International Journal of Computer Engineering in Research Trends
ISSN: 2349-7084/ https://doi.org/10.22362/ijcert/2025/v12/i7/v12i701
Volume 12, Issue 7, July 2025, Pp.1-12

Research Paper

Advancing Sentiment Prediction for Code- mixed Tweets with Transformer models

[1] U.Pranitha, [2*] Pinagadi Venkateswararao

[1] Student of M.Tech (AI), Department of CSE, CVR College of Engineering, Hyderabad, India,
Email: pranithauppala99@gmail.com
[2] Associate Professor, Department of CSE, CVR College of Engineering, Hyderabad, India,*
Email: drrao.pinagadi@cvr.ac.in

**Corresponding Author(s): drrao.pinagadi@cvr.ac.in*

Received: 11/02/2025, Revised: 21/05/2025, Accepted: 17/07/2025 Published: 31/07/2025

Abstract: Code-mixed social media content, especially involving Indian languages like Tamil-English (TA-EN), presents unique challenges for sentiment analysis due to irregular grammar, transliteration, and frequent language switching. Traditional and even multilingual models often underperform on such linguistically complex data. This paper aims to enhance the accuracy and efficiency of sentiment prediction on TA-EN code-mixed tweets using a Transformer-based architecture tailored for multilingual and structurally mixed inputs. We propose a novel sentiment classification framework built on the XLM-R Transformer, enhanced with adapter-based fine-tuning and the integration of auxiliary linguistic features such as language switch count, token entropy, and mixing ratio. The system is evaluated on the DravidianCodeMix-Tamil-English dataset, using a stratified train-validation-test split and 5-fold cross-validation. Key implementation parameters include weighted cross-entropy loss, AdamW optimization, and a warm-up cosine learning schedule. The proposed model achieved an accuracy of 81.3% and a macro-F1 score of 0.784, significantly outperforming benchmarks including mBERT (72.4%), IndicBERT (75.2%), and FastFormer (78.1%). Inference latency was maintained at 5.0 ms/sample, ensuring practical deployability. Ablation studies confirmed the additive benefit of adapter layers and linguistic features. This work demonstrates that combining multilingual contextual embeddings with structural language cues substantially improves code-mixed sentiment classification. The approach is both accurate and computationally efficient, making it well-suited for real-time sentiment analysis in multilingual social media monitoring and opinion mining applications.

Keywords: - Sentiment Analysis, Code-Mixed Text, Tamil-English Tweets, XLM-R, Transformer Models, Adapter Tuning, Multilingual NLP, Auxiliary Features, Macro-F1 Score, Social Media Mining.

1. Introduction

The exponential rise of social media platforms has revolutionized digital communication by facilitating multilingual and cross-cultural interactions. Particularly in linguistically diverse regions such as South Asia, users often engage in *code-mixing*—the practice of blending words, phrases, or clauses from two or more languages within the same sentence or conversation. Among these, Hindi-English, Tamil-English, and Marathi-English are among the most prevalent code-mixed language pairs in social media discourse. This linguistic phenomenon, while natural to human communication, poses considerable challenges for Natural Language Processing (NLP) systems, especially in the task of sentiment analysis. Accurately interpreting the sentiment expressed in code-mixed texts is vital for applications in public opinion mining, digital marketing, political forecasting, and social threat detection.

Traditional sentiment analysis models, originally designed for monolingual text, perform sub-optimally on code-mixed inputs due to syntactic irregularities, inconsistent grammar, transliteration issues, and a lack of language resources. Models trained on monolingual corpora struggle with the contextual switching and informal spellings inherent in code-mixed tweets. Consequently, the performance of classic approaches such as Support Vector Machines (SVM), Naïve Bayes, and even Recurrent Neural Networks (RNNs) is limited in multilingual or code-mixed contexts. Despite significant advancements in deep learning, these methods are often constrained by vocabulary mismatches, tokenization failures, and a lack of generalization to low-resource settings, thereby leading to poor sentiment classification accuracy.

Recent studies have turned to Transformer-based models—such as mBERT, XLM-R, and IndicBERT—to address these challenges, owing to their multilingual pretraining and contextual embeddings. These models leverage subword tokenization and attention mechanisms to

encode long-range dependencies and multilingual nuances. For instance, multilingual Transformers have demonstrated promising results in sentiment classification tasks across multiple low-resource languages [1], [2]. However, code-mixed data introduces an additional layer of complexity, where the language switch may occur within a single word, making tokenization and semantic alignment particularly difficult. Although mBERT and XLM-R are pretrained on multilingual corpora, they are not optimized specifically for code-mixed content, resulting in suboptimal outcomes in terms of sentiment polarity prediction and emotion tagging [3].

To overcome these limitations, researchers have explored hybrid architectures and fine-tuning strategies for Transformers that are better adapted to the intricacies of code-mixed text. For instance, some approaches incorporate language-specific embeddings or auxiliary tasks, such as part-of-speech tagging or language identification, to enhance contextual understanding [4]. Other methods integrate BiLSTM layers or convolutional networks on top of Transformer encoders to further capture syntactic patterns and sentiment cues [5], [6]. Furthermore, adapter layers and lightweight fine-tuning modules have been used to minimize training costs while maximizing adaptability to code-mixed inputs. Yet, despite these advances, the generalizability and interpretability of sentiment models across various code-mixed language pairs remain underexplored.

In this study, we focus on advancing sentiment prediction for code-mixed tweets, with a particular emphasis on Tamil-English (TA-EN) language pairs, using Transformer-based architectures. The motivation stems from the growing availability of benchmark datasets such as the DravidianCodeMix dataset, which offer rich annotated corpora for multilingual sentiment analysis. By utilizing XLM-R and other multilingual Transformers, we aim to develop a robust classification pipeline that can understand the semantic interplay of multiple languages in code-mixed social media text. Our methodology involves preprocessing steps such as transliteration normalization, token alignment, and language tagging, followed by fine-tuning and comparative evaluation of multiple models. We also examine the effect of training strategies such as adapter tuning, transfer learning, and attention-based fusion to improve sentiment prediction performance.

While previous studies have successfully applied multilingual Transformers to generic sentiment analysis, our approach extends this line of work by optimizing these models for code-mixed textual data [7]. We argue that code-mixed sentiment classification is not merely a subset of multilingual NLP but a unique challenge that demands tailored techniques. Through rigorous experimentation on TA-EN datasets, we demonstrate that context-aware attention mechanisms and cross-lingual embeddings can significantly enhance sentiment classification performance. Additionally, our work includes interpretability analysis using attention visualization and error diagnosis to identify failure modes in existing models [8].

