Chapter 6

Hate Speech Detection in Code-Mixed Social Media Conversations

DISCLAIMER: The content of this chapter may contain offensive material (verbatim from social media). Reader's discretion is advised.

Following the exploration of sentence-level sentiment analysis on code-mixed data, we transition to another research objective outlined in Section 1.8 of this thesis. This chapter undertakes an investigation into hate speech and offensive content identification on Hindi-English code-mixed data. The structure of this chapter is organized as follows. Section 6.1 offers an elaborate presentation of the problem statement to be addressed here. The dataset is comprehensively described in Section 6.2, accompanied by an exposition of the pre-processing techniques employed. Section 6.3 provides a detailed account of the proposed methodology and experimental setup. Subsequently, in Section 6.4, we present the results and conduct a comparative analysis of the performance against baseline model, in terms of standard evaluation metrics and ensuing discussion. Ultimately, our findings are encapsulated in Section 6.5, drawing conclusions from the

conducted study.

6.1 Problem Statement

In this chapter, we explore a problem inspired by the Hate Speech and Offensive Content Identification (HASOC) task, conducted at FIRE in 2021, 2022, and 2023. The HASOC shared task aims to identify offensive language to curb abusive behavior on social media. A particularly challenging scenario arises when dealing with conversational code-mixed data, which has become more common on these platforms in recent years. Our focus is specifically on identifying hate speech in conversational Hindi-English code-mixed data, as organized by the HASOC team. Detecting hate speech in such contexts introduces additional complexities due to the conversational nature of the data. Social media posts exhibit a hierarchical structure, typically comprising a post, comment, and reply, as illustrated in Figure 6.1.

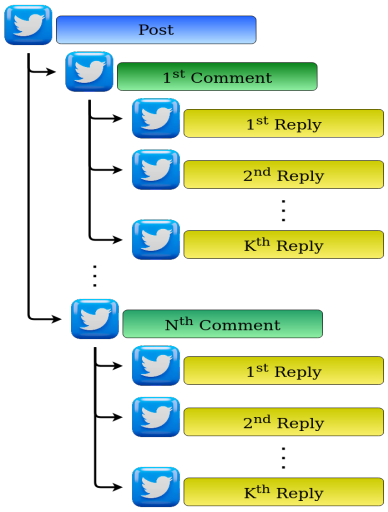

Figure 6.1: The structure of Conversational code-mixed Tweet

Multiple comments can be associated with each post, and each comment may receive several replies. In the context of Hindi-English code-mixed data, each component of this hierarchical structure can exhibit code-mixing between Hindi and English, appear

exclusively in English, exclusively in Hindi, in romanized (transliterated) Hindi, or as a combination of these forms. Consequently, complex input patterns emerge. The labels assign to replies or comments are significantly influenced by the contextual information provided by the parent text. For instance, as illustrated in Figure 6.2, although the content of the second comment appears neutral, it actually reinforces the hateful and offensive (HOF) tone of the post, thereby justifying its HOF designation. Moreover, in a sequence of multiple <post, comment, reply> tuples, identifying the relevant context becomes crucial, posing a significant challenge.

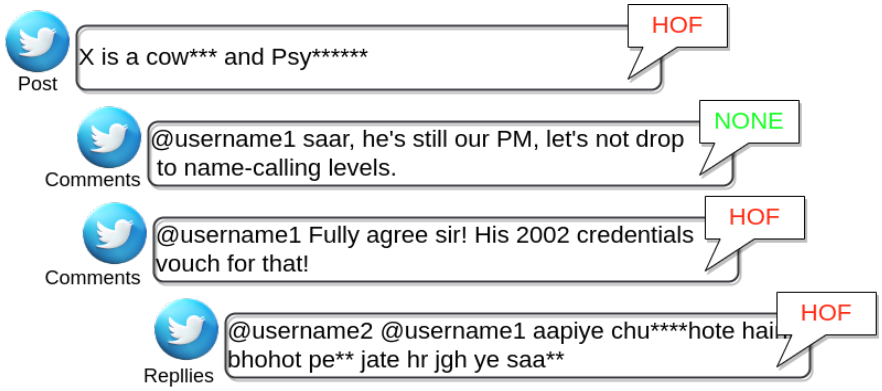

Figure 6.2: Example of conversational code-mixed data from Twitter

6.1.1 Tasks Description

The shared task has been conducted for three consecutive years. In 2021, the organizers named the task *Subtask 2: Identification of Conversational Hate-Speech in Code-Mixed Languages (ICHCL)*. This subtask focus on the binary classification of conversational tweets with tree-structured data into:

- **(NOT) Non Hate-Offensive:** This tweet, comment, or reply does not contain any hate speech, profanity, or offensive content.
- **(HOF) Hate and Offensive:** This tweet, comment, or reply contains hate speech, offensive, and profane content.

