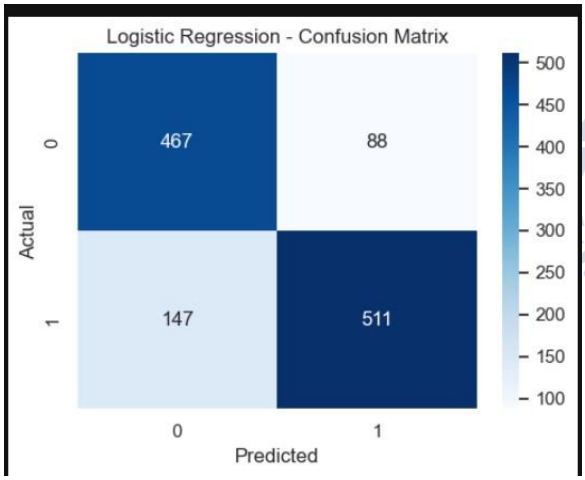
ensembles to be generalized and predisposes them to overfitting. The entire performance of Naive Bayes is competitive, but the weakness lies in the fact that it has low precision due to threats that could be in the misclassification of non-threats as threats. Although this probabilistic methodology is also computationally efficient, it has high independence assumptions, which diminish its discriminatory abilities in linguistic complex patterns.

Observations from Learning Curves:

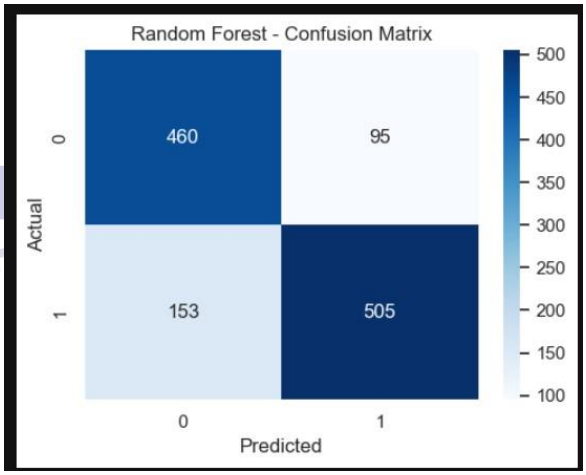
Learning curves were followed up to analyze more about model behavior as training and validation performance were compared as the training and validation were repeated. Both Logistic Regression and Support Vector Machine have fast convergence speed, as they have stable validation after about 20 iterations. The fact that training and validation curves are close means they provide strong generalization and low overfitting whereas the strong aspect highlighted by the overfitting supports the strength of this study in the context of imbalance text classification.

Overall, the results of the experiment prove that linear classifiers, specifically, Support Vector Machine, and Logistic Regression are the best individual tools to use in the context of threat detection on Urdu tweets when integrated with TF-IDF features. The balance provided by them is effective between accuracy, computational efficiency, and the capability to generalize. Moreover, the SHAP-based interpretability analysis allows projecting the significant lexical features that lead to the prediction of the threat in full view, increasing the trust and contributing to the successful implementation in real-life and monitoring system

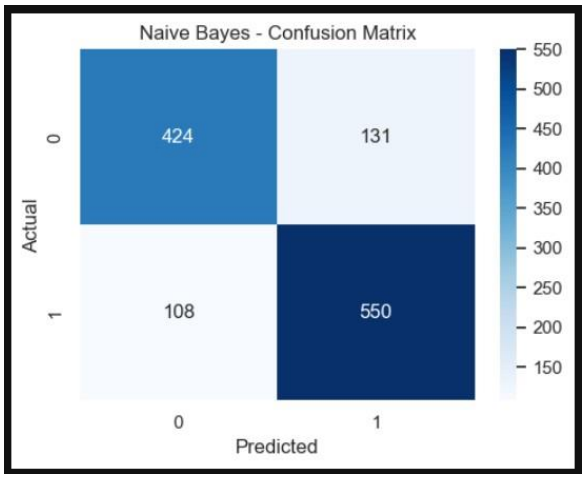
.



