Journal of Computer Applications and Information Technology
Vol. 2, Issue. 3, Sep 2026, pp. 12~23
ISSN: 3107-6424, DOI: 10.32595/jcait/v2i3.2026.38

An Explainable and Cross-Lingual Machine Learning Framework for Offensive Content Detection in Tamil–English Code-Mixed Social Media Text

Dr. Hishamuddin Bin M. Salleh[1], Balqis Nabilah Binti Ramanizan[2]

[1]School of AI Computing and Multimedia, Lincoln University College, Malaysia.
E-mail: hishamuddin@lincoln.edu.my
[2]Lecturer, School of AI Computing and Multimedia, Lincoln University College, Malaysia.
E-mail: balqisnabilah@lincoln.edu.my

Article Info

Article History:

Received Jul 28, 2026
Revised Aug 21, 2026
Accepted Sep 05, 2026

Keywords:

Code-mixed NLP,
Offensive Language Detection,
Tamil–English,
Explainable AI,
Transformer Models,
Cross-Lingual Generalization

ABSTRACT

Social media platforms are rife with multilingual and code-mixed posts, making automatic offensive-content detection difficult, especially for low-resource languages. Tamil-English code-mixed posts present additional challenges, including transliteration, non-canonical spellings, lexical switches, and contextual offensiveness, among other issues. This paper presents an explainable machine-learning framework for detecting offensive content in Tamil-English code-mixed social media posts. This framework relies on the publicly available DravidianCodeMix dataset and employs Logistic Regression, Support Vector Machine (SVM), Random Forest, and the transformer-based XLM-RoBERTa model. Word-level and character-level Term Frequency-Inverse Document Frequency (TF-IDF) representations are evaluated for classical machine-learning models, while XLM-RoBERTa is fine-tuned for classification. A cross-lingual generalization experiment is conducted to evaluate the transfer of models trained on Tamil-English code-mixed posts to Malayalam-English code-mixed posts. Feature-importance analysis is conducted to understand which features have the most significant impact on detection. XLM-RoBERTa achieves the highest accuracy (83.41%) and macro-F1 (77.87%), while character-level SVM attains 82.65% accuracy and 74.09% macro-F1. On the cross-lingual experiment, the macro-F1 scores are 54.24% and 44.94% for XLM-RoBERTa and character-level SVM, respectively. The proposed framework offers a reproducible and interpretable baseline for code-mixed offensive-content detection.

Corresponding Author:

Dr. Hishamuddin Bin M. Salleh,
School of AI Computing and Multimedia,
Lincoln University College, Malaysia.
E-mail: hishamuddin@lincoln.edu.my

1. INTRODUCTION

The rapid growth of social media in multilingual societies has led to an abundance of code-mixed communication, where users communicate in a mixture of two or more languages. In India,

Tamil-English code-mixing is rampant in places such as YouTube, where users comment in Romanized Tamil, native Tamil script, English, or a combination of the three [1], [2]. This poses a problem for automated content-moderation systems, typically trained on datasets of a single language (e.g., English [3]).

Offensive content (e.g., insults, hate-speech, targeted harassment) is a pervasive problem on social media [4], [5], where detecting it in code-mixed content is significantly more challenging than in single-language content due to transliteration and informal spelling variation, and frequent language switches within a single sentence [1], [6]. Detection systems typically fail to generalize to code-mixed content, motivating investigation into this space for Dravidian code-mixed languages [7].

This paper presents an explainable machine-learning framework for detecting offensive content in Tamil-English code-mixed social-media text. The paper compares classical machine learning with word- and character-level TF-IDF features to a transformer-based, multilingual model XLM-R [8] in the task of offensive-content detection, and further investigates cross-lingual generalization by training and testing the model on Malayalam-English code-mixed data. The proposed framework compares classical and transformer based approaches for offensive-content detection in Tamil-English code-mixed text, while performing cross-lingual generalization on Tamil-English and Malayalam-English language pairs. In addition, the importance of each feature for the task of offensive-content prediction is analyzed to improve the interpretability of the model for human moderators.