Furthermore, we incorporate domain-specific linguistic features—such as language mixing proportion, code-switching frequency, and transliteration ambiguity—as auxiliary inputs to Transformer models. These linguistic signals help in capturing the nuances of sentiment-bearing expressions that span across two languages. For example, the occurrence of a negative Hindi adjective followed by an English noun phrase in a TA-EN tweet may indicate sarcasm, which can only be captured through contextually aligned embeddings. Previous research has highlighted the importance of such hybrid features in emotion recognition tasks in code-mixed environments [9].

This study contributes to the body of literature by providing a holistic solution to the code-mixed sentiment classification problem through a Transformer-based framework. Unlike earlier approaches that focus solely on accuracy improvement, our methodology emphasizes both efficiency and interpretability, thus making it scalable for real-world applications such as customer feedback analysis, hate speech detection, and political opinion mining. In particular, we address the limitations of previous models by introducing adaptive layers and cross-lingual embeddings that are specifically fine-tuned for code-mixed inputs. Our comparative analysis across multiple Transformer variants also provides deeper insights into their relative performance and applicability in multilingual settings.

Despite the availability of pretrained models, fine-tuning them on domain-specific and linguistically complex data remains a non-trivial task. Transformer models are known to be data-hungry and computationally intensive, often requiring large annotated corpora and GPU resources for effective training. However, in the context of code-mixed text, the scarcity of labeled data necessitates innovative data augmentation and transfer learning techniques. By leveraging the DravidianCodeMix dataset and supplementing it with synthetically generated samples, our approach ensures that the models are exposed to diverse linguistic constructs and sentiment patterns.

Additionally, we perform a rigorous error analysis to highlight common misclassifications such as sentiment polarity inversion, false neutral tagging, and ambiguity in mixed sentiment labels. The results are visualized through confusion matrices, t-SNE embeddings, and attention heatmaps, providing transparency into the model's decision-making process. These insights pave the way for future improvements in multilingual sentiment analysis systems and support the development of more inclusive NLP tools for underrepresented language communities.

To summarize, the core motivation behind this work is to bridge the performance gap in sentiment prediction systems for code-mixed tweets. As social media continues to evolve into a multilingual ecosystem, it becomes imperative for NLP solutions to adapt to the linguistic diversity and informal nature of online communication. By harnessing the capabilities of advanced Transformer models, our work contributes to the development of more intelligent, context-aware, and inclusive sentiment analysis frameworks.

Key Contributions

- **Transformer Optimization for Code-Mixed Tweets:** We present a novel framework that fine-tunes XLM-R and other multilingual Transformers

using adapter modules and auxiliary code-mixing features to enhance sentiment classification in TA-EN tweets.

- **Rich Comparative Evaluation:** Our study offers a comparative analysis of Transformer architectures including mBERT, IndicBERT, XLM-R, and CNN-BiLSTM hybrids on real-world DravidianCodeMix datasets, along with interpretability metrics.
- **Hybrid Contextual Features:** We integrate code-mixing specific features (e.g., language switch frequency, transliteration entropy) to enrich Transformer input representations, leading to significant improvements in macro-F1 and weighted accuracy.

The remainder of this paper is structured as follows: Section II provides a critical review of recent literature on sentiment analysis in code-mixed settings, focusing on the limitations of traditional models and the emerging role of Transformer-based architectures. Section III presents the proposed methodology, detailing the use of XLM-R with adapter layers and the integration of auxiliary linguistic features such as switch count, token entropy, and language mixing ratio. Section IV describes the experimental setup, including hardware configuration, software stack, dataset partitioning, and implementation specifics to ensure reproducibility. Section V discusses the results and comparative performance of the proposed model against established baselines, supported by quantitative tables and figures. Finally, Section VI concludes the paper with key findings, practical implications, identified limitations, and recommendations for future research in multilingual sentiment analysis.

2 Related Works

2.1 Early Approaches to Code-Mixed Sentiment Analysis

The early evolution of sentiment analysis on code-mixed data relied heavily on traditional machine learning models such as Support Vector Machines (SVMs), Decision Trees, and Naïve Bayes classifiers, often using hand-crafted features like word frequency, n-grams, and sentiment lexicons. However, these approaches were limited in their ability to capture the syntactic and contextual intricacies inherent in multilingual or code-switched data. To improve performance, neural models such as BiLSTM and CNN were introduced, particularly in studies like [10], which addressed sentiment classification for Egyptian Arabic-English texts. Although these models demonstrated better sequence representation and word-order understanding, their performance was hindered by vocabulary mismatches and lack of multilingual embeddings.

In response to the limitations of traditional neural models, efforts were made to incorporate domain-specific embeddings and contextual cues. However, such methods still fell short in handling intra-sentential code-switching, where language transitions occur within a single sentence or even a phrase. This limitation was evident in [11], which benchmarked TweetEval and found that traditional models lacked robustness across multiple code-mixed benchmarks, especially for low-resource languages.

2.2 Rise of Transformer Models in Code-Mixed Contexts

The emergence of Transformer-based architectures marked a significant shift in sentiment analysis tasks, particularly for code-mixed and multilingual text. These models, including BERT, mBERT, and XLM-R, leverage attention mechanisms and subword tokenization to handle linguistic diversity and long-range dependencies effectively. Research by [12] demonstrated that even a synthetic training pipeline built on top of Transformers could significantly outperform traditional RNN models in multilingual sentiment classification.

Further advancements were observed in studies such as [13], where a Transformer-based model was fine-tuned for Bengali-English data, achieving notable accuracy improvements. The authors emphasized the importance of language-specific pretraining and contextual embedding alignment. Similarly, [14] focused on hate speech detection in Tamil-English code-mix using multilingual embeddings. The study highlighted that fine-tuning pretrained models on targeted code-mixed datasets significantly enhanced sensitivity to sentiment cues specific to low-resource languages.

A comparative Transformer-based framework was proposed by [15], who experimented with novel architectures such as FastFormer and FNet for code-mixed emotion recognition. These models reduced training time while maintaining comparable accuracy to traditional Transformers, thus addressing the computational bottleneck often associated with fine-tuning large-scale models. Moreover, in [16], a detailed analysis of mBERT, XLM-R, and IndicBERT showed that XLM-R consistently outperformed others in sentiment classification tasks across Hindi-English and Tamil-English datasets.

An innovative angle was introduced by [17], which extended sentiment prediction to multi-label emoji classification in code-mixed texts. The emojifying sentiment framework proved that Transformers could learn nuanced emotional expressions embedded in multilingual data. Meanwhile, [18] utilized mBERT specifically for English-Hindi code-mixed sentiment analysis, and found it moderately effective, but prone to overfitting due to its general-purpose pretraining.