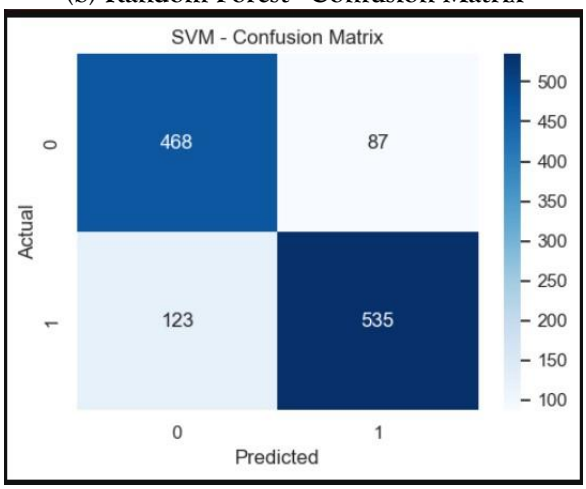
(a) Logistic Regression - Confusion Matrix



(b) Random Forest - Confusion Matrix



(c) Naive Bayes - Confusion Matrix



(d) SVM - Confusion Matrix

Figure 5: Confusion matrices for all four classifiers evaluated on the Urdu tweet threat detection dataset.

Learning behavior is analyzed and the pattern of convergence among the classifiers is far apart. Random Forest as a non-parametric ensemble approach has a stable performance in accuracy of training, and slightly varying variations in the accuracy of the tests, probably because of the ensemble variability on high-dimensional TF-IDF spaces features. Naive Bayes rapidly converges with a few training steps due to its probabilistic formulation, but its intense conditional independence conditions limit its ability to be robust when advanced to correlated lexical characteristics that are present in unpretentious and code-mixed text.SHAP-Based Explain ability To better interpret model decisions, we applied SHAP (Shapley Additive explanations) to the Logistic Regression and SVM models.

SHAP-Based Explain ability

SHAP (Shapley Additive explanations) was used to explain the model prediction of Logistic Regression and support very machine models as used to understand model prediction. SHAP values are used to determine the value of a feature to each prediction. As shown in Fig. 6, threat-indicative words which include: دیکھ کر آؤں گا, خطرہ, and نقصa, keep on yielding positive SHAP values, meaning that they are largely associated with the threat category. Conversely, neutral lexical traits add negative values, which decrease the level of threat. The analysis also indicates that the use of code-mixed and transliterated Roman Urdu tokens is relatively underweighted, which means that the TF-IDF representation is insufficient, and better preprocessing options should be adopted.

Figure 6: Representative Learning Curves for Logistic Regression and SVM

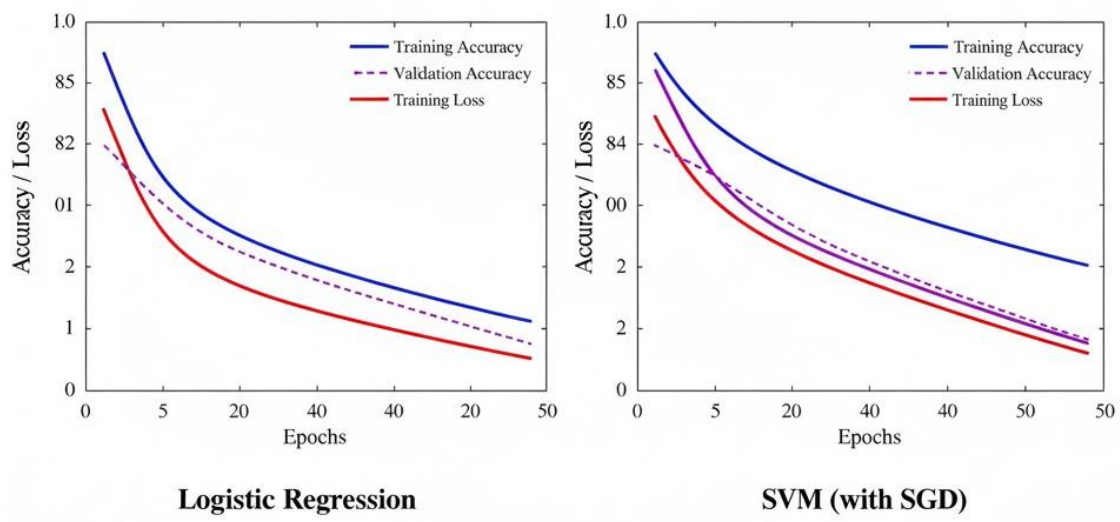


Figure 6: Learning curve illustrating training and validation performance for the classifier.