2. LITERATURE REVIEW

Research on the detection of offensive-language in Dravidian code-mixed languages has witnessed an upsurge in the recent past with the creation of datasets, classical machine learning approaches, and the advent of multilingual transformers. Chakravarthi et al. [1], [9] released the DravidianCodeMix dataset for detecting offensive-language in Tamil-English and Malayalam-English code-mixed texts. Ravikiran and Annamalai [6] created the DOSA dataset comprising of token level offensive-span annotations. In addition, other works [2], [3] have benchmarked classical classifiers such as Logistic Regression, SVM, Naive Bayes, and Random Forest against deep learning and transformer based approaches for detecting offensive-language. These works highlight the efficacy of character level and sub-word representations for transliterated and noisy text. Multilingual transformers [8] have also been leveraged to detect offensive-language with sub-word representations being useful for code-switched text. The shared tasks at HASOC and DravidianLangTech [7] have also helped standardize the evaluation of offensive-language detection approaches.

That said, the cross-lingual generalization capabilities and interpretability of these models are not well studied. Prior works [2], [3] focus on the performance of these models on the same language that they were trained on. There is a lack of analysis on whether the same model can be used to detect offensive-language in any code-mixed languages. More importantly, there has been no analysis on what linguistic features contribute to what extent to the predictions of such models. We conduct a cross-lingual study of code-switched offensive language detection with an emphasis on feature importance in a unified framework.

3. PROPOSED METHODOLOGY

The proposed methodology consists of a pipeline involving five main steps, namely: (1) data collection and cleaning, (2) text pre-processing and normalization, (3) feature extraction, (4) model training/classification, and (5) explainability and cross-lingual evaluation. Figure 1 represents the framework of the proposed methodology.

3.1. Proposed System Architecture

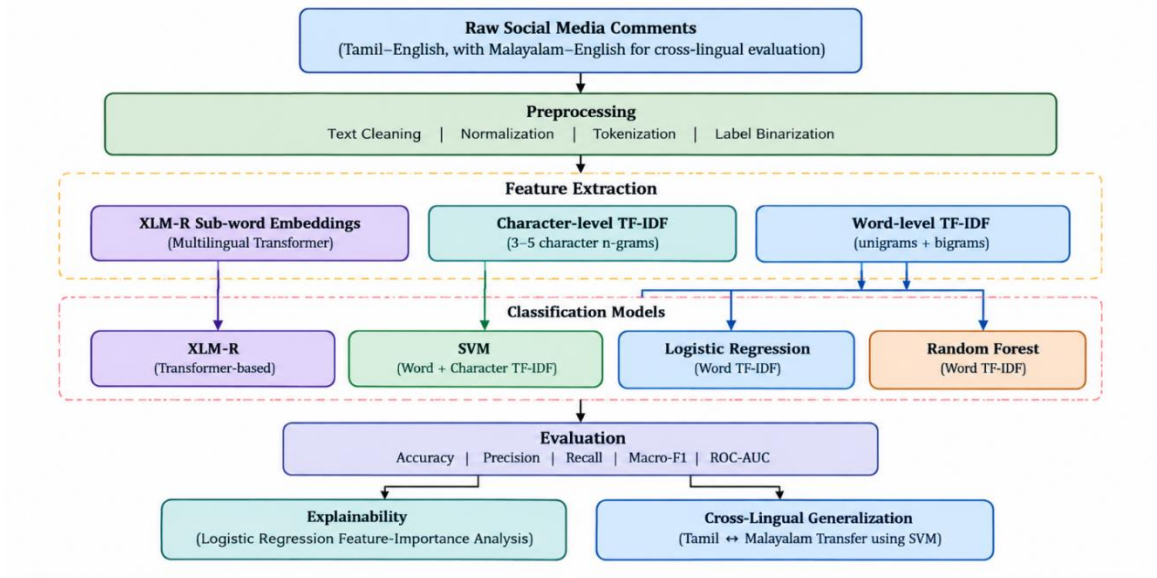


Figure 1. Proposed Architecture Block Diagram

The proposed architecture follows a sequential pipeline for Tamil–English offensive-content detection. Raw social-media comments were preprocessed by cleaning, normalizing, tokenizing, and label binarization. The processed text was represented with word-level TF-IDF, character-level TF-IDF, and XLM-R sub-word embeddings. The representations were classified by Logistic Regression, SVM, Random Forest, and XLM-R classifiers. The models were evaluated by accuracy, precision, recall, macro-F1, and ROC-AUC. Finally, Logistic Regression feature-importance analyzed the most contributing features, while Tamil–Malayalam cross-lingual transfer by SVM validated the framework's generalization.

