BERT-Enhanced Bi-LSTM with weighted cross-entropy for multilingual sentiment classification

Mohammad Mustafa Siddique [a,1,*], Sandeep Kumar [a,2]

[a] Christ University, Bengaluru - 560074, India
[1] mohammad.siddique@res.christuniversity.in; [2] sandeepkumar@christuniversity.in
* corresponding author

ARTICLE INFO

Article history
Received February 21, 2025
Revised June 12, 2025
Accepted July 27, 2025
Available online August 6, 2025

Keywords
BERT
Bi-LSTM
Contextual embeddings
Low-Resource languages
Multilingual sentiment analysis

ABSTRACT

With the increasing volume of multilingual user-generated content across social media platforms, effective sentiment analysis (SA) becomes crucial, especially for low-resource languages. However, traditional models relying on context-independent embeddings, such as Word2Vec, GloVe, and fastText, struggle to handle the complexity of multilingual sentiment classification. To address this, we propose an Automatic Multilingual Sentiment Detection (AMSD) framework that leverages the contextual capabilities of BERT for feature extraction and a Bidirectional Long Short-Term Memory (Bi-LSTM) network for classification. Our method, termed Elite Opposition Cross-Entropy Weighted Bi-LSTM (EOCEWBi-LSTM), integrates elite opposition-based learning to optimize hyperparameters and enhance classification accuracy. A weighted cross-entropy loss function further refines the model's sensitivity to class imbalance, thereby improving its performance. The model is trained and evaluated on the NEP_EDUSET corpus, comprising 45,434 tweets in English, Hindi, and Tamil. Experimental results demonstrate notable improvements in precision, recall, F1-score, and accuracy, highlighting the effectiveness of EOCEWBi-LSTM in multilingual sentiment analysis, especially across both high-resource and low-resource languages. The experimental results show that the proposed EOCEWBi-LSTM achieves a high F1-score ratio of 93.83% and an accuracy ratio of 93.83% compared to other existing methods. EOCEWBi-LSTM provides an effective solution for multilingual sentiment analysis, especially for languages with limited resources.

1. Introduction

Social media platforms, including Facebook, X (formerly Twitter), Instagram, WhatsApp, and TikTok, have become valuable sources of data for sentiment analysis (SA) research [1]. Twitter data, in particular, has been widely used due to its concise format and diverse linguistic expressions, including multilingual and multicultural elements [2]. SA assigns polarity (positive, neutral, or negative) to tweets and is one of the most studied tasks in natural language processing (NLP) and machine learning (ML) [3], [4]. However, SA models are often language-specific and not easily transferable across languages [5].

Multilingual Sentiment Analysis (MSA) aims to analyze sentiment across languages, using resources from well-resourced languages to process data in low-resource ones [6]. Traditional word embeddings, such as Word2Vec and fastText, have failed to consider contextual variations; however, BERT, a state-of-the-art (SOTA) pre-trained model, has significantly improved contextual understanding [7]. Despite

its effectiveness, challenges remain in adapting pre-trained models, such as BERT, to multilingual environments [8]. The development of multilingual tools increasingly relies on artificial intelligence (AI) and deep learning (DL) techniques [9]. DL has been instrumental in advancing MSA, with techniques such as Convolutional Neural Networks (CNN) and LSTM networks being explored for improved classification [10].

Traditional models using static embeddings (e.g., Word2Vec, GloVe) or context-agnostic architectures lack the flexibility to handle polysemy and code-mixed text, which is common in multilingual social media content. Even recent multilingual extensions of BERT, such as mBERT and XLM-R, while promising, often require extensive fine-tuning and do not explicitly address issues like class imbalance or hyperparameter sensitivity in deep classifiers. To address these limitations, we propose a novel hybrid framework, EOCEWBi LSTM, that combines the contextual strength of BERT with a Bi-LSTM classifier optimized using Elite Opposition-Based Learning and a Weighted Cross-Entropy Loss. This approach enhances model performance on low-resource multilingual datasets by addressing both the quality of representation and the dynamics of training.

This paper introduces an Automatic Multilingual Sentiment Detection (AMSD) approach for MSA, focusing on English (high-resource) and Hindi and Tamil (low-resource). AMSD employs BERT for contextual embeddings and an Elite Opposition Cross Entropy Weighted Bi-LSTM (EOCEWBi-LSTM) for classification. This model refines hyperparameters by adjusting the weights and biases, thereby improving sentiment detection accuracy. This study was designed to answer several key questions related to the development of deep learning-based sentiment classification models, as follows:

- Can BERT-enhanced contextual embeddings significantly improve sentiment classification in multilingual settings compared to traditional embeddings?
- Can elite opposition-based optimization improve the performance of Bi-LSTM classifiers in imbalanced multilingual datasets?
- How effectively does the proposed EOCEWBi-LSTM framework at generalizing across different languages with varying resource availability?

The contributions of this research include the creation of the NEP_Eduset dataset, which was mostly collected from platform X (formerly Twitter), the development of the EOCEWBi-LSTM model that integrates the elite opposition-based tuning method with BERT-enhanced Bi-LSTM to improve classification performance, and implementation using Python 3.7 and GPU resources from Google Collaboratory with the application of various deep learning classifiers on the dataset. Additionally, this research demonstrates the superiority of EOCEWBi-LSTM in capturing diverse sentiment tones across languages with greater accuracy. The paper is structured as follows: Section 2 reviews existing DL classification methods and research gaps, Section 3 details the methodology, Section 4 presents findings, and Section 5 summarizes contributions, limitations, and future research directions.

2. Literature Review

Sentiment analysis has been extensively studied using various ML and DL techniques. However, challenges persist in achieving high performance across multiple languages, especially in low-resource contexts. This review organizes related work into three major themes: (1) traditional ML approaches, (2) DL and pre-trained language models, and (3) multilingual and cross-lingual sentiment classification.

A text classification method utilizing BERT and various enhancements for NLP, categorizing tweet sentiments as neutral, negative, or positive. Compared to models using Word2Vec, the results showed significant improvements in Accuracy (Acc), Precision (P), Recall (R), and F1-score, particularly with neural networks such as CNN, RNN, and Bi-LSTM [11]. Four DL models combining BERT with Bidirectional LSTM (Bi-LSTM) and Bidirectional Gated Recurrent Unit (Bi-GRU) were developed. The study optimized accuracy by fine-tuning pre-trained word embeddings. It examined the impact of hybridizing Bi-GRU and Bi-LSTM on DistilBERT and RoBERTa, with Bi-GRU layers yielding the best results [12]. Enhanced sentiment analysis by applying pre-processing techniques to remove noise

from tweets, such as URLs, mentions, and hashtags, was analyzed. Their BERT-based experiments in Italian and English identified optimal pre-processing methods, thereby advancing the state-of-the-art (SOTA) sentiment classification [13].

A BERT-based model incorporating ML algorithms, including SGD, SVM, Decision Tree, Logistic Regression, Random Forest (RF), and XGBoost, was employed to detect hateful content and sentiment in tweets [14]. Mann *et al.* enhanced BERT models for sentiment analysis using the Kaggle SMILE dataset, improving text interpretation by leveraging surrounding contextual information [15]. Azzouza *et al.* employed a four-phase framework that utilized BERT-generated sentence representations, integrating classification models and pre-trained embeddings to enhance sentiment classification [16]. Prottasha *et al.* combined BERT with CNN-BiLSTM to enhance sentiment detection, comparing transfer learning (TL) approaches like GloVe, FastText, and Word2Vec [17].

Jain *et al.* introduced a BERT-DCNN model with parallel Dilated CNN layers for sentiment analysis, preserving dimensional relevance while handling long-term dependencies [18]. Wadud *et al.* developed Deep-BERT, a hybrid CNN-BERT model for offensive text classification in Bengali and English [19]. Cam *et al.* proposed a Turkish SA model that integrates lexicon-based and ML classifiers, examining public sentiment in financial tweets [20]. Nandhini *et al.* introduced a multilingual BERT model classifying sentiments across multiple languages [21].

A model DConvBLSTM-MuRIL was proposed for hate speech detection in Hinglish, Tamil-English, and Malayalam-English, outperforming prior models [22]. Roy suggested an ensemble transformer-based model for sentiment classification in Kannada and Malayalam [23]. Aruna & Vetriselvi [24] demonstrated a cross-lingual transfer learning (TL) approach for Tamil sentiment analysis, yielding higher results in most cases compared to other models. Hossain *et al.* introduced AFuNet, an Attention-Based Fusion Network for low-resource language classification [25]. Nazir *et al.* [26] proposed transformer-based models, including BERT, RoBERTa, and XLMRoBERTa, for Tamil-English YouTube comment sentiment analysis, leveraging synthetic oversampling techniques to enhance performance.

Liu *et al.* [27] suggested the use of DL-based NLP in MSA. At the same time, Prova [28] proposed the GPT and DL-based Meta-Ensemble Model for Multilingual Emotion Classification in E-commerce Customer Reviews. Md Saef Ullah Miah *et al.* [29] recommended a multimodal approach to cross-lingual sentiment analysis using an ensemble of transformers and LLM. Geethanjali and Valarmathi [30] presented modality-enriched and multilingual emotion recognition in social media using IChOA-CNN-LSTM. Dhananjaya *et al.* [31] introduced the lexicon-based fine-tuning of multilingual language models for low-resource language sentiment analysis.

Despite significant advancements in sentiment analysis, several limitations persist in existing approaches. Traditional models employing static word embeddings, such as Word2Vec and GloVe, are unable to capture contextual variations in meaning, which is particularly problematic in informal or multilingual text. While recent methods have incorporated contextualized models like BERT, their application has been limited mainly to high-resource languages, with limited effectiveness observed in low-resource settings such as Hindi and Tamil. Additionally, many studies overlook the issue of class imbalance in multilingual datasets, resulting in biased predictions and degraded performance. Furthermore, hyperparameter selection in deep learning (DL) models is often performed manually or through a simple grid search, which can be inefficient and suboptimal. These gaps highlight the need for a unified, optimized, and scalable framework that can generalize across languages and handle noisy, imbalanced real-world data. Motivated by these challenges, the present study introduces a BERT-enhanced Bi-LSTM model integrated with elite opposition-based learning and weighted cross-entropy loss, designed to improve sentiment classification accuracy across both high- and low-resource languages.

