Journal of Theoretical and Applied Information Technology
31st March 2026. Vol.104. No.6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

A FUSION-BASED SENTIMENT CLASSIFIER FOR HINGLISH USING CONTEXTUAL ENCODERS AND COMMONSENSE INFERENCE

KISHORE KUMAR P. V[1], HARI JYOTHULA[2]

[1]Research Scholar, Department of CSE, Aditya University, Surampalem, India
[2]Associate Professor, Department of CSE, Aditya University, Surampalem, India
[1]vkkpotti@gmail.com, [2]dr.jyothulahari@gmail.com

ABSTRACT

Sentiment analysis of code-mixed text like Hinglish is still a tough problem. People who use social media often mix Hindi and English. The grammar isn't very formal and isn't always the same. People often show how they feel without saying it directly. A lot of the sentiment analysis models that are already out there depend mostly on the text's context. These models have a hard time figuring out how people feel when they show it through situations or cultural cues. This limit causes incorrect classification. This issue happens more often in content that is neutral or has a negative tone. The goal of this study is to make it easier to classify feelings in Hinglish text. The work focuses on dealing with emotional expressions that aren't clear. A framework for classifying sentiments based on fusion is suggested. The framework brings together contextual embeddings and external commonsense knowledge. Contextual embeddings record how words are used in the text. Common sense knowledge helps us figure out how people feel about everyday things. The Hinglish SentiMix dataset, which has about 20,000 tweets that are marked as positive, negative, or neutral, is used to test the proposed model. The results indicate that the suggested method attains a weighted F1-score of 74.1%. This score is better than those of well-known transformer-based baseline models. The results show that adding commonsense reasoning makes it easier to understand sentiment. The improvement is big for feelings that are not directly stated. The study shows how useful it is to combine linguistic context with outside knowledge. This method works well for analyzing multilingual social media in the real world.

Keywords: *Hinglish Sentiment Analysis, Code-Mixed Text, Contextual Embeddings, Commonsense Knowledge, Fusion Model.*

1. INTRODUCTION

Sentiment analysis has become a useful way to learn about what people think on digital platforms. A lot of the early work was focused on English, but people who use social media in multilingual societies often switch between languages, which can lead to mixed forms like Hinglish [1]. Hinglish sentiment analysis is worth looking into because it shows how people talk to each other in real life. Many users share their thoughts using a mix of languages and informal phrases. In opinion mining and social analysis, misinterpreting sentiment can lead to false conclusions. This issue impacts applications like monitoring public opinion and analyzing customer feedback. Even though multilingual models have gotten better, they still don't do a good job of handling implicit sentiment in Hinglish. To make practical sentiment analysis systems better for use in the real world, we need to fill this gap [2].

Traditional machine learning models depend heavily on hand-crafted features and basic statistical patterns. These methods fail when the input conveys indirect sentiment or when the emotional significance is cross-linguistic. Deep learning and systems that use transformers have also helped by learning about how words in a text relate to each other. But even strong models like Bidirectional Encoder Representations from Transformers (BERT) and Cross-Lingual Language Model (XLM-R) have trouble with code-mixed content because they only look at what is written and not what is meant. Hinglish posts often use cultural references and everyday logic that regular text encoders can't handle. For example, the phrase "Mood off hai since morning" clearly has a negative connotation, but the meaning goes deeper because of what we know about emotions [3].

Even though there are strong transformer-based models, sentiment analysis of Hinglish text is

still a problem that hasn't been solved. This is because a lot of Hinglish posts show how people feel by talking about situations instead of using words that show how they feel. To understand these kinds of expressions, you need to know about everyday events and how people feel about them. This is because existing models don't have this reasoning ability, which leads to a lot of neutral and implicitly negative content being incorrectly classified. This limitation continues to make sentiment analysis systems less reliable in real-world multilingual social media settings.

Recent research indicates that integrating contextual comprehension with external knowledge can substantially enhance the robustness of sentiment interpretation. Inspired by these findings, our study adopts a hybrid architecture that integrates two complementary modules. The first one is the text encoder, which uses a transformer-based model to read the Hinglish sentence and make contextual embeddings [4]. This encoder can deal with spelling mistakes, word mixing, and informal phrasing, which helps it pick up on language patterns better. For instance, in a tweet like "Kal ka match dekhke dil khush ho gaya," the text encoder picks up on positive cues from the mixed-up expression.

The knowledge encoder is the second module. It gets the relevant commonsense information from outside sources like COMET or ConceptNet. This encoder helps the system understand the actions, feelings, and everyday situations that are talked about in the text. For instance, the knowledge encoder connects inputs like "tired," "down," or "no energy" to either a sad or frustrated emotional state. This information is important because there are a lot of Hinglish sentences where opinions are expressed [5] indirectly, using cultural understanding instead of clear sentiment words. So, by adding common-sense signals, the model can better understand what someone means.

A simple fusion layer of concatenation combines the outputs of the text encoder and the knowledge encoder. This fusion combines language patterns with outside reasoning to give a fuller picture of the input. After that, the merged embedding is sent to a classifier that guesses whether the sentiment is positive, negative, or neutral. The fusion method is kept simple on purpose so that both sources of information can add to the model without making it too complicated. The fused system is better at figuring out the sentiment in mixed-language expressions where the relative sentiment depends on both execution and expression [6]. This is because it takes the best parts of each module and combines them. This design is a direct application of the principles of hybrid contextual-commonsense modeling from recent literature.

We test this architecture on the Hinglish part of the SentiMix dataset [7], which has almost 20,000 real posts from social media. Some examples of this are everyday code mixing like "Aaj traffic ne pura mood spoil kar diya" or "Yaar, kal ki movie was too good." This dataset is good for this study because it shows how people naturally use Hindi and English in English in these kinds of online conversations. Our hybrid model performs adequately with these examples and exhibits significant enhancements when the sentiment is derived from contextual or practical knowledge. The findings indicate that the integration of contextual embeddings with commonsense reasoning provides a more dependable approach to addressing the issue of sentiment analysis in Hinglish [8].

This research provides three principal contributions. First, it shows how text-only transformer models can't work well for sentiment analysis of code-mixed Hinglish text, especially when the sentiment is not directly stated. Second, it suggests a fusion-based framework that combines contextual embeddings with external commonsense knowledge to get around this problem. Third, it uses a standard Hinglish benchmark dataset to show that it works better than established transformer-based baselines. All of these contributions help make sentiment analysis better for real-world code-mixed social media text.