In 2022, the organizers modify the task as *Task-2*, which includes two sub-tasks:

Task 2A and Task 2B. Task 2A involves Hindi-English and German-English code-mixed binary classification, similar to the 2021 task. Task 2B, offered in Hindi-English code-mixed data, is a multi-class classification task. It further divides the HOF tweets into two sub-classes: SHOF and CHOF.

- **(SHOF) Standalone Hate:** Any tweet (post, comment, or reply) contains hate speech, offensive language, and profane words.
- **(CHOF) Contextual Hate:** Any tweet (comment or reply) reinforces the hate, offense, and profanity expressed in its parent tweet. This encompasses the explicit expression of hatred as well as the endorsement of such sentiments with positive language.
- **(NONE) Non-Hate:** Any tweet does not contain any form of hate speech, profane, offensive content.

In 2023, the task remains similar to the 2022 task, albeit with a different dataset. Throughout its iterations, the task encounters several research issues, prompting the organizers to offer the dataset for investigation and exploration. Each year, the organizers share the dataset, and researchers endeavor to address various research questions (RQs). In this study, we focus on the following RQs:

- *RQ-1:* What are the implications of classifying each tweet (post, comment, and reply) as a standalone sentence?
- *RQ-2:* To what extent does the context of a tweet, including its relationship to the post, comments, and replies, influence classification outcomes? What are the effects of concatenating the post, comment, and reply and treating them as a single sentence during classification?
- *RQ-3:* Can an ensemble model improve performance by assigning varying weights to the post, comment, and reply components?

6.2 Dataset

The HASOC organizers provide a Identification of Conversational Hate-Speech in Code-Mixed Languages (ICHCL) dataset comprising a training, and a test set. The organizers of the study disseminated the training datasets in the format of json (`.json`) files. Organiser shares three files for each tweet: `labels.json`, `binary_labels.json` and `contextual_labels.json`. The structure of `.json` file can be found from the shared task website [1]. The statistics of training, and test data corpus collection and their class distribution are shown in Table 6.1. The example texts from the datasets are shown in Table 6.2.

6.3 Methodology

This section provides a detailed description of the methodology and experimental setup employed in this study for the different RQs presented in Section 6.1. Here we first start with data pre-processing.

6.3.1 Data Pre-processing

Social media data displays a notable level of structural informality and is prone to noise owing to the colloquial style prevalent in Twitter conversations. This inherent attribute presents a potential challenge to the precision of processing techniques. As a result, it has been considered essential to subject all data to pre-processing procedures aimed at alleviating the influence of less informative textual elements.

The expansion of emojis and hashtags are conducted as part of the pre-processing pipeline. Hereafter, we offer a detailed enumeration of the pre-processing steps that are adopted.

- Perform cleaning by removing usernames, punctuation and URLs, mentions and

[1] `https://hasocfire.github.io/hasoc/2022/ichcl.html`

Table 6.1: Statistical overview of the datasets from ICHCL 2021, 2022, and 2023

HASOC 2021 SubTask-2				
Language Pair	**# of sentence**	**NONE**	**HOF**	
HI-EN (train)	5740	2899	2841	
HI-EN (test)	1348	653	695	
HASOC 2022 Task-1				
Language Pair	**# of sentence**	**NOT**	**HOF**	
DE-EN (train)	307	219	88	
DE-EN (test)	81	58	23	
HI-EN (train)	4914	2390	2524	
HI-EN (test)	996	483	513	
HASOC 2022 Task-2				
Language Pair	**# of sentence**	**NONE**	**SHOF**	**CHOF**
HI-EN (train)	4914	2390	1636	888
HI-EN (test)	996	483	350	163
HASOC 2023 Task-2A				
Language Pair	**# of sentence**	**NOT**	**HOF**	
HI-EN (train)	12998	6425	6573	
HI-EN (test)	998	-	-	-
HASOC 2023 Task-2B				
Language Pair	**# of sentence**	**NONE**	**SHOF**	**CHOF**
HI-EN (train)	5910	2873	1986	1051
HI-EN (test)	998	-	-	-

hashtags.

- Use `ekphrasis` which is a text processing tool, geared towards text from social networks, such as Twitter or Facebook. `ekphrasis` performs normalizing hashtags (for example, "#BlackLivesMatters" is segmented into "Black", "Lives",and "Matters").

The cleaned data is then subject to different steps as the research questions demanded. We detail the methodology adopted for each such RQ below.

