

Multi-Level Contextual Semantic Modeling for Implicit Hate Speech Recognition

Rajkumar Panchal^{1*}, Megha Jonnalagedda²

Department of Computer Science and Engineering,

SGGSIE&T, Swami Ramanand Teerth Marathwada University, Nanded, 431606, Maharashtra, INDIA¹

Department of Information Technology,

SGGSIE&T, Swami Ramanand Teerth Marathwada University, Nanded, 431606, Maharashtra, INDIA²

Abstract—Social media platforms are key spaces for sharing opinions, experiences, and daily events. However, the rapid rise in hate speech, especially implicit hate, has become a significant concern. Implicit hate is subtle, context-dependent, and more difficult to detect than explicit hate. Despite the growing body of work on Hate Speech Detection (HSD), most existing models are developed for resource-rich languages like English. Unlike resource-rich languages, Hindi lacks sufficient annotated data and dedicated research, making it a low-resource language. To address this limitation, this work presents a deep learning-driven approach for the classification of implicit and explicit hate speech in Hindi. The approach uses FastText embeddings to capture semantic and subword-level information, which suits noisy social media text. A Bidirectional Long Short-Term Memory (BiLSTM) network models contextual dependencies, while an attention mechanism assigns higher importance to discriminative words and sentences, enabling effective identification of implicit hate. Extensive experiments on the Hindi Hate Speech Dataset (HHSD) demonstrate the model's effectiveness. Compared with the widely used multilingual BERT (mBERT) baseline, the proposed approach improves classification performance, with a 2.51% increase in accuracy and a 17.84% improvement in weighted F1-score. Additionally, the model requires significantly fewer parameters than transformer-based models, making it more parameter-efficient.

Keywords—Attention network; BiLSTM; deep learning; explicit hate speech detection; hate speech detection; implicit hate speech detection; natural language processing

I. INTRODUCTION

Social media has become an integral platform for sharing opinions, daily activities, and connecting with others. While many users engage positively, others disseminate negative content, including hate speech, which can undermine societal harmony. Hate speech encompasses communicative acts expressed through language or behaviour that demean or discriminate against individuals or communities based on attributes such as religion, ethnicity, nationality, race, colour, descent, gender, or other identity markers [1]. To address this issue, governments have introduced regulations that require social media platforms to enforce them. However, to bypass these restrictions, users increasingly resort to indirect or implicit forms of hate, conveyed through subtle textual expressions or multimodal content. Implicit hate speech is characterized by indirect or veiled expressions that convey negative or prejudicial attitudes toward individuals or groups defined by protected identity attributes [2]. Implicit hate speech avoids overtly

abusive language and instead uses context and underlying meaning to convey hostility, which makes detection much more difficult. This complexity is further amplified by cultural and linguistic diversity, where expressions considered acceptable in one context may be offensive in another. In countries like India, where many languages are spoken, people often mix languages, and informal language substantially increases the difficulty of Hate Speech Detection [3], particularly for implicit hate [4]. The slow, tedious process of manual moderation has driven a shift toward automated HSD systems based on machine learning and deep learning techniques. However, most research focuses on English, a high-resource language, while Indian languages remain underexplored [3]. The scale of Hindi usage contrasts sharply with the limited research on implicit hate speech detection in the language, underscoring the need for effective automated solutions.

The main contributions of the proposed work are:

- **Contextual modeling for Implicit Hate Detection:** This work investigates the hierarchical integration of BiLSTM and attention to detect implicit and explicit hate speech in Hindi. The framework models contextual information at two levels: word-level representations capture local semantic dependencies, while sentence-level representations capture broader contextual relationships.
- **Task-Specific Hierarchical Attention:** The proposed framework employs attention at both the word and sentence levels to selectively aggregate contextual representations. The methodological contribution lies in using hierarchical attention to identify salient lexical and sentence-level contextual cues relevant to implicit versus explicit hateful intent. This provides a more targeted representation for fine-grained hate classification.
- **Emoji-Preserving Contextual Processing for Hindi Social Media Text:** Unlike preprocessing pipelines that discard emojis as non-linguistic symbols, the proposed framework preserves emojis as individual tokens because they can provide emotional and pragmatic cues that complement textual meaning. This design is particularly relevant to Hindi social-media content, where emojis may shape interpretations of indirect or context-dependent expressions.
- **Semantic-Preserving GPT-3.5 Backtranslation Framework:** We employ a GPT-3.5-based backtranslation

*Corresponding author.

strategy to address class imbalance and increase linguistic diversity. We validate augmented-data quality using LaBSE similarity and mBERTScore to ensure semantic consistency between original and generated samples.

- **Integrated Low-Resource Hate Speech Detection Framework:** The main methodological contribution is integrating hierarchical contextual modeling, dual-level attention, emoji preservation, and semantically validated augmentation into a single framework for Hindi implicit/explicit hate speech detection.

II. RELATED WORKS

A. Machine Learning Approaches

Initial research on hate speech detection primarily utilized machine learning (ML) methods that relied on handcrafted features, including n-grams, Bag-of-Words, sentiment cues, syntactic dependencies, and hate lexicons. Classical classifiers, including Decision Trees, Bayesian models, Random Forests, Logistic Regression and Support Vector Machines have been widely employed for hate speech detection in English [5], [6], [7]. Similar approaches were later applied to Hindi and code-mixed Hindi-English text using FastText at character-level representations and traditional classifiers [8]. Despite their initial success, ML-based methods suffer from several inherent limitations. These approaches depend on manual feature engineering and domain-specific lexicons, which increases errors, requires significant time, and limits scalability. Furthermore, they are biased toward explicit hate containing abusive words and fail to capture deeper contextual semantics, sarcasm, and implicit hate expressions. Their performance also degrades significantly in code-mixed and multilingual environments, which are common in Indian social media platforms. As a result, ML techniques are insufficient for robust implicit/explicit hate speech detection in Hindi.

B. Deep Learning Approaches

To overcome the limitations of ML-based models, researchers increasingly adopted deep learning (DL) architectures that can automatically extract features and improve contextual representation. Neural models such as hierarchical networks, BiLSTMs, LSTMs, CNNs, and attention-based architectures demonstrated enhanced semantic understanding and contextual modeling [9], [10], [11]. Recent work has explored multimodal toxicity detection in memes by combining PaddleOCR, KOSMOS-2, and SigLIP to analyse textual and visual cues jointly [12]. For Indian languages, several studies have incorporated subword representations, phrase embeddings, sentiment and emotion features, and auxiliary linguistic resources to handle linguistic diversity and morphological richness better.

Recent studies have increasingly adopted pretrained transformer-based models, such as mBERT, MuRIL, BERT, and HateBERT, word clouds, and ensemble transformer architectures for hate speech detection across diverse languages, including Hindi, Marathi, Tamil, and English [13], [14], [15], [16]. Although these models achieve strong empirical performance, they introduce critical challenges, including large parameter sizes, high computational costs, and dependency on

GPUs/TPUs and large annotated datasets. Additionally, many transformer-based approaches exhibit bias toward majority classes, struggle with unknown tokens, sarcasm, and code-mixed text, and often fail to distinguish implicit hate from explicit hate effectively. These challenges highlight the need for parameter-efficient deep learning models better suited to low-resource language settings.

C. Implicit and Explicit Hate Speech Detection

Implicit hate speech detection has gained increased attention as it relies on indirect expression, nuanced meaning, and strong contextual dependence. Unlike explicit hate, implicit hate avoids overtly abusive language, making it difficult for lexicon-based and surface-level models to detect. Elshierief et.al. [4] introduced benchmark datasets and annotation guidelines that distinguish between implicit and explicit hate, laying the foundation for further research in this area. Subsequent studies explored background knowledge integration, contrastive learning-based fine-tuning strategies [17], prompt-based learning approaches, and large language models to improve interpretability and performance [18].

For Indian languages, research on implicit hate detection remains limited. Kapil et. al. [3] introduced the Hindi Hate Speech Dataset (HHSD) and evaluated implicit and explicit hate using transformer-based architectures.