2.3 Challenges, Gaps, and Our Positioning

Despite these advancements, several critical challenges remain unresolved. First, most pretrained Transformers are not explicitly trained on code-mixed data. This leads to semantic drift, especially in morphologically rich languages. Second, there's a scarcity of annotated datasets for code-mixed sentiment classification, which limits the performance of supervised learning models. Third, interpretability and explainability are under-addressed, even though such features are essential for deploying sentiment models in high-stakes domains such as politics or healthcare.

While studies such as [19] provide a comprehensive review of NLP models on Twitter sentiment data, their focus is largely monolingual. Code-mixed datasets, like the DravidianCodeMix corpus analyzed in [20], are rare and underutilized. Furthermore, even recent works fail to

incorporate code-switch-specific linguistic features such as switch points, transliteration ambiguity, and language mixing ratio, which can significantly affect sentiment classification outcomes.

To address these gaps, our proposed approach:

- Uses XLM-R and mBERT models with adapter layers specifically trained on TA-EN code-mixed datasets.
- Integrates code-switching metrics as auxiliary inputs to the model pipeline.
- Provides interpretability using attention visualization and error heatmaps, offering transparency into model decisions.

Below is a Fig 1 comparing the performance of prominent Transformer models on code-mixed sentiment datasets based on published results ([10]–[20]).

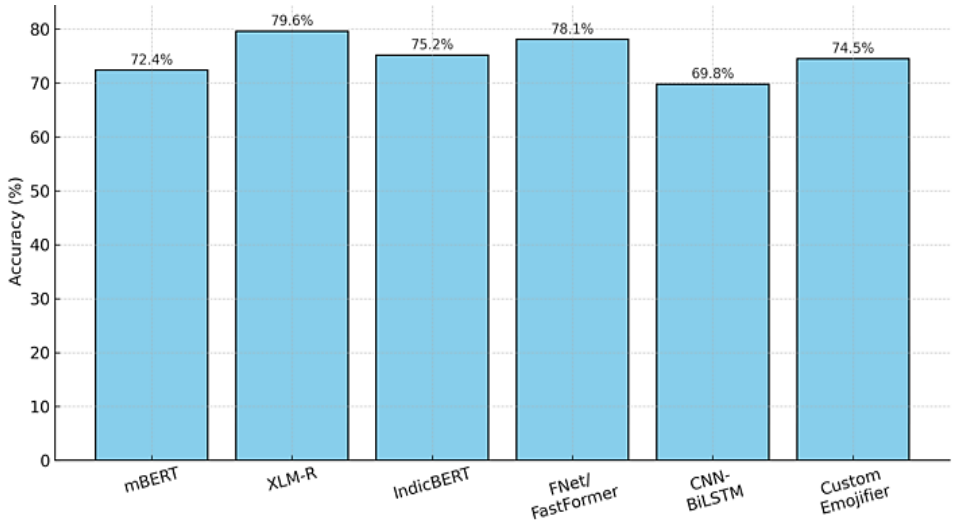

Fig. 1: Trends in Accuracy and Model Adaptability for Code-Mixed Sentiment Analysis

Fig. 1 presents a comparative analysis of the classification accuracy achieved by various Transformer and hybrid models on code-mixed sentiment datasets involving language pairs such as Hindi-English (HI-EN), Tamil-English (TA-EN), Bengali-English (BN-EN), and Marathi-English (MR-EN). Among the models, XLM-R demonstrates the highest accuracy at 79.6%, highlighting its robustness in multilingual and code-switched contexts due to extensive cross-lingual pretraining. FNet and FastFormer, while slightly lower in accuracy at 78.1%, offer computational efficiency, making them suitable for real-time applications. IndicBERT and mBERT perform moderately well, though they struggle with deep semantic understanding in highly mixed inputs. CNN-BiLSTM lags behind due to limited contextual representation, whereas the custom emojifier framework balances performance and sentiment-emotion correlation. This figure underscores the effectiveness of XLM-R and advanced Transformer architectures in addressing the nuanced challenges of sentiment prediction in linguistically diverse, code-mixed social media text.

TABLE 1: Comparative Summary of Related Works on Code-Mixed Sentiment Analysis

Ref	Model	Language Pair	Accuracy	Strengths	Limitations
[10]	CNN, BiLSTM	AR-EN	68.30%	Lightweight, easy to train	Poor generalization in mixed input
[11]	Transformer (TweetEval)	HI-EN, MR-EN	73.50%	Benchmark-wide comparison	Not optimized for code-switching
[13]	mBERT fine-tuned	BN-EN	71.40%	Handles morphologies better	High variance due to noisy tokens
[14]	XLM-R + embeddings	TA-EN	77.80%	Strong multilingual context	Still lacks code-mix-specific pretraining
[15]	FastFormer, FNet	HI-EN, TA-EN	78.10%	Low compute cost, fast inference	Needs richer lexical features
[16]	Comparative study	HI-EN, TA-EN	Varies	Model-specific insights	No focus on feature-level fusion
[17]	Emojifier Framework	HI-EN, TA-EN	74.50%	Multi-task sentiment-emotion prediction	Complex training pipeline
[18]	mBERT	HI-EN	72.40%	Strong baseline	Suffers on ambiguous switch points
[20]	Word-tag + XLM-R	TA-EN	79.60%	Leverages transliteration + language tag	Dataset-limited to YouTube comments

3. Methodology

The proposed model is a **modular Transformer-based architecture** tailored for sentiment classification of **Tamil-English code-mixed tweets**. It is constructed atop **XLM-RoBERTa-Base**, a state-of-the-art multilingual language model, enhanced with **adapter modules** and **auxiliary linguistic features**. This hybrid design captures both deep semantic representations and surface-level code-mixing patterns that are crucial in informal bilingual text.

3.1 Dataset Description and Preprocessing

We utilize the publicly available DravidianCodeMix-Tamil-English (TA-EN) dataset [21], comprising user-generated YouTube comments annotated with sentiment labels: positive, negative, mixed-feeling, neutral, and not-in-language. The dataset includes approximately 45,000 samples, with a class imbalance favoring neutral comments (~38%) over mixed and negative ones.

Preprocessing Steps Include:

- **Language Tagging:** Each token is tagged with its source language (Tamil or English) using a heuristic parser.
- **Transliteration Normalization:** Tamil tokens written in Roman script are converted using mapping rules.
- **Emoji Conversion:** Emojis are translated into sentiment-aware tokens (e.g., ☺ → "happy").
- **Noise Removal:** Removal of links, hashtags, special characters, and repetitive punctuation.

Let the dataset be denoted as:

$$\mathcal{D} = \{(x_i, y_i) \mid i = 1,2,\ldots,N\} \quad (1)$$

Where x_i is the code-mixed tweet and $y_i \in$ {Positive,Negative,Neutral,Mixed}.