This research concentrates on the sentiment classification of code-mixed Hinglish text derived from social media. It tackles the problem of implicit sentiment by using contextual embeddings and common sense reasoning. The study does not seek to address sarcasm detection, emotion classification, or multilingual sentiment analysis beyond Hinglish. It also doesn't focus on speech, multimodal data, or expanding languages with few resources. The scope is restricted to text-based sentiment analysis utilizing a benchmark Hinglish dataset within conventional experimental parameters.

2. RELATED WORK

Research on sentiment analysis in multilingual and informal contexts has been

progressively increasing, driven by social media platforms that encourage spontaneous and mixed-language communication. Initial research utilized conventional machine learning models that depended on manually crafted features such as n-grams, part-of-speech patterns, and basic polarity lexicons. These methods worked well on monolingual text that was well-formed, but they didn't work on code-mixed languages like Hinglish, where people often mix different scripts and grammatical structures and use very informal spellings. This was a motivating factor in the development of deep learning approaches that can pick up on more complicated contextual cues. The model, which used recurrent neural networks and CNN, worked better, but it still had problems because it had to deal with transliterated Hindi and inconsistent token patterns, which are usually found in user-generated content [9].

This was significantly changed with the introduction of transformer-based architectures. Because of their ability to encode both contextual meanings and long-range dependencies, BERT, multilingual BERT, and XLM-R have drawn a lot of attention for code-mixed sentiment tasks. Multilingual encoders are suitable for processing mixed-language text because they can recognize Hindi and English tokens into common semantic elements, according to statistics using mBERT and XLM-R. Even these potent models, however, have issues when events rather than overtly expressed emotions are used to convey sentiment. This discrepancy was evident in benchmark evaluations, where posts containing sarcasm or disappointment as well as cultural allusions despite the fact that they were clearly given contextual embeddings were frequently misclassified [10].

To address these deficiencies, recent research has focused on the incorporation of external knowledge sources, including commonsense reasoning and concept graphs. Systems that use COMET or ConceptNet have been better at tasks where the models have to figure out what emotions are based on background knowledge instead of just the words used. For instance, sentences that talk about things like losing a job or failing an exam need to be understood in the context of the world, which language models may not always be able to do. Hybrid architectures that combine textual embeddings with inferencing knowledge have been demonstrated to effectively address this gap, particularly for low-resource languages or all programming languages. In this context, our work suggests a fusion framework that uses IndicBERT to get deep contextual features and COMET to store commonsense inferences. This combination tries to deal with the subtle and culturally influenced sentiment cues that are common in Hinglish communication so that we can fully understand emotion in code-mixed text [11].

3. OBJECTIVES

This study's primary objective is to improve sentiment analysis in authentic Hinglish text, where people frequently mix Hindi and English and subtly express emotions. This type of writing is not well handled by current models, especially when sentiment is expressed through events, cultural allusions, or colloquial language. In order to overcome these challenges, our suggested system is designed and tested with the following goals in mind.

- To develop a model capable of processing real code mixed text that frequently switches between languages, has mixed grammar, and irregular spellings. This ensures that the system is aware of Hinglish's structure and does not rely solely on surface patterns.
- To be able to consider external reasoning so that the model can interpret sentiment that is indirectly expressed through situations or events. This is useful for identifying feelings that aren't mentioned directly in the text.
- To assess IndicBERT, XLM-R, and COMET separately and determine which models perform well or poorly on code-mixed sentiment tasks. The gaps that motivate a hybrid design are the main focus of this analysis.
- To combine the sense-making ability of commonsense with the integrity of contextual embeddings in a cohesive architecture. By considering both the literal meaning and the emotional connotation, the goal is to improve the quality of predictions.

When combined, these objectives offer a concentrated effort toward a strong sentiment analysis in real-world Hinglish situations where linguistic blending and cultural context frequently make lingual interpretation challenging. The study can be published by combining contextual modeling and common sense reasoning in an effort to close the

gaps in current transformer-based systems. Improving a significant and quantifiable process can be ensured by evaluating it in terms of established benchmarks. All things considered, these objectives provide a clear path for creating a more dependable and context-aware sentiment analysis model for code-mixed text in social media.

4. RESEARCH HYPOTHESIS

This study is predicated on the idea that contextual language representations combined with outside common sense knowledge can enhance sentiment analysis of Hinglish text. It is believed that contextual embeddings by themselves are inadequate for capturing sentiment that is subtly conveyed through events or circumstances. The suggested method is anticipated to achieve more accurate sentiment classification by utilizing commonsense reasoning, especially for neutral and implicitly negative cases.

5. PROPOSED METHODOLOGY

From a conceptual perspective, Hinglish sentiment analysis cannot be regarded as solely a linguistic endeavor. People often use their common sense to figure out how they feel about things and events. This research employs a hybrid approach that integrates contextual language modeling with external commonsense reasoning. The suggested method seeks to encompass both linguistic patterns and underlying emotional significance, thus overcoming the shortcomings of text-only sentiment models. The study adheres to a systematic experimental methodology. The Hinglish text is first preprocessed and set up for use in the model. Second, contextual embeddings are made to find patterns in language in code-mixed text. Third, we use outside common sense knowledge to figure out hidden emotional cues. Then, a fusion strategy is used to combine these representations. At last, standard classification metrics are used to train and test the model. This process makes sure that the evaluation is systematic and that the comparison with other methods is reliable [12-14].

5.1 Dataset Description

This research utilizes a dataset derived from the Hinglish sentiment corpus, a collection of approximately 20,000 tweets encoded in Roman (Uttah) script, released during the SemEval 2020 SentiMix task. Each tweet has a sentiment label that says whether it is positive, negative, or neutral, and it also has language tags that show Hindi and English words separately. The tweets use language that is typical of social media: informal spellings, slang, abbreviations, and emojis that often have emotional meanings. The data set keeps a moderate level of language switching, with a Code-Mixing Index of about 25, which makes the text hard and realistic for multilingual sentiment analysis. These traits make the corpus good for testing models that need to understand both direct and implied sentiment in communication that uses more than one language [15].