Despite significant progress in hate speech detection, existing approaches for Hindi largely rely on traditional machine learning methods or computationally intensive pretrained transformer models, both of which struggle to handle implicit hate speech effectively. Furthermore, most studies focus solely on textual content and overlook emoji-based contextual cues, even though emojis are frequently used on social media to convey sentiment, sarcasm, and indirect intent. Robust experimental evaluation of implicit and explicit hate classification remains limited, with many studies relying on single train-test splits rather than comprehensive validation strategies. In addition, data imbalance and limited dataset size remain challenges, as augmentation techniques are rarely explored in the context of implicit hate detection. These limitations highlight the need for efficient, context-aware, and well-evaluated models tailored to the Hindi language (a low-resource language).

Existing Hindi hate speech detection methods primarily rely on transformer-based architectures or keyword-driven approaches that struggle to capture implicit hateful intent. Furthermore, limited attention has been given to multi-level contextual modeling, interpreting model focus, emoji-aware contextual understanding, and semantically validated augmentation strategies. To address these limitations, this work develops a task-specific hierarchical contextual framework that combines word- and sentence-level BiLSTM representations with dual-level attention for implicit/explicit hate discrimination. The framework additionally preserves emojis as contextual tokens and incorporates GPT-3.5-based back-translation with semantic validation using LaBSE and mBERTScore.

This study introduces a novel approach to address the challenges of identifying both implicit and explicit hate speech in Hindi. Emojis are widely used on social media to convey emotions and implicit intent; incorporating emoji information is essential for accurately interpreting user-generated content

[19]. Accordingly, we retain and process emojis during pre-processing to enhance contextual understanding. The proposed model uses FastText embeddings to generate efficient, semantically rich representations of the input text [20]. These embeddings flow into a Bidirectional Long Short-Term Memory network, which uncovers contextual relationships from both past and future perspectives. Furthermore, an attention mechanism assigns higher importance to salient words or sentences, enabling the model to focus on critical contextual cues for detecting implicit hate speech [21].

III. PROPOSED METHODOLOGY

The proposed technique classifies hate posts on social media as either implicit or explicit hate speech. Fig. 1 shows the system architecture. The proposed system takes Hindi text as input, which is then preprocessed. Using an embedding technique, the text is converted into a vector and, finally, classified as implicit/explicit hate speech. The detailed explanation is as follows.

A. Dataset

The proposed method uses the Hindi Hate Speech Dataset (HHSD). HHSD contains 14782 tweets, each labelled as hate or non-hate. 7311 hateful tweets are present, which are further labelled as implicit or explicit hate speech. Of these, 2432 are implicit, and 4879 are explicit. Because the proposed work focuses on binary classification, we applied the method only to tweets with hate labels. Table I presents detailed information about the dataset.

TABLE I. LABEL DISTRIBUTION

Label	Label	Size
Hate	Implicit	2432
	Explicit	4879
Nonhate		7471
Total		14782

B. Motivation and Design Rationale

The proposed approach's key design consideration is a hierarchical contextual modeling framework that captures context at both the word and sentence levels. Unlike traditional hate speech detection models that rely primarily on offensive keywords, the framework identifies subtle and indirect forms of hate speech by focusing on contextual semantics and inter-sentence relationships. At the word level, attention is used to highlight critical terms that signal hateful intent, even when explicit abusive language is absent. Additionally, at the sentence level, attention identifies which sentences carry the most discriminatory meaning within multi-sentence posts. This hierarchical attention structure provides an interpretable view of the words and sentences that influence predictions, aiming to improve detection of challenging implicit hate speech scenarios. By combining bidirectional contextual learning with multi-level attention, the model provides complementary contextual representations of textual content compared with conventional single-level attention or keyword-based approaches.

The methodological contribution therefore lies in integrating these established components for task-specific contextual discrimination of implicit and explicit hate speech.

1) *Context-aware implicit hate speech detection*: Traditional hate speech detection models often rely on the presence of explicit offensive terms (e.g., slurs, abusive words, derogatory expressions). However, implicit hate speech is expressed indirectly through stereotypes, sarcasm, coded language, insinuations, or contextual references. Since no explicit hate-related keywords may be present, detecting implicit hate requires understanding the surrounding linguistic context.

For example:

- Explicit Hate: “आप जैसे हारामी को भी अवसर मिलेगा” (Translation: Even a scoundrel like you will get an opportunity.)
- Implicit Hate: “मानसिक दिवालियापन है 🤪🤪” (Translation: It is mental bankruptcy 🤪🤪)

The second statement contains no obvious hate words, but the context implies discrimination toward a particular group. Therefore, the proposed model utilizes contextual representation learning through BiLSTM and attention mechanisms.

The meaning of a potentially hateful expression depends heavily on its surrounding context. A word or phrase may appear neutral on its own but may express discriminatory intent when interpreted in a broader linguistic context. Therefore, contextual representation learning is essential for accurately distinguishing implicit hate speech from explicit hate speech.

Let the input sentence be represented as in Eq. (1):

$$S = \{w_1, w_2, w_3, \dots, w_l\} \quad (1)$$

where, w_i denotes the i^{th} word and l represents the sentence length. Each word is transformed into a dense vector representation using Eq. (2):

$$x_i = \text{Embedding}(w_i) \quad (2)$$

A Bidirectional Long Short-Term Memory (BiLSTM) network processes the embedded sequence. The forward hidden state is computed using Eq. (3):

$$\vec{h}_i = \text{LSTM}_f(x_i, \vec{h}_{i-1}) \quad (3)$$

and the backward hidden state is computed as defined in Eq. (4):

$$\overleftarrow{h}_i = \text{LSTM}_b(x_i, \overleftarrow{h}_{i+1}) \quad (4)$$

The contextual representation of each word is obtained Eq. (5) by concatenating the forward and backward hidden states :

$$h_i = [\vec{h}_i; \overleftarrow{h}_i] \quad (5)$$

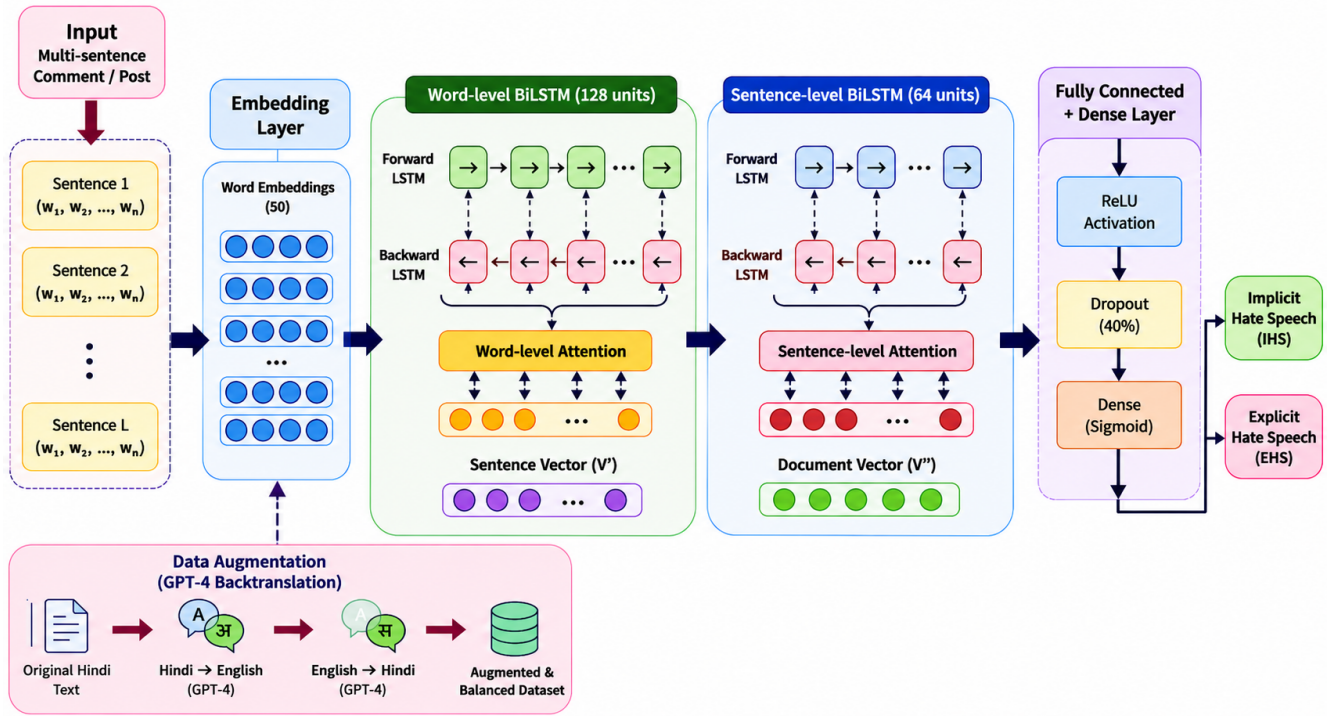


Fig. 1. Architecture diagram.

