Scalable Computing: Practice and Experience, ISSN 1895-1767, http://www.scpe.org
© 2023 SCPE. Volume 24, Issues 4, pp. 941–955, DOI 10.12694/scpe.v24i4.2469

HATE SPEECH DETECTION IN LOW-RESOURCE BODO AND ASSAMESE TEXTS WITH ML-DL AND BERT MODELS

KOYEL GHOSH*, APURBALAL SENAPATI†, MWNTHAI NARZARY‡ AND MAHARAJ BRAHMA §

Abstract. Hate speech detection research is a recent sizzling topic in natural language processing (NLP). Unburdened uses of social media platforms make people over-opinionative, which crosses the limit of leaving comments and posts toxic. A toxic outlook increases violence towards the neighbour, state, country, and continent. Several laws have been introduced in different countries to end the emergency problem. Now, all the media platforms have started working on restricting hate posts or comments. Hate speech detection is generally a text classification problem if considered a supervised observation. To tackle text in terms of computation perspective is challenging because of its semantic and complex grammatical nature. Resource-rich languages leverage their richness, whereas resource scarce language suffers significantly from a lack of dataset. This paper makes a multifaceted contribution encompassing resource generation, experimentation with Machine Learning (ML), Deep Learning (DL) and state-of-the-art transformer-based models, and a comprehensive evaluation of model performance, including thorough error analysis. In the realm of resource generation, it adds to the **N**orth-**E**ast **I**ndian **H**ate **S**peech tagged dataset (NEIHS version 1), which encompasses two languages: Assamese and Bodo.

Key words: Hate Speech Detection, Assamese, Bodo, Natural Language Processing, NLP, Machine Learning, Deep Learning, Word2Vec, NB, SVM, LSTM, BiLSTM, CNN, BERT.

1. Introduction. People can use the internet to learn new skills, engage in knowledgeable debates, and share information. Still, they also engage in anti-social activities such as cyberbullying, trolling, spreading hate, and so on [1]. Spreading hate can lead people to communal riots [2, 3], can be responsible for mental illness [4] etc. United Nations Strategy and Plan of Action on hate speech [5] has been introduced and defines hate speech as *any kind of communication in speech, writing or behaviour, that attacks or uses pejorative or discriminatory language regarding a person or a group based on their religion, ethnicity, colour, gender or other identity factors.* Indian government law is also introduced against hate speech [6]. Several social media platforms revised their community guidelines to eradicate hate, automatically detecting hate comments and posts and giving users access to report posts and comments[1][2]. English and other popular languages benefit from their global popularity. Few studies are done on Indian languages in hate speech detection [7, 8, 9] like in Hindi, Bangla, Marathi, etc. India has 22 official languages and about 1,000 living languages from various language groups [10]. North-East Indian languages are under-resourced, and no tagged hate speech detection task data is available. As an associate official language of the Indian state of Assam, Bodo is widely spoken in the Bodoland Territorial Region[3]. Among the official languages of India, it has gained some recognition[4] [11]. The 2011 Indian Census[5] estimates a total of 1,482,929 Bodo speakers, including 1,454,547 native speakers. Assamese is spoken by 15,311,351 people, which is a huge number. Assamese and Bodo suffer from a serious lack of resources for Natural Language Processing (NLP). Native researchers are trying to build a sufficient

*Department of Computer Science and Engineering, Central Institute of Technology, Kokrajhar, Assam, India (ghosh.koyel8@gmail.com)
†Department of Computer Science and Engineering, Central Institute of Technology, Kokrajhar, Assam, India (a.senapati@cit.ac.in)
‡Department of Computer Science and Engineering, Central Institute of Technology, Kokrajhar, Assam, India (mwnthainarzary123@gmail.com)
§Department of Computer Science and Engineering, IIT Hyderabad (mraj.brahma@gmail.com)
[1]https://web.facebook.com/communitystandards/ (Access on 02.10.2023)
[2]https://www.youtube.com/howyoutubeworks/policies/community-guidelines/(Access on 02.10.2023)
[3]Formerly known as the Bodoland Territorial Autonomous District (BTAD)
[4]Scheduled languages of India: Bodo is one of the 22 scheduled languages, added in 2004
[5]https://censusindia.gov.in/census.website/data/census-tables (Access on 02.10.2023)

corpus for the research [12], describing a process for creating a monolingual Bodo corpus using Google Keep for OCR to scan text from different books. Major contributing factors are the paucity of language models labelled datasets and effective machine learning (ML) approaches for a wide range of NLP jobs. Due to unfettered access and usage of social media and digitalization, hate speech is rising in North-East India, too, just as in other major languages like English. To the best of our knowledge, no hate speech detection dataset is available in Assamese or Bodo. This is our first attempt at creating a North-East Indian Hate Speech (NEIHS) dataset for binary text classification with *2K* in Bodo and *4K* in Assamese respectively, labelled as *'hate'* and *'non-hate'*. We found that state-of-the-art models like transformers are more effective than ML or DL approaches. Our contribution to this paper :

1. We create the North-East Indian Hate Speech dataset (NEIHS). NEIHS (version 1) is a binary text classification task-purpose human annotate dataset.
2. Trained Naïve Bayes (NB), Support Vector Machines (SVM), Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (BiLSTM), Convolutional Neural Network (CNN), and state-of-the-art transformer-based models on the NEIHS dataset.
3. Detailed analysis of the results of all models to check how a model works on the language-specific hate speech data.
4. At the end, present a detailed error analysis for the outcomes.