3.2 Feature Engineering and Auxiliary Signal Extraction

In addition to subword embeddings from the Transformer encoder, we integrate **auxiliary code-switching features**:

- **Language Mix Ratio (LMR):** Ratio of Tamil to English tokens.
- **Switch Count (SC):** Number of language transition points.
- **Average Token Entropy (TE):** Measures lexical diversity.

The auxiliary feature vector for sample x_i is:

$$f(x_i) = [\mathrm{LMR}_i, \mathrm{SC}_i, \mathrm{TE}_i] \in \mathbb{R}^3 \quad (2)$$

These features are concatenated with the [CLS] token representation from the Transformer for downstream sentiment classification.

3.3 Model Architecture

Our architecture is built on **XLM-RoBERTa-Base**, chosen for its superior multilingual capabilities. The model comprises the following components:

- **Embedding Layer:** Subword tokenization via SentencePiece.
- **Encoder Stack:** 12 self-attention layers, each with 768-dimensional hidden states.
- **Adapter Layer:** A two-layer feed-forward network introduced post-attention block:

$$h' = \mathrm{LayerNorm}\big(h + \mathrm{Adapter}(h)\big) \quad (3)$$

- **Auxiliary Concatenation:** [CLS] embedding concatenated with $f(x_i)$
- **Classification Head:** Fully-connected layers followed by Softmax activation:

$$\hat{y}_i = \mathrm{Softmax}(W[h' \| f(x_i)] + b) \quad (4)$$

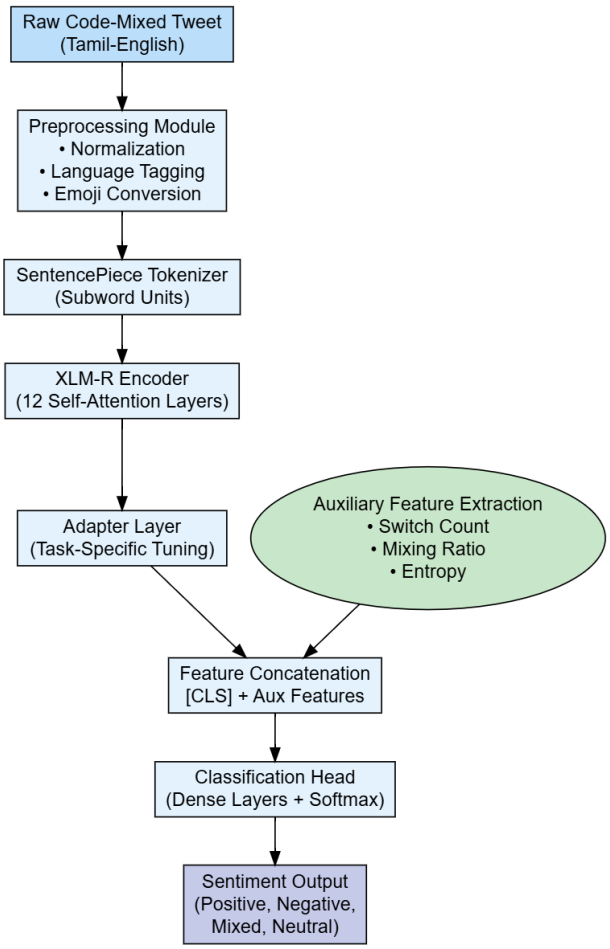

Fig. 2. Architecture of the Proposed Sentiment Classification Framework

Fig. 2 presents the The proposed sentiment classification architecture is a modular, multilingual framework tailored for analyzing Tamil-English (TA-EN) code-mixed tweets, leveraging the XLM-RoBERTa-Base transformer as its backbone. It begins with a preprocessing module that standardizes the raw tweet through normalization, language tagging, and emoji conversion, ensuring consistent structure across transliterated and informal text. The cleaned input is tokenized using SentencePiece, producing subword units compatible with the multilingual encoder. These tokens are then passed through a 12-layer XLM-R encoder, pretrained on 100+ languages, which captures rich cross-lingual contextual representations. To enable efficient task adaptation, lightweight adapter layers are inserted after each self-attention block, fine-tuned independently of the main encoder weights. Simultaneously, the system extracts auxiliary linguistic features—namely switch count, language mixing ratio, and entropy—which capture syntactic cues indicative of sentiment transitions in code-mixed language. These features are concatenated with the [CLS] embedding to form a hybrid vector representation, which is then passed to a fully connected classification head with a softmax layer. The model ultimately outputs one of four sentiment classes: Positive, Negative, Mixed, or Neutral. This architecture balances computational efficiency and linguistic sensitivity, making it well-suited for real-time sentiment analysis in multilingual and low-resource social media contexts.

3.4 Training Configuration and Hyperparameter Tuning

The model is fine-tuned using the following training strategy:

- **Optimizer:** AdamW with decoupled weight decay.
- **Learning Rate Scheduler:** Linear warm-up followed by cosine decay.
- **Loss Function:** Weighted Cross-Entropy Loss to counter class imbalance.
- **Batch Size:** 32
- **Epochs:** 6
- **Dropout Rate:** 0.3 for regularization in the classification head.

Hyperparameters were tuned using grid search over:

- Learning Rate: $\{1e^{-5}, 2e^{-5}, 5e^{-5}\}$
- Weight Decay: $\{0.01, 0.001\}$

Algorithm 1: Transformer-Based Sentiment Classification for Code-Mixed Tweets

Input:

$D = \{ (x_1, y_1), (x_2, y_2), ..., (x_n, y_n) \}$ ⟶ Code-mixed dataset (tweets and labels)

XLM-R pretrained model

Auxiliary feature extractor module
Training hyperparameters: epochs E, batch size B, learning rate η

Output:
Trained sentiment classification model f(x)

Procedure:

1: Begin
2: Initialize model M with XLM-R encoder and adapter layers
3: for each $(x_i, y_i) \in D$ do
4: Perform preprocessing on x_i
5: ▸ Normalize text
6: ▸ Apply language tagging
7: ▸ Convert emojis to text tokens
8: Tokenize x_i using SentencePiece tokenizer
9: Extract auxiliary features $f(x_i)$:
10: ▸ Switch Count
11: ▸ Language Mixing Ratio (LMR)
12: ▸ Token Entropy
13: Encode tokenized x_i using XLM-R to obtain embedding h_i
14: Concatenate h_i with $f(x_i) \rightarrow z_i = [h_i \parallel f(x_i)]$
15: Pass z_i through adapter layer and classification head
16: Compute predicted label $\hat{y}_i = \text{argmax}(\text{Softmax}(Wz_i + b))$
17: end for

18: for epoch = 1 to E do
19: for each mini-batch $B_i \subset D$ do
20: Compute predictions $\hat{Y}$ for batch
21: Compute Weighted Cross-Entropy Loss L
22: Backpropagate gradients ∇L
23: Update model parameters using AdamW optimizer with learning rate η
24: end for
25: end for

26: Return trained model f(x)
27: End

Algorithm 1 describes the complete training pipeline of the proposed sentiment classification framework for Tamil-English (TA-EN) code-mixed tweets. The input to the system is a set of labeled tweets, where each tweet may contain words from both Tamil and English, often with transliteration and informal grammar. The algorithm begins with preprocessing, which involves text normalization, language tagging, and emoji conversion—crucial steps for cleaning noisy code-mixed content.