where, $h_i \in \mathbb{R}^{2d}$ and d denotes the hidden-state dimension. The BiLSTM output matrix is represented using Eq. (6):

$$OB_{BiLSTM}^{(w)} = [h_1, h_2, \dots, h_l] \in \mathbb{R}^{l \times 2d} \quad (6)$$

Unlike explicit hate speech, implicit hate often lacks direct offensive words. Therefore, contextual understanding becomes essential. The BiLSTM captures semantic relationships among words, enabling the model to identify hidden hateful intent even when explicit abusive vocabulary is absent.

2) *Hierarchical multi-level context modeling*: Hateful intent may span multiple sentences rather than being expressed within a single sentence. Consequently, effective hate speech detection requires modeling both local semantic relationships among words and global contextual relationships among sentences.

At the word level, contextual dependencies among tokens are captured using a BiLSTM Eq. (7):

$$H_w = BiLSTM(EH) \quad (7)$$

where, EH denotes the FastText embedding representation. The resulting sentence representations are then processed through a sentence-level BiLSTM Eq. (8):

$$H_s = BiLSTM(V) \quad (8)$$

where, V represents the sentence vectors generated from the word-level attention mechanism. This hierarchical architecture enables the model to capture:

- Local semantic dependencies among words.
- Global discourse-level relationships among sentences.
- Contextual information distributed across multi-sentence posts.

By jointly modeling lexical and discourse-level context, the framework provides richer semantic representations for implicit hate speech detection.

3) *Attention-guided feature selection*: Not all words and sentences contribute equally to hate speech classification. In implicit hate speech, only a small subset of words may reveal the underlying hateful intent. Similarly, within a multi-sentence post, only certain sentences may contain discriminatory cues.

Overall, the proposed framework combines context-aware representation learning, hierarchical multi-level context modeling, and attention-guided feature selection to capture subtle and context-dependent hate expressions. Furthermore, we incorporate semantic-preserving back-translation to address data imbalance and increase linguistic diversity, supporting robust hate speech detection in the low-resource Hindi setting.

C. Preprocessing

During preprocessing, we cleaned the data. We removed content that does not contribute to classification, such as usernames, mentions, handles, hashtags, and hyperlinks. We removed leading, trailing, and multiple spaces from the input data. Special characters do not contribute to classification, so we also removed them from the input. Suffixes are not useful in classification tasks; therefore, we also removed them from the input text. To retain contextual knowledge, some stop

words play an important role, so preprocessing didn't remove them. FastText embedding converts the input into a vector representation using contextual knowledge, so keeping some stop words is important.

Many users across platforms, such as social media, use emojis to express their feelings in posts. Emojis provide a more expressive and attractive way to convey feelings or add meaning to textual posts. Emojis may be used to remove ambiguity from ambiguous sentences. Emojis are used to complement relatively negative text [19]. Emojis greatly contribute to understanding emotion, whether positive/negative and direct/indirect. Many researchers have considered emotion/sentiment as an important feature in HSD, specifically in implicit HSD [22], [23], [24].

Emojis benefit implicit/explicit HSD, so the proposed method does not remove them from the input data. Preprocessing uses them to understand the sentence's context better. In tokenization, each unique emoji is assigned a separate token, which is then used in embedding and other layers.

D. Tokenizer

The tokenizer converts input text into numerical representations, enabling more efficient processing by the NLP model. The tokenizer takes text and emojis as input, identifies unique words (w_i) and emojis, and builds a vocabulary. The vocabulary contains words, emojis, and their corresponding index entries, which are used to convert input into numbers. If any out-of-vocabulary words are present, they are handled using an out-of-vocabulary (OOV) token. In the proposed method, we replace OOV with the <UNK> token.

The tokenizer converts each word and emoji into a token t_i . This creates a numeric vector t of varying length, depending on the input sentence. The input vector t is padded to produce a fixed-length vector P of length l , $|P| = l \geq |t|$. The first $|t|$ entries are filled with t , and the remaining $(l - |t|)$ entries are filled with zeros as padding. In the proposed model, the maximum sentence length (l) is 50, so padding converts all input vectors to size 50. The resulting vector with fixed length $P \in \mathbb{R}^{1 \times l}$ is provided to the next layer (embedding layer) for further processing.

E. Embedding Layer

Embedding converts input into an efficient representation, so this work uses FastText embeddings. Facebook provides pre-trained FastText embeddings [20] for 157 languages, including Hindi. FastText embeddings provide more efficient word representations. FastText provides an efficient data representation, which also makes the classification task easier. FastText captures word morphology and handles out-of-vocabulary words better. FastText extends word2vec, in which words are represented as a collection of subword embeddings. Because of these features, FastText is widely used in NLP tasks such as sentiment analysis and classification. Each token t_i of the input vector is embedded using FastText. This embedding creates a vector $E_H \in \mathbb{R}^{l \times 300}$. The embedding vector is 50x300. This vector proceeds to the next layer, the BiLSTM layer. It handles word morphology and out-of-vocabulary words better.

F. BiLSTM Layer

A bidirectional LSTM (BiLSTM) processes information in both directions. In the forward direction, an LSTM processes the input from left to right, and in the backward direction, from right to left. Due to this bidirectionality, the model captures context from the previous and next words relative to the current word. This provides a better representation of each word by considering the context of nearby words. So, the proposed method uses a BiLSTM to process the input in both directions and capture context from nearby words.

The embedding layer produces a vector that is fed into a BiLSTM. In implicit/explicit hate speech, knowing contextual information is important. Specific word-based identification is not available for detecting implicit hate. The implicit nature of the input makes the detection task much harder, so understanding context and representing it efficiently is important. In the proposed model, we use 128 BiLSTM units. BiLSTM provides b_i hidden state for each token t_i . By processing all tokens, we create a matrix using Eq. (9).

$$B_{BiLSTM} \in \mathbb{R}^{l \times 2d} \quad (9)$$

Where, d is the dimension of the hidden state.

l : length of the input vector (50).

This vector representation is then forwarded to the attention layer for further processing.

G. Attention Layer

The model learns word-specific importance weights through a lexical-level attention mechanism. This process assigns different importance scores to different words. Assigning importance scores allows the model to prioritize the most relevant words within sentences [21]. The resulting attention weights show the relative importance assigned to different tokens during classification.

For implicit/explicit HSD, the focus should be on the words. In explicit hate speech, the use of offensive/abusive words directly represents hate, but in implicit hate, there is no direct word present. Indirect targeting of individuals or groups complicates the identification of implicit hate speech, as it does not rely on explicit offensive or abusive language. So, we need to focus on the words in sentences that signal implicit hate speech. Attention mechanisms focus on words/sentences to identify important terms that signal implicit hate speech. The attention network accepts the B_{BiLSTM} matrix as input and assigns relative importance to tokens by focusing on words that contribute to implicit/explicit hate classification.