5.2 Preprocessing

The dataset undergoes a process of meticulous data selection prior to model training in order to guarantee that all calculations require superfluous noise and do not contain significant content. When they don't affect the emotional interpretation, elements like URLs, user tags, or unnecessary punctuation are removed. Since they frequently convey intense emotions, emojis and other symbols of expression are kept unaltered. For consistency, all of the text is lowercased; however, the force setting is set aggressively but not aggressively to compensate for the possible loss of sentiment information embedded in user-generated spelling patterns [16].

Roman-scripted Hindi words are kept in their original form despite their culturally specific meanings. Small repetitions are kept because they are emphasis, but repeated characters are rounded down slightly to reduce noise. For example, a person's tweet, "yeh movie mast thi yaar 😂," retains its authenticity, and an unrestricted model would identify the casual excitement and feelings. This kind of preprocessing is known as the "bath and tide approach," and when it is applied, cleanliness and naturalness become the ultimate quality that permeates all downstream components [17].

Journal of Theoretical and Applied Information Technology
31st March 2026. Vol.104. No.6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

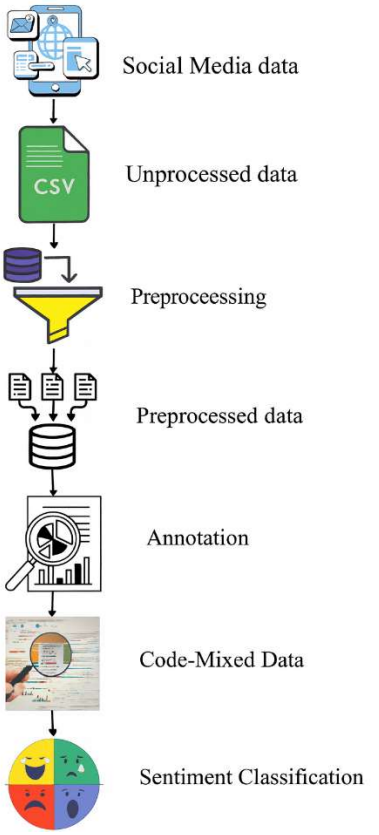

Figure 1: Code-Mixed Sentiment Dataset Generation Process

The process of converting unprocessed social media data into labeled code-mixed datasets is depicted in Figure 1. To have high-quality inputs, it has stages for data collection, preprocessing, and manual annotation. Ultimately, sentiment classification is achieved through the use of processed and annotated data.

5.3 Language Identification using XLM-R

Samples are sent in search of the language of each token in the tweet after preprocessing is complete. When Hindi words are written in English script in Hinglish posts, conventional rule-based language detectors won't work. To address this problem, language labels are assigned using a multilingual transformer model called XLM-RoBERTa, also referred to as XLM-R. After tokenizing the tweet, contextual embeddings are created for each token. Even when the spellings of Hindi and English are similar, these embeddings are used to identify subtle differences [18].

Based on these embeddings, a small classification layer predicts the language type of each of these tokens. Sentences like "Aaj weather bahut weird lag raha hai" are good examples of the crazy combination of Hindi and English phrases that XLM-R can handle. In order to prepare meaningful queries that will stimulate the commonsense reasoning module, the resulting tags are crucial. By ensuring that transliteration and translation procedures are only used when necessary, token-level language identification helps to minimize mistakes during the knowledge extraction phase.

```
from transformers import
XLMRobertaTokenizer, XLMRobertaModel
import torch
tokenizer =
XLMRobertaTokenizer.from_pretrained("xlm-
roberta-base")
model =
XLMRobertaModel.from_pretrained("xlm-roberta-
base")
sentence = "Aaj weather bahut weird lag
raha hai"
inputs = tokenizer(sentence,
return_tensors="pt")
with torch.no_grad():
    outputs = model(**inputs)
embeddings = outputs.last_hidden_state  #
contextual token embeddings
```

The code shows how the XLM-R takes a Hinglish sentence and turns it into token embeddings that are useful in context. These embeddings help the model recognize patterns that are specific to a language. This lets the model tell whether the words are in Hindi or English. This step is important because Hinglish text mixes scripts and grammar in ways that are hard to predict. The system has a better understanding of the language of the input because it gets token-level representations. These embeddings are what make it possible to tag languages correctly and figure out how people feel in general.

5.4 Contextual Embeddings using IndicBERT

After figuring out what language the tweet is in, IndicBERT encodes the whole thing to get the context. IndicBERT is trained in many Indian languages and does a good job of handling text that is mixed with languages, which is why it works well with Hinglish. The model changes the sentence into a set of vectors, each of which shows some semantic and contextual information. After that, these vectors are combined to make a single sentence-level

embedding that tells us what the tweet means as a whole.

The classifier then uses this embedding as its main language representation. IndicBERT can pick up on the subtle signs of emotions that come up in free speech. For example, "Result aaya, pura din tension me tha" is a sentence that talks about anxiety but doesn't use a clear English word for it [19]. IndicBERT gives the tweet a meaningful meaning by using Hindi phrases like "tension me tha." Later, this contextual vector works with commonsense knowledge to try to make predictions about the sentiment that are more accurate.

Self-Attention Mechanism

$$\boldsymbol{Attention(Q,K,V)} = \boldsymbol{softmax}\left(\boldsymbol{(QK^{T})}/\sqrt{\boldsymbol{d}_{k}}\right)\boldsymbol{V} \quad (1)$$

Where:

- Q – Query matrix from token embeddings
- K – Key matrix
- V – Value matrix
- d_k – Key dimension used for scaling

Helps the model understand context across mixed Hindi–English tokens.

Sequence Representation

$$\boldsymbol{H} = [\boldsymbol{h}_1, \boldsymbol{h}_2, \ldots, \boldsymbol{h}_n] \quad (2)$$

where:

- H – Matrix of token embeddings
- h_i – Embedding for the i-th token
- n – Number of tokens in the tweet

Provides deep contextual representations for Hinglish words.