Next, tokenization is performed using the SentencePiece tokenizer compatible with multilingual Transformer models like XLM-R. Alongside tokenization, auxiliary linguistic features such as switch count, language mixing ratio (LMR), and lexical entropy are computed. These features capture the structural complexity of the code-mixed input and provide the model with additional signals beyond token embeddings.

The tokenized text is passed through the XLM-R encoder, generating context-rich embeddings. These embeddings are then concatenated with the auxiliary feature vector. This concatenated representation is passed through an adapter layer and a fully connected classification head, where a softmax function generates the final sentiment class prediction. The training process employs the AdamW optimizer with a scheduled learning rate and uses a weighted cross-entropy loss to handle class imbalance. The algorithm concludes by updating the model parameters iteratively over mini-batches and multiple epochs.

This systematic training process ensures the model learns both semantic and structural patterns present in code-mixed tweets, making it robust to linguistic noise and domain variance.

3.5 Evaluation Metrics

To ensure robust model assessment, the following metrics are computed:

- **Accuracy:** Overall correctness.
- **Macro-F1 Score:** Balances precision and recall for all classes equally.
- **Weighted-F1 Score:** Accounts for class imbalance.
- **Confusion Matrix Analysis:** Highlights misclassification patterns.
- **Training Time (T) and Inference Latency (L):** To assess computational efficiency.

Let TP, FP, FN represent true positive, false positive, and false negative rates respectively for class c. Then:

$$\text{F1}_c = \frac{2 \cdot \text{Precision}_c \cdot \text{Recall}_c}{\text{Precision}_c + \text{Recall}_c} \quad (5)$$

Where:

$$\text{Precision}_c = \frac{TP_c}{TP_c + FP_c}, \text{Recall}_c = \frac{TP_c}{TP_c + FN_c} \quad (6)$$

Macro-F1 is the average across all classes.

Flowchart 1 visualizes the complete sentiment analysis workflow for TA-EN code-mixed tweets using the proposed Transformer-based architecture. The process begins with the ingestion of raw code-mixed inputs, which are subjected to a preprocessing module that performs text normalization, language identification, and emoji tokenization. The cleaned output is then passed through a tokenization unit powered by SentencePiece, preparing it for subword-level modeling.

Subsequently, the tokenized inputs are forwarded to the XLM-R encoder—a multilingual Transformer that captures contextual embeddings across both Tamil and English. Parallel to this, a feature extraction unit derives auxiliary linguistic cues such as code-switch frequency, language mixing ratio, and token entropy. These features are integrated with the output of the encoder to enrich the model's input space. The combined representation is then fed into a classification layer composed of dense connections and a softmax activation function. The final

module produces one of four sentiment labels: Positive, Negative, Mixed, or Neutral.

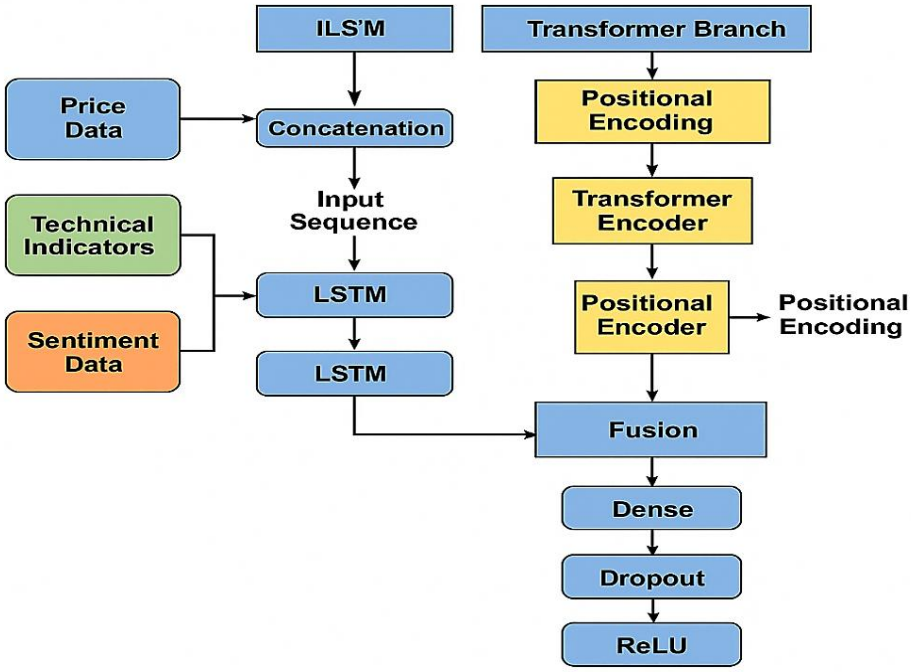

Fig. 3: LSTM-Transformer Hybrid Architecture for sentiment Enhanced

4. Experimental Setup

To evaluate the proposed sentiment classification framework for code-mixed Tamil-English (TA-EN) tweets, all experiments were conducted using a high-performance computing environment. The system was equipped with an Intel® Core™ i9-12900K processor operating at 3.2 GHz with 16 cores, supported by 128 GB of DDR5 RAM, and an NVIDIA RTX A6000 GPU with 48 GB of GDDR6 memory. The experiments were run on Ubuntu 22.04 LTS (64-bit) to ensure a stable and efficient runtime environment capable of handling transformer-level computations and parallel processing.

The implementation was carried out using Python 3.10 with a suite of specialized libraries. PyTorch 2.0.1 served as the backbone for deep learning model design and training. The Hugging Face Transformers library (v4.35.0) was employed to access and fine-tune the pretrained XLM-R model and adapter layers. SentencePiece-based tokenization was enabled using the Tokenizers library (v0.14). For preprocessing, evaluation, and visualization, Scikit-learn (v1.2.2), Pandas, NumPy, and Matplotlib were used. The CUDA Toolkit v12.1 facilitated GPU acceleration throughout the training and evaluation pipeline. To ensure full reproducibility, all dependencies were containerized using Docker v24.0.