To automatically identify informative textual units, we apply an attention mechanism over the contextual representations using Eq. (10). First, a hidden representation is computed as:

$$U^{(w)} = \tanh(B_{BiLSTM}^{(w)} W^{(w)} + b^{(w)}) \quad (10)$$

where, $B_{BiLSTM}^{(w)}$ represents the contextual word representations generated by the word-level BiLSTM, $W^{(w)}$ is a trainable weight matrix, and $b^{(w)}$ is the bias vector. The

importance score for each word is calculated as shown in Eq. (11):

$$s_i^{(w)} = U_i^{(w)} u^{(w)} \quad (11)$$

where, $u^{(w)}$ is the trainable word-level attention score vector. Attention weights are obtained using the softmax function, as defined in Eq. (12):

$$\alpha_i^{(w)} = \frac{\exp(s_i^{(w)})}{\sum_{j=1}^l \exp(s_j^{(w)})} \quad (12)$$

such that:

$$\sum_{i=1}^l \alpha_i^{(w)} = 1 \quad (13)$$

The context-aware sentence representation is then computed using Eq. (14):

$$V^{(s)} = \sum_{i=1}^l \alpha_i^{(w)} h_i \quad (14)$$

The word-level attention mechanism assigns different weights to contextual word representations, allowing the model to emphasize informative words within each sentence. These attention weights provide an interpretable view of the model's focus, but they should not be regarded as a complete explanation of the model's decision.

H. Sentence-Level BiLSTM Layer

The representations produced by the preceding layer are propagated in both forward and backward directions, enabling the model to capture complex inter-sentential relationships. This sentence-level BiLSTM provides contextual sentence representations by leveraging surrounding sentences in multi-sentence comments/posts. In the proposed model, we use 64 BiLSTM units.

I. Sentence-Level Attention Layer

The BiLSTM outputs feed into the sentence-level attention layer. This attention layer produces a single vector from multiple sentences, assigning higher weight to important sentences. This layer identifies relationships among sentences and focuses on the most important ones to form a single multi-sentence vector. Using this weighted vector, identifying implicit hate speech becomes easier.

The attention network creates a hidden representation using the $\tanh(\cdot)$ activation function, as defined in Eq. (15).

$$U^{(s)} = \tanh(B_{BiLSTM}^{(s)} \cdot W^{(s)} + b^{(s)}) \quad (15)$$

Where, $w^{(s)}$ and $b^{(s)}$ are the trainable sentence-level attention parameters.

$B_{BiLSTM}^{(s)}$: Input matrix of size $\mathbb{R}^{l \times 2d}$.

Using Eq. (16), we compute the sentence potential importance score.

$$S_j^{(s)} = U_j^{(s)} u^{(s)} \quad (16)$$

Where, $u^{(s)}$ is the trainable sentence-level attention score vector.

Eq. (17) calculates attention weights using $\text{softmax}(\cdot)$ function.

$$\alpha_j^{(s)} = \frac{\exp(s_j^{(s)})}{\sum_{k=1}^m \exp(s_k^{(s)})} \quad (17)$$

Eq. (18) calculates the context vector as the sum of the products of the attention network input and the sentence-level attention weights.

$$V^{(d)} = \sum_{j=1}^m \alpha_j^{(s)} h_j^{(s)} \quad (18)$$

The sentence-level attention mechanism assigns different weights to contextual sentence representations, allowing the model to emphasise informative sentences among multiple sentences. The outcome of this layer is a sentence vector, which will be used for further processing in the model.

J. Fully Connected and Dense Layer

After attention-based feature extraction, a fully connected layer with ReLU non-linearity performs classification, with regularization via a 40% dropout strategy. Final predictions are generated using a softmax-based decision layer to distinguish between implicit hate speech (IHS) and explicit hate speech (EHS).

K. Model with Augmentation

To examine the impact of data augmentation and class balancing, we apply back-translation to the entire dataset. The size of the dataset after complete dataset augmentation and class balancing is shown below in Table II.

TABLE II. AUGMENTED DATASET DETAILS

Detail	Size	
	Implicit	Explicit
Class Balancing	4864	4879
Complete Augmentation	4864	9651

The HHSD dataset is highly imbalanced. As shown in Table I, the dataset contains 2432 tweets with implicit labels and 4879 with explicit labels. Using back-translation, a data augmentation technique, we balanced the dataset. In backtranslation, we first translate the input data from Hindi to English and then back to Hindi. For back-translation, we use GPT-3.5. Fig. 2 shows the back-translation process.



Fig. 2. Back-translation.

1) *GPT-based back-translation*: To augment the Hindi dataset, we performed back-translation using the OpenAI API with the `gpt-3.5-turbo` model. We first translated each selected Hindi instance into English and then translated it back into Hindi. This process aimed to generate semantically preserving variations of the original instances.

The translation was performed using the following prompt template:

Translate the following text to [target language]:
[text]

We also provided a system instruction identifying the model as a professional translator. We used the API's default settings. We used the resulting Hindi sentences as augmented instances while preserving the original class labels. To prevent possible data leakage during augmentation, we generated back-translated samples only within the training portion of each cross-validation fold.

To validate the augmented dataset, the quality of translation is evaluated using the Semantic Cosine similarity score and BERTScore.

2) *Cosine similarity score*: It is a semantic similarity metric. We use cosine similarity with LaBSE to verify the quality of the translated text. This verifies how much core meaning the translation retains. Because the proposed work uses Hindi, we use LaBSE to assess the validity of the back-translated text. It uses Eq. (19) to calculate cosine similarity.

$$\text{Cosine Similarity}(V_{orig}, V_{back}) = \frac{V_{orig} \cdot V_{back}}{\|V_{orig}\| \|V_{back}\|} \quad (19)$$

The average semantic similarity score using cosine similarity with LaBSE is 0.8450 for the implicitly balanced text and 0.7928 for the complete augmented dataset. The LaBSE cosine similarity scores of 0.8450 and 0.7928 indicate substantial semantic similarity between the original and back-translated text. High scores indicate that, despite phrasing variations, back-translation preserved the original text's meaning.

3) *BERT score*: BERT score is also a semantic similarity metric that computes sentence-level similarity. mBERT computes cosine similarity between sentences using contextualised token embeddings. It captures paraphrases and semantic equivalence. The final mBERT score is calculated using contextual embeddings for tokens in the original and back-translated sentences. The final F1 score measures semantic similarity between the original and back-translated sentence. A higher mBERT score indicates that the original text is semantically close to the final back-translation. The mBERT score for the back-translated text on the implicit balanced dataset is

0.8084, and on the complete augmented dataset it is 0.7824. This higher score indicates that the backtranslated text is semantically close to the original.

IV. EXPERIMENT

This section presents the experimental setup and the baseline and state-of-the-art (SOTA) methods used for comparison. Except for the BiGRU+Capsule Network, all other baseline and SOTA models were implemented and reproduced based on the architectural and experimental details reported in HHSD [3]. We followed the same HHSD dataset and preprocessing procedure for these reproduced baseline and SOTA experiments. The details of the same are as follows:

We compare the following baseline and state-of-the-art (SOTA) methods:

- LaBSE [25], [3]: It is used to produce vector embeddings for 109 languages using language-agnostic BERT.
- MuRIL[26], [3]: MuRIL was developed specifically for Indian languages. To improve performance on Indian languages, this model is trained on 16 of them.
- IndicBERT [27], [3]: It uses fewer parameters than some BERT variants. IndicBERT is a multilingual language model developed using data from 12 Indian languages.
- XLM-RoBERTa [28], [3]: This model was trained on 100 languages and demonstrates good cross-lingual classification performance compared to mBERT.
- mBERT [29], [3]: mBERT led the development of Multilingual BERT, a pre-trained transformer model supporting 104 languages.
- BiGRU + Capsule Network [30]: The BiGRU + Capsule Network [30] baseline consists of a 128-unit bidirectional GRU layer, followed by a capsule layer containing 16 capsules of dimension 32. The capsule layer uses five routing iterations. The capsule representations are aggregated using global average pooling, followed by a dropout layer (rate 0.3) and a two-class softmax output layer. We optimised the model using Adam with a learning rate of 0.001. Training was performed for 10 epochs with a batch size of 512.
- CNN [31]: The CNN baseline was implemented using a Conv1D layer with 100 filters and a kernel size of 4, followed by max pooling. A fully connected layer with 32 ReLU units followed before the output layer. The model was optimized with Adam at a learning rate of 0.001. Training was performed for 3 epochs with a batch size of 30.
- BiLSTM [32]: The input representation was processed by a BiLSTM with 100 hidden units, followed by MaxPooling1D with a pool size of 2. A fully connected layer with 32 ReLU units preceded the output layer. The model was optimized with Adam (learning rate 0.001), a batch size of 30, and 2 epochs of training.

