Social Network Analysis and Mining (2022) 12:87
https://doi.org/10.1007/s13278-022-00920-w

ORIGINAL ARTICLE

Fighting hate speech from bilingual hinglish speaker's perspective, a transformer- and translation-based approach.

Shankar Biradar[1] · **Sunil Saumya**[1] · **Arun chauhan**[2]

Received: 27 February 2022 / Revised: 28 June 2022 / Accepted: 4 July 2022 / Published online: 24 July 2022

Abstract
Many people have begun to use social media platforms due to the increased use of the Internet over the previous decade. It has a lot of benefits, but it also comes with a lot of risks and drawbacks, such as Hate speech. People in multilingual societies, such as India, frequently mix their native language with English while speaking, so detecting hate content in such bilingual code-mixed data has drawn the larger interest of the research community. The majority of previous work focuses on high-resource language such as English, but very few researchers have concentrated on the mixed bilingual data like Hinglish. In this study, we investigated the performance of transformer models like IndicBERT and multilingual Bidirectional Encoder Representation(mBERT), as well as transfer learning from pre-trained language models like ULMFiT and Bidirectional encoder Representation(BERT), to find hateful content in Hinglish. Also, Transformer-based Interpreter and Feature extraction model on Deep Neural Network (TIF-DNN), is proposed in this work. The experimental results found that our proposed model outperforms existing state-of-art methods for Hate speech identification in Hinglish language with an accuracy of 73%.

Keywords Hinglish · Code-mixed · mBERT · Transformers

1 Introduction

India is one of the largest and fastest-growing marketplace for digital consumers. Urban consumers mostly drove this rapid expansion of the digital economy. However, with the government's push for digital India, rural India has also begun to embrace the digital economy. People adopting their regional language or blending it with English during conversation or information exchange has become a typical phenomenon as the number of Internet users from rural India has grown. Furthermore, because of lower literacy levels and a lack of cyber awareness, these people are readily motivated to spread toxic content such as Hate speech, fake news, and so on through social media platforms. Further, India has 22 different languages in which people communicate; research into controlling the transmission of hate content in these code-mixed regional languages has taken a major step in multilingual and multicultural countries like India. It will also provide an ideal test bed for the research community to investigate the spread of harmful content through social media platforms. Furthermore, the government lacks media policies to prevent the spread of such harmful content. Though the government has recently attempted to impose certain regulations, enforcing these laws in a country with a huge and diverse population is extremely difficult. There is a definite need to develop automated solutions to tackle the spread of hate content across regional languages or code mixed languages. Bokamba (1989) defines code-mixing as mixing words, phrases, and sentences from two grammatical subsystems inside the same utterance.

Some of the recent events that occurred due to the spread of hate news through social media have given a glimpse of the gravity of the problem. The recent 2020 Delhi riots, which resulted in the deaths of 53 innocent people, are living proof of how this Hate news may be exploited to destabilize

✉ Shankar Biradar
shankar@iiitdwd.ac.in

Sunil Saumya
sunil.saumya@iiitdwd.ac.in

Arun chauhan
aruntakhur@gmail.com

[1] Department of Computer Science and Engineering, Indian Institute of Information Technology, Dharwad, Karnataka, India

[2] Department of Computer Science and Engineering, Graphic Era University, Dehradun, Uttarakhand, India

the region or ruin social harmony. In addition, a new trending hashtag[1] appeared on Twitter at the end of March 2020, blaming a religious group for Covid-19's spread in India. It had been seen more than 300,000 times on Twitter by the beginning of April, with a potential audience of 165 million. Demonstrates how quickly Hate speech can spread on social media. Also, hate-mongering on social media reached its peak during the recent West Bengal, India elections, which ended in violence between workers from two opposing parties, resulting in the deaths of several innocent people. More recently, northeast Indian inhabitants have suffered racial discrimination during the Covid-19 rise due to hateful propaganda published on social media against those individuals. These facts have emphasized the need to prevent the transmission of hate news on social media, gaining larger attention among researchers and academics.

Because of their widespread availability, previous research on Hate speech identification has mainly focused on high-resource monolingual languages such as English Khan et al. (2020); Mossie and Wang (2020); Senarath and Purohit (2020). Among these, neural network-based models have obtained state-of-the-art outcomes for various natural language processing applications. Several neural architectures, including RNN Bisht et al. (2020), CNN Khan et al. (2020), and transformer Banerjee et al. (2021); Biradar et al. (2021); Biradar and Saumya (2022) models, have been explored for Hate speech identification. However, code-mixing has recently been popular in social media, especially in multilingual countries like India, where more than 350 million individuals speak Hinglish[2]. Due to a lack of grammar Lal et al. (2019) and informal transliteration Singh et al. (2018), identifying language cues and establishing a contextually robust representation for code-mixed texts remain a fundamental difficulty in NLP. The following are some examples of code-mixed texts.

T1: 'neeraj ka nam humesha yaad rahega because he won the first gold medal for India in athletics!!!..'.

T2: 'muje apane manager se bahut nafarat hai, I want to kill him.'

From the preceding examples, T1 contains normal speech; however, T2 is a Hate speech instance.