2.1. LSTM Classifier

An LSTM retains all textual data throughout time. A bidirectional RNN integrates two RNNs, enabling them to convey data from both forward and backward sequences. LSTM has enhanced sequence

processing abilities and can identify long-range dependencies within sequences. An LSTM layer comprises a series of LSTM cells, with the sequential data input processed in a forward manner. The LSTM cell is illustrated in Fig. 1.

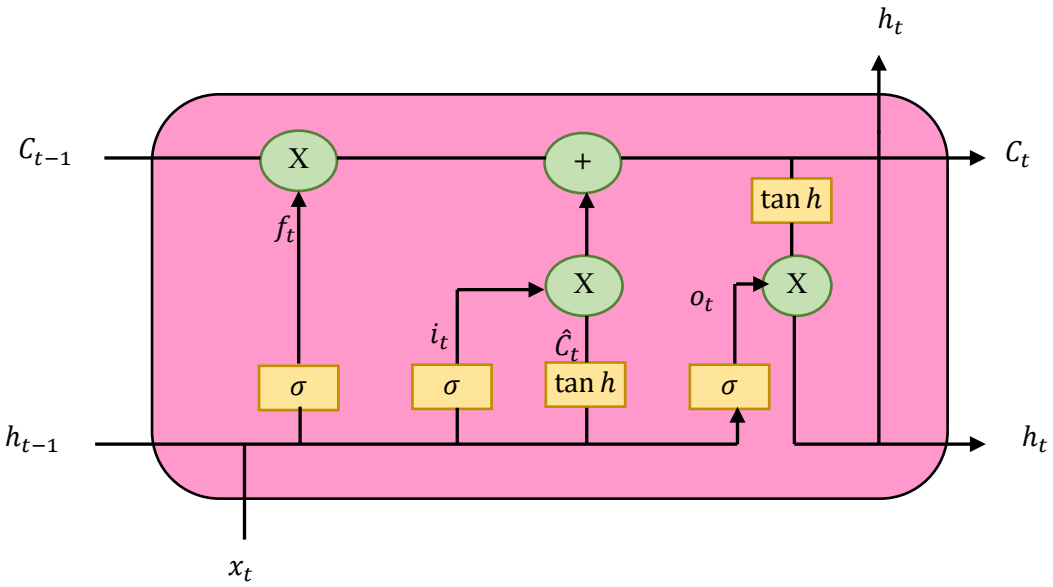

Fig. 1. LSTM cell

The following transformations are carried out while taking into account the current value x_t, the previous hidden state h_{t-1}, and the previous state C_{t-1} in below equation (1-6).

$$f_t = \sigma(We_f.[h_{t-1}, x_t] + b_f) \quad (1)$$

$$i_t = \sigma(We_i.[h_{t-1}, x_t] + b_i) \quad (2)$$

$$\hat{C}_t = \tanh(We_C.[h_{t-1}, x_t] + b_c) \quad (3)$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \quad (4)$$

$$o_t = \sigma(We_o.[h_{t-1}, x_t] + b_o) \quad (5)$$

$$h_t = o_t * \tanh(C_t) \quad (6)$$

In this case, σ represents the sigmoid function. Then, tanh is a representation of the hyperbolic tangent function. The forget gate is represented by f_t. The input gate is indicated by i_t. The output gate is denoted by o_t. The weight matrix is denoted as W_e, and the bias vector's value is b. The concatenation operator is $[\cdot]$. The dot product is indicated by $*$. Since unidirectional LSTM can only take in data from the past, it can only remember information that has already occurred. Another option is to use a Bi-LSTM, which allows for both forward and backwards processing of inputs. Because it integrates two latent states, this strategy is significantly more successful in the present and future, enabling one to learn from both past and future experiences.

2.2. Bi-LSTM Classifier

In Bi-LSTM, contextual data is analyzed in both backward and forward directions, facilitating the retention of knowledge from both past and future, as proposed by [32], [33]. The suggested framework illustrates the Bidirectional LSTM architecture in Fig. 2. In the equations (1-6), there are two hidden values, namely $\overrightarrow{h}_{t-1}$ and $\overleftarrow{h}_{t-1}$ associated with the sequence in the Bi-LSTM: $\overrightarrow{h}_{t-1}$ is involved in the forward calculation and $\overleftarrow{h}_{t-1}$ is involved in the reverse calculation. In Equation (1-6), the bias vector is represented by b, and the weight matrix by we. It has a major impact on improving the accuracy of the SA.

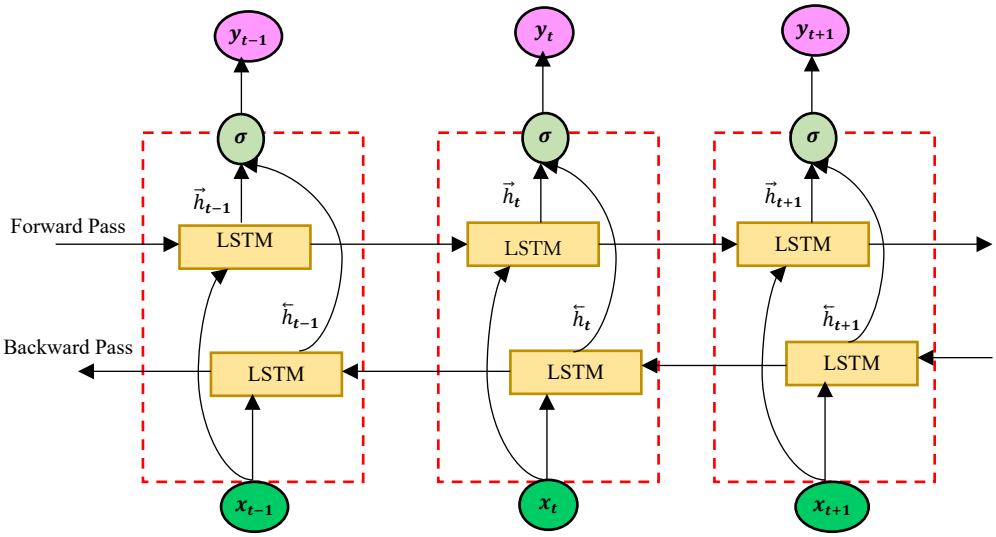

Fig. 2. Bidirectional LSTM (Bi-LSTM) architecture

3. Method

Data collection and labeling constituted the preliminary phases of this classification study. The social media data that were extracted were meticulously labeled, and an expert in the corresponding specific domain verified the manual labeling. The datasets were then pre-processed intransitively, wherein spelling mistakes were corrected, missing points and noise were removed, and both feature extraction and dimensionality reduction were performed. After that, the dataset was divided into 70:30 training and testing subsets. Supervised learning is utilized to train and assess this model. The efficiency of the trained model was then tested on the assessment information and compared with the actual value. The overall process of the proposed system is illustrated in Fig. 3.

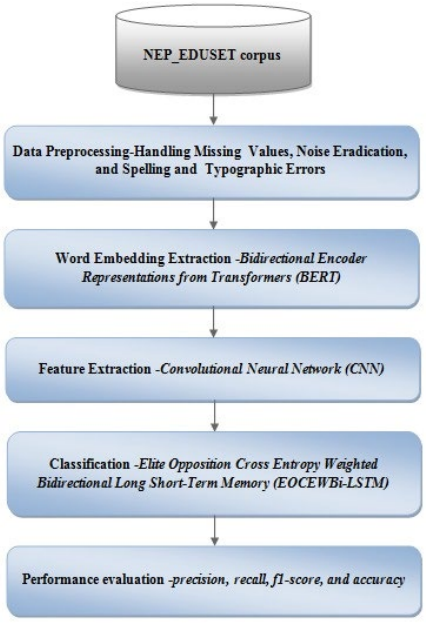

Fig. 3. The overall process of the proposed system

3.1. Data Pre-processing

The dataset used in this study, NEP_EDUSET, consists of 45,434 tweets in English, Hindi, and Tamil, representing both high- and low-resource languages. Pre-processing is crucial for ensuring the consistency and quality of the input data. The following steps were applied.

3.1.1. Data Preparation

This step is crucial in ML classification because the model's accuracy largely relies on the input fed to it.

3.1.2. Handling Missing Values

The first stage in the data processing step involved handling the missing values present in the dataset. The reason was either completely discarding the data points or low data availability. If a sample did not contain the required information, we discarded the entire row. In cases corresponding to the lack of information to some extent, the missing value was replaced with a string from a similar observation. This approach maintains the dataset in its original and holistic form for further analysis.

3.1.3. Noise Eradication

Subsequently, after imputing missing values, noise was removed from the dataset. Noise refers to text or characters from a different language, unrelated special symbols, and emoticons, all of which lack relevance to the language context. While emoticons express a wide range of emotions, their limited appearance in the dataset justified excluding them from the data cleaning process. During this noise-reduction phase, punctuation, numbers, emoticons, and stopwords were removed. This process was crucial in enhancing the quality and relevance of the data, thereby facilitating more effective subsequent analysis.

3.1.4. Spelling and Typographic Errors

Some terms in the text data, taken from various users, are mistyped or misspelled, which is present in the dataset used. The AD (Available Dictionary) database was used to verify the building possibilities of each word and correct those errors. The sentiment corpus, represented a $sSC = d_1, d_2, d_3, \ldots, d_n$ where $d_1, d_2, d_3, \ldots, d_n$ denotes the individual text entries, was scrutinized for spelling accuracy.This is done using a specified sentiment corpus, known as, for verifying the spelling of every text input (texts). Those words found in the corpus but not in the AD were marked as misspelled. Thereafter, the appropriate term was obtained from the AD and utilized to log in to the inappropriate term in the data collection. The quality of textual data is ensured at this stage for further processing.

3.2. Feature Extraction

The methodology proposed by Misra utilizes feature extraction through word embedding to represent words contextually [34]. BERT receives tokens for embedding, which a CNN then processes to extract relevant features. The proposed EOCEWBi-LSTM module structures this information sequentially, and a Feed-forward Neural Network (FNN) computes the loss for accurate predictions.

BERT, a bidirectional and unsupervised pre-trained language model, utilizes Transformer architecture to enhance semantic representation [35]. Unlike unidirectional models, BERT interprets text bi-directionally, allowing deeper contextual understanding. Vaswani *et al.* introduced an attention mechanism to track sentence semantics, while Rush highlighted how BERT's encoder layers comprising self-attention and feed-forward layers, process input sequences effectively [36].