Table 6.2: Example tweets from ICHCL 2021, ICHCL 2022 and ICHCL 2023 dataset for all classes (HI for Hindi, EN for English and Lang for Language)

Language	Sample tweet from ICHCL 2021	SubTask-2	
Hindi-English	@Samriddhi0809 We can count bewakoof, berozagar , anpad, jhahil, chu*** ya ss in the clip	HOF	-
	@Samriddhi0809 @police_haryana plzz sir look into this matter and do something to right BCOZ they spread hate	NONE	-
	Sample tweet from ICHCL 2022	**Task 1**	**Task 2**
	@Joydas @NSaina 2 rs k liye tweet kiya isne samjho bhai national hero hogi phle ab to izzat gawa di.. Andhbhakt ban gayi didi	HOF	CHOF
	@itsoutrageeyash @NSaina Yet another man telling a woman what to do and yet another telling her to #shutup. Shame on you Yash! #Feminism #Mansplaining #shethepeople	HOF	SHOF
	#Islamophobia Harish Ramkali, leader of the Bajrang Dal in Jind, Haryana posted a video descreating a Muslim mazar on the same land where a temple was built. As seen in the video he is using hateful and violent language against Muslims and the religion. https://t.co/q9dCjdFpne	NOT	NONE
	Sample tweet from ICHCL 2023	**Task 2A**	**Task 2B**
	@Joydas @NSaina 2 rs k liye tweet kiya isne samjho bhai national hero hogi phle ab to izzat gawa di.. Andhbhakt ban gayi didi	HOF	CHOF
	The only thing I want to say to the Islamophobic State.#SharjeelImam #Shaheen_Bagh #JNU #Chakaa_jaam_is_not_sedition #releaseallpoliticalprisoners @URL	HOF	SHOF
	@AUTHOR CAA also not against muslims	NOT	NONE

6.3.2 Methodology for RQ-1

We utilize the 2021 dataset, which involves a binary classification task, where text must be classified into two categories. We treat each tweet as an individual sentence for text classification. The implementation leverages HuggingFace's transformers library. We employ a pre-trained multilingual BERT model for tokenization and fine-tune it for the classification task. The maximum sequence length is set to 256. The classifier is trained with a batch size of 32 for 15 epochs. The dropout rate is set to 0.1, and we use the AdamW optimizer with a learning rate of 2e-5. The prediction file for the text data is submitted to the organizers under the name `CM_submission_1`.

6.3.3 Methodology for RQ-2

The 2022 dataset includes two subtasks. Task 1 mirrors the 2021 task, while Task 2 involves multi-class classification. To generate the final text sequence, we concatenate the tweet with its comments and replies. Specifically, for posts, we use only the post; for comments, we concatenate the post and the comment; and for replies, we concatenate the post, comment, and reply. Task 1 consists of two language pairs: Hindi-English and German-English. We fine-tune pre-trained language models (mBERT, XLM-RoBERTa, GermanBERT[2]) for binary and multi-class classification tasks. For each subtask, we submit different models. Below are the descriptions of each submission, identified by a unique submission name.

6.3.3.1 Submission for Task 1 (Binary classification)

1. **submission-task1-1:** For German-English code-mixed data, we use mBERT with a maximum sequence length of 256 tokens and a batch size of 32. For Hindi-English code-mixed data, we use XLM-RoBERTa with a maximum sequence length of 512 tokens and a batch size of 16.
2. **submission-task1-2_t:** For German-English code-mixed data, we use GermanBERT with a maximum sequence length of 128 tokens and a batch size of 32. For Hindi-English code-mixed data, we use XLM-RoBERTa with a maximum sequence length of 512 tokens and a batch size of 16.

6.3.3.2 Submission for Task 2 (Multi-class classification)

1. **submission-task2-1_t:** We use XLM-RoBERTa with a maximum sequence length of 512 tokens, a batch size of 16, and 10 epochs.
2. **submission-task2-2:** We use mBERT with a maximum sequence length of 512 tokens and a batch size of 16. All parameters of the pre-trained model are frozen,

[2] `https://www.deepset.ai/german-bert`

and early stopping criteria are applied. To address class imbalance, we use balanced classweight.

3. **submission-task2-3:** We use mBERT with a maximum sequence length of 512 tokens and a batch size of 16. A focal loss function is utilized in this submission.
4. **submission-task2-4:** We use XLM-RoBERTa with a maximum sequence length of 512 tokens and a batch size of 16.
5. **submission-task2-5:** We use XLM-RoBERTa with a maximum sequence length of 512 tokens, a batch size of 32, and 2 epochs.

6.3.4 Methodology for RQ-3

For the binary classification task, the 'HOF' labels are converted to integer '1', representing instances of harmful or offensive content, while the 'NOT' labels are converted to integer '0', indicating non-harmful content.