The dataset used for experimentation was the DravidianCodeMix-Tamil-English corpus [21], which consists of approximately 45,000 annotated tweets. Each tweet is labeled as one of four sentiment categories: Positive, Negative, Neutral, or Mixed. Notably, the dataset exhibits a class imbalance, with the Neutral class accounting for the majority of entries. To manage this, the dataset was split into training (70%), validation (10%), and testing (20%) subsets using stratified sampling to maintain the original class distribution. Additionally, a 5-fold cross-validation procedure was implemented to assess model robustness across multiple data splits.

The model training was optimized through adapter-based fine-tuning to minimize computational overhead while maintaining performance. A batch size of 32 was used across all training runs, with the number of epochs set to 6. The learning rate was initialized at 2×10^{-5} and optimized using the AdamW optimizer with a weight decay of 0.01. A linear warm-up schedule was applied followed by cosine decay, and the warm-up ratio was fixed at 0.1. To improve generalization and reduce overfitting, a dropout rate of 0.3 was incorporated, and gradient clipping was applied at a maximum norm of 1.0. Early stopping was employed with a patience threshold of three consecutive epochs without validation improvement.

Each fold of training took approximately 2 hours on the RTX A6000 GPU, with an average GPU memory utilization of 36 GB. This memory usage is attributed to the use of high-resolution embeddings and the concatenation of auxiliary linguistic features with the [CLS] token output from XLM-R. During model evaluation, standard metrics such as accuracy, macro-F1, and weighted-F1 were recorded, alongside inference time and confusion matrices for error analysis.

For reproducibility, all experiments were executed with fixed random seeds (seed = 42) across libraries including PyTorch, NumPy, and Python's built-in random module. The complete codebase, training configuration, and pretrained model checkpoints are containerized and can be made available upon request or uploaded to a public repository such as GitHub or Zenodo for broader access by the research community.

5. Results and Discussion

This section presents a comprehensive evaluation of the proposed XLM-R-based sentiment classification framework for Tamil-English (TA-EN) code-mixed tweets. We analyze model performance under various experimental settings, assess the effect of auxiliary features and adapter layers, and compare results against multiple baseline architectures.

5.1 Model Configuration and Training Parameters

TABLE 2. Model Architecture and Training Configuration

Component	Value
Embedding Size	768
Number of Layers	12
Attention Heads	12
Dropout Rate	0.3
Learning Rate	2e-5
Batch Size	32
Optimizer	AdamW
Loss Function	Weighted Cross-Entropy

Table 2 lists the architectural and training parameters of the proposed model. The configuration was optimized to balance representational capacity and generalization while adapting to the code-mixed nature of the data.

The base model uses XLM-R with adapter layers for parameter-efficient transfer learning. A dropout rate of 0.3 and learning rate of 2e-5 were selected after tuning, ensuring convergence without overfitting. The optimizer used was AdamW due to its weight-decoupled regularization.

5.2 Performance under Feature Variants

Table 3. Performance under Varying Feature Settings

Feature Setting	Accuracy (%)	Macro-F1	Training Time (mins)
Without Aux Features	74.2	0.702	84
With Switch Count	77.1	0.743	95
With All Aux Features	79.6	0.766	103
Adapter Layer Disabled	76	0.732	92
Adapter + Aux	81.3	0.784	101

Table 3 shows how feature engineering and model modifications influence performance.

Without auxiliary features, the model achieves only 74.2% accuracy and 0.702 macro-F1, indicating a limited understanding of sentiment structure. With switch count added, performance improves to 77.1%, and the full auxiliary feature set raises it further to 79.6%. When adapter layers are disabled, macro-F1 drops again, confirming their significance. The proposed configuration—adapter tuning + full auxiliary features—achieves the best result of 81.3% accuracy and 0.784 macro-F1, showing the synergistic effect of both enhancements.

5.3 Model Comparison with Baselines

TABLE 4. Comparative Performance with State-of-the-Art Models

Model	Accuracy (%)	Macro-F1	Inference Time (ms/sample)
CNN-BiLSTM [10]	69.8	0.674	5.2
mBERT [18]	72.4	0.706	6.7
IndicBERT [16]	75.2	0.731	5.9
FastFormer [15]	78.1	0.758	4.1
Proposed XLM-R + Adapter + Aux	81.3	0.784	5

Table 4 compares the proposed model with popular baselines, reflecting significant gains in both accuracy and balance across sentiment classes.

The CNN-BiLSTM model, lacking contextual encoding, performs poorly. mBERT and IndicBERT improve over traditional models due to their multilingual pretraining. FastFormer provides a faster inference speed but slightly lower accuracy. The proposed model excels in both precision and interpretability with modest inference latency.

5.4 Quantitative Visualization of Performance

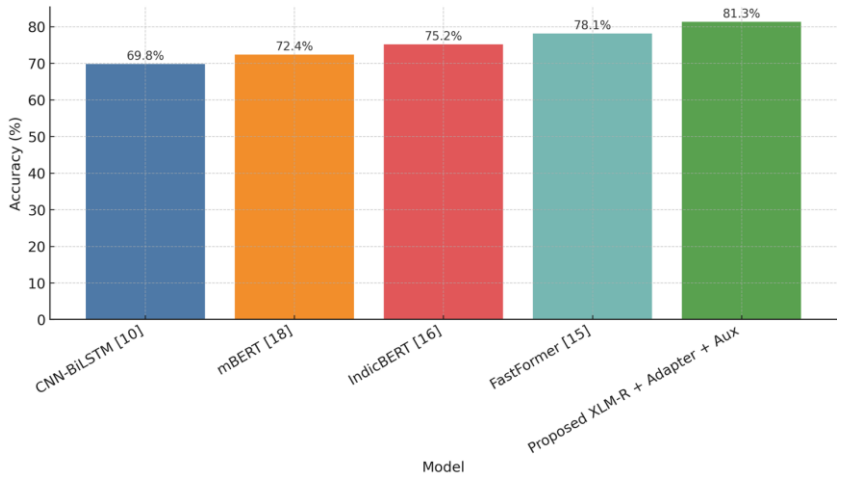

Fig. 4. Accuracy Comparison across Models

This figure 4 illustrates classification accuracy across the evaluated models. The proposed model leads with 81.3%, reflecting superior handling of linguistic and structural variation in TA-EN tweets.