A. Experimental Setup of Proposed Technique

We developed the proposed model in TensorFlow and initialized it using the FastText embedding technique. We used pre-trained 300-dimensional Hindi FastText vectors to initialize the embedding matrix. The embedding layer was kept trainable during model training to allow task-specific adaptation of the representations. Performance was evaluated with 5-fold cross-validation. Training used a fixed 5 epochs per fold; we optimized model with the Adam optimizer at a learning rate of 9e-5, keeping the remaining optimizer parameters at their default TensorFlow/Keras settings. We used binary cross-entropy as the objective function and trained with mini-batches of 8 samples. We selected the hyperparameter values through preliminary experiments on the training data, considering validation performance and convergence stability. We varied the learning rate, batch size, number of training epochs, and hidden-unit configurations over a small set of empirically reasonable values. We selected the configuration that provided stable validation performance. We did not perform exhaustive hyperparameter optimization. We used the same hyperparameters for both the fully augmented and balanced datasets. We used a fixed random seed of 42 to ensure reproducibility. Input sequences were padded or truncated to a maximum sentence length of 50 tokens. The proposed architecture employed 128 hidden units in the word-level BiLSTM and 64 hidden units in the sentence-level BiLSTM. We applied a dropout rate of 0.4 after the fully connected layer to reduce overfitting during training. The attention mechanism used trainable attention weights with softmax normalization to identify important contextual information from the input sequence. In the attention layer, the trainable attention weight matrix and attention score vector were initialized from a random-normal distribution, while the attention bias was initialised to zeros.

We conducted the experiments on an NVIDIA RTX A4000 GPU with 16 GB GPU memory and 32 GB system RAM. Across the 5-fold cross-validation, the average training time was 22.29±2.65 min per fold. The average model evaluation time across the five folds was 12.20±9.39 sec per fold.

B. Evaluation Metrics

We evaluate the proposed technique using accuracy and weighted F1. The accuracy is calculated using Eq. (20)

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (20)$$

Here, TP and TN stand for true positives and true negatives (successful hits and correct misses), while FP and FN denote false alarms and overlooked cases.

The dataset exhibits significant class imbalance; therefore, standard accuracy is not an appropriate evaluation metric, as it may produce inflated scores by favouring the majority class. For highly imbalanced datasets, the weighted F1-score is more appropriate because it averages F1 scores across all classes, assigning weights that reflect the number of true examples in each class. We evaluated model performance using the weighted F1-score metric Eq. (21).

$$\text{Weighted F1} = \frac{\sum_{i=1}^N (F1_{class_i} \times S_{class_i})}{\sum_{i=1}^N S_{class_i}} \quad (21)$$

where, N: Total number of classes.

$F1_{class_i}$: F1 score for class i.

S_{class_i} : Support (number of actual instances) for class i.

C. Results

On the original dataset, the model correctly identifies 301 samples from the implicit class (0) and 796 samples from the explicit class (1), as summarized in Fig. 3.

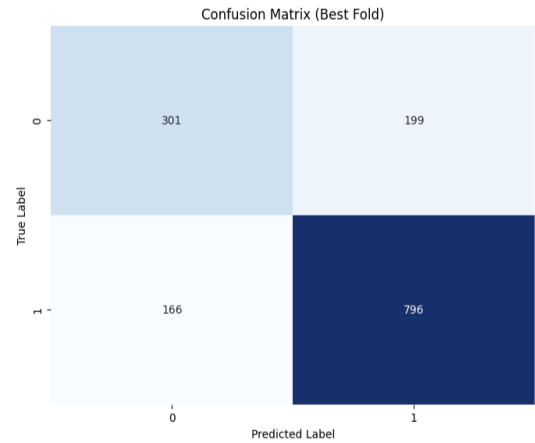


Fig. 3. Confusion matrix.

For the best-performing fold, the proposed model achieved an overall accuracy of 75.03% and a weighted F1-score of 74.81%. The per-class analysis shows that the model achieved 64.45% precision, 60.20% recall, and a 62.25% F1-score for implicit hate speech. In contrast, the corresponding values for explicit hate speech were 80.00%, 82.74%, and 81.35%, respectively. The macro F1-score was 71.80%. Table III presents the class-wise Precision, Recall, and F1-score for the best-performing fold.

These results reflect balanced predictive behaviour across both categories. The Best accuracy and loss of the model are shown in Fig. 4 and 5 for the original dataset. Test-set accuracy continued to improve through 5 epochs.

TABLE III. CLASS-WISE CLASSIFICATION PERFORMANCE FOR THE BEST-PERFORMING FOLD

Class	Precision (%)	Recall (%)	F1-score (%)	Support
Implicit Hate Speech (IHS)	64.45	60.20	62.25	500
Explicit Hate Speech (EHS)	80	82.74	81.35	962
Macro Average	72.23	71.47	71.80	1462
Weighted Average	74.69	75.03	74.82	1462

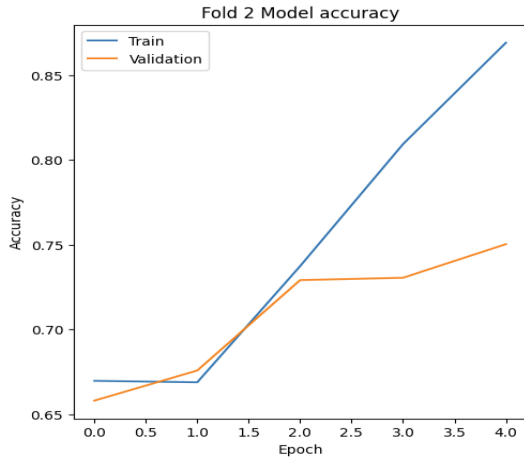


Fig. 4. Model accuracy.

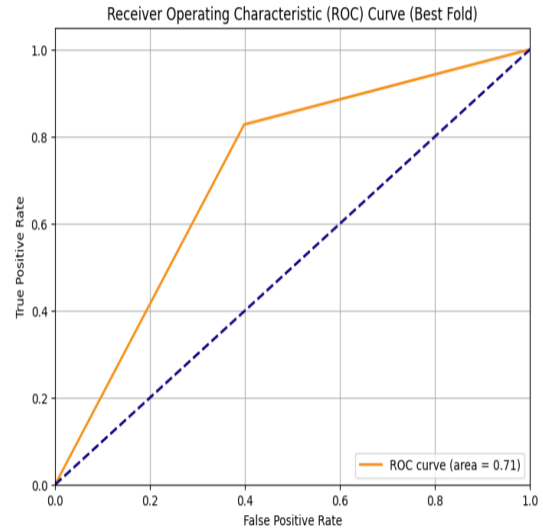


Fig. 6. ROC curve.

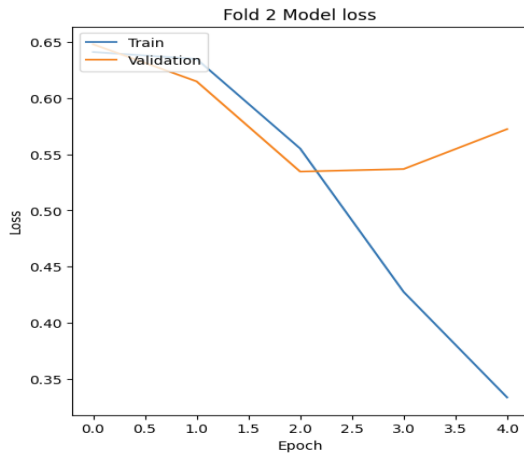


Fig. 5. Model loss.