This section delineates the methodology employed for Task 2A and Task 2B of the HASOC shared task, with a focus on the ICHCL 2023. Task 2A presents a binary classification of conversational tweets with tree-structured data to determine whether the content contains hate speech, offensive language, or profanity (HOF), or if it falls under the category of non-hate and offensive (NOT). Unlike conventional classification problems, we can not rely on pre-trained models due to the unique nature of this task, where the contextual relevance of preceding posts or replies is of paramount importance. Each classification decision is contextually driven: comments are evaluated in the context of their parent posts, and replies are contextualized within the broader framework of the main post, alongside the specific comment to which the reply pertains. Therefore, we fine-tune the mBERT model, as discussed below.

6.3.4.1 Fine-tuning mBERT

This section details the fine-tuning approach employed for Task 2A, a task that requires specialized techniques due to its unique characteristics. Unlike conventional classification tasks, where pre-trained models are often directly used, this task necessitates a fine-tuning process tailored specifically for pairwise tweet classification.

- **Dataset Construction:** To fine-tune the model effectively, we first curate a specialized dataset based on anchor points, pairing each anchor tweet with a positive or negative counterpart. The pairs are constructed as follows: if a parent tweet and any of its child tweets share the same label, the pair is assigned a label of 0 (denoting semantic similarity), while pairs with differing labels are assigned a label of 1 (indicating semantic dissimilarity). This dataset construction methodology is crucial for training the model to distinguish between similar and dissimilar tweet pairs (as depicted in Figure 6.3).

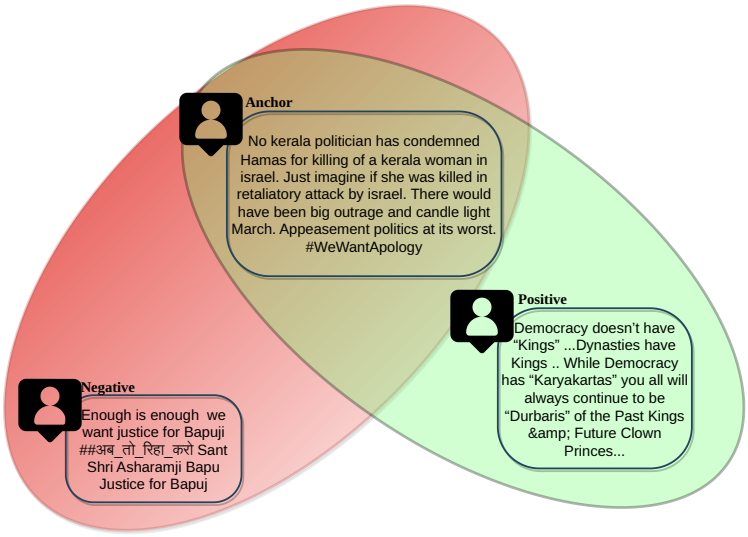

Figure 6.3: Creation of datasets using anchor-positive and anchor-negative pair

- **Siamese Network Architecture:** For the fine-tuning process, we utilize a pre-trained mBERT model and apply a Siamese network architecture. A Siamese network is particularly well-suited for tasks requiring pairwise comparisons, as it enables the comparison of two input sequences (in our case, tweet pairs) by passing them through the same neural network. The network consists of a 768-dimensional contextual embedding layer (output from mBERT), which serves as input to a dense layer of 128 units (as illustrated in Figure 6.4). Both tweet pairs (the parent tweet and its child tweet) are processed through this identical network structure, ensuring that the representations for each pair are derived in the same manner.

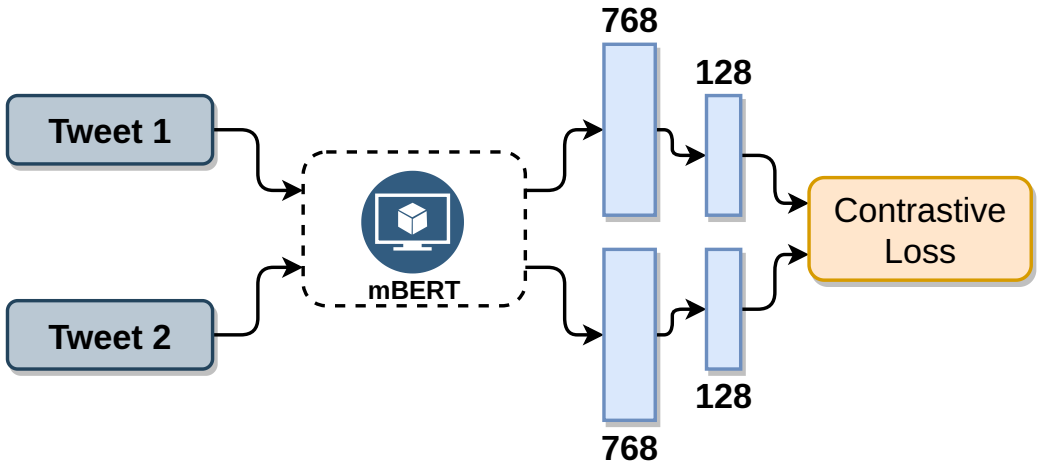

Figure 6.4: Architecture diagram for fine-tuning mBERT

The goal of the network is to generate similar contextualized embeddings (CLS tokens) for tweet pairs labeled as 0 (semantically similar) and distinct embeddings for pairs labeled as 1 (semantically dissimilar). This approach allows the model to learn nuanced semantic differences between tweets, making it more effective in distinguishing positive and negative tweet pairs.