Several off-the-shelf tools like IndicNLP Kunchukuttan et al. (2020), iNLTK Arora (2020), and stanza Qi et al. (2020) have been developed in recent times to handle regional languages. However, they work mainly on monolingual data like Hindi, Bengali, Marathi, etc., but they cannot handle code-mixed bilingual text. Finding Hate speech in code mixed data is more difficult because of shorter forms of words and spelling changes. More recently, few researchers have tried to develop a model for handling Code-mixed text, such as CS-ELMO, a neural architecture for transfer learning from an ELMO model, pre-trained English to code-mixed texts Aguilar and Solorio (2020). Also, the Bilingual word embedding model is proposed by Pratapa et al. (2018) to handle code-mixed data. All these models handle different aspects of NLP tasks; however, no comprehensive work has been done to identify Hate content in code-mixed data. Hence to explore the challenges of code-mixed scenarios, in this work, we have experimented with various language as well transformer models; we also proposed Transformer-based Interpreter and Feature extraction model on Deep Neural Network (TIF-DNN) as explained in Sect. 3.

The main contribution to the paper includes:

1. TIF-DNN, a Transformer-based Interpreter and Feature extraction model on Deep Neural Network for Hate speech identification in code-mixed Hinglish language, has been developed.
2. The efficiency of the proposed model is demonstrated by comparing the proposed method with existing ones.
3. The performance of different off-the-shelf language and Transformer models are demonstrated for Hate speech identification on code-mixed data.

The remainder of the article is structured as follows: Sect. 2 gives a summary of the background literature. Next, sect. 3 contains specifics on the suggested approach and data set. Further, sect. 4 discusses the experimental results as well as the experimental setup. Finally, we conclude our article with Discussion and limitations.

2 Literature review

Most prior work on sentiment analysis has been done primarily on high-resource languages such as English. However, code-mixed languages have received little attention due to their non-standard writing style and a shortage of data sets to train the models. As a result, researchers have just lately begun to investigate code-mixed data. The following are some of the approaches used to handle Code-mixed data.

2.1 Using handcrafted linguistic features

The first such attempt was performed by Bohra et al. (2018), they provided an annotated corpus of Hindi–English code-mixed text, comprising tweet ids and the accompanying annotations. They also demonstrated the supervised method for detecting Hate speech in code-mixed text. They employed character n-grams, word n-grams, punctuation's, negation words, and hate lexicons as classification features.

[1] https://strongcitiesnetwork.org/en/wp-content/uploads/sites/5/2020/06/CoronaJihad.pdf.

[2] https://en.wikipedia.org/wiki/.

Furthermore, some researchers used Logistic Regression and multinomial Naïve Bayes to analyze statistical features such as char n-gram, word uni-gram, and word bi-gram. The experimental investigations revealed that character n-grams and word uni-gram performed better when classified using Logistic Regression on a Hindi data set. In addition, the authors used pre-trained Word2Vec embeddings for the English data set Samghabadi et al. (2018).

Ghosh et al. performed sentiment identification on code-mixed text data derived from social media. Their experiment used two code-mixed data sets, English–Bengali and English–Hindi. They classified the data according to the polarity contradiction in the statement, such as positive, negative, or neutral. SentiWordNet, opinion lexicon, and Part-of-speech (POS) tags are employed, and the multilayer perception model is used to classify the polarity, with 68.5% accuracy Ghosh et al. (2017). Si et al. used statistical features such as TF-IDF and linguistic features like emoji, part of speech, and emotion score to evaluate the performance of machine learning classifiers like XGBoost Classifier, Gradient Boosting Classifier (GBM), and Support Vector Machine (SVM) on three different datasets: English, Hindi, and Hinglish code-mixed. They obtained F1-scores of 68.13%, 54.82%, and 55.31% for the English, Hindi, and code-mixed datasets, respectively Si et al. (2019).

2.2 Using deep learning models

Recently, deep learning-based models have improved the performance of handcrafted feature models. A substantial amount of work has been done using deep learning models to detect hatred and inflammatory content. Mathur et al. used a CNN-based transfer learning approach to detect abusive tweets. They also introduced the HEOT dataset and the Profanity Lexicon Set Mathur et al. (2018). In addition, Mathur et al. (2018) classified Hate speech in Hinglish using a Multi-Input Multi-Channel transfer learning architecture based on a CNN-LSTM network. (Kamble and Joshi 2018; Kumar et al. 2020) have built a domain-specific word embedding to detect Hate speech in Hindi code mixed data and applied CNN, LSTM, and BiLSTM as a classifier and found that word-level feature is the most contributing feature for detecting Hate speech.

Santosh et al. worked with existing code-mixed datasets for Hate speech identification using two architectures: sub-word level LSTM model and Hierarchical LSTM model with attention based on phonemic sub-words(Santosh and Aravind 2019). Chopra et al. demonstrated how targeted hate embeddings combined with social network-based features outperform existing state-of-the-art models, both quantitatively and qualitatively (Chopra et al. 2020). (Chakravarthi et al. 2020; Kumar et al. 2020; Saumya et al. 2021) presented a code mixed data set for Malayalam–English language obtained from offensive comments on YouTube and Twitter, also achieved a baseline result of 75% F1 score using BERT's transformer model.

2.3 Using transformer models

Transformers are deep learning models developed with the attention mechanism in mind. The transformer model relies heavily on self-attention and a feed-forward neural network. The transformer employs a series of self-attention, feed-forward networks, and layer normalization's to encode the provided input text Vaswani et al. (2017). They have achieved the state of the art results on various natural language processing tasks.