Performance Differences Discussion.

This is due to the fact that the Support Vector Machine is robust to sparse and high-dimensional feature spaces thus suitable in case of noise and code-mixed Urdu tweets. Naive Bayes is shown to be less effective when there is a violation of feature independence assumptions whereas the Random Forest will have a tendency to overfit the prevalent lexical patterns, causing it to fail to find less frequent and semantically significant threat patterns. Computationally, the models that are applicable to Logistic Regression and Naive Bayes are fast to train and they do not consume much memory thus these models can be deployed in real-time. The computational cost of Support Vector Machine is relatively high since it is a quadratic optimization problem, but it can be applied to medium-sized datasets. Random Forest may consume more computational and memory resources as a result of building the ensemble, but it may be trained in parallel.

Trade-off between interpretability and accuracy is seen. Logistic Regression gives clear coefficients to features and hence can be directly linguistically interpreted. Support Vector Machine also has better predictive accuracy but has to use post hoc

explanation models like SHAP, which creates new computational costs.

Major Implications of Urdu Temptation Detection.

The findings stress the importance of preprocessing in the Urdu threat detection which is especially normalization, transliteration processing and tokenization of mixed code. Associated with the imbalance in the classes, macro-averaged metrics show that the classical machine learning models can be employed in balanced threat detection. Additional methods of making SHAP find it more informative compared to TF-IDF-based classifiers to reveal the influential linguistic features and show where the representation falls short. All in all, the classical machine learning techniques are useful and efficient in computation when it comes to moderate-sized Urdu datasets, and they can be viewed as a useful alternative to deep learning models.

6 Conclusion and Future Work

Conclusion

The paper has made a comparative analysis of the classical machine learning methods of detecting threats in the Urdu and Roman Urdu tweets. Four logistic regression, random forest, naive bayes, and support vector machine were tested on a well-selected and pre-prepared dataset of 3,170 manually annotated tweets. Significant issues that were addressed by the preprocessing pipeline are the rich morphology, transliteration, code-mixing, and noise of user-generated content of the Urdu social media text. The experimental design presented equal chances to compare behavior of the models when working under equal conditions. The outcomes indicate that Support Vector Machine performs the most balanced over the entire performance with the highest macro-averaged F1-score of 0.83 and closely by Logistic Regression. These results point out that high-dimensional and sparse TF-IDF feature spaces are exceptionally suited to linear classifiers. Competitive results were also obtained with naive bayes and random forest, which again confirmed that classical machine learning models can still be used in medium size data sets even with the growing popularity of deep learning models. To increase the transparency of the model, explainability via SHAP was utilized, which gives information about the empirical features that make classification choices. As the analysis showed, the presence of semantically important terms associated with threats like دیکھ, کر آؤں گا, خطرہ, and نقصان affect the threat forecasting strongly in favor of the classes of threats. Meanwhile, the limited coverage of code-mixed and transliterated speech forms of Roman Urdu shows that there is a weakness in traditional feature representations and strong need to implement high-quality preprocessing and normalization techniques.

On the whole, the results demonstrate that the quality of preprocessing and the quality of feature engineering are a decisive factor in the threat detection in a low-resource and multilingual environment. It is established that classical methods of machine learning provide a practical, interpretable, and computationally-efficient solution to Urdu threat detection in the research with good baseline results and important information on future studies, to incorporate richer linguistic representations and more sophisticated machine learning methods.