3.2. Working Principle

The proposed system processes each social-media comment through a sequential text-classification pipeline. Let x represent an input comment. First, preprocessing produces a normalized representation $P(x)$ by removing irrelevant elements while preserving Tamil characters and code-mixed linguistic information. The processed text is then converted into numerical representations $F(x)$ using word-level TF-IDF, character-level TF-IDF, or XLM-R sub-word representations. The selected classifier C processes these features and generates a binary prediction $\hat{y}$, where $\hat{y} \in \{\text{Offensive}, \text{Non - Offensive}\}$. For the classical models, feature coefficients are further examined to identify influential linguistic patterns. The complete operation can therefore be represented as:

$$x \rightarrow P(x) \rightarrow F(x) \rightarrow C\big(F(x)\big) \rightarrow \hat{y} \quad (1)$$

Model performance is evaluated using accuracy, precision, recall, macro-F1, and ROC-AUC. Macro-F1 is given particular importance because the dataset contains an unequal distribution

between offensive and non-offensive classes. The same pipeline is additionally evaluated across Tamil–English and Malayalam–English data to examine cross-lingual generalization.

3.3. Text Preprocessing and Normalization

The preprocessing step is aimed at reducing the amount of irrelevant information that may appear in social-media comments. The comments are full of URLs, mentions, punctuation, repeated characters, and other artifacts. It is crucial to remove them to lower the dimensionalities of the models and increase the relevance of features extracted from the text.

At the same time, the preprocessing step should not overly normalize the data. Specifically, in the case of code-switching texts, it is essential to maintain language-specific characters and transliterations, which often carry critical information. Thus, the preprocessing step focuses on removing irrelevant characters and leaves Tamil Unicode and English textual information intact.

The resulting comments are then tokenized for word representation and converted into character sequences for character-based TF-IDF representation. Finally, in the case of XLM-R, the normalized text is input to the multilingual tokenizer, which outputs sub-word tokens for representation in the classifier.

3.4. Word-Level TF-IDF Representation

The word-level representation describes a document according to the importance of individual terms. TF-IDF assigns greater importance to terms that occur frequently in a particular document but occur less frequently across the complete corpus. This representation is useful when offensive expressions appear consistently as complete words or tokens. Logistic Regression, SVM, and Random Forest can directly operate on the resulting sparse feature matrix. In the present implementation, the dimensionality is restricted to 10,000 features to maintain a practical balance between vocabulary coverage and computational complexity. However, word-level representation may suffer when users intentionally modify spellings or use transliterated variants. Two semantically equivalent expressions can become separate features if their spelling differs substantially. This limitation motivates the inclusion of character-level TF-IDF.

3.5. Character-Level TF-IDF Representation

Character-level TF-IDF utilizes character sequences as a fundamental unit for representing text, as opposed to using whole words. This implementation is especially suitable for encoding social-media texts, as users tend to create many spelling variants, transliterations, abbreviations, and other linguistic phenomena. Character n-grams can encapsulate partially overlapping patterns of related words' spellings. As a result, two different words can have character sequences that are partially overlapping, allowing increased robustness to inconsistent transliterations and other noise. The character-level TF-IDF encoding is primarily evaluated on the SVM classifier in the proposed experiments. The comparison of results of word-level and character-level encoding methods is indicative of sub-word level patterns being more informative than any word's lexical tokenization for the given classification task.

3.6. XLM-R Based Classification

The transformer-based component uses XLM-R to secure multilingual contextual representations. Unlike the sparse TF-IDF ones, the transformer processes token sequences while accounting for contextual relationships between the words. The model is further fine-tuned on the Tamil-English training data to perform a binary offensive-content classification. Specifically, during the fine-tuning stage, its parameters are updated to minimize the prediction error for the given labels. As a result, for each test comment, the model outputs a prediction. Finally, XLM-R's

utilization also allows for a meaningful comparison to the classical approaches. While TF-IDF models mainly rely on either lexical or character patterns, the XLM-R can leverage rich contextual sub-word representations. Thus, the comparison helps establish whether the contextual representations benefit the code-mixed task.

4. DATASET DESCRIPTION

Table 1. Dataset distribution across splits and languages

Dataset	Total	Offensive	Non-Offensive
Tamil–English (train)	26,947	6,608	20,339
Tamil–English (test)	6,737	1,652	5,085
Malayalam–English (cross-lingual test)	18,403	706	17,697

Table 1 summarizes the data splits used in this study. After filtering out non-Tamil comments and aggregating offensive sub-categories, the resulting Tamil–English dataset comprises 33,684 comments, which were split 80/20 between train and test sets, stratified by class. The Malayalam–English dataset (18,403 comments) comes from the same DravidianCodeMix corpus family [9] and is used in the cross-lingual generalization experiment of Section 6.3.