Fig. 6 depicts the ROC curve obtained for the best validation fold. The curve's proximity to the upper-left region reflects the model's stronger ability to correctly distinguish explicit hate instances while keeping the false positive rate low. With an AUC (Area Under the Curve) of 0.71, the model shows a fair capacity to separate explicit from implicit hate speech. This value indicates a 71% chance that a randomly selected explicit hate instance will be ranked as explicit. Because detecting implicit hate is very difficult, an AUC of 0.71 is fair for classifying implicit/explicit HSD.

Table IV shows the number of parameters required for pretrained transformer models.

TABLE IV: Parameter comparison

Model	# Parameters
IndicBERT [27]	$\geq 18\text{M}$
MuRIL [26]	236M
LaBSE [25]	$\geq 110\text{M}$
XLM-RoBERTa [28]	270M
mBERT [29]	172M
Proposed model	12M

mBERT has 172 million parameters, while XLM-RoBERTa has approximately 270 million, which is very large. This large parameter count implies a larger model footprint and potentially greater computational requirements. The proposed model requires approximately 12 million parameters, resulting in a substantially smaller parameter footprint. Thus, the proposed architecture is relatively lightweight in terms of parameter count.

This method outperforms CNN, BiLSTM, BiGRU+Capsule Network, IndicBERT, MuRIL, LaBSE, XLM-RoBERTa, and mBERT. The proposed model achieves a best accuracy of 75.03%, which is 2.51% higher than mBERT. The best weighted F1-score is 74.81%, which is 17.84% higher than that of mBERT.

Table V reports the accuracy and weighted F1-scores

achieved by the proposed method.

TABLE V: Result on implicit HHSD

Model	Accuracy	Weighted-F1
CNN [31]	66.10%	52.20%
BiLSTM [32]	68.72%	54.30%
BiGRU + Capsule Network [30]	69.92%	53%
IndicBERT [27]	66.50%	52.80%
MuRIL [26]	70.95%	56.33%
LaBSE [25]	69.17%	54.54%
XLM-RoBERTa [28]	66.30%	40.50%
mBERT [3]	72.52%	56.97%
Proposed	75.03%	74.81%

D. Discussion of Key Findings

1) *Finding 1*: The proposed model achieves the highest weighted F1-score (74.81%), outperforming mBERT by 17.84%.

2) *Finding 2*: The small gap between Accuracy (75.03%) and Weighted F1-score (74.81%) indicates that the overall performance is reasonably consistent when accounting for class-wise F1 contributions. However, the lower F1-score for implicit hate speech demonstrates that this class remains more challenging to classify.

3) *Finding*: The proposed hierarchical contextual framework achieves competitive performance against the evaluated transformer-based baselines, while class-wise results indicate that implicit hate remains harder to distinguish than explicit hate.

4) *Finding 4*: Semantic-preserving augmentation improves performance, while class balancing provides an additional improvement in weighted F1-score.

E. Ablation Study

The proposed work investigates the impact of back-translation-based augmentation and class balancing specifically for implicit hate speech detection, where contextual ambiguity and limited annotated data significantly affect classification performance. We ran experiments using 5-fold cross-validation and computed the mean weighted F1-score and standard deviation across folds. Results in Table VI indicate that back-translation-based augmentation improves contextual diversity and enhances model generalization. Furthermore, balancing implicit and explicit hate samples significantly improves classification stability and minority class detection performance. The balanced augmented dataset achieved the best overall performance with the highest mean weighted F1-score.

Augmenting the complete dataset further enhances model performance. Experimental results show that data augmentation improves model performance compared with training on the original dataset.

TABLE VI. ABLATION STUDY USING 5-FOLD CROSS-VALIDATION

Variant	Weighted F1-score (Mean $\pm$ Std)
Proposed	73.33 $\pm$ 0.97
Proposed+augmented	74.95 $\pm$ 0.55
Proposed+augmented balanced	79.04 $\pm$ 1.44

In 5-fold cross-validation, the best fold achieves a weighted F1-score of 75.81% and an accuracy of 76.42%. These results represent an 18.84% improvement in weighted F1-score and a 3.9% increase in accuracy compared to mBERT. Relative to the original dataset, the weighted F1-score increases by 1% and accuracy by 1.39%.

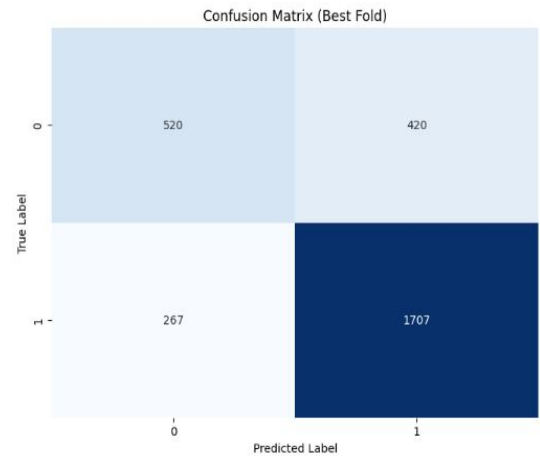


Fig. 7. Confusion matrix of complete augmentation.

Fig. 7 presents the confusion matrix, Fig. 8 displays the accuracy and Fig. 9 illustrates the ROC curve for the fully augmented dataset.

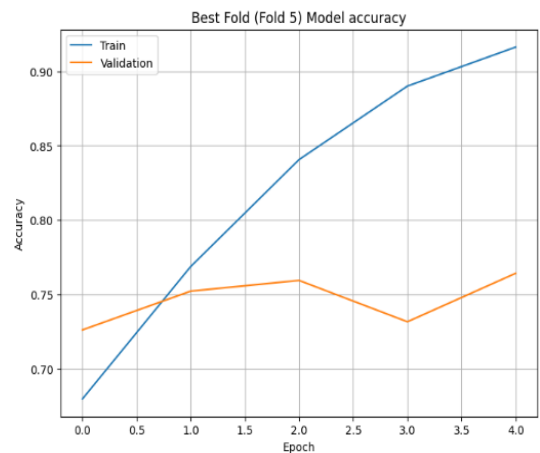


Fig. 8. Accuracy of complete augmentation.

To correct the dataset imbalance, we used data augmentation with backtranslation, as described in Section III-K. Only implicit data was augmented for balancing. The model was then evaluated on the balanced dataset, which showed significantly improved performance compared to the imbalanced

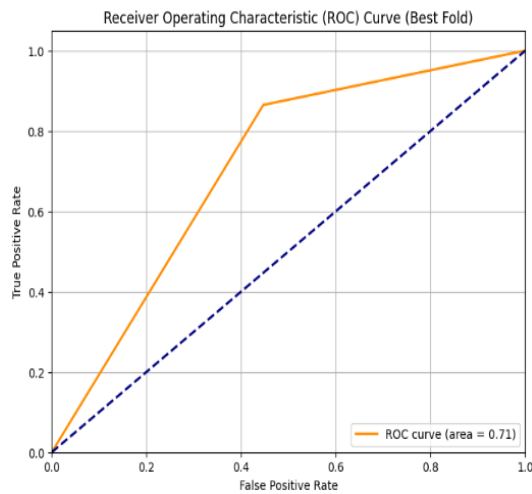


Fig. 9. ROC curve of complete augmentation.

version. In 5-fold cross-validation, the highest weighted F1-score of 80.70% and accuracy of 80.70% were achieved, reflecting improvements of 8.18% and 23.73% over mBERT. The weighted F1-score increased by 5.89% and accuracy by 5.67% compared to the original dataset. The balanced dataset is evaluated using multiple performance indicators. Class-wise results, accuracy trends, and ROC analysis are shown in Fig. 10, 11 and 12.