BERT employs two key techniques: Next-sentence prediction (NSP) and Masked Language Modeling (MLM). MLM randomly masks 15% of tokens, replacing 80% with [MASK], 10% with random tokens, and leaving 10% unchanged. This allows the model to learn contextual relationships. NSP helps identify whether a given sentence follows another in a sequence, improving performance across tasks.

The input undergoes tokenization, numericalization, and embedding. Tokenization assigns unique numbers, and sequences are padded for uniformity. During encoding, the input matrix takes dimensions (Input length) × (Embedding dimension) [33]. Multi-head attention enables multiple weighted computations to be performed in parallel, producing a concatenated matrix of dimensions Input_Length × (h * d_v). A final linear layer refines the output to Input_Length × Embedding_Dimension, ensuring optimized representations for NLP tasks. In mathematical terms, Equation (7).

$$\text{Multihead}(A, B, C) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)w^0 \quad (7)$$

where $\text{head}_i = \text{Attention}(Aw_i^A, Bw_i^B, Cw_i^C)$ and A, B and C are placeholder for various input matrix. Every head corresponds to three distinct projections (matrix multiplication) identified by the matrix present in a scaled Dot-Product mechanism.$w_i^B$with the dimension $d_{emb\_dim} \times d_B$, $w_i^A$with the dimension $d_{emb\_dim} \times d_A$,$w_i^C$with the dimensions $d_{emb\_dim} \times d_C$. Each head estimate, therefore, projects the input matrix X through its respective weight matrix. Then, the resultant matrix is as follows as Equation (8).

$XW_i^B = B_i$ with the dimension input_length×dB

$XW_i^A = A_i$ with the dimension input_length ×dA

$XW_i^C = C_i$ with the dimension input_length ×dC

A_i , B_i and C_i to calculate the scaled dot product attentions,

$$\text{Attention}(A, B, C) = \text{softmax}\left(\frac{AB^T}{\sqrt{d_k}}\right)C \quad (8)$$

These different elements could include the query sequence (A), key-value pairs (B and C), and so on. The dot products of this projection is used to measure the similarity between token projections. Where m_i and n_j are the ith and jth token's projection via $B_i$and A_i. Hence, the dot-product is represented in Equation (3).

$$m_i . n_j = \cos(m_i, n_j)\|m_i\|^2\|n_j\|^2 \quad (9)$$

It depicts the relationship between m_i and n_j. The resulting matrix is divided by the square root of d_k for scaling and partitioning into elements. Softmax is applied to each row, ensuring values range between 0 and 1, summing to 1 in above Equation (9). Finally, $V_i$multiplies this value to determine the head [37]. The text sequence E is split into token embeddings, segment embeddings, and position embeddings. Adding vectors of the same dimension produces the text sequence T for model comprehension. BERT is built on the transformer's bidirectional language model [38], [39], using masked LM to predict missing words based on context. The trained word vector Tn and transformer structure Trm process input data. Token embedding divides words into unique tokens, adding special symbols ([CLS], [SEP]) to mark sentence boundaries. Segment embedding differentiates sentences, while position embedding represents word locations [37], as shown in Fig. 4.

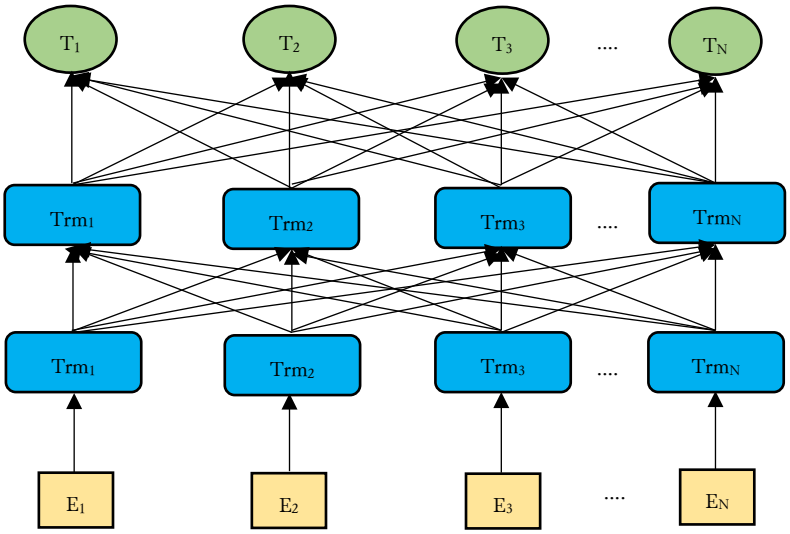

Fig. 4. BERT model structure diagram

3.3. Convolutional Neural Network (CNN)

CNN is a type of deep feed-forward artificial neural networks. Generally, a CNN consists of one or more convolution layers with associated weights and pooling layer followed by a fully connected layer. It is used to extract the features of local correlation from local information [40].

3.3.1. Convolution Layer

A kernel is used in the convolution layer to calculate a dot product (or convolutions) of every sliding window of the input text corpus, followed by an addition of a bias, which is fed forward through an activation function to form feature maps on the next layer. Suppose the input vectors for text samples is $x_i^0 = [x_1, x_2, \ldots, x_n]$, where n is the number of text samples. The output value is then computed utilizing Equation (10).

$$C_i^{l,j} = h\left(b_j + \sum_{m=1}^{M} w_m^j x_{i+m-1}^{0j}\right) \tag{10}$$

Here l is the layer index, h is the activation function that introduces non-linearity to this layer, and b is the bias terms for the j feature maps. M is the kernel/filter size, w is the weights for the jth feature maps and m filter index.

3.3.2. Batch Normalization

Intuitively, data is collected in batches for training. Due to the need to match the batch distributions with the network parameters during each training cycle, the model's convergence is significantly delayed because the distributions are not uniform and unstable. The concept of batch normalization is based on computing the μ_D and variance σ_D^2 of every mini-batch of training data and normalizing the actual data to zero-mean and unity-variance. Furthermore, the shifted data is fed to it with weights and bias that enhance its expressive powers. The equations (11) - (14) present the calculations. The change to a batch normalization approach makes it much easier to coordinate updates across the layers of the neural network.

$$\mu_D = \frac{1}{m}\left(\sum_{i=1}^{m} x_i\right) \tag{11}$$

$$\sigma_D^2 = \frac{1}{m}\sum_{i=1}^{m}(x_i - \mu_D)^2 \tag{12}$$

$$\hat{x}_l = \frac{x_i - \mu_D}{\sqrt{\mu_D^2 + \epsilon}} \tag{13}$$

$$y_i = \gamma \hat{x}_l + \beta \tag{14}$$

3.3.3. Max Pooling Layer

The pooling layer is also known as a subsampling layer. The max-pooling approach is employed in this study to obtain the maximum value among neighboring inputs. Equation (15) defines the pooling of a feature map in a layer.

$$p_i^{l,j} = \max_{r \in R} {}_{i \times T + r}^{l,j} \tag{15}$$

where R is the pooling window size, and T is the pooling stride.

3.4. Elite Opposition Cross Entropy Weighted Bidirectional Long Short-Term Memory (EOCEWBi-LSTM)

The core innovation of this study lies in the design of the Elite Opposition Cross Entropy Weighted Bi-LSTM (EOCEWBi-LSTM) classifier, which combines the sequential modeling capabilities of Bi-LSTM with elite opposition-based learning (EOBL) for dynamic hyperparameter optimization. In the proposed work, if the error rate of the algorithm is higher due to the random selection of weights and bias in the classifier, it results in incorrect detection of MSA. Thus, the hyperparameters of the Bi-

LSTM classifier are tuned using elite opposition-based learning (EOBL). The weighted cross-entropy function weights the negative class by the same weight. The positive class is weighted by a factor of 1.Given the current values x_t, the previous hidden states h_{t-1} and the previous states C_{t-1}, the subsequent changes are employed.

$$f_t = \sigma\left(\widetilde{EWe}_f . [h_{t-1}, x_t] + \widetilde{Eb}_f\right) \quad (16)$$

$$i_t = \sigma\left(\widetilde{EWe}_i . [h_{t-1}, x_t] + \widetilde{Eb}_i\right) \quad (17)$$

$$\hat{C}_t = CE\left(\widetilde{EWe}_c . [h_{t-1}, x_t] + \widetilde{Eb}_c\right) \quad (18)$$

$$C_t = f_t * C_{t-1} + i_t * \hat{C}_t \quad (19)$$

$$o_t = \sigma\left(\widetilde{EWe}_o . [h_{t-1}, x_t] + \widetilde{Eb}_o\right) \quad (20)$$

$$h_t = o_t * CE(C_t) \quad (21)$$

$\widetilde{EWe}_i, \widetilde{EWe}_f, \widetilde{EWe}_c, \widetilde{EWe}_o$is the elite opposition weight matrix, according to the input gate, forget gate, cell gate, and output gate.$\widetilde{Eb}_i, \widetilde{Eb}_f, \widetilde{Eb}_c, \widetilde{Eb}_o$is the elite opposition bias vector according to input gate, forgot gate, cell gate, and output gate in Equation (16) – (21). Finally, the sentiment analysis detection is performed by Feed-forward Neural Network (FNN). It is characterized by a one-directional flow of data between its layers. The term "feed-forward" refers to a model where information flows in one direction, with no cycles or loops; the data moves only forward from the input nodes to the output nodes through the hidden nodes.