Ayan et al. used the transformer model BERT and hierarchical attention network to investigate the relationship between five offensive features: anger, hatred, sarcasm, humor, and stance in Hindi–English code mixed social media content. They also used Pseudo-labeling techniques to create a combined annotated data set Sengupta et al. (2021). Mustafa Farooqi et al. (2021) used transformer-based models to identify hate material in a Hate speech and Offensive Content Identification (HASOC 2021) Hinglish code-mixed data set. They approached the problem in three ways: by using neural networks, by utilizing the transformer's cross-lingual embeddings, and finally by fine-tuning the transformers using transliterated Hindi text. Experimental results conclude that the ensembled setup of IndicBERT, XLM-RoBERTa, and mBERT performed better, with a weighted F1 score of 72%.

Ananya et al. classified hateful content in code-mixed data using context-based embedding from ELMo, FLAIR, and the transformer BERT models[31]. Saha et al. (2021) give a thorough examination of various transformer models, as well as a genetic algorithm technique is used to ensemble the result of various models. The authors employed a genetic algorithm to obtain optimal weights during the ensemble process. We found some limitations in the existing work which are stated in Table 1.

Most of the models listed in Table 1 tried to detect hate speech in the code-mixed text by using a few statistical constructed features that are extremely difficult to identify without domain expertise. Furthermore, few models tried to use transfer learning from models trained on monolingual text. These models, however, failed to detect hate features in the code-mixed text because it does not follow the same grammatical norms and syntax as the monolingual text. To overcome the difficulties caused by code-mixed text, our proposed model first attempted to convert the code-mixed text to monolingual text through a series of translation and transliteration processes before applying the classification. Since, due to the availability of pre-trained models trained on larger corpora, Hate speech detection can be performed better on monolingual data.

Table 1 Limitations of the existing models

Author	Model	Limitations
Bohra et al.	SVM with handcrafted features	They have not considered the context of the word; also model is trained using a few hand-picked features
Smghabadi et al.	LR with statistical features	They only worked with a monolingual Hindi data set
Shukrity Si et al.	SVM and ensemble classifier	Used manual dictionary-based approach
Abhinav et al.	CNN and LSTM	Statistical N-gram and TF-IDF are used for feature extraction
Santosh et al.	LSTM	They used sub-word level features extracted from models trained on monolingual text
Kamble et al.	LSTM and Word2Vec	The author's used statistical Word2Vec embeddings

Table 2 Samples from dataset

Text	Label
Sir phansi nahi sirf looted money wapas chaiya	No
Tujhe murder kar dungi main	Yes
I hate weddings ye madrchod log 1 ghante se ghar k niche dhol baja rhe h	Yes

3 Methodology

This section discusses how various models for evaluating Hate speech data are developed and tested using different approaches. We first discuss the data set used in the study, and later, its pre-treatment and different hate news classifiers are explained in detail in this section

3.1 Problem definition:

Let $\mathbf{S} = \{s_1, s_2, s_3, ...s_n\}$ be the set of input tweets, and $\mathbf{L} = \{l_1, l_2, l_3, ...l_n\}$ be the corresponding **n** labels for input **S**, where **S**∈{Hate, Non-hate} denotes the presence and absence of Hate speech, respectively. The goal of the proposed model is to predict the conditional label **'l'** for the given input **'s' i.e., P(l/s)**.

3.2 Data set description

The data set used to validate the proposed model was obtained from Bohra et al. (2018). The data set include both normal and Hate speech. The data collection contains 4575 code-mixed tweets, of which 1661 contain Hate speech, and the remaining 2914 code-mixed tweets in the data set consist of Non-Hate speech. All of these tweets were scraped from Twitter using the Twitter Python API. The data set is slightly unbalanced and consists of two fields: Text and Label[3]. Table 2 provides some of the sample sentences from the data set.

[3] https://github.com/deepanshu1995/HateSpeech-Hindi-English-Code-Mixed-Social-Media-Text.

3.3 Data preprocessing

As we all know, social media data contains noise. Several pre-treatment methods were carried out on the text fields to eliminate noise from the data set. The textual corpus had URLs, hyperlinks, emojis, stop words, and capital characters. Various preprocessing steps were carried out to simplify details, such as replacing punctuation with white spaces removing URLs and Twitter account names that could not be used to identify hate news. Texts were also lower-cased to avoid duplication problems. Further, the data lemmatization was carried out to translate tweets' words into their useful basic form. We have used a wordnet lemmatizer from NLTK to convert words into their basic form.

3.4 Baseline transformer model without translation

We performed experiments with two different transformer models trained in Indian languages. On top of these models, we applied traditional machine learning classifiers such as Logistic Regression (LR) and Support Vector Machine (SVM) for classification. The input from the preprocessing step is tokenized into several tokens before being passed to the padding layer, which turns uneven length sentences into equal length sentences. These padded tokens are then passed through a transformer model for feature selection; in our study, we employed two distinct transformer models; we used multilingual BERT for preliminary trials. mBERT is a bidirectional model that is built on the Transformer architecture (Devlin et al. 2018). The Transformer again relied on the attention mechanism (Vaswani et al. 2017). We employed multilingual BERT trained on 104 distinct languages from Wikipedia articles in the proposed model for feature extraction. mBERT, like BERT, holds 12 attention heads and 12 transformer blocks. On top of mBERT, we implement the aforementioned conventional algorithms.