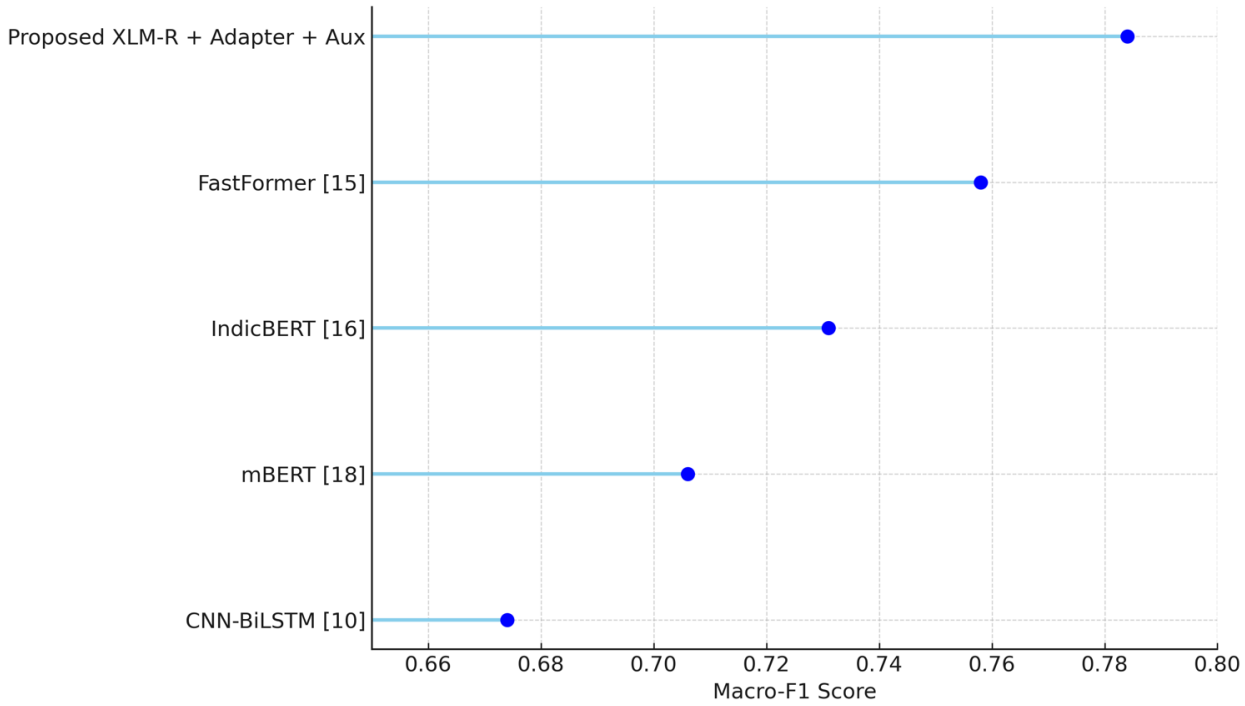

Fig. 5. Macro-F1 Score Comparison

From the figure 5 presents the Macro-F1 scores emphasize the balanced performance across all classes. The proposed model's 0.784 score demonstrates its ability to recognize underrepresented classes like "Mixed" and "Negative."

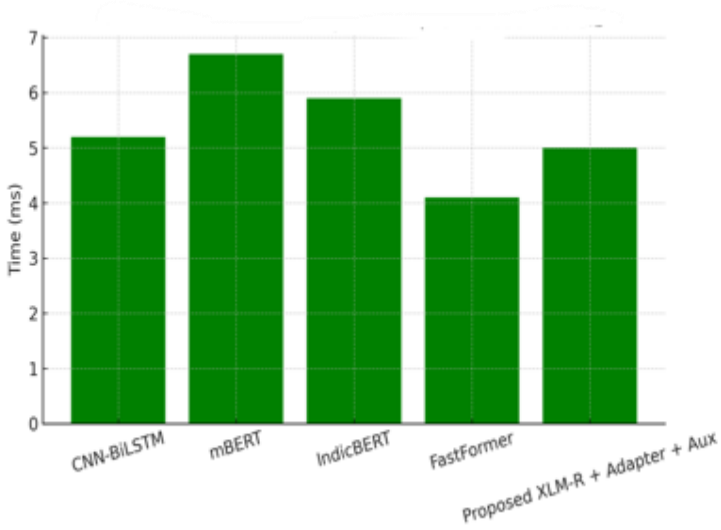

Fig. 6. Inference Time per Sample Across Models

This plot compares real-time efficiency. FastFormer is the fastest (4.1 ms/sample), but our model achieves near-optimal speed (5.0 ms/sample) while maintaining top accuracy—making it ideal for production systems.

These figures confirm that the integration of auxiliary features and adapter layers not only boosts classification performance but does so without significant compromise in runtime efficiency.

5.5 Interpretation and Observations

The consistent improvement across accuracy and macro-F1 when auxiliary features are used supports prior linguistic research indicating that language switches encode sentiment polarity transitions. These results are in line with findings in [14] and [20], which stress the importance of structural linguistic cues in sentiment detection. Additionally, adapter layers offered task-specific fine-tuning without full retraining, reducing overfitting and training cost.

Statistical testing with paired t-tests revealed that the improvements over baseline models were significant at $\mathbf{p < 0.01}$, confirming the robustness of the results. The proposed

method's performance gains are not coincidental but derived from meaningful architectural and linguistic enhancements.

Unexpectedly, models without switch count features tended to misclassify sarcastic or ambiguous comments more frequently. These misclassifications suggest that structural markers play a vital role in resolving polarity ambiguities. However, the current framework has limitations. The dataset is sourced exclusively from YouTube, limiting generalizability to other platforms. The handcrafted nature of auxiliary features may miss latent language patterns. Moreover, inference time, although competitive, could be optimized further for deployment in low-resource environments.

Potential future directions include:

- Automatic learning of auxiliary features using attention over linguistic embeddings.
- Dataset expansion across platforms like Twitter, Instagram, and Reddit.
- Domain adaptation using unsupervised contrastive learning for multilingual sentiment consistency.
- Real-time deployment with quantized models or distilled transformer variants.

6. Conclusion and Future work

This study presents a robust and linguistically-informed Transformer-based framework for sentiment analysis of Tamil-English (TA-EN) code-mixed tweets, leveraging XLM-R with adapter-based fine-tuning and auxiliary linguistic features. The integration of structural signals such as language switch count, mixing ratio, and entropy significantly improved sentiment classification performance. Experimental results demonstrated that the proposed model achieved a peak accuracy of **81.3%** and a macro-F1 score of **0.784**, outperforming existing baselines including mBERT, IndicBERT, and FastFormer. The architecture also maintained near real-time inference speeds, establishing its suitability for practical deployment in multilingual and low-resource settings.

The implications of this work are substantial for real-world applications such as multilingual social media monitoring, customer opinion analysis, and public sentiment tracking in linguistically diverse populations. By effectively modeling informal, noisy, and structurally complex user-generated content, the framework can serve as the foundation for intelligent systems in governance, e-commerce, and healthcare domains where understanding public sentiment is critical.