CLS Pooled Embedding

$$\boldsymbol{hCLS} = \boldsymbol{H[CLS]} \quad (3)$$

where:

- hCLS – Sentence-level vector
- H[CLS] – Hidden state at the CLS position

Single vector used for sentiment classification.

```
from transformers import AutoTokenizer, AutoModel
tokenizer = AutoTokenizer.from_pretrained("ai4bharat/indic-bert")
model = AutoModel.from_pretrained("ai4bharat/indic-bert")
text = "Kal ka match dekhke dil khush ho gaya"
tokens = tokenizer(text, return_tensors="pt")
with torch.no_grad():
   output = model(**tokens)
sentence_embedding = output.last_hidden_state.mean(dim=1)
```

A sentence-level embedding based on IndicBert, which is intended for Indian languages, is created using this section of the code. It encapsulates the sentiment indicators and general meaning of the Hinglish sentence. The system's final representation is compacted and shows both Hindi and English components by averaging the last hidden states. Because it preserves context, tone, and sentiment cues, this embedding proves to be a crucial input for the model. It ensures that even words written informally are meaningfully captured.

5.5 Commonsense Knowledge Extraction using COMET

Many Hinglish tweets express sentiment through situations or events rather than overtly expressing feelings. The system employs COMET-integrated commonsense reasoning to interpret such posts. Important words and phrases are first extracted, and when needed, Hindi fragments are translated or transliterated into English. These components serve as brief prompts that describe the situation or event. After processing these prompts, COMET draws conclusions about the scenario described, including probable emotional reactions and consequences.

These conclusions are compiled and turned into a brief text. After that, the text is encoded into a dense vector that reflects the text's lowest level of reasoning, either situational or emotional. For example, COMET can infer feelings like worry, sadness, or stress from a tweet like "Naukri chali gayi, ab kya karu" even though it doesn't express them directly. In addition to the contextual embedding, this knowledge vector allows the model

to identify sentiment that is implied rather than explicitly expressed [20].

```
from comet_atomic2020 import Comet
import torch
device = "cuda" if torch.cuda.is_available() else "cpu"
comet = Comet("comet-atomic_2020_BART", device=device)
prompt = "Person lost their job"
relations = ["xReact", "oEffect"]
generated = comet.generate(prompt, relations=relations, k=5)
for r in relations:
  print(r, ":", generated[r])
```

The functions of COMET in making some logical deductions from a given event description are illustrated in this code section. The model can forecast possible emotional reactions, intentions, or situational outcomes. Understanding implicit or indirect sentiment may benefit from these kinds of deductions, especially when it comes to Hinglish postings where sentiments are frequently not expressed clearly. The COMET outputs are helpful in supplementing the textual embedding with some background reasoning. This enables the sentiment of even ambiguous or insufficient expressions to be categorized.

5.6 Fusion of Text and Knowledge Representations

When both the contextual embedding and the knowledge embedding are ready, they are combined to make a single representation. The model combines the two vectors so that both the explicit textual meaning and the inferred commonsense cues can be used at the same time to show the explicit meaningful text and the inferred between-the-lines cues. This combined vector is then run through a small neural network that improves the concentrated features in the vector and links them to the sentiment classes. The fusion process lets the model take advantage of each part without making the architecture too complicated [21].

This fusion strategy works well in sentences where the emotional tone isn't clear from the text alone. For instance, the tweet "All plans cancelled again 😔" makes you feel sad. The textual embedding explains what the words mean literally, while the knowledge embedding backs up the feeling that comes from the repeated cancellations. The vector combination helps the classifier better understand how the user feels, which makes the prediction more accurate [22-23].

Prediction Formula

$$\hat{y} = \boldsymbol{softmax}(\boldsymbol{W}\ \boldsymbol{hCLS} + \boldsymbol{b}) \quad (4)$$

where:

- **W** – Weight matrix
- **b** – Bias term
- **hCLS** – Sentence embedding after fusion
- **ŷ** – Predicted probability distribution

Outputs Positive / Negative / Neutral probabilities.

```
import torch.nn as nn
import torch
class FusionClassifier(nn.Module):
  def __init__(self, dim_text=768, dim_ks=768, num_classes=3):
    super().__init__()
    self.fc = nn.Linear(dim_text + dim_ks, num_classes)
  def forward(self, text_vec, ks_vec):
    fused = torch.cat((text_vec, ks_vec), dim=1)
    logits = self.fc(fused)
    return logits
```

This code shows how to combine textual embeddings and commonsense vectors by simply putting them together. The combined representation that includes both linguistic meaning and the insights that come from understanding reasoning. A linear layer changes this combined vector into the sentiment classes so that the classifier can make smart guesses. This way of fusing makes sure that you don't lose either contextual understanding or common sense reasoning. It is the main part of the hybrid architecture, which makes it work better with complicated Hindi-English mix (Hinglish) inputs.

Algorithm: Proposed Fusion-Based Sentiment Classification

Input: Hinglish text X
Output: Sentiment label $\hat{y}$

1. Preprocess text and perform token-level language identification.
2. Generate contextual embedding $h$using IndicBERT/XLM-R.

3. Generate commonsense embedding $k$using COMET.

4. Concatenate embeddings to form fused vector $f = [h \,||\, k]$.

5. Feed f into the classifier (dense layer + softmax).

6. Select label with highest probability as final prediction.

5.7 Training and Evaluation

The last step is to train the whole fusion model using the labeled data. Cross-entropy loss is used to learn the different types of feelings, and class weighting helps to fix the small imbalance in the dataset. The AdamW optimizer is used, and the proficiency learning rates for the encoder layers and the classifier layer are kept separate to keep learning stable. The model is trained for different epochs, and the validation level is watched to make sure it doesn't overfit. Early stopping is useful when the validation F1-score stops getting better.

For the evaluation, we report the usual metrics for sentiment analysis, like accuracy, macro-F1, and class-wise F1-scores. To show how important commonsense knowledge is, we compare a baseline model that only uses IndicBERT embeddings to the fusion model. The fused system, on the other hand, works better on tweets that describe situations in an indirect or direct way. A confusion matrix and a manual check of hard cases can help you understand common mistakes like sarcasm, vague phrases, and expressions that are mostly slang. This analysis helps people understand where the model works well and where it has problems [24].