4.1. Data Preparation

The original dataset contains code-mixed comments with different language categories and offensive-language labels. For the current experiment, the Tamil – English part was selected for the primary classification task. The comments that did not meet the required Tamil-language criterion were excluded from the main Tamil – English experiment.

The offensive subcategories were aggregated into the binary offensive class, while the remaining comments were treated as non-offensive. This binary formulation simplifies the classification task and allows a consistent comparison of Logistic Regression, SVM, Random Forest, and XLM-R. After preprocessing and filtering, the 33,684 Tamil – English comments remained. The data were divided into training and test subsets using an 80/20 stratified split. Stratification maintains the relative distribution of offensive and non-offensive examples in both subsets and reduces the chance that the evaluation will be unduly influenced by an unusual class distribution.

4.2. Class Imbalance Consideration

The dataset contains a significantly higher number of non-offensive comments than offensive ones. This class imbalance can lead to optimism about the performance of our classifier (i.e., a classifier can predict all comments as non-offensive and achieve decent accuracy on the test set).

To address this issue, classical machine-learning models were trained using class weights, a standard cost-sensitive strategy for learning from imbalanced data [10]. Class weights allow us to increase the contribution of the minority class to the loss function without generating extra samples via up-sampling.

Besides, we used macro-averaged precision, recall, and F1-score to assess the performance of our classifier. Macro-averaging is more informative in case of imbalanced classes (see the discussion above). Using ROC-AUC is also helpful in this scenario as it measures the ability of a

classifier to distinguish between classes: AUC close to 1 means that the classifier almost always ranks a positive example higher than a negative one.

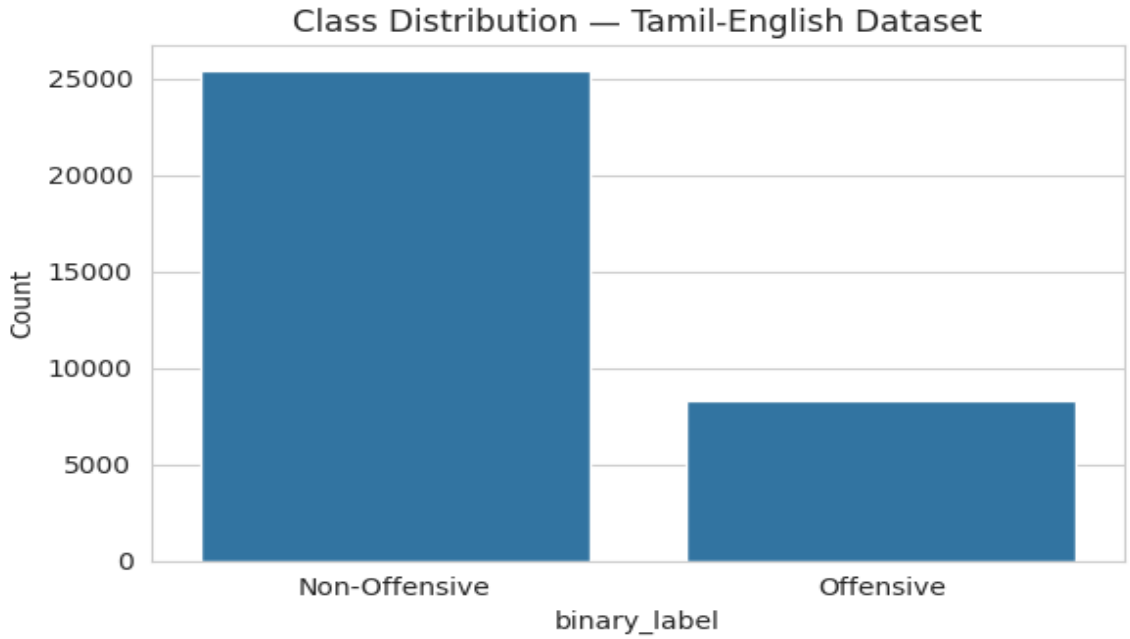


Figure 2. Class distribution of the Tamil–English dataset (75.5% / 24.5% imbalance)

Figure 2 shows that the distribution of comments in the Tamil-English dataset is imbalanced. In this case, there are approximately 25,000 Non-Offensive comments and 8,200 Offensive comments. For this reason, using macro-F1, precision, and recall alongside accuracy can provide a more reliable performance metric.