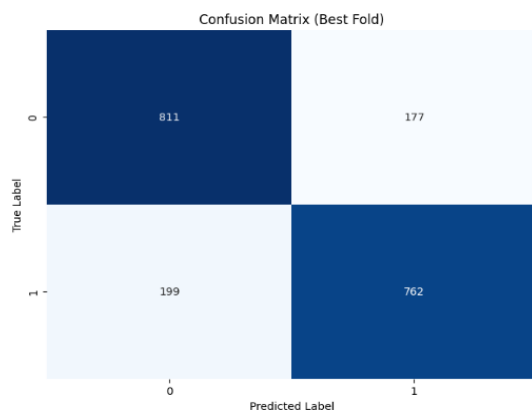


Fig. 10. Confusion matrix of balanced dataset.

F. Error Analysis

The study provides detailed qualitative error analysis demonstrating how contextually sensitive lexical terms can confuse deep learning models between implicit and explicit hate speech categories. The proposed model's confusion matrix shows that many implicit hate speech instances are misclassified. Unlike explicit hate speech, implicit hate often lacks direct offensive vocabulary or explicit slurs and instead conveys hateful intent through indirect references, sarcasm, stereotypes, or hidden contextual meaning. These linguistic characteristics make implicit hate speech much harder to identify. In addition, the relatively smaller number of implicit hate samples in the dataset also affects the model's classification capability and contributes to prediction bias toward the majority class.

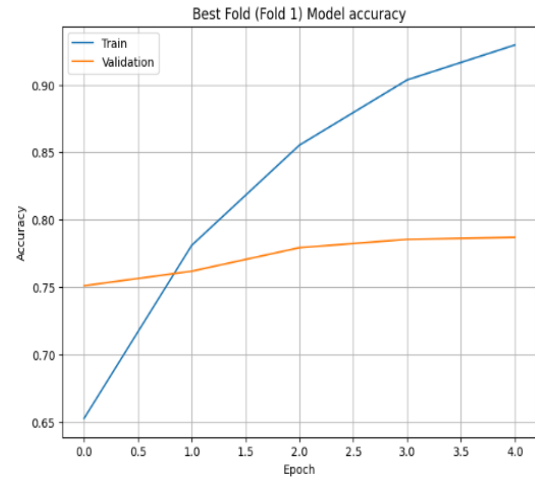


Fig. 11. Accuracy of balanced dataset.

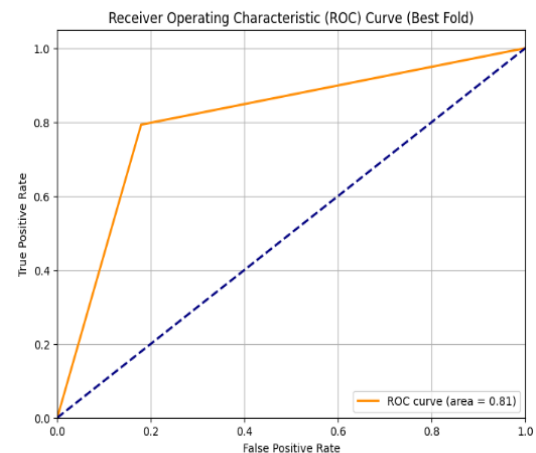


Fig. 12. ROC curve of balanced augmentation.

For example, the statement “माफ करिए 🙏 पैसे देकर खरीदते हैं हराम का तो नहीं खाते”

(Translation: “Sorry 🙏, we buy it with our own money; we do not consume anything forbidden or unlawful.”) was misclassified as explicit hate speech due to the presence of contextually sensitive terms such as हराम (forbidden). Although the statement does not contain direct abusive language, the implicit sarcastic tone and indirect targeting may cause the model to associate such terms with explicit hateful expressions. This demonstrates the difficulty of distinguishing implicit hate from explicit hate when contextual and cultural understanding is required.

The proposed model correctly identified most explicit hate speech instances; however, it still misclassified a small number of samples. Explicit hate speech detection primarily relies on the presence of offensive vocabulary, abusive expressions, and explicit slurs, which are generally easier for the model to recognize. However, in some cases, abusive or sensitive words are used indirectly within an implicit contextual setting rather than as direct hate expressions. In such situations, the model tends to associate these terms with explicit hate speech and may incorrectly classify implicit hate instances as explicit hate.

For instance, the statement “इतिहास में आज जब अंटार्कटिका पर पहली बार पड़े इंसान के कदम साथ में कुत्ते भी वहां पहुंचे थे.” (Translation: “On this day in history, when humans first stepped onto Antarctica, dogs had also reached there along with them.”) was an implicit hate speech instance. Still, the model incorrectly classified it as explicit hate speech. The misclassification may have occurred because of the word “कुत्ते” (dog), which is frequently associated with abusive or offensive expressions in hateful content. Although the statement uses the term indirectly rather than as a direct insult, the model likely relied on lexical associations and interpreted it as explicit hate speech. This shows that contextually sensitive words can confuse the model when distinguishing between implicit and explicit hate speech.

V. CONCLUSION AND FUTURE WORK

The proliferation of hateful discourse on social media platforms poses increasing risks to social harmony. It highlights the need for effective automated Hate Speech Detection (HSD), particularly for low-resource languages such as Hindi. This study developed a hierarchical BiLSTM–attention-based framework for Hindi hate speech detection, emphasizing the identification of both implicit and explicit forms of hate.

The principal contributions of this study include the task-specific integration of a hierarchical BiLSTM architecture with attention-based feature selection for contextual discrimination of implicit and explicit hate speech, the selective retention of stop words and emojis to preserve contextual and emotional information, and the incorporation of semantic-preserving data augmentation and class balancing to address class imbalance and increase linguistic diversity. The proposed framework does not rely on large transformer-based language models and has an approximately 12 million parameter footprint, compared with approximately 172 million parameters for mBERT.

Its ability to identify both implicit and explicit hate expressions can help human moderators prioritize potentially harmful content for further review. Thus, the proposed approach provides a parameter-efficient alternative for Hindi hate speech detection while retaining the ability to model hierarchical contextual information.

The present study has some limitations. First, the experimental evaluation is conducted on the HHSD dataset. The dataset’s characteristics, including its topic distribution, annotation scheme, and domain-specific expressions, may influence model performance and limit the generalizability of the findings. Second, publicly available Hindi datasets generally use different annotation schemes, including hate/non-hate, hate/offensive/profane, hostility-related, or target-based categories. Because equivalent implicit/explicit hate-speech annotations are limited, directly evaluating this specific task on other Hindi corpora is challenging. Third, the applicability of the proposed approach to multilingual settings and cross-domain environments requires further investigation.

Future research will evaluate the proposed approach on additional Hindi corpora, cross-domain datasets, and multilingual resources to assess its robustness and generalizability. Future research will also explore integrating contextual and region-

specific cultural knowledge to improve model performance further.

Overall, the proposed work shows that hierarchical contextual representation learning can effectively support Hindi hate speech detection, including identifying implicit hate expressions, without relying solely on large transformer-based architectures. The combination of hierarchical context modelling, semantic-preserving augmentation, class balancing, and parameter efficiency provides a practical foundation for developing more robust and deployable Hindi hate speech detection systems.

DECLARATIONS

Ethics Approval and Consent to Participate

This study does not involve human participants or animal subjects requiring ethical approval.

Competing Interests

The authors declare no conflict of interest.