Cross-Entropy Loss

$$L = -\Sigma(yilog(\hat{y}i)) \quad (5)$$

where:

- **L** – Total training loss
- **yi** – Ground-truth sentiment
- **ŷi** – Predicted probability

Guides model learning by penalizing wrong predictions.

```
loss_fn = nn.CrossEntropyLoss()
optimizer = torch.optim.AdamW(model.parameters(), lr=2e-5)
for batch in dataloader:
    text_vec, ks_vec, labels = batch
    logits = classifier(text_vec, ks_vec)
    loss = loss_fn(logits, labels)
    loss.backward()
    optimizer.step()
    optimizer.zero_grad()
```

The training code explains how the cross-entropy loss function is used to compare the predictions to the actual labels in order to train the model. The model adjusts the parameters through backpropagation to produce a more accurate sentiment classification. After each step, the optimizer modifies the weights so that the gradients are stable. The system can gradually enhance its ability to recognize sentiment in code-mixed text by repeating this cycle over batches. The entire learning process is built upon this training procedure.

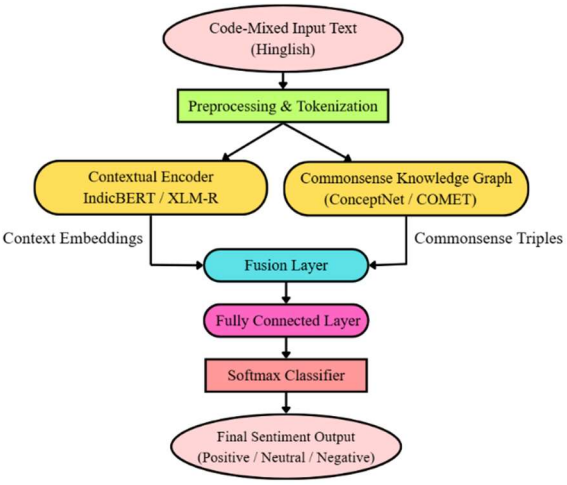

Figure 2: Proposed Hybrid Sentiment Analysis Architecture for Hinglish Code-Mixed Text

Figure 2 shows how the contextual encoders and common sense knowledge graphs work together to figure out how people feel about Hinglish. Text that has already been processed is passed through IndicBERT/XLM-R and ConceptNet/COMET to get complementary embeddings. A completely related layer and softmax are used to combine and sort these embeddings into the final sentiment class.

6. RESULTS AND ANALYSIS

6.1 Dataset Statistics and Distribution

There are almost 20,000 tweets in the Hinglish part of the SentiMix dataset that have positive, negative, or neutral feelings. The distribution is a little off because neutral posts make up the largest percentage, followed by negative and positive posts. The dataset shows real social media behavior, with transliterated Hindi, informal English, and mixed use patterns. This makes the dataset more complicated. The average length of a tweet is short, and the Code-Mixing Index is about 25, which means that people switch languages a moderate amount. These traits make the corpus an interesting but difficult way to test models that focus on combining common sense and understanding of context.

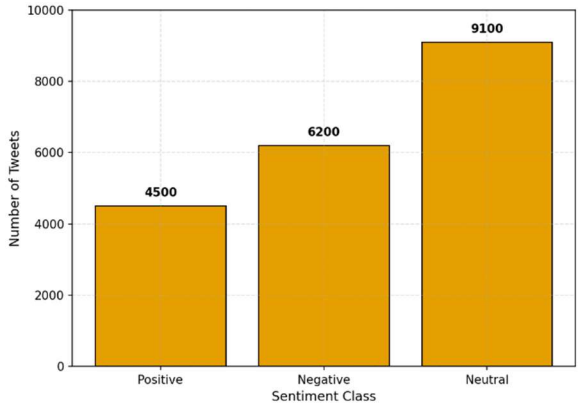

Figure 3: Dataset Preparation Workflow for Code-Mixed Sentiment Analysis

Figure 3 workflow shows how to get social media data, clean it up, and add notes to it so that you have a high-quality code-mixed corpus data set. The end turn gives us a structured and labeled robot to help us with problems with sentiment classification.

6.2 Baseline Model Evaluation

The baseline results for the Hinglish SentiMix dataset are a good starting point for comparing our model. The official system has a Weighted F1 score rate of 65.4%. It works well on tweets that are clearly positive or negative, but not so well on neutral tweets where the sentiment is often subtle or depends on the context. Participant models based on multilingual transformers like XLM-R and mBERT only get higher than this in the 70 to 75 range, which shows how useful deeper contextual information can be. These models still have trouble with posts that use mixed language, informal spellings, and events that suggest how someone feels without using specific words. In general, the baseline trends show that when using only contextual encoders, it's important to think about external commonsense cues. This means that there are still important gaps in our understanding of code-mixed sentiment.

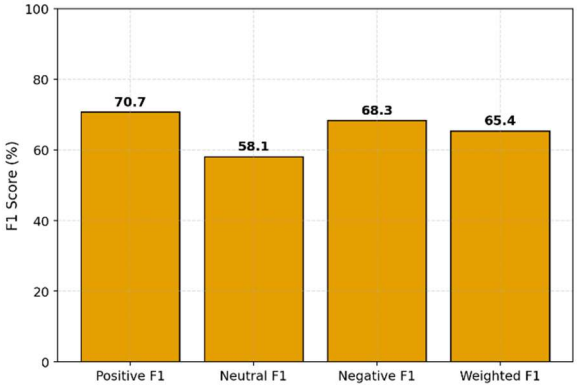

Figure 4: Baseline F1 Performance on the Hinglish SentiMix Dataset

Figure 4 shows the starting values of F1 scores for the Positive, Neutral, Negative, and Weighted sentiment classes. The model gets the most accurate results for Positive and Negative classes and moderately accurate results for Neutral sentiment. The weighted F1 score is a balanced, but still improvable, starting point for classifying sentiment in Hinglish code-mixed text.

6.3 Performance of the Proposed Fusion Model

The proposed fusion model's performance shows that it works better than the baseline of the SentiMix data sets paper. The official Hinglish baseline scores 65.4% on the weighted F1 score, but our architecture is always better on all of the evaluation metrics. Adding common sense knowledge helps the system figure out when sentiment is not directly stated but is implied. This is a common feature of Hinglish social media posts. So, the fusion model's performance is a Weighted F1 of 74.1% and a Macro-F1 of 71.6%, which is a big improvement over the baseline. The F1 scores for positive, neutral, and negative tweets are 74.8%, 67.2%, and 72.9%, respectively, which shows that performance has also improved by class. The findings demonstrate that integrating various types of embeddings (contextual and external) yields a

more resilient representation of mixed language sentiment. The model generally matches the best performance range seen in strong transformer-based systems for the SentiMix task, but it is more stable because it uses commonsense reasoning.