5. EXPERIMENTAL SETUP

All classical models were trained using Scikit-learn [11] with the parameter class_weight = 'balanced', given the target class distribution imbalance 75.5%/24.5%. TF-IDF features were reduced to 10,000 dimensions. The XLM-R model was fine-tuned using the Transformers library from Hugging Face [12] on a T4 GPU with batch_size = 16, weight_decay = 0.01, and epochs = 3. All models were evaluated on the same test set of 6737 comments using Accuracy, Macro-Precision, Macro-Recall, Macro-F1, and ROC-AUC metrics for comparison.

5.1. Experimental Comparison Strategy

The setup of the experiment was aimed at obtaining a fair comparison between the traditional sparse–text representations and multilingual contextual representations. The logistic regression, SVM, and random forests classifiers were evaluated using the TF-IDF–based features, while the XLM-R model was fine-tuned on the textual input directly.

The same Tamil-English test set was used for all the primary model evaluation. In such a way, the differences in the reported results can be attributed mainly to the learning paradigm and the type of representation used.

Two types of TF-IDF representations were considered for classical machine learning. The word-level TF-IDF aims to capture the whole words, while the character-level TF-IDF was used to account for smaller discriminative textual patterns. The comparison of such approaches is

especially relevant for code-mixed social-media texts, as the spelling variations and transliterations might be present, thus rendering the word boundaries detection ambiguous.

5.2. Evaluation Metrics

Accuracy represents the ratio that shows how many test examples were classified correctly. The metric can be misleading for imbalanced data sets because the majority class can swamp out the minority class. Precision is the ratio of positive instances (offensive comments) that are correctly identified as positive, meaning that high precision means fewer false positives. On the other hand, recall is the ratio of positive instances that are correctly identified as positive by the classifier. For content moderation, the recall is valuable because it is critical not to miss any offensive comments. Finally, the F1-score is the harmonic mean of precision and recall, but Macro-F1 is utilized for calculating the F1 metric separately for each class and averaging the results to avoid overemphasizing the majority class. In turn, the ROC-AUC score is a measure of how well a classifier can distinguish between positive and negative instances, so the higher the score, the better the classifier. Overall, considering multiple evaluation metrics allows a more objective assessment of offensive comment detection performance.

6. RESULTS AND DISCUSSION

6.1. Model Performance Comparison

Table 2. Main performance comparison across all five models on the Tamil–English test set

Model	Accuracy (%)	Precision (%)	Recall (%)	Macro-F1 (%)	ROC-AUC
Logistic Regression (Word TF-IDF)	78.64	72.85	77.74	74.23	0.859
SVM (Word TF-IDF)	81.16	75.68	68.81	70.94	0.847
SVM (Character TF-IDF)	82.65	77.68	72.08	74.09	0.876
Random Forest (Word TF-IDF)	80.53	74.01	69.51	71.13	0.838
XLM-R (Transformer)	83.41	77.53	78.24	77.87	0.888

Table 2 reports the results of all the considered models on the Tamil-English test set. Notably, XLM-R achieves the best performance across all the metrics with an accuracy of 83.41%, and a macro-F1 score of 77.87%.

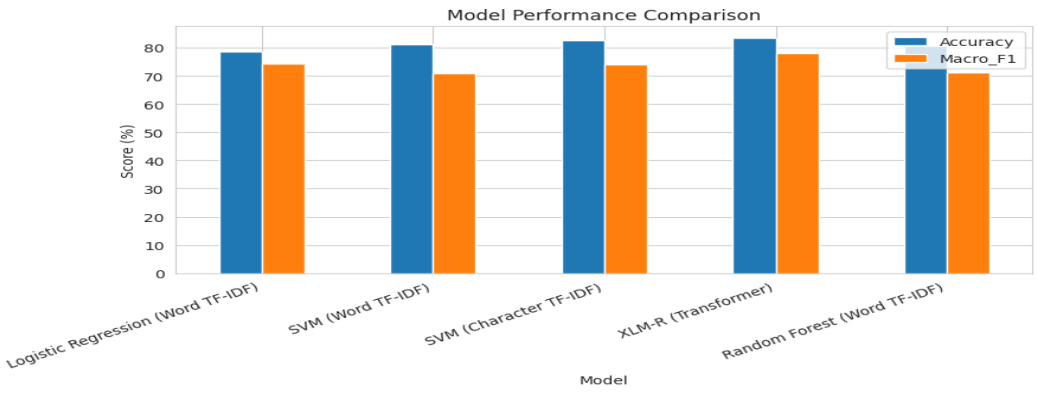


Figure 3. Accuracy and Macro-F1 comparison across all five models

Figure 3 compares the accuracy and macro-F1 of all five models. An interesting observation is that SVM (Word TF-IDF) has the second-highest accuracy (81.16%) but the worst macro-F1 (70.94%) of all models. This emphasizes that accuracy can be a misleading metric in the case of imbalanced classes.