In addition, we also experimented with IndicBERT, a multilingual ALBERT model trained on 12 different Indian languages which includes Assamese, Bengali, English, Gujarati, Hindi, Kannada, and Marathi. IndicBERT also

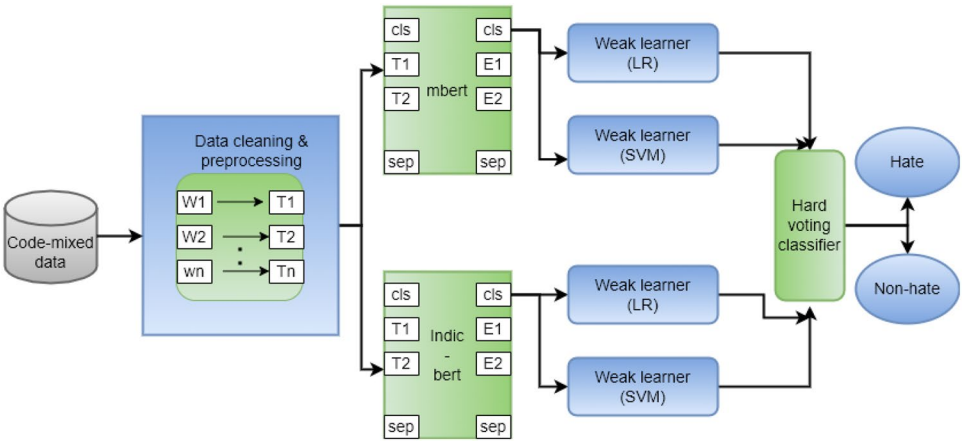

Fig. 1 Transformer-based model without translation

supports Oriya, Punjabi, Tamil, and Telugu. IndicBERT has fewer parameters than other BERT variants but still provides state-of-the-art performance Kakwani et al. (2020). The architecture of IndicBERT is similar to that of other BERT variants. Again, on top of IndicBERT, we add a conventional machine learning classifier, which takes input from the CLS token of dimension 768. To improve the model performance, we also experimented with their ensembled setup. We add weak learners LR and SVM classifiers in an ensemble model on top of the transformer models. The output from the weak learners is then routed through the Hard voting classifier model. The model takes input from each weak learner and predicts the class with maximum votes. For example, suppose predictions from the classifiers are('A,' 'A,' 'B'), so most of the classifiers have predicted 'A' as output. Hence 'A' will be the final prediction. The detailed architecture of the model is illustrated in Fig. 1.

3.5 Pre-trained language models

We also use state-of-the-art language model classifiers like BERT and ULMFiT. We used a simple classification API supplied by the developers to create the BERT and ULMFiT models. Our underlying option models for BERT and ULMFiT, respectively, are 'bert-base-uncased' and ASGD Weight Dropped LSTM (AWD-LSTM). AWD-LSTM is a state-of-the-art language model made out of ordinary LSTM without any attention. The AWD-LSTM process is divided into three stages: LM pre-training is the process of training a language model with a large wikitext-103 corpus to capture general language properties. Then, the model is fine-tuned to the target task data during LM fine-tuning. Finally, using the BPTT for Text Classification (BPT3C) language model, the classifier is fine-tuned. BPT3C Language models are trained with backpropagation through time (BPTT) to enable gradient propagation for large input sequences Howard and Ruder

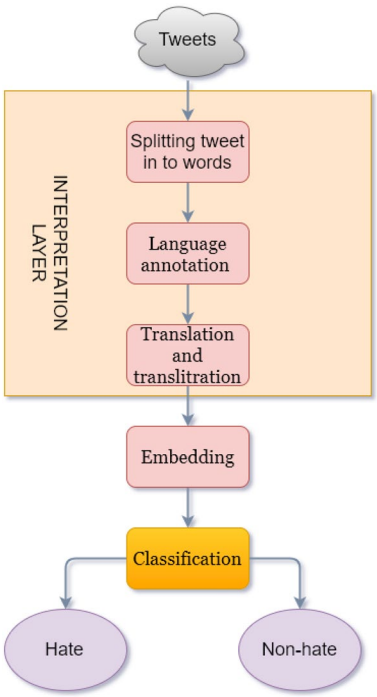

Fig. 2 Pipeline architecture of proposed model

(2018). The huggingface[4] in python is used to build BERT model while the fastai[5] is used to build ULMFiT.

3.6 Proposed TIF-DNN model

The proposed model (TIF-DNN) is built on three-layer architecture: the interpretation layer, the feature extraction layer, and the classification layer. Figure 2 illustrates the complete pipeline of the architecture.

[4] https://huggingface.co/docs/transformers/.

[5] https://docs.fast.ai/tutorial.text.

3.7 Interpretation layer

The interpretation layer forms the first layer in our proposed model; cleaned and lemmatized tweets are input to this layer. The Interpretation is performed in the following four steps:

1. Each tweet is separated into several words at the start using Python's split() function.
2. The Microsoft LID-tool[6] is used to annotate each word with its matching Lang-id. Language tags such as English, Hindi are used to annotate words.
3. Each annotated word is compared to its language id; if the Lang-id is English, the word is translated into the matching Devanagari term using Python's *Englishtohindi* module[7]. On the other hand, if Lang-id is Hindi, the word is transliterated to the Devanagari script using the Indic-transliteration function from the Indic-nlp-library[8].
4. Concatenates all transliterated and translated words to produce an original phrase that will be used as input to the feature extraction layer.