- **Contrastive Loss Function:** To train the Siamese network, we employ a contrastive loss function, which is designed to minimize the distance between em-

beddings of similar tweet pairs and maximize the distance between dissimilar pairs. The mathematical formulation of the contrastive loss function is provided in Equation 6.1. This loss function is ideal for tasks involving pairwise comparisons, as it directly optimizes the model to differentiate between semantically similar and dissimilar tweet pairs.

$$\mathcal{L}(x_1, x_2, y) = \frac{1}{2}(1-y) \cdot d(x_1, x_2)^2 + \frac{1}{2} y \cdot \max(0, m - d(x_1, x_2))^2 \tag{6.1}$$

where,

$$\begin{aligned} x_1, x_2 &= \text{Input vectors / features (128-d vectors),} \\ y &= \text{Binary similarity label,} \\ d(x_1, x_2) &= \text{Distance (Euclidean) between } x_1 \text{ and } x_2, \\ m &= \text{Margin parameter,} \end{aligned}$$

- **Hyperparameter Tuning:** Due to computational limitations, we opt for a batch size of 8 during training. After experimentation, we determine that a learning rate of 0.001 is optimal for fine-tuning the model. This balance between computational constraints and model performance ensures that the network is trained effectively within the available resources.

6.3.4.2 Submission for Task 2A

Expanding upon the fine-tuned mBERT model, we augment our primary architecture to enhance the classification of hate speech and offensive content in Task 2A of the HASOC shared task. This architecture synergistically leveraged the capabilities of mBERT with the sentence transformer [102], with a maximum token length of 160 tokens taken into consideration.

In our model architecture, we adhere to the structure of mBERT, integrating a sequence of LSTM layers. Specifically, we employ a bidirectional LSTM with 512 units, succeeded by two unidirectional LSTM layers with 512 and 256 units, respectively. The output from the final LSTM layer and the output from SentenceBERT are concatenated. Because dataset encompass three distinct data types: Posts, comments, and replies, each characterized by its own unique attributes. To accommodate these variations, our architecture seamlessly integrate all three data channels, as delineated in the architectural diagram (Figure 6.5). Post, comment and reply are multiplied with their important score (α for post, β for comment and γ for reply). Afterwords a feed-forward network is introduced comprising a dense layer with 1024 neurons, followed by a dropout layer with a probability of 0.3, and subsequent dense layers with 256 and 32 neurons. Finally, a single-layer neuron with a sigmoid activation function is utilized for the output. This configuration is devised to optimize both contextual and semantic understanding within our research framework.

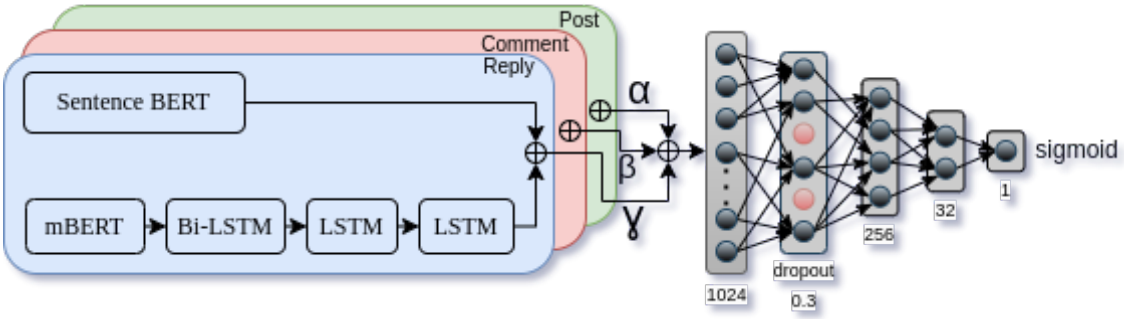

Figure 6.5: Architecture of proposed methodology for Task 2A

For training purposes, the data is divided into an 80% training set and a 20% validation set. A batch size of 8 is selected during training. To make classification decisions, a threshold of 0.5 is applied to the model's output probabilities.