6.2. Confusion Matrix Analysis

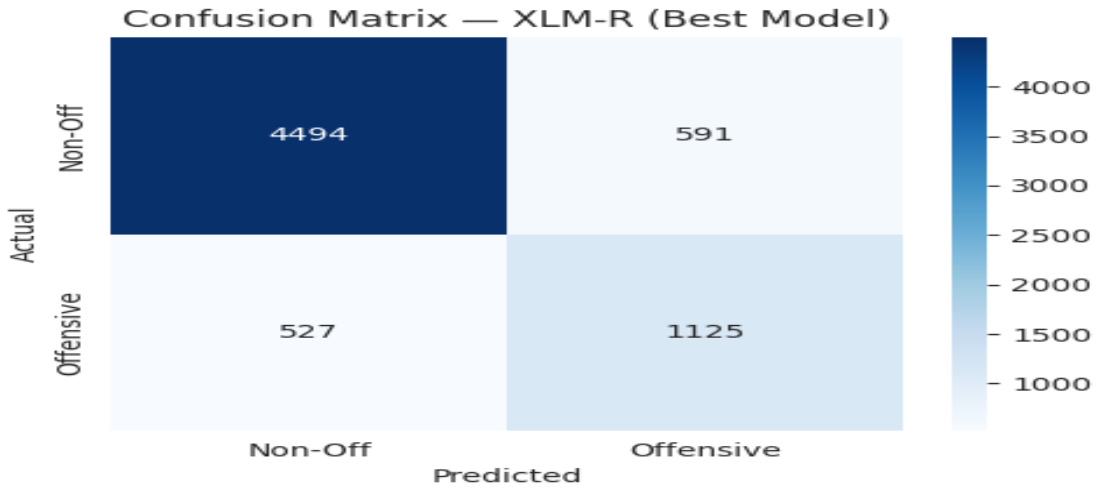


Figure 4. Confusion matrix for XLM-R on the Tamil–English test set

Figure 4 represents the confusion matrix for the best model, XLM-R, on the Tamil-English test set. Thus, given 1,652 offensive comments, the algorithm correctly recognized 1,125 of them or 68.1%, predicting 527 comments as non-offensive. At the same time, 5,085 non-offensive comments were supposed to be 4,494 (88.4%) correctly identified and 591 false positive. Therefore, it could be concluded that XLM-R significantly outperforms the classical models; however, there is still room for improvement, especially with offensive content recall to reduce false negatives, which could be disastrous in practice.

6.3. Cross-Lingual Generalization

To evaluate whether offensive-language patterns are generalizable across Dravidian code-mixed languages, we trained a character-level SVM on the full Tamil-English dataset and tested it on the Malayalam-English subset, and vice versa. The results are shown in Table 3.

Table 3. Cross-lingual generalization performance between Tamil–English and Malayalam–English

Train → Test	Accuracy (%)	Macro-Precision (%)	Macro-Recall (%)	Macro-F1 (%)
Tamil → Malayalam	83.61	54.21	65.37	54.24
Malayalam → Tamil	75.80	76.38	50.86	44.94

Both directions show a significant macro-F1 drop as compared to within-language testing (77.87% for XLM-R, 74.09% for the best classical model). This decrease can be explained by two factors: the differing offensive-class base rates between the two data sets (24.5% for Tamil–English versus only 3.8% for Malayalam–English) and real lexical differences in offensive vocabulary between the two languages.

6.4. Explainability and Feature-Importance Analysis

Explainability is an important component of the proposed framework because offensive-content detection systems can influence content-moderation decisions. It is useful to understand

what textual patterns contributed to the classification in addition to whether a comment is offensive, a motivation shared by model-agnostic explanation methods such as LIME [13].

For the classical Logistic Regression model, feature coefficients can be examined to identify influential TF-IDF features. Positive coefficients indicate features that contribute toward the offensive class while negative coefficients indicate stronger association with the non-offensive class. This provides a direct mechanism for examining the relationship between textual patterns and classification decisions.