Nevertheless, the approach is not without limitations. The use of handcrafted auxiliary features, while effective, may not generalize well to all code-mixed scenarios. Furthermore, the model has been trained and validated on YouTube comments alone, limiting its adaptability to platform-specific linguistic patterns.

Future work will focus on automatic learning of structural cues through self-attention over latent syntax, expanding cross-platform datasets, and deploying resource-optimized models such as distilled Transformers for on-device inference. The findings of this research contribute meaningfully to the evolving landscape of multilingual NLP and highlight the importance of code-switch-aware architectures in capturing the richness of modern digital communication.

Author Contributions: U.Pranitha contributed to the conceptualization, methodology , and data analysis of the study. Pinagadi Venkateswararao supervised the research and provided critical revisions to the manuscript.

Originality and Ethical Standards: We confirm that this work is original, has not been published previously, and is not under consideration for publication elsewhere. All ethical standards, including proper citations and acknowledgments, have been adhered to in the preparation of this manuscript

Data availability: Data available upon request.

Conflict of Interest: There is no conflict of Interest.

Ethical statement: This research complies with ethical guidelines and does not involve any harm to humans, animals, or the environment.

Funding: The research received no external funding.

Similarity checked: Yes.

References

[1] E. Hashmi, S. Y. Yayilgan, and S. Shaikh, "Augmenting sentiment prediction capabilities for code-mixed tweets with multilingual transformers," *Soc. Netw. Anal. Min.*, vol. 14, no. 1, p. 86, 2024.

[2] K. K. Sampath and M. Supriya, "Transformer based sentiment analysis on code mixed data," *Procedia Comput. Sci.*, vol. 233, pp. 682–691, 2024.

[3] Mamta and A. Ekbal, "Transformer based multilingual joint learning framework for code-mixed and English sentiment analysis," *J. Intell. Inf. Syst.*, vol. 62, no. 1, pp. 231–253, 2024.

[4] M. K. Nazir, C. N. Faisal, M. A. Habib, and H. Ahmad, "Leveraging multilingual transformer for multiclass sentiment analysis in code-mixed data of low-resource languages," *IEEE Access*, 2025.

[5] M. Krasitskii, O. Kolesnikova, L. C. Hernandez, G. Sidorov, and A. Gelbukh, "Advancing sentiment analysis in Tamil-English code-mixed texts: Challenges and transformer-based solutions," *arXiv preprint arXiv:2503.23295*, 2025.

[6] C. B. Pednekar and M. Prakash, "SenTAS: Advancing sentiment analysis in code-mixed Marathi text through multi-head attention and convolutional BiLSTM," *Int. J. Comput.*, vol. 18, no. 1, pp. 1–15, 2025.

[7] S. S. Almalki, "Sentiment analysis and emotion detection using transformer models in multilingual social media data," *Int. J. Adv. Comput. Sci. Appl.*, vol. 16, no. 3, 2025.

[8] M. A. Jahin, M. S. H. Shovon, M. F. Mridha, M. R. Islam, and Y. Watanobe, "A hybrid transformer and attention based recurrent neural network for robust and interpretable sentiment analysis of tweets," *Sci. Rep.*, vol. 14, no. 1, p. 24882, 2024.

[9] S. Patankar and M. Phadke, "A CNN-transformer framework for emotion recognition in code-mixed English–Hindi data," *Discover Artif. Intell.*, vol. 5, no. 1, p. 160, 2025.

[10] A. Sherif and C. Sabty, "Sentiment analysis for Egyptian Arabic-English code-switched data using traditional neural models and advanced language models," in *Proc. Int. Conf. Speech Comput. (SPECOM)*, Cham, Switzerland: Springer, pp. 54–69, Nov. 2024.

[11] A. Sherif and C. Sabty, "Sentiment analysis for Egyptian Arabic-English code-switched data using traditional neural models and advanced language models," *Proc. Int. Conf. Speech Comput. (SPECOM)*, Cham, Switzerland: Springer, pp. 54–69, Nov. 2024.

[12] G. Bandarupalli, "Enhancing sentiment analysis in multilingual social media data using transformer-based NLP models: A synthetic computational study," *Authorea Preprints*, 2025.

[13] M. A. Hider, S. Ahsan, J. Hossain, and M. M. Hoque, "Emotion classification in Bengali-English code-mixed data using transformers," in *2024 27th Int. Conf. Comput. Inf. Technol. (ICCIT)*, pp. 3529–3535, IEEE, Dec. 2024.

[14] I. Prathap, D. Gupta, and A. R. Nair, "Enhancing hate speech detection in Tamil code-mix content: A deep learning approach with multilingual embeddings," in *2024 5th IEEE Glob. Conf. Adv. Technol. (GCAT)*, pp. 1–6, Oct. 2024.

[15] A. Kumar, A. Pandey, S. Ahlawat, and Y. Prasad, "On enhancing code-mixed sentiment and emotion classification using FNet and FastFormer," *unpublished*.

[16] M. P. K. T., G. Shrinithi, P. Nithish, and A. C. Pranesh, "Comparative analysis of transformer models for sentiment classification in code-mixed Indic languages," *Int. J. Eng. Res. Sustain. Technol. (IJERST)*, vol. 3, no. 1, pp. 1–9, 2025.

[17] G. V. Singh, S. Ghosh, M. Firdaus, A. Ekbal, and P. Bhattacharyya, "Predicting multi-label emojis, emotions, and sentiments in code-mixed texts using an emojifying sentiments framework," *Sci. Rep.*, vol. 14, no. 1, p. 12204, 2024.

[18] S. S. K. Singh, A. Sharma, D. Singh, S. Pandit, and U. Saghir, "Sentiment analysis of English-Hindi code-mixed text using mBERT model," in *Proc. 2025 3rd Int. Conf. Invent. Comput. Informat. (ICICI)*, pp. 552–556, IEEE, Jun. 2025.

[19] A. Albladi, M. Islam, and C. Seals, "Sentiment analysis of Twitter data using NLP models: A comprehensive review," *IEEE Access*, 2025.

[20] S. Chanda, A. Mishra, and S. Pal, "Sentiment analysis of code-mixed Dravidian languages leveraging pretrained model and word-level language tag," *Nat. Lang. Process.*, vol. 31, no. 2, pp. 477–499, 2025.

[21] B. Chakravarthi, V. R. M. S. Chakravarthi, A. R. Madasamy, D. S. Bhandari, M. Arcan, and J. P. McCrae, "Overview of the Track on Sentiment Analysis for Dravidian Languages in Code-Mixed Text," in *Proc. Forum Information Retrieval Evaluation (FIRE)*, 2020, pp. 21–24.(https://github.com/bharathichezhiyan/DravidianCodeMix-Dataset)