3.8 Feature extraction layer

The feature extraction layer receives a monolingual tweet with the Devanagari script from the previous layer. The tokenizer is then given a Devanagari tweet to turn each tweet into several tokens, with each word in the tweet considered a separate token. Padding and masking for variable-length phrases were also performed in conjunction with tokenization. The proposed model uses of transformer's mBERT tokenizer. Padded tokens are then passed through mBERT transformer model for feature extraction. We only drew embedding from the CLS token, which gives full-sentence embedding of 768 vector dimensions. These embeddings are then passed through the classification layer for stance detection.

3.9 Classification layer

We used two classifiers in our suggested model on Hate speech data. First, we tested conventional machine learning classifiers like Support Vector Machine (SVM), Logistic Regression (LR), Random Forest (RM), Naïve Bayes (NB), and K Nearest Neighbors (KNN) on translated and transliterated Devanagari script using mBERT embeddings. Later, we experimented with the Deep Neural Network (DNN) model, which acts as the second model in our suggested approach. DNN model comprises

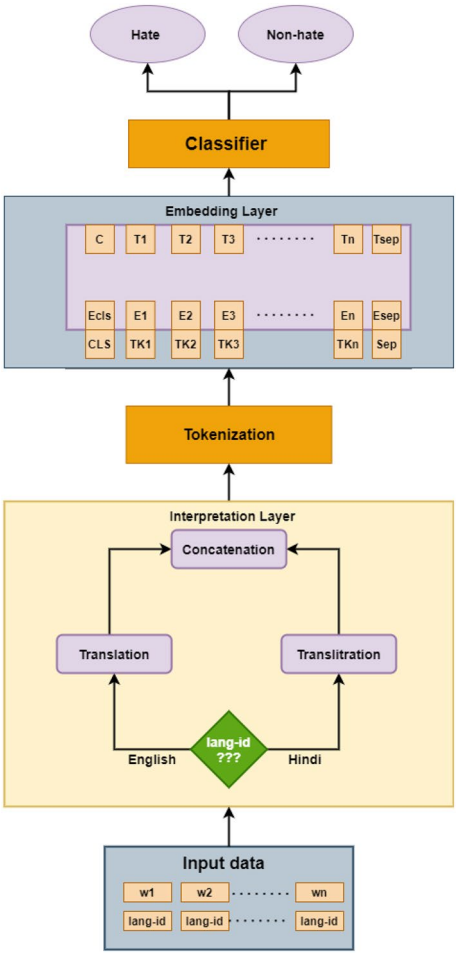

Fig. 3 Proposed TIF-DNN-based architecture

multiple dense layers, which aim to shape and compress the input in a meaningful fashion. Dense layers are those that are fully connected. A dropout layer follows each dense layer to avoid over-fitting problems. We also used a batch normalization layer to normalize activation values; the normalization layer calculates new activation values as follows.

$h_{ij}^{norm} = (h_{ij} - j) / \sigma_j$

$h_{ij}^{final} = \gamma_j . h_{ij}^{norm} + \beta_j$ where 'γ' is mean, and 'σ' is the standard deviation. The detailed architecture of the proposed model is illustrated in Fig. 3

4 Results and implementation

The experiment is started with transformer models; on top of the transformer, we implemented several traditional machine learning classifiers like LR, SVM, RF, NB, and KNN. The experimental trials found that the mBERT model and traditional machine learning classifiers have performed better

[6] https://github.com/microsoft/LID-tool.

[7] https://pypi.org/project/English-to-Hindi/.

[8] https://github.com/anoopkunchukuttan/indic_nlp_library.

Table 3 Classifier's parameters

Classifier	Hyper-parameter
Logistic regression	C=1, max-iter=500
Random forest	no-of-estimators=200
Naïve bayes	var-smoothing=1e-09
Support vector machine	c=1, solver='lbfgs,' kernel='linear'
K nearest neighbors	n-neighbors=24
TIF-DNN model	lr=1e-4, loss='binary-cross-entropy,' optimizer=adam

than IndicBERT for hate content detection in the code-mixed Hinglish Twitter data set. On the other hand, IndicBERT failed to achieve a better result because the model is trained using Devanagari script; hence it failed to identify context information from Romanized script. Parameters used during the training of the aforementioned classifiers are indicated in Table 3. The parameters such as learning rate, loss function, and optimizers are selected from experimental trials, while the others are selected through optimal grid search. These algorithms were implemented using the Scikit-learn library, using a train-test ratio of 70:30. The outcomes of these algorithms provide baseline results for the proposed model. According to the results shown in Table 4, LR has achieved a better result for IndicBERT with 65% accuracy, and SVM performed better on mBERT embeddings with 67% accuracy. We also experiment with language models such as pre-trained BERT and ULMFiT, the results of which are shown in Table 5. From the Table, BERT outperformed ULMFiT in language models, with an accuracy of 71%. However, even language models failed to produce improved results since they were primarily trained on high-resource languages such as English.

4.1 Proposed model results

The proposed model uses the interpretation layer to transform multilingual input data to a monolingual form. We

Table 4 Baseline transformer model results

	Model	Accuracy (%)	F1-Hate (%)	F1-Non-hate (%)
IndicBERT embeddings	LR	65	37	75
	SVM	64	35	74
	KNN	63	21	74
	RF	62	28	71
	NB	58	51	62
mBERT embeddings	LR	66	44	76
	SVM	67	46	76
	KNN	64	22	77
	RF	64	21	77
	NB	54	49	58
	Ensemble	**68**	**52**	**77**

Highlighted values indicated best-performing models.