In summary, feature-importance analysis can reveal domain-specific criticism vocabulary, informal address terms, and recurring linguistic structures. Such observations can be useful for human moderators because they provide a qualitative interpretation of the model's decisions rather than presenting the prediction as an unexplained output.

The analysis also provides a mechanism for identifying potential weaknesses. If a model depends excessively on a small number of words, it may be vulnerable to spelling changes or deliberate obfuscation. Conversely, distributed importance across character patterns may indicate greater robustness to lexical variation. The explainability component should therefore be interpreted as a complementary analysis rather than as proof that every prediction is linguistically correct.

6.5. Computational and Practical Considerations

The investigated models demonstrate different practical trade-offs. Classical TF-IDF based classifiers require only simple feature extracting and provide an easily interpretable outcome such as coefficients or features weights. It makes them appealing for moderation practice or other light-weight applications where computational resources are a primary constraint. XLM-R, on the other hand, demonstrates the best performance but requires more involved model fine-tuning and significant amount of computational resources to process. The model was trained using T4 GPU with batch size of 16 and 3 epochs. It shows that it is possible to use such heavy models for the given task, but the resources required are noticeably higher than in the case of classical classifiers.

For a real-world application, these trade-offs might be more important than overall performance gains. For instance, a server-based moderation application with enough computational resources may prefer XLM-R. At the same time, a lighter-weight solution such as character-based SVM may be considered for less resource-heavy applications.

Finally, note that the considerations are not exhaustive. The goal of the analysis is not to identify a single best model but rather to understand how to design an effective offensive content detection framework. In particular, a reasonable choice should consider predictive performance, general behavior across different classes, computational footprint and interpretability where applicable.

6.6. Error Analysis and Limitations

Despite the promising results, several limitations remain. The first limitation is the binary formulation of offensive-content classification task. Aggregating multiple offensive categories into single class simplified the problem but at the cost of ignoring distinction between different types of offensive behaviors. A second limitation comes from the inherently variable nature of code-mixed social-media text. Users may deliberately alter offensive words with spelling mistakes, extra repeated characters, abbreviations, or transliteration. These variations might not always be captured by the learned representations.

The confusion matrix further reveals that offensive comments can still slip through the best model's detection. Thus, improving the recall of the offensive class also remains an important challenge. Future works can explore threshold optimization, richer contextual representations, and additional training data to tackle this issue. Another limitation comes from the cross-lingual transfer. The drop in macro-F1 implies that models trained on one code-mixed language pair cannot be automatically assumed to generalize well to another. The language-specific vocabulary and class distribution should be taken into account when deploying such systems. Finally, the feature-importance analysis is primarily available for interpretable classical models. Predictions of transformer are more challenging to interpret, suggesting the need of additional explainability techniques for future works.

7. CONCLUSION

This paper describes an explainable machine-learning framework for the task of offensive-content detection in a code-mixed setting. We compare classical TF-IDF based classifiers with a fine-tuned XLM-R transformer model. The latter achieves the best overall performance (83.41% accuracy, 77.87% macro-F1). We observe that character level TF-IDF features consistently improve upon word level features for classical models. A cross-lingual generalization experiment highlights that offensive-language detection models are not easily transferable between code-mixed Tamil-English and Malayalam-English, with significant performance degradation observed on either target language. Finally, we analyze feature importance and show that model predictions are consistent with linguistically informed expectations, including the presence of domain specific criticism vocabulary and informal address terms.

7.1. Implications of the Study

The results of the experiments carry several implications for code-mixed offensive-content detection. First, character-level representations remain competitive with respect to state-of-the-art transformer-based multilingual models, which is vital in social-media contexts due to the prevalence of spelling variation and transliteration. Second, the results emphasize the need to report performance across multiple metrics, as the difference between accuracy and macro-F1 for some models demonstrates that a classifier can be competitive in one metric while underperforming on a second. Third, the cross-lingual experiment highlights the necessity to explicitly evaluate language transfer, rather than assuming positive transfer across Dravidian languages. Explainability provides another benefit for offensive-language classifiers, with the potential to aid both human moderators and model developers. Identifying influential linguistic features can highlight patterns that merit further investigation and provide a mechanism to evaluate classifier predictions in context.