3.4.1. EOBL

The concept of EOBL involves generating opposites of the current particles located within the dimension of the best weight values of each feature vector based on the elite individual's current weights and bias. As a result, they will pull those particles to specific areas in the end, which is a better area for them to find a suitable label. Thus, the EOBL method would achieve a better diversity of weights bias and further promote the exploration of Bi-LSTM. For the weight and bias $X_k = (x_{k1}, x_{k2}, \dots, x_{kD})$ in the current population $X_i = (x_{i1}, x_{i2}, \dots, x_{iD})$; thus, the elite opposite location will be $\tilde{X}_k = (\tilde{x}_{k1}, \tilde{x}_{k2}, \dots, \tilde{x}_{kD})$ formulated as Equation (22).

$$\tilde{x}_{k,j} = F\times \left(dy_j + dz_j\right) - x_{k,j} \quad (22)$$

where $F \in [0, 1]$ and F is a generalization factor. $dy_j$and dz_j are dynamic boundaries and can be articulated as in Equation (23)

$$dy_j = \min\left(x_{k,j}\right), dz_j = \max(x_{k,j}) \quad (23)$$

However, the resulting opposite can exceed the search boundary $[y_k, z_k]$. Random values are assigned to the relocated weight, and bias in $[y_k, z_k]$, as in Equation (24),

$$\tilde{x}_{k,j} = \text{rand}\left(y_j + z_j\right), \text{if}\tilde{x}_{k,j} < y_j \| \tilde{x}_{k,j} z_j \quad (24)$$

Elite Opposition-Based Learning (EOBL) is a metaheuristic optimization technique inspired by the principle of simultaneously considering a candidate solution and its opposite to accelerate convergence toward optimal values. In our proposed framework, EOBL is applied to optimize the hyperparameters of the Bi-LSTM classifier, such as learning rate, number of hidden units, dropout rate, and batch size. Instead of relying on conventional manual tuning or computationally expensive grid/random search methods, EOBL dynamically explores the search space more efficiently by generating elite candidates and their opposites, thereby increasing the likelihood of escaping local optima. The novelty of using EOBL in our work lies in its integration with a weighted cross-entropy loss function, forming the

EOCEWBi-LSTM framework. This combination not only enhances classification accuracy but also effectively addresses class imbalance in multilingual sentiment datasets. To the best of our knowledge, this is the first application of EOBL in optimizing Bi-LSTM parameters for multilingual sentiment classification, especially in low-resource language settings.

3.4.2. Cross-Entropy Function

From this best weight is attained which gives lower bias when compared to previous work. The tuned weight value is validated by weighted cross-entropy function expressed as Equation (25),

$$H = -\sum_{i=1}^{N} Weac_i \log(\widehat{pr}_i) + (1 - ac_i) \log(\widehat{pr}_i) \quad (25)$$

where N is the number of text instances, ac_i is the actual label, and $pr_i$is the prediction for text data i. We is the weight generated by the EOBL. If We is equal or less than 1, the exact sentiment is detected; if We is smaller than 0.5, bias increases, leading to incorrect sentiment detection. The proposed AMSD classification method determines sentiment based on context using DL, specifically the BERT model a pre-trained, bidirectional, unsupervised language representation method. A multilingual BERT model is trained on a large corpus of multilingual tweets. The obtained word embeddings pass through two deep architecture layers: CNN and EOCEWBi-LSTM (Fig. 5). After intermediate output from the second layer, a feature vector is formed by concatenating output neurons for each word. These vectors are then fed into a densely connected neural network to reduce dimensions. The final reduced vector is classified using the softmax function. Finally, the EOCEWBi-LSTM classifier is introduced for AMSD, detailed in Algorithm 1.

Algorithm 1: EOCEWBi-LSTM
Input: Dataset D containing n samples, each with m features.
Output: Predicted sentiments for the dataset D.
Step 1: Data Pre-processing
- **Tokenization:** Convert raw text data into tokens.
- **Padding/Truncation:** Normalize sequences to a fixed length L.

Step 2: Embedding Layer
- **Embeddings:** Transform tokens into dense vectors using a pre-trained model like BERT.

Step 3: Bidirectional LSTM Layer
- **Bidirectional Processing:** Process embedded vectors using a bidirectional LSTM layer.

Step 4: Attention and Gating Mechanisms
- **Attention Weights:** Compute attention weights using a dense layer with a softmax function.
- **Gating:** Apply sigmoid gating on LSTM outputs to control information flow.

Step 5: Feature Fusion
- **Concatenation:** Concatenate attention outputs and gated outputs.
- **Dense Layer:** Pass concatenated features through a dense layer for classification.

Step 6: Output
- **Softmax Activation:** Apply a softmax layer to get probabilities of each class.

Prediction: Derive final predictions from the softmax probabilities

The input layer receives sequences of words from the dataset. Each word in a sentence or text input (Word1 to Word N) is processed individually. It acts as the gateway for feeding textual data into the model. BERT comprises multiple encoder layers (E0 to E10, EA, EB) that refine word vectors using transformer blocks, attending to different input parts. It includes:

- Token embeddings: Converts words into vectors using BERT's pre-trained embeddings, capturing context and meaning.
- Segment embeddings: Differentiates text segments, useful for tasks like question answering.
- Position embeddings: Encodes word positions to help BERT understand word order and structure

3.4.3. EOCEWBi-LSTM layer

Combines Bi-LSTM with EOBL. The Bi-LSTM processes data in both forward and reverse directions, capturing dependencies from both ends of the sequence, which is critical for understanding the overall context and sentiment of the text. EOBL technique is integrated to enhance the parameter tuning process. By considering opposing or elite solutions during the training, EOBL helps in escaping local optima, improving the generalization of the model.

3.4.4. Activation layer

Introduces non-linearities into the model, which are essential for learning complex patterns in the data.

3.4.5. Concatenation layer

Merges features from different sources (like outputs from different LSTM cells or different feature maps from CNNs), combining all learned features into a single vector that can be used for final classification.

3.4.6. Dense and Sigmoid layers

1) Dense layer: Fully connected, using concatenated features for final classification. 2) Sigmoid layer: Converts logits into probabilities, useful for binary classification.

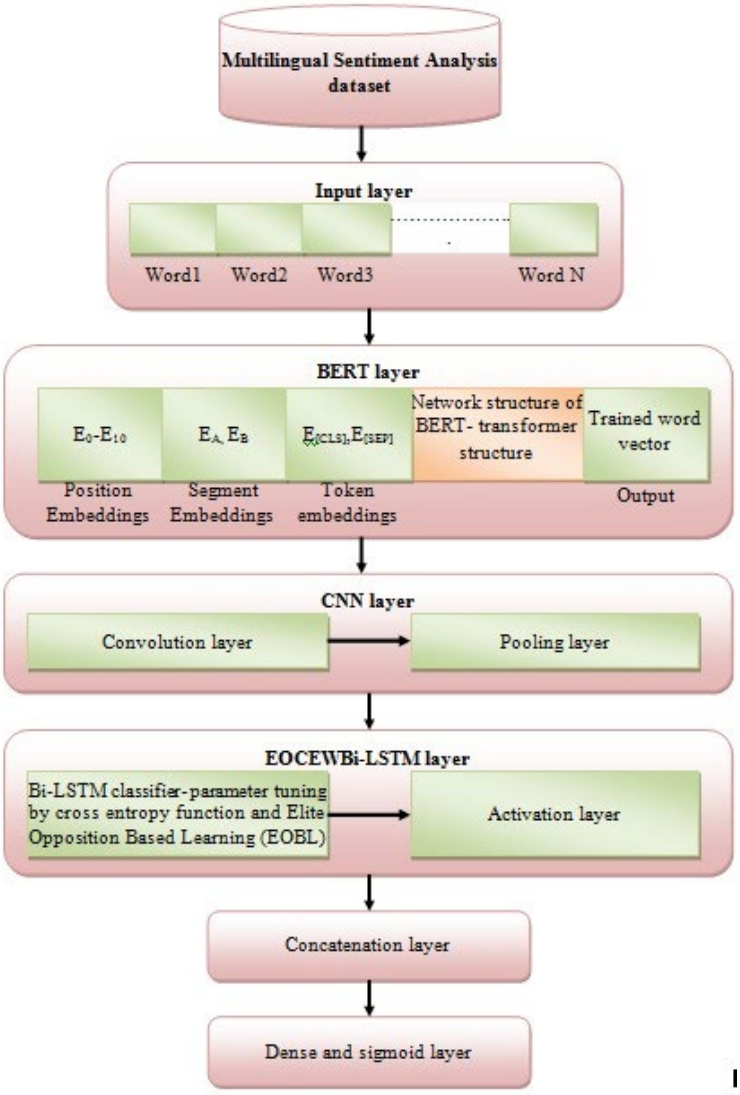

Fig. 5. Flow diagram of proposed EOCEWBi-LSTM method

4. Results and Discussion

The results, which have been fully implemented in Python 3.9.13, are discussed in this section. Google Collaboratory was used for GPU support, as the dataset was significantly larger than typical, and to deploy various DL models. Sci-Kit Learn and Keras (with a TensorFlow backend) were utilized for the ML and DNN models. To provide an FT model with comparable evaluation metrics, the training set, crucial Hyperparameter, and CE are adjusted using the elite opposition. Performance is improved by the suggested architecture's removal of data limitations and the aggregation of weights. Adjusting the Bi-LSTM classifier's parameters improves the outcomes when the classification method is fine-tuned with BERT, as suggested by the method.

4.1. Dataset

Initially, MSA in the tweet dataset contains tweets generated from the real time. Data gathered for the corpus, named NEP_EDUSET, with a total of 45,434 tweets in English, Hindi and Tamil languages was extracted mainly from the social networking Twitter platform (now named X), via Twitter comments. The languages are combined into new form like Tamil+English, Tamil+Hindi, Hindi+English and Tamil+Hindi+English. The dataset was split into training (70%), validation (15%), and testing (15%) subsets to ensure comprehensive evaluation. The dataset's training, development or validation, corresponding to 31804, 6815, and 6815 tweets respectively. The tweets are labeled into three sentiment classes: positive, negative, and neutral using BERT label generator.

While EOCEWBi-LSTM demonstrates strong classification performance, it incurs a higher computational cost compared to simpler models. We evaluated the training efficiency of EOCEWBi-LSTM and compared it with standard Bi-LSTM and CNN models on a system equipped with an NVIDIA RTX 3090 GPU and 64GB RAM. EOCEWBi-LSTM required approximately 1.8× more training time than the baseline Bi-LSTM model, primarily due to the additional BERT embedding layer and elite opposition-based hyperparameter optimization. Additionally, GPU memory usage increased by roughly 1.5×, owing to the larger parameter space and complex training dynamics.

4.2. Performance Evaluation Metrics

Models were assessed using measures such as accuracy, precision, recall, and F1-score based on the test set.