Table 5 Comparative study of proposed model with baseline results

	Model	Accuracy (%)	F1-Hate (%)	F1-Non-hate (%)
Baseline	Ensemble	68	50	77
Language models	BERT	71	59	72
	ULMFiT	68	48	75
Proposed Translation & Transliteration-based model	LR	70	48	77
	SVM	72	55	76
	KNN	68	44	78
	RF	66	28	77
	NB	56	57	55
	TIF-DNN	**73**	**56**	**78**

Highlighted values indicated best-performing models.

Table 6 Sample test cases

Text	Ensemble	BERT	TIF-DNN	Target
Fake hina besaram arshi ki sakal se hi nafrat ho gyi hme	Non-hate	Hate	Hate	Hate
Thank you geeta ma am insaan nafrat me sahi aur galat hi bhool jata	Non-hate	Non-hate	Non-hate	Non-hate
Murder karne waale ko aachaarsahita ki wajha se action se dur kiya hua hai har	Non-hate	Non-hate	Non-hate	Hate

found that Hate speech identification on monolingual data sets has better accuracy from the literature. An mBERT subsequently processes the translated data for feature extraction. First, classical machine learning models are used on top of the mBERT model for classification; among these, SVM achieves a higher accuracy of 72 percent. Later, we experimented with a Deep Neural Network-based model to improve the performance of existing techniques. In the DNN model, the mBERT input is initially passed through dense layers of sizes 1000, 500, 100, and 50; batch normalization and dropout of 0.4 are added to avoid over-fitting problems. The dropout value of 0.4 is decided based on the experimental trials. Finally, the output from the last dense layer is passed through the sigmoid layer for stance detection. Table 5 compares the outcomes of our proposed method with the baseline classifiers. As shown in the table proposed model outperformed top-performing baseline models with an accuracy of 73%.

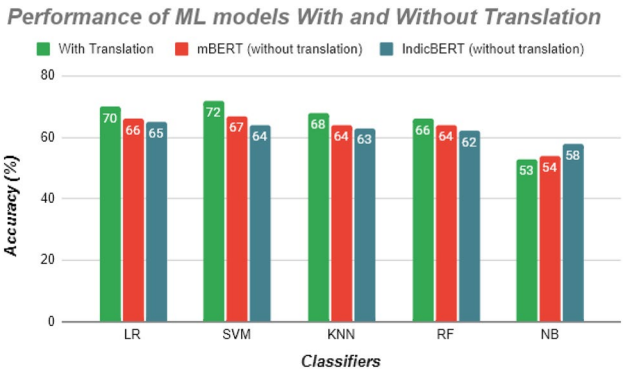

Fig. 4 Performance of ML models with and without translation

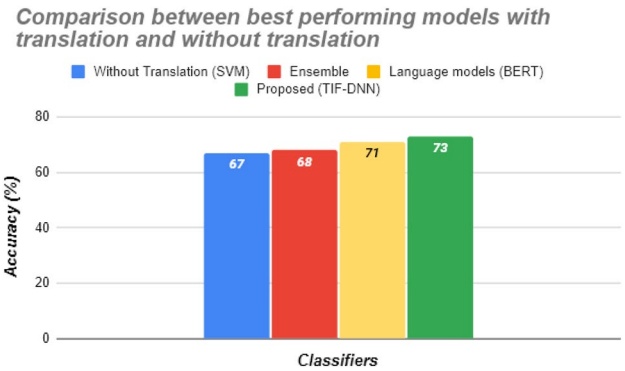

Fig. 5 Comparison of best performing models

5 Discussion

To examine the behavior of individual models, we selected sample phrases from test data. Then, we passed them through the best-performing models for stance identification, and the results are included in Table 6. According to the table, most models correctly recognized non-hate sentences, but only BERT and the proposed model correctly classified Hate speech. The reason for our suggested model's better performance is that we first transformed code-mixed data to monolingual, then extracted features using Transformer models trained on monolingual text. On the other hand, other models failed to capture the hateful features from the data set because they were not trained on code-mixed text.

To evaluate the efficacy of our suggested translation and transliteration model, we compared our benchmark findings on both translated and untranslated data, and the results are summarized in Fig. 4. The observations from the comparative study indicate the translation process improves model performance significantly. The main conclusion that can be made from our experimental data is that the process of translating code-mixed text to monolingual text during classification will increase model performance. Another interesting observation from our experiments is that SVM outperforms all other machine learning models on both translated and original text, with an accuracy of 72%. Other contribution of experimental trials includes, among the pre-trained transformer-based embeddings, mBERT outperforms IndicBERT for hate speech identification in code-mixed text. Further, we also compared the outcomes of the best-performing models, and the results are summarized in Fig. 5. The experimental results show that our suggested TIF-DNN outperforms all baseline models by considerably improving classification accuracy. Furthermore, to validate our claims we compared the proposed work with existing models, and the results are shown in Table 7. On Twitter data, the proposed model outperformed existing methods for Hate speech recognition.