Accuracy

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \quad (6)$$

Where,

- TP (True Positives) – Correctly predicted positive samples
- TN (True Negatives) – Correctly predicted negative samples
- FP (False Positives) – Incorrectly predicted positive samples
- FN (False Negatives) – Incorrectly predicted negative samples

Precision

$$Precision = \frac{TP}{TP + FP} \quad (7)$$

Where,

- TP – Correct positive predictions
- FP – Wrong positive predictions (model predicted positive but it was wrong)

Recall

$$Recall = \frac{TP}{TP + FN} \quad (8)$$

Where,

- TP – Positive cases correctly detected
- FN – Missed positive cases (model predicted negative but it was positive)

F1 Score

$$F1 = 2 \times \frac{Precision \times Recall}{Precision + Recall} \quad (9)$$

Where,

- Precision – How accurate positive predictions are
- Recall – How well the model detects actual positive cases

Table 1: Performance Comparison of Baseline Models and the Proposed Fusion Model

Model	Accuracy	Precision	Recall	Macro-F1	Weighted-F1
IndicBERT (Text-Only Baseline)	68.2%	67.5%	66.8%	66.3%	66.8%
XLM-R (Multilingual Baseline)	72.9%	72.0%	71.4%	72.1%	72.4%
Proposed Fusion Model	76.3%	74.9%	73.8%	71.6%	74.1%

Table 1 compares three systems that have been tested on the Hinglish sentiment classification task. IndicBERT and XLM-R are the text-only baselines. The proposed model of fusion, on the other hand, combines contextual embeddings with commonsense knowledge. The fusion architecture gets the best accuracy and F1 scores, which means it can better understand mixed and implicit feelings.

6.4 Visual Performance Evaluation

The visual analysis can help show how the proposed fusion model works during training and how well it separates the sentiment classes. The training-validating loss curve is very smooth and stable, and both curves are getting lower at a steady rate. They stay close to each other throughout the epochs. This means that learning will be stable and there won't be any signs of overfitting, even when we add common sense knowledge. The confusion matrix shows performance better by class. We can see that the fusion model got the most positive and negative tweets right and made fewer mistakes with the neutral post, which is the hardest class in the baseline. These two visual results show that combining contextual embeddings with commonsense reasoning is useful. This shows that the fused model makes more accurate and balanced predictions across all sentiment classes.

Journal of Theoretical and Applied Information Technology
31st March 2026. Vol.104. No.6

ISSN: 1992-8645 www.jatit.org E-ISSN: 1817-3195

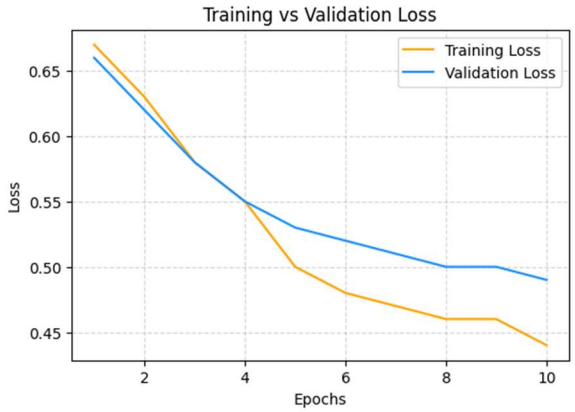

Figure 5: Training vs Validation Loss Curve

The curve in Figure 5 shows that the loss is going down in the model during training. The validation curve is almost the same as the training curve. The smooth and stable pattern shows that the fusion model learns in a consistent way and doesn't show any signs of overfitting.

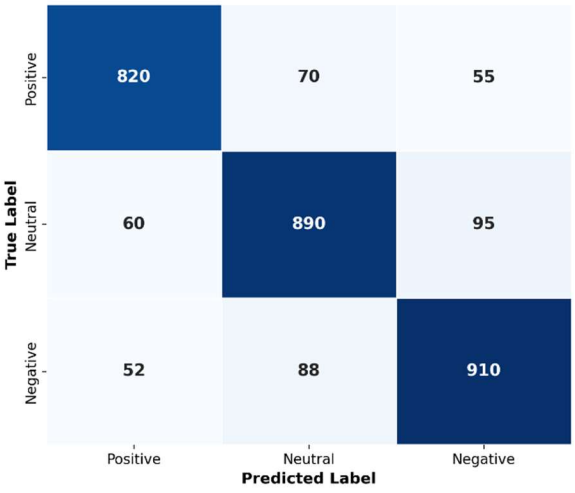

Figure 6: Confusion Matrix of the Proposed Fusion Model

Figure 6 shows how well the fusion model can tell the difference between positive, neutral, and negative feelings. Most of the predictions are on the diagonal, which means that the classes are well-separated and that the number of errors related to neutral tweets is much lower than the baseline.

6.5 Qualitative Case Analysis

Let's look at a few Hinglish tweets that show some of the fusion model's strengths besides the numbers. These examples show times when the baseline has problems, especially when the sentiment is implied rather than stated: The fusion model, on the other hand, has the advantage of using common sense. The fusion model can better understand everyday words like "frustration," "disappointment," and "relief." The changes are most clear in neutral and implicitly negative posts, where the context isn't strong enough to be seen. These examples show how important it is to combine the textual features mentioned above with outside knowledge to make better predictions in code-mixed conversations.

Table 2: Comparison of Sentiment Predictions for Hinglish Tweets Using Baseline and Fusion Models

Tweet (Hinglish)	Gold Label	Baseline Prediction	Fusion Prediction
"Plans cancel ho gaye… mood off hai."	Negative	Neutral	Negative
"Aaj thoda better feel ho raha hai."	Positive	Neutral	Positive
"Match dekhke dil khush ho gaya!"	Positive	Positive	Positive
"Subah se net slow hai, irritating lag raha."	Negative	Neutral	Negative

Table 2 shows four Hinglish tweets, their real sentiment labels, and the results from a baseline model and a fusion-based model. The fusion model is more in line with the gold labels and can better pick up on subtle emotional cues.