7.2. Future Enhancement Directions

Future work could extend the proposed framework in several directions. First, a larger and more diverse collection of code-mixed comments could be used to better represent different spelling conventions, transliteration schemes, and dialects. Additionally, more language pairs could be added for a broader assessment of cross-lingual generalization. A multilingual training strategy could also be explored to see if jointly training on multiple Dravidian code-mixed languages improves transfer performance. Domain adaptation can also be explored if the source and target datasets have significantly different class distributions. Another future direction could be to perform fine-grained offensive-content classification rather than simply predicting offensive or non-offensive comments. Future models could distinguish between insults, harassment, hate-related content, and other types of offensive communication. Explainability could also be expanded

beyond coefficient-based feature importance, for example through local model-agnostic explanations [13].

REFERENCES

[1] Chakravarthi, B. R., Muralidaran, V., Priyadharshini, R., & McCrae, J. P. (2020). Corpus creation for sentiment analysis in code-mixed Tamil-English text. In *Proceedings of the 1st Joint Workshop on Spoken Language Technologies for Under-resourced languages (SLTU) and Collaboration and Computing for Under-Resourced Languages (CCURL)* (pp. 202-210). European Language Resources Association.

[2] Thavareesan, S., & Mahesan, S. (2019). Sentiment analysis in Tamil texts: A study on machine learning techniques and feature representation. In *2019 14th Conference on Industrial and Information Systems (ICIIS)* (pp. 320-325). IEEE. https://doi.org/10.1109/ICIIS47346.2019.9063341

[3] Priyadharshini, R., Chakravarthi, B. R., Vegupatti, M., & McCrae, J. P. (2020). Named entity recognition for code-mixed Indian corpus using meta embedding. In *2020 6th International Conference on Advanced Computing and Communication Systems (ICACCS)* (pp. 68-72). IEEE. https://doi.org/10.1109/ICACCS48705.2020.9074379

[4] Schmidt, A., & Wiegand, M. (2017). A survey on hate speech detection using natural language processing. In *Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media* (pp. 1-10). Association for Computational Linguistics. https://doi.org/10.18653/v1/W17-1101

[5] Zampieri, M., Malmasi, S., Nakov, P., Rosenthal, S., Farra, N., & Kumar, R. (2019). Predicting the type and target of offensive posts in social media. In *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers)* (pp. 1415-1420). Association for Computational Linguistics. https://doi.org/10.18653/v1/N19-1144

[6] Ravikiran, M., & Annamalai, S. (2021). DOSA: Dravidian code-mixed offensive span identification dataset. In *Proceedings of the First Workshop on Speech and Language Technologies for Dravidian Languages* (pp. 10-17). Association for Computational Linguistics.

[7] Chakravarthi, B. R., Kumaresan, P. K., Sakuntharaj, R., Anand Kumar, M., Thavareesan, S., Premjith, B., Sreelakshmi, K., Subalalitha, S. C., McCrae, J. P., & Mandl, T. (2021). Overview of the HASOC-DravidianCodeMix shared task on offensive language detection in Tamil and Malayalam. In *Working Notes of FIRE 2021 - Forum for Information Retrieval Evaluation* (CEUR Workshop Proceedings, Vol. 3159, pp. 589-602). CEUR-WS.org.

[8] Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2020). Unsupervised cross-lingual representation learning at scale. In *Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics* (pp. 8440-8451). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.acl-main.747

[9] Chakravarthi, B. R., Priyadharshini, R., Muralidaran, V., Jose, N., Suryawanshi, S., Sherly, E., & McCrae, J. P. (2022). DravidianCodeMix: Sentiment analysis and offensive language identification dataset for Dravidian languages in code-mixed text. *Language Resources and Evaluation*, *56*(3), 765-806. https://doi.org/10.1007/s10579-022-09583-7

[10] He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering*, *21*(9), 1263-1284. https://doi.org/10.1109/TKDE.2008.239

[11] Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., & Duchesnay, É. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, *12*(85), 2825-2830.

[12] Wolf, T., Debut, L., Sanh, V., Chaumond, J., Delangue, C., Moi, A., Cistac, P., Rault, T., Louf, R., Funtowicz, M., Davison, J., Shleifer, S., von Platen, P., Ma, C., Jernite, Y., Plu, J., Xu, C., Le Scao, T., Gugger, S., Drame, M., Lhoest, Q., & Rush, A. (2020). Transformers: State-of-the-art natural language processing. In *Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing: System Demonstrations* (pp. 38-45). Association for Computational Linguistics. https://doi.org/10.18653/v1/2020.emnlp-demos.6

[13] Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). "Why should I trust you?": Explaining the predictions of any classifier. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining* (pp. 1135-1144). Association for Computing Machinery. https://doi.org/10.1145/2939672.2939778