4.2.1. Precision

The number of total positive predictions that are accurate is known as precision. This statistic is calculated by dividing the total number of predicted positives by the total number of classified positives. A well-performing model should have a high degree of precision. The definition of precision is expressed as equation (26).

$$\text{Precision} = \frac{\sum TP}{\sum(TP+FP)} \tag{26}$$

4.2.2. Recall

The number of accurately predicted courses with positive results, or the ratio of correctly identified positively classified classes to all positively classified classes, is known as the recall. A high recall rate is a sign of a solid model. The definition of recall is expressed as equation (27).

$$\text{Recall} = \frac{\sum TP}{\sum(TP+FN)} \tag{27}$$

4.2.3. F1-score

Because the F1-score includes information regarding P and R, a high score suggests high P and R. It can be defined as follows in Equation (28),

$$F1 - Score = \frac{\sum 2TP}{\sum(2TP+FP+FN)} \tag{28}$$

4.2.4. Accuracy

The ratio of correctly predicted instances to all instances is known as accuracy. The most popular statistic for classification tasks is accuracy. It is expressed as equation (29),

$$ACC = \frac{\sum TP+TN}{\sum(TP+TN+FP+FN)} \quad (29)$$

TP, TN, FP, and FN represent True Positives, True Negatives, False Positives, and False Negatives. As seen in Table 1, the CM is a 2x2 matrix that lists the ratio of accurate and inaccurate samples that a classifier predicted.

Table 1. Confusion Matrix (CM)

		Actual class	
		True	***False***
Predicted class	True	TP(True Positive)	FP (False Positive)
	False	FN(False Negative)	TN(True Negative)

The number of positive samples that the classifier accurately predicts as positive is denoted by TP. FP is a measure of the number of negative samples the classifier incorrectly predicts as positive. The number of negative cases that FN denotes the classifier mispredicted. TN denotes the classifier's accurate prediction of the number of negative samples. Table 2 compares classifier performance across unilingual, bilingual, and multilingual settings using evaluation metrics such as P, R, F1-score, and Accuracy. The analysis of ML and DL models highlights their distinct performance characteristics.

Table 2. Classifiers Comparison for NEP_EDUSET dataset

Classifiers	**Tamil**			
	Precision	***Recall***	***F1-Score***	***Accuracy***
RF	0.72441	0.73295	0.72866	0.72868
XGBOOST	0.75757	0.75972	0.75865	0.75865
LGBM	0.76216	0.78860	0.77516	0.77538
CNN	0.82582	0.83142	0.82862	0.82862
LSTM	0.85473	0.85761	0.85617	0.85617
Bi-LSTM	0.86169	0.86675	0.86422	0.86422
WBi-LSTM	0.87851	0.88418	0.87821	0.88156
EOWBi-LSTM	0.89256	0.90244	0.89854	0.89565
EOCEWBi-LSTM	0.91623	0.92319	0.91972	0.91971
Classifiers	**English**			
	Precision	***Recall***	***F1-Score***	***Accuracy***
RF	0.72305	0.74016	0.73150	0.73160
XGBOOST	0.75045	0.76699	0.75863	0.75872
LGBM	0.77839	0.78126	0.77982	0.77983
CNN	0.82692	0.83415	0.83052	0.83053
LSTM	0.85633	0.85411	0.85521	0.85522
Bi-LSTM	0.86380	0.85465	0.85920	0.85923
WBi-LSTM	0.87251	0.86874	0.87147	0.87145
EOWBi-LSTM	0.89574	0.88861	0.89361	0.89266
EOCEWBi-LSTM	0.91406	0.93955	0.92663	0.92680
Classifiers	**Hindi**			
	Precision	***Recall***	***F1-Score***	***Accuracy***
RF	0.73887	0.74182	0.74034	0.74035
XGBOOST	0.75980	0.76618	0.76298	0.76299
LGBM	0.76055	0.78257	0.77140	0.77156
CNN	0.82736	0.81276	0.82000	0.82006
LSTM	0.85280	0.84278	0.84776	0.84779
Bi-LSTM	0.85954	0.86211	0.85578	0.85583
WBi-LSTM	0.86454	0.87451	0.87154	0.87892
EOWBi-LSTM	0.88475	0.89418	0.89475	0.89789
EOCEWBi-LSTM	0.91906	0.92989	0.92444	0.92447

Table 2. (cont…)

Classifiers	Tamil+English			
	Precision	*Recall*	*F1-Score*	*Accuracy*
RF	0.72776	0.73208	0.72991	0.72992
XGBOOST	0.74745	0.75616	0.75178	0.75181
LGBM	0.77362	0.77977	0.77668	0.77669
CNN	0.82516	0.83773	0.83140	0.83144
LSTM	0.85228	0.86324	0.85772	0.85776
Bi-LSTM	0.85540	0.86508	0.86021	0.85824
WBi-LSTM	0.87441	0.88353	0.87985	0.87542
EOWBi-LSTM	0.90143	0.92145	0.91256	0.90541
EOCEWBi-LSTM	0.93186	0.94480	0.93828	0.93833
Classifiers	**Tamil+Hindi**			
	Precision	*Recall*	*F1-Score*	*Accuracy*
RF	0.73746	0.73654	0.73700	0.73700
XGBOOST	0.74264	0.75014	0.74637	0.74639
LGBM	0.77399	0.78571	0.77981	0.77985
CNN	0.82547	0.83435	0.82765	0.82971
LSTM	0.85143	0.83510	0.84318	0.84326
Bi-LSTM	0.85740	0.86508	0.86122	0.85824
WBi-LSTM	0.86925	0.88152	0.87841	0.88545
EOWBi-LSTM	0.89546	0.91257	0.89715	0.91547
EOCEWBi-LSTM	0.93261	0.93367	0.93314	0.93314
Classifiers	**Hindi+English**			
	Precision	*Recall*	*F1-Score*	*Accuracy*
RF	0.72380	0.74021	0.73191	0.73200
XGBOOST	0.75065	0.75497	0.75280	0.75281
LGBM	0.76950	0.77518	0.77233	0.772346
CNN	0.82843	0.83100	0.82971	0.82971
LSTM	0.85377	0.84356	0.84863	0.84866
Bi-LSTM	0.85016	0.86833	0.85915	0.85924
WBi-LSTM	0.86115	0.87569	0.87688	0.87581
EOWBi-LSTM	0.87569	0.88735	0.90253	0.90458
EOCEWBi-LSTM	0.93281	0.93394	0.93366	0.93385
Classifiers	**Tamil+Hindi+English**			
	Precision	*Recall*	*F1-Score*	*Accuracy*
RF	0.73344	0.74128	0.73734	0.73736
XGBOOST	0.75650	0.76141	0.75895	0.75895
LGBM	0.76106	0.77162	0.76630	0.76634
CNN	0.82775	0.81812	0.82291	0.82294
LSTM	0.85143	0.83510	0.84318	0.84326
Bi-LSTM	0.85984	0.85126	0.85553	0.85555
WBi-LSTM	0.88254	0.87554	0.87821	0.87541
EOWBi-LSTM	0.89369	0.90251	0.89922	0.89364
EOCEWBi-LSTM	0.91193	0.92259	0.91723	0.91726

Among them, EOCEWBi-LSTM excels across all metrics, with detailed examination and interpretation provided. Traditional ML models, including ensemble methods like RF and gradient boosting (XGBoost and LGBM), showed moderate performance. LGBM led this group, achieving balanced precision and recall (0.773), which ensured consistent case identification. CNN outperformed traditional models, particularly in precision (0.829), minimizing false positives and enhancing feature extraction. LSTM and BiLSTM further improved upon CNN, effectively capturing temporal dependencies. BiLSTM, with a precision of 0.862, demonstrated superior contextual understanding, reducing false positives. The consistent performance across different language combinations demonstrates the proposed model's robustness. The highest accuracy is achieved for Tamil + English (93.83%), showcasing the model's ability to handle code-mixed data effectively.

Fig.6 compares classifier performance for Tamil using P, R, F1-score, and Acc. The EOCEWBi-LSTM model outperformed all, achieving the highest precision (0.91623), recall (0.92319), F1-score (0.91972), and accuracy (0.933997). Results indicate a performance boost with increasing model complexity, as DL classifiers surpass traditional ML models across all metrics. While RF, XGBoost, and LGBM provide strong baselines, CNN, LSTM, and Bi-LSTM exhibit superior pattern recognition in Tamil text. EOCEWBi-LSTM excels by effectively capturing contextual and linguistic complexities, making it highly reliable for applications like SA and hate speech detection. Despite computational costs, DL, especially EOCEWBi-LSTM, proves optimal for Tamil text interpretation.

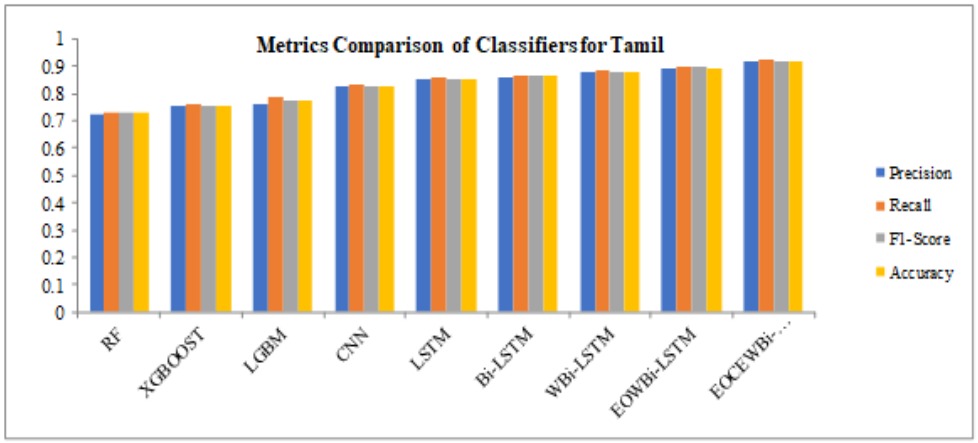

Fig. 6. Metrics comparison of classifiers for Tamil language

Fig. 7 compares classifier performance in English using P, R, F1-score, and Acc. EOCEWBi-LSTM outperformed all, achieving the highest precision (0.91406), recall (0.93955), F1-score (0.92663), and accuracy (0.92680). The results show classifier efficacy improves with model complexity. While RF, XGBoost, and LGBM perform well, DL models consistently achieve superior results. CNN surpasses traditional models by identifying patterns, while LSTM and Bi-LSTM further enhance sequence comprehension. Advanced models like Bi-LSTM, EOWBi-LSTM, and EOCEWBi-LSTM improve performance, with EOCEWBi-LSTM excelling overall. Its architecture enhances material understanding, making it highly effective for English text analysis, ensuring precise and reliable outcomes.