Throughout the training phase, 50 epochs are executed with a learning rate of 0.09, utilizing the Stochastic Gradient Descent (SGD) optimizer. We get a validation accuracy of 83.19%, underscoring the efficacy of our architecture in identifying hate

speech and offensive content within code-mixed conversations on social media. Based on the best-performing model, we generate the prediction file for the text data and submit it to the organizers.

6.3.4.3 Submission for Task 2B

An ensemble approach is employed, leveraging the collaborative operation of two distinct models. The first model utilise `BertForSequenceClassification`, specifically selected for its proficiency in identifying abusive language in code-mixed data within sentences, as evidenced by its applicability to the hate speech Hindi code mixed abusive dataset. For our task which might contain hate speech without an explicit use of abusive word could still be a HOF because of presence of sarcastic comments to a race, people or gender etc. For this we need an additional classification model (see Figure 6.6) to better understanding the context of post, comment and reply.

The second model underwent a comprehensive architectural transformation, distinguishing itself from the initial model. This architectural evolution encompasses the integration of several pivotal elements. It commence with the inclusion of word-level embeddings from MuRIL, subsequently pass to a Bi-LSTM layer featuring 512 units, thereby enabling the model to effectively capture bidirectional contextual information. After Bi-LSTM, two LSTM each with 256 units are added to capture sequential patterns. Expanding upon the architectural framework, a dense layer comprising 32 units is incorporated, facilitating the extraction and abstraction of features. This is succeeded by the inclusion of a final dense layer housing a single neuron and employing the sigmoid activation function, primarily responsible for generating the second model's output. Throughout the training phase, the model underwent fine-tuning, with a learning rate set at 5e-6, and was trained using a batch size of 8. It should be noted that throughout the training process the last two layers of MuRIL used in second model were kept unfrozen.

The ensemble strategy rely on the synergistic collaboration between these two models, each contributing its unique strengths to effectively address the classification task. The output from both the models are added. If total sum is >=1 then the output of ensemble model is 1 (means abusive) else 0 (means not abusive). Because both model give output from 0 to 1 so if a sentence have either just abusive word (in Hindi or English written in any script) or if it just contain a hate speech without any abusive word in both cases total sum is greater than 1 making the final output of ensemble model as HOF.

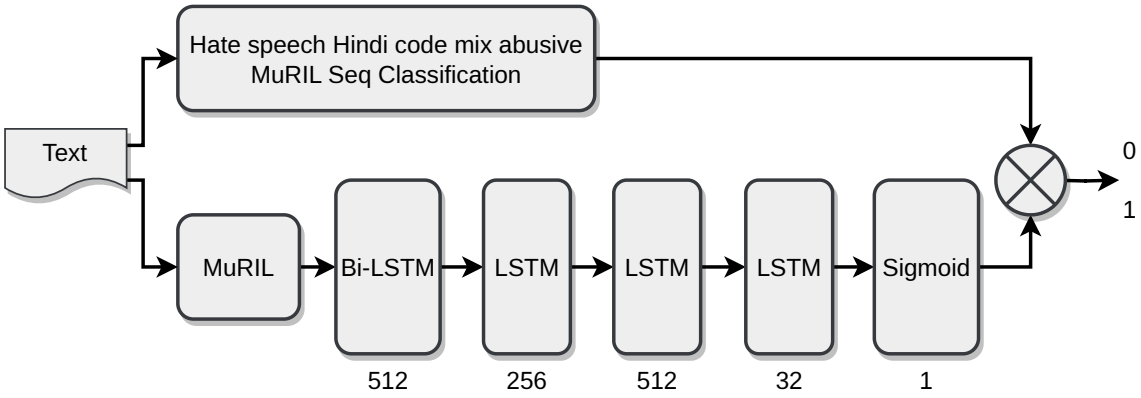

Figure 6.6: Architecture of proposed methodology for Task 2B

Now for the final prediction, a rule-based approach is adopted between the ensemble model given here and the model stated in Task 2A. The Table 6.3 describes the rule-based approach used to get final prediction.

Table 6.3: Rule-based approach for Task 2B

Prediction from Task 2A Model	**Prediction from Task 2B Model**	**Final Prediction**
NOT	NOT	NOT
NOT	HOF	SHOF
HOF	NOT	CHOF
HOF	HOF	CHOF

The process involve evaluating predictions from both models to ascertain the definitive classification. For instance, in cases where model 2A predict “NOT” while model

2B predict "HOF," the final prediction is categorized as "SHOF." Conversely, if model 2A predict "HOF," regardless of model 2B's prediction, the output is designated as "CHOF." This underscores the precedence of contextual hate over standalone hate within our classification framework. We create the submission file on text data and submit that to organiser for evaluation.

Both the models are validated on the training and development sets due to the limited amount of data available for training. Subsequently, the prediction files are submitted on the test data to obtain the final results.