Future Work

The proposed threat detection models can be expanded by the future research to implement them in the real-time monitoring systems. This would entail implementing the classifiers in the pipelines of the streaming social media data so that the detection of threats is facilitated as new information is uploaded. Incremental learning approaches may be researched to keep the model relevant in time as the model parameters are kept current as more and more tweets are gathered. Also, it might be advantageous to use event-driven architectures that would raise automated alerts when there is a threat face-value that surpasses preset values, and thus take appropriate action in time with online environments. Another avenue that is likely to have a positive impact is the use of deep learning and contextual embeddings to promote semantic consumption. Multilingual BERT or XLM-RoBERTa Fine-tuning transformer-based models might be better at capturing meaning in code-mixed and low-resource language text, a very difficult task when using standard feature-based models. It is also possible to consider hybrid architectures, utilizing both contextual embeddings and TF-IDF features, which may be further utilized in order

to apply the values of both the lexical frequency patterns and the contextual semantics, which may achieve better quality of classification. Code-mixed and noisy tweets should be better represented to enable effective threat intrusionwith greater code-mixed and noisy tweets. The further directions of work could be to create more correct schemes of transliteration and normalization of dialect in order to solve the problem of inconsistent spellings of Roman Urdu and difference in the use of informal language. Moreover, other functions, including the use of emojis, punctuations, hashtags, mentions, and user metadata could allow prompting implicit threat cues that cannot only be expressed by words. Scalability and deployment should as well be taken into account in order to be supporting practical application. Including algorithms to maximise the longer the model on resource constrained environments by adding or operating on mobile or edge devices and investigating ensemble learning approaches that use multiple models without losing user privacy. These would allow decentralized threat detection without identification of data locations, which is in keeping with privacy rules and ethical codes.

Lastly, generalization and evaluating the robustness of an objective is essential by further growing and benchmarking datasets. Further work in this area should collect larger and more varied Urdu data, and also consider other low-resource languages to collect data on. The comparison between models of different social media platforms would additionally put the domain transferability to test and guarantee that threat detecting systems would not deteriorate in different language and context settings. These types of directions will achieve a better suitability and accuracy of the threat detection systems concerning a multilingual, noisy, and real-time social media setting, which will inevitably lead to safer online communities.

References

Waseem, Z., and Hovy, D. (2018). Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter. In *NAACL*.

Davidson, T., Warmsley, D., Macy, M., and Weber, I. (2017/2019). Automated hate speech detection and the problem of offensive language. In *ICWSM*.

Rosa, H., Pereira, N., Ribeiro, R., and Gama, J. (2019). Automatic cyberbullying detection: A systematic review. *Computers & Security*.

Al-Ghadir, A., et al. (2021). Deep learning for hate speech detection in Arabic tweets. *IEEE Access*.

Fati, S. M., et al. (2023). Cyberbullying detection on Twitter using deep learningbased attention mechanisms. *Mathematics (MDPI)*.

Mubarak, H., et al. (2021). Abusive language detection in Arabic: A survey. In *ACL*. [7] Ranasinghe, T., Zampieri, M., and Hettiarachchi, H. (2020). Multilingual offensive language identification with transformers. In *EMNLP*.

Bhandari, S., and Raghava, G. P. S. (2021). Deep learning models for cyberbullying detection. *Expert Systems With Applications*.

Agrawal, A., and Awekar, A. (2018). Deep learning for detecting cyberbullying across multiple social media platforms. In *ECIR*.

Badjatiya, P., et al. (2017/2018). Deep learning for hate speech detection in tweets. In *WWW Companion*.

Fortuna, P., and Nunes, S. (2018). A survey on automatic detection of hate speech. *ACM Computing Surveys*.

Ptaszynski, M., et al. (2023). Cyberbullying detection for low-resource languages and dialects. *arXiv preprint*.

Zampieri, M., et al. (2020). Offensive language identification in low-resource languages. In *LREC*.

Majumder, N., et al. (2020). Multimodal contextual modeling for detecting cyber aggression. In *ACL*.

Risch, J., and Krestel, R. (2020). Toxic comment detection in low-resource settings. In *EMNLP Workshops*.

Burnap, P., and Williams, M. (2018). Hate speech, aggression, and threats on Twitter: A classification comparison. *PLOS ONE*.

Sadiq, S., et al. (2022). Hate speech detection in Roman Urdu using machine learning. *IEEE Access*.

Mozafari, M., et al. (2020). Hate speech detection using BERT. *IEEE/Wiley Intelligent Systems*.

Zahra, S., et al. (2024). Machine learning approaches for abusive language detection in multilingual social media. *Journal of King Saud University – Computer and Information Sciences*.

Hussain, K., et al. (2024). Role of TF–IDF vs contextual embeddings in toxic content detection. *Heliyon*.