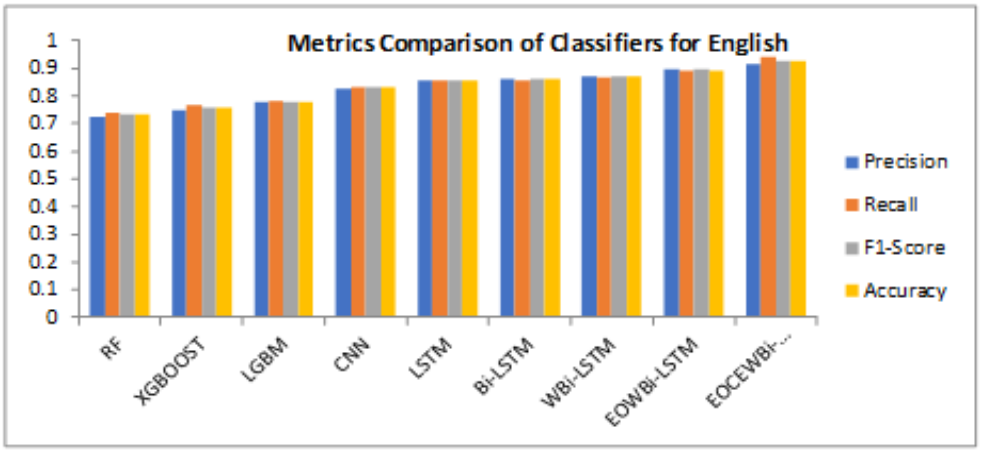

Fig. 7. Metrics comparison of classifiers for English language

A comparison of classifiers using evaluation metrics such as P, R, f1-score, and accuracy for the Hindi language is illustrated in Fig. 8. The EOCEWBi-LSTM model consistently achieved higher scores than all others, attaining the highest scores in precision (0.91906), recall (0.92989), f1-score (0.92444), and accuracy (0.92447). The efficacy of classifiers in Hindi text analysis, demonstrating a marked

enhancement as models evolve. Conventional models such as RF, XGBOOST, and LightGBM (LGBM) deliver dependable baseline performance, with LGBM exhibiting marginally superior results. DL approaches surpass these, with CNNs efficiently capturing textual patterns, while sequential models such as LSTM and Bi-LSTM enhance outcomes by learning word associations. Advanced models such as WBi-LSTM, EOWBi-LSTM, and EOCEWBi-LSTM demonstrate increasingly superior performance, with EOCEWBi-LSTM attaining the highest scores across all criteria. The results demonstrate the effectiveness of DL models, specifically EOCEWBi-LSTM, in achieving precise and equitable outcomes for Hindi text analysis.

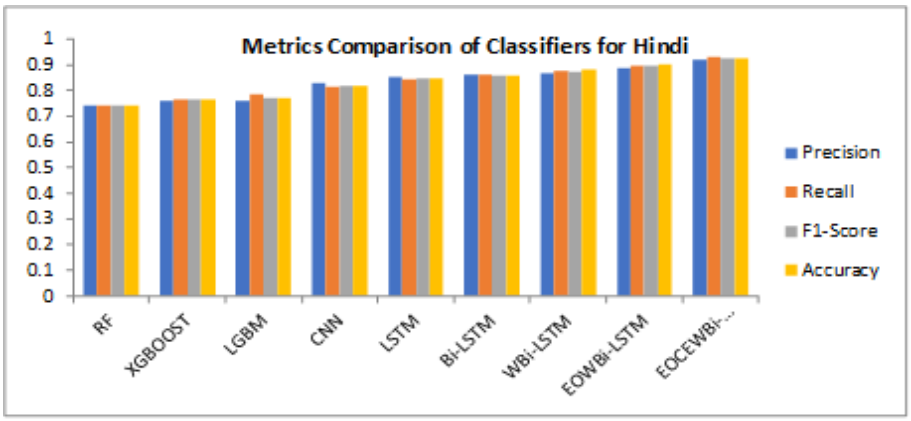

Fig. 8. Metrics comparison of classifiers for Hindi language

Evaluation metrics like P, R, f1-score and Acc with respect to Tamil+English languages among classifiers are illustrated in Fig. 9. The EOCEWBi-LSTM model showed measurable improvement all others, achieving the highest scores in precision (0.93186), recall (0.94480), f1-score (0.93828), and accuracy (0.93833). The graphic illustrates the efficacy of classifiers in analyzing Tamil and English texts, showing enhancements as models progress in sophistication. Conventional models such as RF, XGBOOST, and LGBM deliver robust baseline performance; nevertheless, they are surpassed by DL models. CNN effectively captures bilingual text patterns, while sequential models, such as LSTM and Bi-LSTM, enhance performance by leveraging word associations. Advanced models, such as WBi-LSTM, EOWBi-LSTM, and EOCEWBi-LSTM, demonstrate increasingly superior performance, with EOCEWBi-LSTM yielding the highest results across all measures. The results underscore the efficacy of DL models, especially EOCEWBi-LSTM, in providing precise and equitable analysis for bilingual text data.

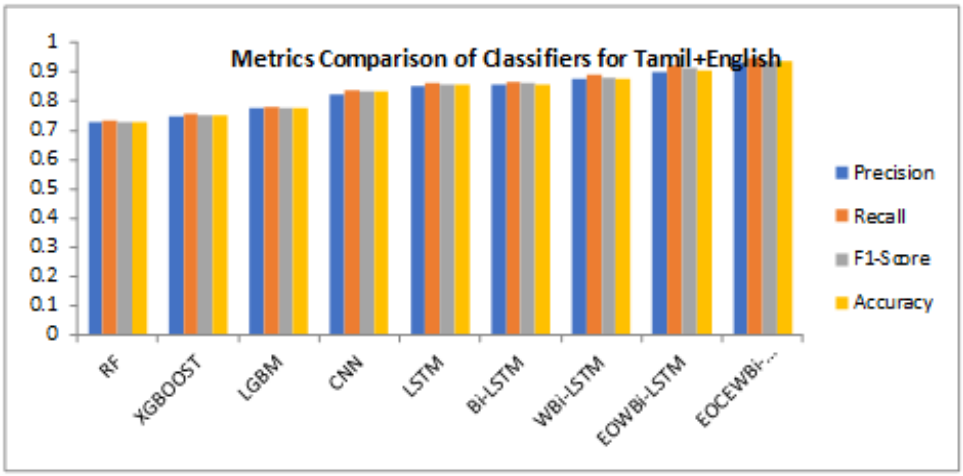

Fig. 9. Metrics comparison of classifiers for Tamil+English languages

Values of P, R, f1-score and Acc with respect to Tamil+Hindi languages among classifiers are shown in Fig. 10. The EOCEWBi-LSTM model achieved increased accuracy, especially in low-resource

settings, all others, achieving the highest scores in precision (0.93261), recall (0.93367), f1-score (0.93314), and accuracy (0.93314). The illustration demonstrates a notable enhancement in performance when classifiers evolve for Tamil and Hindi text analysis. Conventional models such as RF, XGBOOST, and LGBM yield robust baseline outcomes but are frequently surpassed by DL models. CNN exhibits superior pattern recognition, whereas sequential models, such as LSTM and Bi-LSTM, enhance performance by capturing word dependencies. Advanced models, such as WBi-LSTM, EOWBi-LSTM, and EOCEWBi-LSTM, achieve increasingly superior scores, with EOCEWBi-LSTM yielding the highest performance in terms of Precision, Recall, F1-Score, and Accuracy. The results underscore the efficacy of DL models, especially EOCEWBi-LSTM, in delivering precise and equitable performance for bilingual text analysis.

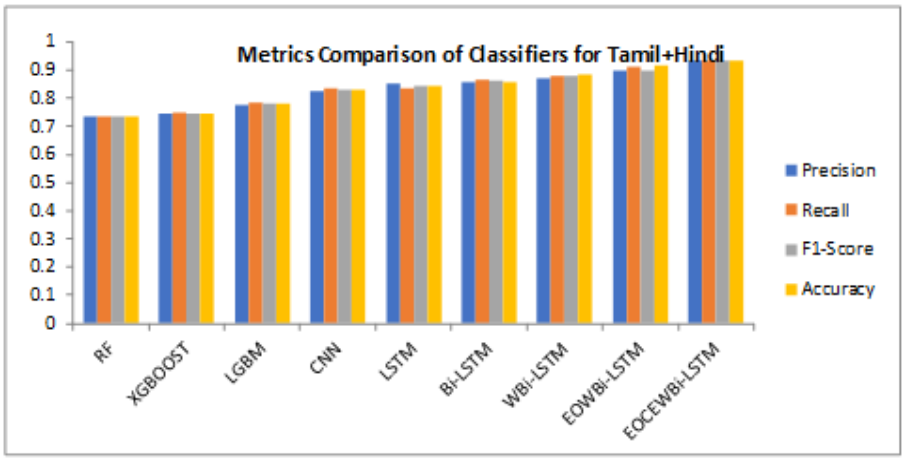

Fig. 10. Metrics comparison of classifiers for Tamil+Hindi languages

Classifiers' performance among evaluation metrics like recall, precision, f1-score, and accuracy concerning Hindi+English languages is illustrated in Fig. 11. The EOCEWBi-LSTM model compared favorably with baseline models across all languages evaluated, achieving the highest scores in precision (0.93281), recall (0.93394), f1-score (0.93366), and accuracy (0.93384). The illustration depicts the efficacy of classifiers in analyzing Hindi and English texts, showing steady enhancements as models evolve. Conventional models, such as RF, XGBoost, and LGBM, yield robust baseline outcomes but are surpassed by DL models. CNN demonstrates superior pattern recognition, whereas sequential models, such as LSTM and Bi-LSTM, enhance performance by capturing word dependencies. Advanced variations, such as WBi-LSTM, EOWBi-LSTM, and EOCEWBi-LSTM, attain ever superior scores, with EOCEWBi-LSTM yielding the optimal outcomes across all criteria. The results underscore the efficacy of DL, specifically EOCEWBi-LSTM, in delivering precise and equitable analysis for bilingual text data.