6.4 Results and Discussion

This section reports experimental results conducted to address the RQs on the Hindi-English language pairs followed by discussion on the findings. We evaluate and compare every model's performance in terms of macro F_1 -score.

6.4.1 Results for RQ-1

The primary objective of RQ-1 is to explore the implications of classifying each tweet as a standalone sentence. As shown in Table 6.4, our model outperforms the official baseline model provided by the organizers.

Table 6.4: Evaluation results on ICHCL 2021 test data (Submission number in bracket)

Language Pair	**Subtask #**	**Team Name**	**Macro F_1 score**
English-Hindi	2	HASOC Baseline	0.55
		IRLab@IITBHU (1)	**0.68**

6.4.2 Results for RQ-2

In this section, we examine the effect of concatenating posts with comments, or posts with comments and replies. Table 6.5 displays the performance of our model for both

tasks. For binary classification (Task 1), our model surpasses the baseline model. However, for multi-class classification (Task 2), our model does not exceed the baseline performance.

Table 6.5: Evaluation results on ICHCL 2022 test data (Submission number in bracket)

Language Pair	**Subtask #**	**Team Name**	**Macro F_1 score**
HI-EN, DE-EN	Task 1	HASOC Baseline	0.58
		irlab@iitbhu (1)	0.62
		irlab@iitbhu (2)	**0.63**
HI-EN	Task 2	HASOC Baseline	0.49
		irlab@iitbhu (1)	**0.44**
		irlab@iitbhu (2)	0.31
		irlab@iitbhu (3)	0.39
		irlab@iitbhu (4)	0.39
		irlab@iitbhu (5)	0.38

6.4.3 Results for RQ-3

In 2023, Task 2 focus on Hindi-English code mixed data, and our team's performance is notable. Table 6.6 shows our official performances on the test data as shared by the organizers [127]. In task 2A, we achieve a macro F_1 score of 0.7008. In task 2B, we achieve a score of 0.5631. The factor contributing to laser score for task 2B, is the limited training data for this task as compared to task 2A. "IRLab@IITBHU" demonstrate competitive performance in both subtasks. The organizers of this task also shared the complete results of both tasks, along with the names of the participating teams, which can be found in Table 6.7 and Table 6.8, respectively.

6.4.4 Discussion

The findings from our research provide valuable insights into hate speech detection in code-mixed conversational data and offer practical implications for handling such datasets effectively. The three research questions are interconnected and collectively

Table 6.6: Evaluation results for Task 2A and 2B on Hindi-English test data (ICHCL 2023)

Language Pair	Subtask #	Team Name	Macro F_1 score
Hindi-English	Task 2A	HASOC Baseline	0.3743
		IRLab@IITBHU (1)	**0.7008**
Hindi-English	Task 2B	HASOC Baseline	0.2495
		IRLab@IITBHU (1)	**0.5631**

Table 6.7: ICHCL Task 2A results published by organiser

Rank	Team Name	Submission Name	F_1 score	Precision	Recall
1	FiRC-NLP	parfirst2_all_folds	0.80791	0.80844	0.80741
2	IRLab@IITBHU	IRLab@IITBHU_Task2A_1	0.70079	0.70255	0.69949
3	Chetona	chetona-2a-def2	0.61551	0.62525	0.61425
4	AiAlchemists	task2_binary_test_pred_2	0.61466	0.63351	0.60820
5	MUCS_3	MUCs_run_2	0.43474	0.38456	0.500
6	HASOC	BASELINE	0.37429	0.29909	0.500

Table 6.8: ICHCL Task 2B results published by organiser

Rank	Team Name	Submission Name	F_1 score	Precision	Recall
1	FiRC-NLP	parfirst_top3_top7_task2b	0.65414	0.64334	0.67178
2	IRLab@IITBHU	IRLab@IITBHU_Task_2B_1	0.56316	0.56872	0.56685
3	AiAlchemists	task_multiclass_1	0.38243	0.39198	0.39212
4	HASOC	BASELINE	0.24952	0.19939	0.33333
5	Chetona	chetona_2b_def2	0.17263	0.20795	0.15883

contribute to a comprehensive understanding of hate speech detection in code-mixed datasets.

In RQ-1, we apply mBERT and report a 23.6% improvement in performance over the baseline. Figure 6.7 shows the confusion matrix on ICHCL 2021 test data for English-Hindi code-mixed language. It is evident that a significant number of tweet actually labeled as HOF are predicted as NONE by our model. This misclassification occurs because we did not consider the context; each tweet was treated as a standalone instance. Consequently, if a tweet supports any hate content but does not explicitly contain hateful or offensive words, it is classified as NONE.