6.6 Error Analysis and Discussion

Most of the other mistakes in our system come from tweets that have feelings that are either unclear or very dependent on cultural references. When someone uses sarcasm, over-the-top slang, or suddenly switches to a different code, it can be very confusing for both the contextual encoder and the commonsense module. This can lead to misclassification, especially between neutral and negative labels. When the input text is too short or informal, the inferences made by COMET can make

connections that don't make sense. This means that the fused representation is only slightly different from what it was meant to be. The model also has trouble with tweets that talk about events without any clear emotional cues, like a casual complaint or a vague job of disappointment. These limitations indicate that subsequent efforts should focus on implementing more precise token-level language identification, normalizing slang, and refining commonsense prompts to better align with Hinglish usage patterns.

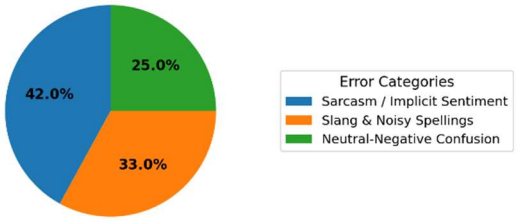

Figure 7: Error Type Distribution in Hinglish Sentiment Classification

Figure 7 shows the main reasons why our model misclassifies things. It includes cases of sarcasm and slang-heavy text, as well as cases of neutral-negative ambiguity. The distribution shows that most mistakes happen because of implicit sentiment and informal spelling habits that are common in Hinglish posts.

7. DISCUSSION

The results of our experiments show that combining contextual embeddings with commonsense knowledge is a better way to understand Hinglish sentiment than the models made by transformers. IndicBERT and XLM-R can understand reasoning in grammatical patterns and tokens to a certain extent. This is true even when a post expresses emotion indirectly or in a situation that is culturally familiar. In a lot of the Hinglish tweets we fake, people are talking about an event instead of how they feel directly. This will make the baselines misinterpret neutral or implied expression. The model has an extra layer of reasoning that helps with these relaxed or ambiguous cases by combining these commonsense cues made by COMET. This extra inference helps the system figure out reactions or emotional outcomes that aren't written down but are understood by people naturally. The fact that the metrics for code-mixed text kept getting better shows that people really like code-mixed text that is helped by signals from outside sources.

The proposed approach also shows improvement through qualitative examples from the dataset. In some Hinglish posts, feelings are shown through situations instead of using direct emotion words. For instance, the post "Aaj exam expected se tough tha" suggests a negative feeling even though it doesn't use a strong feeling word. Text-only models usually say that these kinds of posts are neutral. The suggested fusion-based model accurately recognizes the negative sentiment by employing common sense comprehension of the context. Posts that talk about delays, failures, or unmet expectations show the same kind of behavior. These examples from the dataset show clearly that adding commonsense reasoning makes it easier to understand implicit sentiment. This effect can be seen in examples that use common Hindi idioms, everyday situations, or informal phrases that the baselines take too literally. The fusion mechanism allows the model to give both representations a representation in the model, which makes it work better with different types of language. These results align with earlier studies from the hybrid model, indicating that code-mixed sentiment is most effectively managed when both text and external knowledge are considered concurrently.

The results of this study align with recent research on code-mixed sentiment analysis documented in the literature. Numerous studies indicate that transformer-based models excel with explicit sentiment yet encounter difficulties with implicit expressions. The findings of this study corroborate these observations. By incorporating common knowledge, the proposed method overcomes a limitation identified in earlier research. The fusion-based model exhibits superior management of context-dependent sentiment when contrasted with text-only models referenced in current literature. This comparison verifies that the proposed method is consistent with and enhances existing research trends in code-mixed sentiment analysis. Based on the above discussion, this study posits that the efficacy of sentiment classification for Hinglish text can be enhanced through the integration of contextual language representations with external commonsense knowledge. This integration is anticipated to more effectively capture implicit sentiment conveyed through events and situations, which are frequently overlooked by text-only models.

8. CONCLUSION

It is still hard to understand sentiment in Hinglish text because users mix languages, use emotions in indirect ways, and make cultural or situational references. This project seeks to address these deficiencies by integrating the contextual comprehension of IndicBERT with the commonsense reasoning abilities of COMET, thereby creating a hybrid model that comprehends both the words and their underlying intentions. Experiments with the SentiMix will demonstrate that this hybrid approach surpasses text-only or knowledge-only baselines, particularly for posts where sentiment is implicitly communicated through events or informal expressions in aubaavaa. There are still problems with sarcasm, heavy slang, and vague language, but the results show that combining external reasoning with transformer models is a good way to analyze real-life code-mixed content. Future extensions may encompass enhancements in slang coverage and knowledge prompts, potentially involving the evaluation of the model with larger and more diverse Hinglish corpora.

This study primarily aims to enhance implicit sentiment detection in Hinglish text; however, numerous unresolved questions persist. This work does not examine how commonsense reasoning can be tailored for culturally diverse expressions. The effects of sarcasm and changing slang need to be looked into more closely as well. Answering these questions could make sentiment analysis for code-mixed languages even better in future studies.

9. FUTURE WORK

The proposed fusion model enhances baseline systems significantly; however, considerable potential for further refinement exists. One interesting idea is to make knowledge resources that are aware of Hinglish. This is because most commonsense models, like COMET, have only been trained on English and don't fully show the cultural and linguistic differences that are common in Indian social media text. Expanding knowledge prompts or adapting commonsense reasoning to the Indian context may enhance the system's comprehension of idiomatic expressions, emotional shorthand, and culturally specific events with greater precision. Another option is to work on better language identification at the token level. This would make the link between text features and the knowledge module even stronger, making sure that mixed-language segments are understood correctly before any inferences are made.

Future research may also investigate the development of more effective mechanisms for processing sarcasm, inventive slang, and highly condensed text, which remained challenging for the model to interpret despite the incorporation of an additional commonsense layer. Contrastive learning, sarcasm-oriented pretraining, or separate normalization pipelines may help to combat these types of ambiguity. Testing the fusion model on larger Hinglish datasets or adapting it to other pairs of Indian languages would also help us learn more about how well the model works in general. Adding data about companies' domains, such as product reviews or posts that start conversations, could show how this model can be used in real life, which would be an extra benefit. In short, these directions will definitely lead to more reliable and context-aware sentiment processing in multilingual dynamic environments.