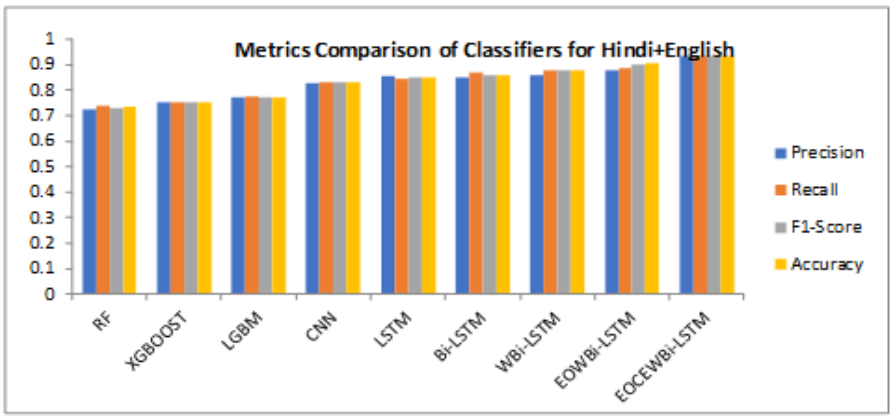

Fig. 11. Metrics comparison of classifiers for Hindi+English languages

The EOCEWBi-LSTM model demonstrated better performance across evaluation metrics than all others, achieving the highest scores in precision (0.91193), recall (0.92259), F1-score (0.91723), and accuracy (0.91726) for multi-language, as shown in Fig. 12.

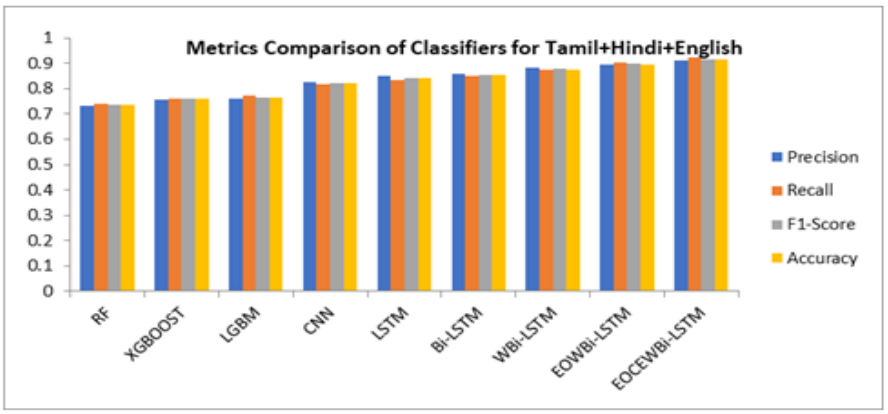

Fig. 12. Metrics comparison of classifiers for Tamil+Hindi+English languages

The figure illustrates the efficacy of classifiers in analyzing Tamil, Hindi, and English texts, indicating a steady enhancement with the advancement of models. Conventional classifiers, such as RF, XGBoost, and LGBM, yield dependable baseline outcomes; however, they are surpassed by DL models. CNN successfully captures multilingual patterns, while sequential models, such as LSTM and Bi-LSTM, enhance performance by learning word dependencies. Advanced versions, including WBi-LSTM, EOWBi-LSTM, and EOCEWBi-LSTM, yield increasingly superior outcomes, with EOCEWBi-LSTM attaining the best scores across all parameters. These findings underscore the preeminence of EOCEWBi-LSTM in delivering precise and equitable outcomes for multilingual text analysis. To validate the performance improvements of the suggested EOCEWBi-LSTM model over baseline methods, we conducted McNemar's test, a non-parametric test commonly used to compare the predictions of two classifiers on the same dataset. The test was applied between EOCEWBi-LSTM and its closest baseline, Bi-LSTM, with BERT embeddings. Results showed a chi-squared value of 6.21 with a corresponding p-value of 0.013, indicating that the observed performance difference is statistically significant at the 95% confidence level.

4.3. Comparison with State-of-the-Art Models

The proposed model was compared against recent SOTA sentiment classification models from the literature. The comparison focuses on models that have demonstrated high efficacy in handling sentiment classification tasks across various datasets. Table 3 summarizes the comparison.

Table 3. Comparison of EOWBi-LSTM with SOTA models

Model	Dataset	F1-Score (%)	Accuracy (%)
mBERT + Fine-Tuning	Twitter	88.0	88.0
Bi-GRU + Attention	Film Reviews	89.0	89.0
Proposed EOCEWBi-LSTM	NEP_EDUSET	93.83	93.83

5. Conclusion

Sentiment analysis (SA) is crucial for understanding public opinion in domains like politics and business. In tweet classification, keywords play a vital role. While BERT has proven effective in MSA, accuracy can be further improved. This paper introduces a three-part technique: Pre-processing, BERT, and EOCEWBi-LSTM. Pre-processing includes checks for missing values, removal of names, whitespaces, hashtags, numbers, punctuation, URLs, and spelling correction. BERT enhances word representation in short texts using dynamic word vectors and a self-attention mechanism. EOCEWBi-LSTM classifies tweets based on sentiment. The analysis demonstrates EOCEWBi-LSTM's superior precision, recall, accuracy, and F1-score, outperforming traditional and DL models in MSA. Compared

to SOTA techniques, the EOCEWBi-LSTM combination excels in performance metrics. Superior handling of class imbalance, particularly in the neutral sentiment class, due to the dynamically weighted cross-entropy loss function. Future research can expand sentiment analysis beyond online data to include emotions such as happiness and surprise. Other transformer models, such as RoBERTa, XLNet, and DistilBERT, warrant further study. EOCEWBi-LSTM's promising results highlight its effectiveness in MSA. The high accuracy and robustness of EOCEWBi-LSTM make it suitable for deployment in several real-world applications: 1) Social Media Sentiment Monitoring: Organizations and governments can use the model to monitor public opinion on global issues, marketing campaigns, or policy decisions; 2) Customer Feedback Analysis: Multinational companies can analyze feedback across multiple languages to understand customer satisfaction and improve services; 3) Crisis Management: The model can provide early warning signals by identifying negative or neutral sentiment during crises, enabling proactive response measures. While the proposed EOCEWBi-LSTM model demonstrates strong performance on the NEP_EDUSET dataset, several limitations warrant further exploration. First, although the dataset includes multilingual content, it is restricted to the social media domain. This raises concerns about the model's domain transferability; its ability to generalize to other text genres, such as news articles, product reviews, or formal discourse, remains untested. A domain adaptation study, measuring performance drop (e.g., F1-score and accuracy variation greater than 10%) across new domains, is necessary to assess the model's robustness. Second, the model operates as a black box, and its lack of interpretability poses a challenge for use in critical domains such as healthcare, policy analysis, or legal reviews. Future work should incorporate interpretability mechanisms, such as attention weight visualization or post-hoc explanation techniques like SHAP (SHapley Additive exPlanations) or LIME (Local Interpretable Model-agnostic Explanations), to provide transparency and gain user trust. Third, the model has been evaluated on only two low-resource languages Hindi and Tamil. To validate its linguistic scalability, it is essential to extend testing to other typologically diverse low-resource languages (e.g., Amharic, Marathi, or Burmese), aiming for consistent performance across at least five such languages with a performance deviation of ≤5%. Additionally, future enhancements may include integrating advanced semantic analysis through external knowledge graphs or transformer-based semantic parsers. The inclusion of transfer learning strategies, such as domain-adaptive pretraining, could further boost generalization to unseen data. Finally, we plan to evaluate the model's scalability on significantly larger datasets (e.g., more than 250,000 instances) to ensure performance stability and computational efficiency.

Declarations

Author contribution. All authors (Mohammad Mustafa Siddique [MMS]; Sandeep Kumar [SK]) contributed significantly to this study. [Abstract: MMS; Introduction: MMS and SK; Literature Review: MMS; Proposed Methodology: MMS; Results and Discussion: MMS and SK; Conclusion and Future Work: MMS and SK; Complete Manuscript Review: SK]
Funding statement. No funding was received for this study
Conflict of interest. The authors declare no conflict of interest.
Additional information. No additional information is available for this paper.

References

[1] A. Ghafoor *et al.*, "The Impact of Translating Resource-Rich Datasets to Low-Resource Languages Through Multi-Lingual Text Processing," *IEEE Access*, vol. 9, pp. 124478–124490, 2021, doi: 10.1109/ACCESS.2021.3110285.

[2] X. Ou and H. Li, "YNU@Dravidian-CodeMix-FIRE2020: XLM-RoBERTa for multi-language sentiment analysis," *CEUR Workshop Proc.*, vol. 2826, pp. 560–565, 2020, [Online]. Available at: https://ceur-ws.org/Vol-2826/T4-13.pdf.

[3] 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*, vol. 13, pp. 7538–7554, 2025, doi: 10.1109/ACCESS.2025.3527710.

[4] M. Wankhade, A. C. S. Rao, and C. Kulkarni, "A survey on sentiment analysis methods, applications, and challenges," *Artif. Intell. Rev.*, vol. 55, no. 7, pp. 5731–5780, Oct. 2022, doi: 10.1007/s10462-022-10144-1.

[5] Muhammad Zulqarnain *et al.*, "Text Classification Using Deep Learning Models: A Comparative Review," *Cloud Comput. Data Sci.*, pp. 80–96, Oct. 2023, doi: 10.37256/ccds.5120243528.

[6] O. Habimana, Y. Li, R. Li, X. Gu, and G. Yu, "Sentiment analysis using deep learning approaches: an overview," *Sci. China Inf. Sci.*, vol. 63, no. 1, p. 111102, Jan. 2020, doi: 10.1007/s11432-018-9941-6.

[7] A. Rogers, O. Kovaleva, and A. Rumshisky, "A Primer in BERTology: What We Know About How BERT Works," *Trans. Assoc. Comput. Linguist.*, vol. 8, pp. 842–866, Dec. 2020, doi: 10.1162/tacl_a_00349.

[8] F. Carneiro, D. Vianna, J. Carvalho, A. Plastino, and A. Paes, "BERTweet.BR: a pre-trained language model for tweets in Portuguese," *Neural Comput. Appl.*, vol. 37, no. 6, pp. 4363–4385, Feb. 2025, doi: 10.1007/s00521-024-10711-3.