REFERENCES

- [1] United Nations, “What is hate speech? understanding hate speech,” <https://www.un.org/en/node/165070>, 2025, accessed: July 4, 2025.
- [2] M. S. Jahan and M. Oussalah, “A systematic review of hate speech automatic detection using natural language processing,” *Neurocomputing*, vol. 546, p. 126232, 2023.
- [3] P. Kapil, G. Kumari, A. Ekbal, S. Pal, A. Chatterjee, and B. N. Vinutha, “HHSD : Hindi hate speech detection leveraging multi-task learning,” *IEEE Access*, vol. 11, pp. 101 460–101 473, 2023.
- [4] M. ElSherief, C. Ziems, D. Muchlinski, V. Anupindi, J. Seybolt, M. De Choudhury, and D. Yang, “Latent hatred: A benchmark for understanding implicit hate speech,” Proceedings of the 2021 Conference on Empirical Methods in Natural Language Processing, Online and Punta Cana, Dominican Republic, pp. 345–363, 2021. [Online]. Available: <https://aclanthology.org/2021.emnlp-main.29/>
- [5] P. Burnap and M. L. Williams, “Cyber hate speech on twitter: An application of machine classification and statistical modeling for policy and decision making,” *Policy & Internet*, vol. 7, no. 2, pp. 223–242, 2015.
- [6] J. A. Siddiqui, S. S. Yuhani, and Z. A. Memon, “Unmasking online hostility: Analysing and mitigating hate speech in social media,” *Baghdad Science Journal*, vol. 22, no. 4, pp. 1393–1408, 2025.
- [7] A. M. U. D. Khanday, Q. R. Khan, and S. T. Rabani, “Detecting textual propaganda using machine learning techniques,” *Baghdad Science Journal*, vol. 18, no. 1, pp. 199–209, 2021.
- [8] A. Sharma, A. Kabra, and M. Jain, “Ceasing hate with moh: Hate speech detection in hindi–english code-switched language,” *Information Processing & Management*, vol. 59, no. 1, p. 102760, 2022.
- [9] A. Z. Miran and H. S. Yahia, “Hate speech detection in social media (twitter) using neural network,” *Journal of Mobile Multimedia*, vol. 19, no. 3, pp. 765–798, 2023.
- [10] M. Subramanian, V. Easwaramoorthy, G. Deepalakshmi, and J. Cho, “A survey on hate speech detection and sentiment analysis using machine learning and deep learning models,” *Alexandria Engineering Journal*, vol. 80, pp. 110–121, 2023.
- [11] A. Kamal, T. Anwar, V. K. Sejwal, and M. Fazil, “Bicapshate: Attention to the linguistic context of hate via bidirectional capsules and hatebase,” *IEEE Transactions on Computational Social Systems*, vol. 11, no. 2, pp. 1781–1792, 2024.
- [12] R. Panchal, P. Shelke, S. Ravandale, and S. Patil, “Shielding social media: Detection of toxic memes for automated moderation,” International Conference on Sustainability, Innovation & Technology (ICSIT), pp. 1–6, 2025, 10.1109/ICSIT65336.2025.11295293.

- [13] P. Kapil and A. Ekbal, "Transformer-based ensemble learning for hate speech detection leveraging sentiment and emotion knowledge sharing," Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, pp. 4485–4496, 2022.
- [14] Y. Shukla, R. V. Panchal, T. Nigade, S. Khodade, and P. Pimpalkar, "A multilingual bert-based framework for robust online hate speech detection," International Conference on Computer Vision and Machine Intelligence (CVMi), pp. 1–8, 2024, 10.1109/CVMi61877.2024.10782783.
- [15] A. Velankar, H. Patil, A. Gore, S. Salunke, and R. Joshi, "L3cube-mahahate: A tweet-based marathi hate speech detection dataset and bert models," 2022. [Online]. Available: <https://arxiv.org/abs/2203.13778>
- [16] V. Ibrahim, J. A. Bakar, N. H. Harun, and A. F. Abdulateef, "A word cloud model based on hate speech in an online social media environment," *Baghdad Science Journal*, vol. 18, no. 2, pp. 937–944, 2021.
- [17] Y. Kim, S. Park, and Y.-S. Han, "Generalizable implicit hate speech detection using contrastive learning," Proceedings of the 29th International Conference on Computational Linguistics, Gyeongju, Republic of Korea, pp. 6667–6679, 2022. [Online]. Available: <https://aclanthology.org/2022.coling-1.579/>
- [18] M. Mozafari, R. Farahbakhsh, and N. Crespi, "Hate speech detection and racial bias mitigation in social media based on bert model," *PLOS ONE*, vol. 15, no. 8, p. e0237861, 2020.
- [19] F. Barbieri, F. Ronzano, and H. Saggion, "What does this emoji mean? a vector space skip-gram model for twitter emojis," Proceedings of the Tenth International Conference on Language Resources and Evaluation (LREC'16), Portorož, Slovenia, pp. 3967–3972, 2016. [Online]. Available: <https://aclanthology.org/L16-1626/>
- [20] W. Ashiq, S. Kanwal, A. Rafique, M. Waqas, T. Khurshaid, E. Caro Montero, A. Bustamante Alonso, and I. Ashraf, "Roman urdu hate speech detection using hybrid machine learning models and hyperparameter optimization," *Scientific Reports*, vol. 14, p. 28590, 2024.
- [21] Q. Sifak and E. B. Setiawan, "Hate speech detection using cnn and bigru with attention mechanism on twitter," International Conference on Communication, Networks and Satellite (COMNETSAT), pp. 170–175, 2023, 10.1109/COMNETSAT59769.2023.10420628.
- [22] A. R. Jafari, G. Li, P. Rajapaksha, R. Farahbakhsh, and N. Crespi, "Fine-grained emotions influence on implicit hate speech detection," *IEEE Access*, vol. 11, pp. 105 330–105 343, 2023.
- [23] K. Maity, A. S. Poornash, S. Bhattacharya, S. Phosit, S. Kongsamlit, S. Saha, and K. Pasupa, "Hatethaisent: Sentiment-aided hate speech detection in thai language," *IEEE Transactions on Computational Social Systems*, vol. 11, no. 5, pp. 5714–5727, 2024.
- [24] K. Chavinda, P. Kalansooriya, T. Weerasuriya *et al.*, "An emotion aware context adaptive machine learning approach for detecting hate speech in social media," *Neural Computing and Applications*, vol. 38, p. 268, 2026.
- [25] F. Feng, Y. Yang, D. Cer, N. Arivazhagan, and W. Wang, "Language-agnostic bert sentence embedding," Proceedings of the 60th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Dublin, Ireland, pp. 878–891, 2022. [Online]. Available: <https://aclanthology.org/2022.acl-long.62/>
- [26] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam, P. Aggarwal, R. T. Nagipogu, S. Dave, S. Gupta, S. C. B. Gali, V. Subramanian, and P. Talukdar, "MuriL: Multilingual representations for indian languages," 2021. [Online]. Available: <https://arxiv.org/abs/2103.10730>
- [27] D. Kakwani, A. Kunchukuttan, S. Golla, G. N.C., A. Bhattacharyya, M. M. Khapra, and P. Kumar, "IndicNLPsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages," Findings of the Association for Computational Linguistics: EMNLP 2020, Online, pp. 4948–4961, 2020. [Online]. Available: <https://aclanthology.org/2020.findings-emnlp.445/>
- [28] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, "Unsupervised cross-lingual representation learning at scale," Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics, Online, pp. 8440–8451, 2020. [Online]. Available: <https://aclanthology.org/2020.acl-main.747/>
- [29] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1 (Long and Short Papers), Minneapolis, Minnesota, pp. 4171–4186, 2019. [Online]. Available: <https://aclanthology.org/N19-1423/>
- [30] P. Rathnayaka, S. Abeysinghe, C. Samarajeewa, I. Manchanayake, and M. Walpola, "Sentylic at iest 2018: Gated recurrent neural network and capsule network based approach for implicit emotion detection," Proceedings of the 9th Workshop on Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, Brussels, Belgium, pp. 254–259, 2018. [Online]. Available: <https://aclanthology.org/W18-6237/>
- [31] Y. Kim, "Convolutional neural networks for sentence classification," Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, pp. 1746–1751, 2014. [Online]. Available: <https://aclanthology.org/D14-1181/>
- [32] A. Graves and J. Schmidhuber, "Framewise phoneme classification with bidirectional lstm and other neural network architectures," *Neural Networks*, vol. 18, no. 5–6, pp. 602–610, 2005.