In RQ-2, we concatenate posts with comments and posts with comments and replies

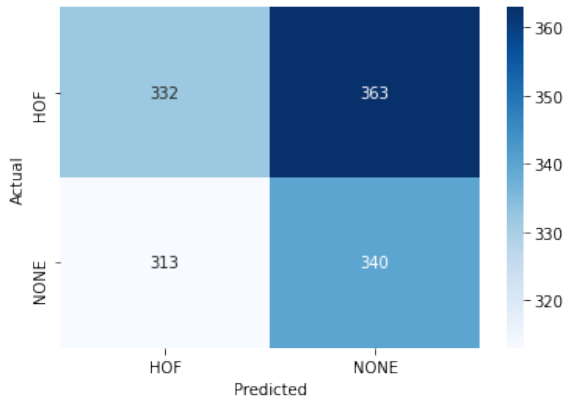

Figure 6.7: Confusion matrix on ICHCL 2021 test data for English-Hindi code mixed language (Submission 1 for Subtask 2)

and conduct experiments for both tasks. For Task 1, our model reports an 8.6% improvement in performance over the baseline. However, for Task 2, we encounter a performance drop from baseline. Our model frequently predicts maximum comments and replies as single-alone hate and offensive (SHOF). This bias arises because concatenating non-hate content with hate content skews the sentence toward hate. Our assumption is that concatenation would help the model better comprehend the context, particularly in cases where the remark or reply is not hateful but supports the hateful parent tweet. However, this assumption do not hold true as expected.

In RQ-3, we conduct a series of experiments to optimize the approach. One crucial aspect explore is the weighting of different components, namely the post, comments, and replies, when making predictions. It is observed that assigning appropriate weights to these components can greatly enhance the model's understanding of context. After experimenting with values between 0.1 and 1.0, with a step size of 0.1, we found that the optimal combination of importance scores is achieved when assigning alpha (contextual weighting coefficient for posts) a value of 0.3, beta (contextual weighting coefficient for comments) a value of 0.1, and gamma (contextual weighting coefficient for replies) a value of 0.3. This finding implies that, when predicting on reply data, it is imperative to afford equal importance to both the post and reply elements while assigning

comparatively less weight to comments. This strategic allocation of weights effectively captures and leverages contextual nuances within the data, ultimately resulting in the optimal scores achieved in this task.

We experiment with different numbers of LSTMs, ranging from 1 to 4. It was observed that there is only a slight improvement with an increasing number of LSTMs. Thus, using one Bi-LSTM and two LSTMs appears to be the best choice when constrained by computational resources.

In our quest for the optimal optimizer, our experimentation indicates that Stochastic Gradient Descent (SGD) with a slightly higher learning rate converges more rapidly. Conversely, the AdamW optimizer with a higher learning rate exhibits a zigzag convergence pattern. Notably, AdamW performs optimally with a lower learning rate, typically around 1e-5. However, it is important to recognize that SGD with a marginally higher learning rate can be a pragmatic choice for quick model testing, particularly in the context of Transformer-based models. This approach provides insights into a model's convergence tendencies before committing to more computationally intensive optimization methods.

6.5 Summary of this Chapter

The exponential growth of user-generated data in the era of Web 2.0 has necessitated advanced techniques for detecting hate speech and offensive content. While numerous studies have focused on hate speech detection in monolingual datasets, there is a scarcity of research on code-mixed datasets, particularly in the context of code-mixed conversational data. Our research reveals that treating each tweet as a standalone sentence is not the most effective solution. Subsequently, we applied the concatenation of posts, comments, and replies, which also proved suboptimal. We then explored an ensemble method, which showed more promising results.

Through our investigation into the detection of conversational hate speech, several

key insights emerged. The inherent challenge of managing text inputs of variable lengths led us to adopt a strategic approach involving distinct feature extraction from posts, comments, and replies. This pragmatic methodology effectively addressed the linguistic diversity within our dataset, thereby mitigating potential limitations associated with conventional models such as BERT, which imposes a maximum token limit of 512 tokens for both input and output.

The exponential growth of user generated data by social media users in the era of Web 2.0 has necessitated advanced techniques for hate speech and offensive content detection. While numerous studies have focused on hate speech detection in monolingual datasets, there is a scarcity of research in code-mixed datasets, specially in code-mixed conversations data. We observe that classifying each tweet as a single, standalone sentence is not the best solution. Next we apply concatenation of post, comments and reply, which also not a best solution. We then proceed with an ensemble method, which shows somewhat promising results. Throughout our investigation into the detection of conversational hate speech, several insights have come to light. The challenge inherent in this task, namely the management of text inputs of variable lengths, prompted the adoption of a strategic approach: distinct feature extraction from posts, comments, and replies. This pragmatic methodology effectively addressed the linguistic diversity within our dataset, thereby mitigating potential limitations associated with conventional models such as BERT, which impose a maximum token limit of 512 tokens for both input and output.