Table 7 Comparison with existing work

Model	Accuracy (HS)%
Mathur et al. (2018) (CNN-LSTM)	72
Bohra et al. (2018) (Random Forest)	65
Bohra et al. (2018) (SVM)	71
Santosh and Aravind (2019) (Sub-word level LSTM)	71
Proposed TIF-DNN model	**73**

Highlighted values indicated best-performing models.

5.1 Limitations of our model

To understand the limitations of our model, we examine the outcomes of our models, which inspire new research directions. Some of these limitations are as follows.

1. As shown in Table 5, when compared to Hate speech, the proposed model exhibits higher accuracy for Non-Hate speech identification. Unbalanced data used during the training process might be one of the reasons.
2. If the performance of the translator model used during interpretation is improved further, the performance of the proposed model can be improved. Our translation model struggles to find the exact translated term in Devanagari script for the few equivalent English words in a code-mixed tweet. As a result, there are significant mistranslations in our translated tweets.

6 Conclusion and future enhancements

The proposed approach investigates Hate speech identification in a Hindi–English code-mixed Twitter data set. In this article, we proposed the TIF-DNN model for Hate speech identification. We proved the efficacy of the proposed model by comparing the results of the suggested model to baseline classifiers and past work. The findings also revealed that the proposed translator-based models outperform several baseline classifiers and existing work. However, better results may be obtained if a more powerful translation model is included in future studies. Furthermore, experiments detailed in this study can be repeated on other regional languages as part of future research, which is essential because India is a multilingual society with numerous local languages.

Funding No funding was received to assist with the preparation of this manuscript.

Declarations

Conflict of interest No potential competing interest was reported by the authors.

Ethical approval • This article does not contain any studies with animals performed by any of the authors. • This article does not contain any studies with human participants or animals performed by any of the authors.

References

Aguilar G, Solorio T (2020) From english to code-switching: transfer learning with strong morphological clues. In: Proceedings of the 58th annual meeting of the association for computational linguistics, pp. 8033–8044. https://doi.org/10.18653/v1/2020.acl-main.716

Arora G (2020) inltk: Natural language toolkit for indic languages. In: Proceedings of second workshop for NLP open source software (NLP-OSS), pp. 66–71. https://doi.org/10.18653/v1/2020.nlposs-1.10

Banerjee S, Sarkar M, Agrawal N, Saha P, Das M (2021) Exploring transformer based models to identify hate speech and offensive content in english and indo-aryan languages. arXiv preprint arXiv:2111.13974. https://doi.org/10.48550/arXiv.2111.13974

Biradar S, Saumya S (2022) Iiitdwd@ tamilnlp-acl2022: Transformer-based approach to classify abusive content in dravidian code-mixed text. In: Proceedings of the second workshop on speech and language technologies for Dravidian languages, pp. 100–104 . https://doi.org/10.18653/v1/2022.dravidianlangtech-1.16

Biradar S, Saumya S, Chauhan A (2021) Hate or non-hate: translation based hate speech identification in code-mixed hinglish data set. In: 2021 IEEE international conference on big data (big data), IEEE, pp. 2470–2475. https://doi.org/10.1109/BigData52589.2021.9671526

Bisht A, Singh A, Bhadauria H, Virmani J (2020) Detection of hate speech and offensive language in twitter data using lstm model. Recent Trends Image Signal Process Comput Vis. https://doi.org/10.1007/978-981-15-2740-1_17

Bohra A, Vijay D, Singh V, Akhtar SS, Shrivastava M (2018) A dataset of hindi-english code-mixed social media text for hate speech detection. In: Proceedings of the second workshop on computational modeling of people's opinions, personality, and emotions in social media, pp. 36–41. https://doi.org/10.18653/v1/W18-1105

Bokamba EG (1989) Are there syntactic constraints on code-mixing? World Engl 8(3):277–292. https://doi.org/10.1111/j.1467-971X.1989.tb00669.x

Chakravarthi BR, Jose N, Suryawanshi S, Sherly E, McCrae JP (2020) A sentiment analysis dataset for code-mixed malayalam-english. arXiv preprint arXiv:2006.00210 (2020). https://doi.org/10.48550/arXiv.2006.00210

Chopra S, Sawhney R, Mathur P, Shah RR (2020) Hindi-english hate speech detection: author profiling, debiasing, and practical perspectives. In: Proceedings of the AAAI conference on artificial intelligence, vol. 34, pp. 386–393. https://doi.org/10.1609/aaai.v34i01.5374

Devlin J, Chang M-W, Lee K, Toutanova K (2018) Bert: Pre-training of deep bidirectional transformers for language understanding. arXiv preprint arXiv:1810.04805

Ghosh S, Ghosh S, Das D (2017) Sentiment identification in code-mixed social media text. arXiv preprint arXiv:1707.01184. https://doi.org/10.48550/arXiv.1707.01184

Howard J, Ruder S (2018) Universal language model fine-tuning for text classification. In: Proceedings of the 56th annual meeting of the association for computational linguistics (Volume 1: Long Papers), pp. 328–339. https://doi.org/10.48550/arXiv.1801.06146

Kakwani D, Kunchukuttan A, Golla S, Gokul N, Bhattacharyya A, Khapra MM, Kumar P (2020) Indicnlpsuite: Monolingual corpora, evaluation benchmarks and pre-trained multilingual language models for indian languages. In: Findings of the association for computational linguistics: EMNLP 2020, pp. 4948–4961. https://doi.org/10.18653/v1/2020.findings-emnlp.445