[9] G. Manias, A. Mavrogiorgou, A. Kiourtis, C. Symvoulidis, and D. Kyriazis, "Multilingual text categorization and sentiment analysis: a comparative analysis of the utilization of multilingual approaches for classifying twitter data," *Neural Comput. Appl.*, vol. 35, no. 29, pp. 21415–21431, Oct. 2023, doi: 10.1007/s00521-023-08629-3.

[10] A. Amrullah, "Advanced Sentiment Analysis Using Deep Learning: A Comprehensive Framework for High-Accuracy and Interpretable Models," *Intellithings J.*, vol. 1, no. 1, pp. 21–31, 2025. [Online]. Available at: https://e-jurnal.unisda.ac.id/index.php/intellithings/article/view/8972.

[11] A. Bello, S.-C. Ng, and M.-F. Leung, "A BERT Framework to Sentiment Analysis of Tweets," *Sensors*, vol. 23, no. 1, p. 506, Jan. 2023, doi: 10.3390/s23010506.

[12] A. S. Talaat, "Sentiment analysis classification system using hybrid BERT models," *J. Big Data*, vol. 10, no. 1, p. 110, Jun. 2023, doi: 10.1186/s40537-023-00781-w.

[13] M. Pota, M. Ventura, H. Fujita, and M. Esposito, "Multilingual evaluation of pre-processing for BERT-based sentiment analysis of tweets," *Expert Syst. Appl.*, vol. 181, p. 115119, Nov. 2021, doi: 10.1016/j.eswa.2021.115119.

[14] T. S. Sai Kumar, K. Arunaggiri Pandian, S. Thabasum Aara, and K. Nagendra Pandian, "A Reliable Technique for Sentiment Analysis on Tweets via Machine Learning and BERT," in *2021 Asian Conference on Innovation in Technology (ASIANCON)*, Aug. 2021, pp. 1–5, doi: 10.1109/ASIANCON51346.2021.9545013.

[15] S. Mann, J. Arora, M. Bhatia, R. Sharma, and R. Taragi, "Twitter Sentiment Analysis Using Enhanced BERT," in *Lecture Notes in Electrical Engineering*, vol. 959, Springer, Singapore, 2023, pp. 263–271, doi: 10.1007/978-981-19-6581-4_21.

[16] N. Azzouza, K. Akli-Astouati, and R. Ibrahim, "TwitterBERT: Framework for Twitter Sentiment Analysis Based on Pre-trained Language Model Representations," in *Advances in Intelligent Systems and Computing*, vol. 1073, Springer, Cham, 2020, pp. 428–437, doi: 10.1007/978-3-030-33582-3_41.

[17] N. J. Prottasha *et al.*, "Transfer Learning for Sentiment Analysis Using BERT Based Supervised Fine-Tuning," *Sensors*, vol. 22, no. 11, p. 4157, May 2022, doi: 10.3390/s22114157.

[18] P. K. Jain, W. Quamer, V. Saravanan, and R. Pamula, "Employing BERT-DCNN with sentic knowledge base for social media sentiment analysis," *J. Ambient Intell. Humaniz. Comput.*, vol. 14, no. 8, pp. 10417–10429, Aug. 2023, doi: 10.1007/s12652-022-03698-z.

[19] M. Anwar Hussen Wadud, M. F. Mridha, J. Shin, K. Nur, and A. Kumar Saha, "Deep-BERT: Transfer Learning for Classifying Multilingual Offensive Texts on Social Media," *Comput. Syst. Sci. Eng.*, vol. 44, no. 2, pp. 1775–1791, Jun. 2023, doi: 10.32604/csse.2023.027841.

[20] H. Cam, A. V. Cam, U. Demirel, and S. Ahmed, "Sentiment analysis of financial Twitter posts on Twitter with the machine learning classifiers," *Heliyon*, vol. 10, no. 1, p. e23784, Jan. 2024, doi: 10.1016/j.heliyon.2023.e23784.

[21] P. Nandhini, R. Karunamoorthi, P. Mariappan, and S. Revathi, "Multilingual Offensive Language Detection In Social Media Content Using BERT-Base-Multilingual-Cased Model," in *2024 15th International Conference on Computing Communication and Networking Technologies (ICCCNT)*, Jun. 2024, pp. 1–6, doi: 10.1109/ICCCNT61001.2024.10726015.

[22] P. Kakati and D. Dandotiya, "Automatic detection of hate speech in code-mixed Indian languages in twitter social media interaction using DConvBLSTM-MuRIL ensemble method," *Soc. Netw. Anal. Min.*, vol. 14, no. 1, p. 108, May 2024, doi: 10.1007/s13278-024-01264-3.

[23] P. K. Roy, "Deep Ensemble Network for Sentiment Analysis in Bi-lingual Low-resource Languages," *ACM Trans. Asian Low-Resource Lang. Inf. Process.*, vol. 23, no. 1, pp. 1–16, Jan. 2024, doi: 10.1145/3600229.

[24] A. G. A. and V. V., "Sentiment analysis on a low-resource language dataset using multimodal representation learning and cross-lingual transfer learning," *Appl. Soft Comput.*, vol. 157, p. 111553, May 2024, doi: 10.1016/j.asoc.2024.111553.

[25] M. R. Hossain, M. M. Hoque, M. A. A. Dewan, E. Hoque, and N. Siddique, "AFuNet: an attention-based fusion network to classify texts in a resource-constrained language," *Neural Comput. Appl.*, vol. 37, no. 9, pp. 6725–6748, Mar. 2025, doi: 10.1007/s00521-024-10953-1.

[26] D. R. Kubal and A. S. Nagvenkar, "Leveraging Multilingual Models for Robust Grammatical Error Correction Across Low-Resource Languages," in *Proceedings of the 31st International Conference on Computational Linguistics: Industry Track*, 2025, pp. 505–510. [Online]. Available at: https://aclanthology.org/2025.coling-industry.43/.

[27] J. Liu *et al.*, "Application of Deep Learning-Based Natural Language Processing in Multilingual Sentiment Analysis," *Mediterr. J. Basic Appl. Sci.*, vol. 08, no. 02, pp. 243–260, 2024, doi: 10.46382/MJBAS.2024.8219.

[28] N. Prova, "Multilingual Emotion Classification in E-Commerce Customer Reviews Using GPT and Deep Learning-Based Meta-Ensemble Model," *SSRN*. p. 19, Mar. 02, 2025, doi: 10.2139/ssrn.5161505.

[29] M. S. U. Miah, M. M. Kabir, T. Bin Sarwar, M. Safran, S. Alfarhood, and M. F. Mridha, "A multimodal approach to cross-lingual sentiment analysis with ensemble of transformer and LLM," *Sci. Rep.*, vol. 14, no. 1, p. 9603, Apr. 2024, doi: 10.1038/s41598-024-60210-7.

[30] R. Geethanjali and A. Valarmathi, "A novel hybrid deep learning IChOA-CNN-LSTM model for modality-enriched and multilingual emotion recognition in social media," *Sci. Rep.*, vol. 14, no. 1, p. 22270, Sep. 2024, doi: 10.1038/s41598-024-73452-2.

[31] V. Dhananjaya, S. Ranathunga, and S. Jayasena, "Lexicon-based fine-tuning of multilingual language models for low-resource language sentiment analysis," *CAAI Trans. Intell. Technol.*, vol. 9, no. 5, pp. 1116–1125, Oct. 2024, doi: 10.1049/cit2.12333.

[32] A. Onan and M. A. Tocoglu, "A Term Weighted Neural Language Model and Stacked Bidirectional LSTM Based Framework for Sarcasm Identification," *IEEE Access*, vol. 9, pp. 7701–7722, 2021, doi: 10.1109/ACCESS.2021.3049734.

[33] Y. Zhang, W. Yang, J. Wang, Q. Ma, and J. Xiong, "CAMEF: Causal-Augmented Multi-Modality Event-Driven Financial Forecasting by Integrating Time Series Patterns and Salient Macroeconomic Announcements," *Proc. Make sure to enter correct Conf. title from your rights confirmation emai (Conference Acron. 'XX)*, vol. 1, pp. 1–12, 2018, [Online]. Available at: https://arxiv.org/pdf/2502.04592.

[34] J. Misra, "autoNLP: NLP Feature Recommendations for Text Analytics Applications," *arxiv Artif. Intell.*, pp. 1–11, 2020, [Online]. Available at: http://arxiv.org/abs/2002.03056.

[35] X. Ding, X. Zhang, C. Liang, B. Liu, and L. Niu, "A Composite Recognition Method Based on Multimode Mutual Attention Fusion Network," *Appl. Artif. Intell.*, vol. 39, no. 1, Dec. 2025, doi: 10.1080/08839514.2025.2462371.

[36] A. Rush, "The Annotated Transformer," in *Proceedings of Workshop for NLP Open Source Software (NLP-OSS)*, Jun. 2018, pp. 52–60, doi: 10.18653/v1/W18-2509.

[37] B. Jlifi, C. Abidi, and C. Duvallet, "Beyond the use of a novel Ensemble based Random Forest-BERT Model (Ens-RF-BERT) for the Sentiment Analysis of the hashtag COVID19 tweets," *Soc. Netw. Anal. Min.*, vol. 14, no. 1, p. 88, Apr. 2024, doi: 10.1007/s13278-024-01240-x.

[38] A. Razaq, Z. Halim, A. Ur Rahman, and K. Sikandar, "Identification of paraphrased text in research articles through improved embeddings and fine-tuned BERT model," *Multimed. Tools Appl.*, vol. 83, no. 30, pp. 74205–74232, Feb. 2024, doi: 10.1007/s11042-024-18359-w.

[39] M. Badrani, A. Marouan, N. Kannouf, and A. Chetouani, "Personalized Guidance for Moroccan Students: An Approach Based on Machine Learning and Big Data," *Int. J. Eng. Pedagog.*, vol. 15, no. 1, pp. 125–136, Jan. 2025, doi: 10.3991/ijep.v15i1.51985.

[40] D. Naik and C. D. Jaidhar, "A novel Multi-Layer Attention Framework for visual description prediction using bidirectional LSTM," *J. Big Data*, vol. 9, no. 1, p. 104, Nov. 2022, doi: 10.1186/s40537-022-00664-6.