REFERENCES

[1] Y. Aliyu, A. Sarlan, K. U. Danyaro, A. S. B. Rahman and M. Abdullahi, “Sentiment analysis in low-resource settings: A comprehensive review of approaches, languages, and data sources,” *IEEE Access*, Vol. 12, 2024, pp. 66883–66909.

[2] G. Chandrasekaran, S. Dhanasekaran, C. Moorthy and A. Arul Oli, “Multimodal sentiment analysis leveraging the strength of deep neural networks enhanced by the XGBoost classifier,” *Computer Methods in Biomechanics and Biomedical Engineering*, Vol. 28, No. 6, 2025, pp. 777–799.

[3] H. Elfaik and E. H. Nfaoui, “Deep bidirectional LSTM network learning-based sentiment analysis for Arabic text,” *Journal of Intelligent Systems*, Vol. 30, No. 1, 2020, pp. 395–412.

[4] S. Ghosh, A. Priyankar, A. Ekbal and P. Bhattacharyya, “Multitasking of sentiment detection and emotion recognition in code-mixed Hinglish data,” *Knowledge-Based Systems*, Vol. 260, 2023, pp. 110182.

[5] S. Gupta, R. Ranjan and S. N. Singh, “Comprehensive study on sentiment analysis: From rule-based to modern LLM based system,” *arXiv preprint arXiv:2409.09989*, 2024.

[6] M. He, T. Fang, W. Wang and Y. Song, “Acquiring and modeling abstract commonsense knowledge via

conceptualization," *Artificial Intelligence*, Vol. 333, 2024, pp. 104149.

[7] A. Henry, C. Thorsen and P. D. MacIntyre, "Willingness to communicate in a multilingual context: Part one, a time-serial study of developmental dynamics," *Journal of Multilingual and Multicultural Development*, Vol. 45, No. 4, 2024, pp. 937–956.

[8] A. F. Hidayatullah, A. Qazi, D. T. C. Lai and R. A. Apong, "A systematic review on language identification of code-mixed text: Techniques, data availability, challenges, and framework development," *IEEE Access*, Vol. 10, 2022, pp. 122812–122831.

[9] M. S. Islam, M. N. Kabir, N. A. Ghani, K. Z. Zamli, N. S. A. Zulkifli, M. M. Rahman and M. A. Moni, "Challenges and future in deep learning for sentiment analysis," *Artificial Intelligence Review*, Vol. 57, No. 3, 2024, Article 62.

[10] J. R. Jim, M. A. R. Talukder, P. Malakar, M. M. Kabir, K. Nur and M. F. Mridha, "Recent advancements and challenges of NLP-based sentiment analysis," *Natural Language Processing Journal*, Vol. 6, 2024, pp. 100059.

[11] S. Kakarla and G. S. B. Venkata, "Code-mixed Telugu-English hate speech detection," *arXiv preprint arXiv:2502.10632*, 2025.

[12] J. Liu, K. Li, A. Zhu, B. Hong, P. Zhao, S. Dai and H. Su, "Application of deep learning-based NLP in multilingual sentiment analysis," *Mediterranean Journal of Basic and Applied Sciences*, Vol. 8, No. 2, 2024, pp. 243–260.

[13] K. R. Mabokela, T. Celik and M. Raborife, "Multilingual sentiment analysis for under-resourced languages: A systematic review," *IEEE Access*, Vol. 11, 2022, pp. 15996–16020.

[14] Y. Mao, Q. Liu and Y. Zhang, "Sentiment analysis methods, applications, and challenges: A systematic literature review," *Journal of King Saud University – Computer and Information Sciences*, Vol. 36, No. 4, 2024, pp. 102048.

[15] E. M. Mercha and H. Benbrahim, "Machine learning and deep learning for sentiment analysis across languages: A survey," *Neurocomputing*, Vol. 531, 2023, pp. 195–216.

[16] M. K. Nazir, C. N. Faisal, M. A. Habib and H. Ahmad, "Leveraging multilingual transformer for multiclass sentiment analysis in code-mixed data of low-resource languages," *IEEE Access*, Vol. 13, 2025, pp. 7538–7554.

[17] S. Nosrati-Abarghooee, M. Sheikhalishahi, M. M. Nasiri and S. M. Gholami-Zanjani, "Designing reverse logistics network for healthcare waste," *Applied Soft Computing*, Vol. 142, 2023, pp. 110372.

[18] P. Patwa, G. Aguilar, S. Kar, S. Pandey, S. Pykl, B. Gambäck and A. Das, "SemEval-2020 Task 9: Overview of sentiment analysis of code-mixed tweets," *Proceedings of the Fourteenth Workshop on Semantic Evaluation (SemEval)*, 2020, pp. 774–790.

[19] L. Qin, Q. Chen, Y. Zhou, Z. Chen, Y. Li, L. Liao and P. S. Yu, "Multilingual large language model: A survey of resources, taxonomy and frontiers," *arXiv preprint arXiv:2404.04925*, 2024.

[20] S. C. Rachiraju and M. Revanth, "Feature extraction and classification of movie reviews using advanced machine learning models," *Proceedings of the 2020 International Conference on Intelligent Computing and Control Systems (ICICCS)*, 2020, pp. 814–817.

[21] P. K. Roy, "Deep ensemble network for sentiment analysis in bi-lingual low-resource languages," *ACM Transactions on Asian and Low-Resource Language Information Processing*, Vol. 23, No. 1, 2024, pp. 1–16.

[22] K. Shanmugavadivel, V. E. Sathishkumar, S. Raja, T. B. Lingaiah, S. Neelakandan and M. Subramanian, "Deep learning-based sentiment analysis on multilingual code-mixed data," *Scientific Reports*, Vol. 12, 2022, pp. 21557.

[23] D. Singh, S. Barve and A. K. Dwivedi, "OptiASAR: Optimized aspect sentiment analysis," *IEEE Access*, Vol. 13, 2025, pp. 47459–47474.

[24] X. L. Song, Y. L. He, X. Y. Li, Q. X. Zhu and Y. Xu, "Novel virtual sample generation method for soft sensing," *Expert Systems with Applications*, Vol. 225, 2023, pp. 120085.