Kamble S, Joshi A (2018) Hate speech detection from code-mixed hindi-english tweets using deep learning models. arXiv preprint arXiv:1811.05145 . https://doi.org/10.48550/arXiv.1811.05145

Khan MU, Abbas A, Rehman A, Nawaz R (2020) Hateclassify: a service framework for hate speech identification on social media. IEEE Internet Comput 25(1):40–49. https://doi.org/10.1109/MIC.2020.3037034

Kumar A, Saumya S, Singh JP (2020) Nitp-ai-nlp@ hasoc-dravidian-codemix-fire2020: A machine learning approach to identify offensive languages from dravidian code-mixed text. In: FIRE (Working notes), pp. 384–390. http://ceur-ws.org/Vol-2826/T2-38.pdf

Kunchukuttan A, Kakwani D, Golla S, Bhattacharyya A, Khapra MM, Kumar P *et al.* (2020) Ai4bharat-indicnlp corpus: monolingual corpora and word embeddings for indic languages. arXiv preprint arXiv:2005.00085. https://doi.org/10.48550/arXiv.2005.00085

Lal YK, Kumar V, Dhar M, Shrivastava M, Koehn P (2019) De-mixing sentiment from code-mixed text. In: Proceedings of the 57th annual meeting of the association for computational linguistics: student research workshop, pp. 371–377. https://doi.org/10.18653/v1/P19-2052

Mathur P, Sawhney R, Ayyar M, Shah R (2018) Did you offend me? classification of offensive tweets in hinglish language. In: Proceedings of the 2nd workshop on abusive language online (ALW2), pp. 138–148 (2018). https://doi.org/10.18653/v1/W18-5118

Mathur P, Shah R, Sawhney R, Mahata D (2018) Detecting offensive tweets in hindi-english code-switched language. In: Proceedings of the sixth international workshop on natural language processing for social media, pp. 18–26 (2018). https://doi.org/10.18653/v1/W18-3504

Mossie Z, Wang J-H (2020) Vulnerable community identification using hate speech detection on social media. Inf Process Manag 57(3):102087. https://doi.org/10.1016/j.ipm.2019.102087

Mustafa Farooqi Z, Ghosh S, Ratn Shah R (2021) Leveraging transformers for hate speech detection in conversational code-mixed tweets. arXiv e-prints, 2112. https://doi.org/10.48550/arXiv.2112.09986

Pratapa A, Choudhury M, Sitaram S (2018) Word embeddings for code-mixed language processing. In: Proceedings of the 2018 conference on empirical methods in natural language processing, pp. 3067–3072. https://doi.org/10.18653/v1/D18-1344

Qi P, Zhang Y, Zhang Y, Bolton J, Manning CD (2020) Stanza: a python natural language processing toolkit for many human languages. In: Proceedings of the 58th annual meeting of the association for computational linguistics: system demonstrations, pp. 101–108. https://doi.org/10.18653/v1/2020.acl-demos.14

Saha D, Paharia N, Chakraborty D, Saha P, Mukherjee A (2021) Hate-alert@ dravidianlangtech-eacl2021: Ensembling strategies for transformer-based offensive language detection. In: Proceedings of the first workshop on speech and language technologies for Dravidian languages, pp. 270–276. https://aclanthology.org/2021.dravidianlangtech-1.38

Samghabadi NS, Mave D, Kar S, Solorio T (2018) Ritual-uh at trac 2018 shared task: aggression identification. COLING 2018, 12 . https://doi.org/10.48550/arXiv.1807.11712

Santosh T, Aravind K (2019) Hate speech detection in hindi-english code-mixed social media text. In: Proceedings of the ACM India joint international conference on data science and management of data, pp. 310–313. https://doi.org/10.1145/3297001.3297048

Saumya S, Kumar A, Singh JP (2021) Offensive language identification in dravidian code mixed social media text. In: Proceedings of the first workshop on speech and language technologies for dravidian languages, pp. 36–45. https://aclanthology.org/2021.dravidianlangtech-1.5

Senarath Y, Purohit H (2020) Evaluating semantic feature representations to efficiently detect hate intent on social media. In: 2020 IEEE 14th International Conference on Semantic Computing (ICSC), IEEE, pp. 199–202. https://doi.org/10.1109/ICSC.2020.00041

Sengupta A, Bhattacharjee SK, Akhtar MS, Chakraborty T (2021) Does aggression lead to hate? detecting and reasoning offensive traits in hinglish code-mixed texts. Neurocomputing. https://doi.org/10.1016/j.neucom.2021.11.053

Si S, Datta A, Banerjee S, Naskar SK (2019) Aggression detection on multilingual social media text. In: 2019 10th international conference on computing, communication and networking technologies (ICCCNT), IEEE, pp. 1–5 (2019). https://doi.org/10.1109/ICCCNT45670.2019.8944868

Singh R, Choudhary N, Shrivastava M (2018) Automatic normalization of word variations in code-mixed social media text. arXiv preprint arXiv:1804.00804

Srivastava A, Hasan M, Yagnik B, Walambe R, Kotecha K (2021) Role of artificial intelligence in detection of hateful speech for hinglish data on social media. In: Applications of artificial intelligence and machine learning, p. 83. https://doi.org/10.1007/978-981-16-3067-5_8

Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I (2017) Attention is all you need. Adv Neural Inf Process Syst 30