The rest of the paper is arranged in a way where Section 2 shall explain the related work; Section 3 describes the details of the NEIHS dataset. Section 4 will be the methods used on the datasets, and detailed results with error analysis will be shown in section 5. Lastly, we conclude our work in section 6.

2. Related works. Here, we shall discuss existing hate speech detection research. To correctly and reliably identify hate speech from widely spoken languages like English, many works have been proposed [13, 14]. Traditional algorithms like SVM, NB, Logistic Regression (LR), Decision Trees (DT), Random Forest (RF), and Gradient-Boosted Trees (GBT) have relied on manual feature engineering for quite some time. On the other hand, CNN or LSTM networks form the backbone of Deep Learning (DL) based techniques that learn multilayers of abstract features from raw texts. Since linear models have proven less accurate and scalable when dealing with billions of such texts, these methods pale compared to DLs. When extracting features, CNN excels, while LSTM excels at modelling neatly sequenced learning issues. For example, CNNs can extract n-grams, sequences of words or characters, and LSTM may learn the long-term relationships between words or characters in texts. Each network architecture has benefits, but various works have investigated what would happen if combined [15]. With Conv-LSTM, the class of a word sequence depends on previous word sequences, making it a powerful architecture that captures long-term correlations between characteristics retrieved by CNN. This architecture is more effective than structures exclusively based on CNN or LSTM. Variation of BERT (Bidirectional Encoder Representations from Transformers) [16], RoBERTa (Robustly optimized BERT) [17], ALBERT (A Lite BERT) [18], DistilBERT (Distilled version of BERT) [19] and their pre-trained models such as mBERT [16], MuRILBERT [20], RoBERTaHindi[6], Indic Bert [22], MahaBERT [23], MahaRoBERTa [24], XLM-RoBERTa [25], BanglaBert [26] etc. is used for hate speech detection experiments. This section mainly focuses on hate speech detection in Indian languages. Nowadays, mainly code-mixed languages like Hindi-English [27], Tamil-English [28] etc. are preferred for the research as social media users use complex code-mixed language, like their native language as well as English in between sentences, but we shall stick to the monolingual experiments for the discussion. HASOC (Hate Speech and Offensive Content Identification), a shared task organized by FIRE (Forum for Information Retrieval Evaluation)[7], which published hate datasets in Indian languages such as Hindi, Marathi, etc. HASOC offers four subtracks, one of which is relevant to us: **HASOC - English and Indo-Aryan Languages**. Datasets are distributed in tab-separated format. HASOC and most other collections require mechanisms to detect hateful content from the text of a post. In 2019, the HASOC-Hindi dataset offered three tasks [7]. The first task is binary classification, i.e., subtask A. The second task is to find whether the hate comment was profane or abusive (multiclass), i.e., subtask B. The third is to predict whether the hate comment is targeted or untargeted (multiclass), i.e., subtask C. In the Hindi language, ninety-three runs were submitted across three sub-tasks. Regarding the Hindi subtask A, the winner

[6]https://huggingface.co/flax-community/roberta-hindi (Access on 02.10.2023)
[7]http://fire.irsi.res.in/fire/2022/home (Access on 02.10.2023)

team, QutNocturnal [29], employed a CNN base technique with Word2vec embedding and got better Marco F1 and Weighted F1 values, 0.8149 and 0.8202, respectively. The second team LGI2P [30], trained a fastText model for the proposed Hindi language and later used BERT for classification. The system achieved 0.8111 Marco-F1 and 0.8116 Weighted-F1 values. For sub-task B on the Hindi Dataset, 3Idiots [31] scores 0.5812 and 0.7147 in Marco-F1 and Weighted-F1 utilizing BERT. Team A3-108 [32] achieves a high Marco-F1 score on sub-task C Hindi Dataset, which is 0.5754. According to them, Adaboost [33] was the best-performing classifier among the three classifiers, i.e., Adaboost or Adaptive Boosting (AB), RF, Linear SVM. They merge multiple weak classifiers to construct a robust prediction model, but an ensemble of SVM, RF, and AB with hard voting performed even better. This classifier used TF-IDF features of word unigrams and characters 2, 3, 4, and 5 grams with an additional feature of the length of every tweet. In HASOC 2020, two Hate Speech detection tasks [8], sub-task A (binary class) and sub-task B (multiclass) are proposed with another Hindi dataset in the research area. NSIT_ML_Geeks [34] outperforms other competing teams, scoring Marco-F1 0.5337 and 0.2667 in sub-task A and sub-task B, respectively, utilizing CNN and BiLSTM. In 2021, HASOC published a Hindi dataset [9] with sub-tasks A and B again. The best submission was achieved Macro F1 0.7825 in sub-task A with a fine-tuned Multilingual-BERT (20 epochs) with a classifier layer added at the final phase. The second team also fine-tuned Multilingual-BERT and scored Macro F1 0.7797. NeuralSpace [35] got Macro F1 0.5603 in sub-task B. They use an XLM-R transformer, vector representations for emojis using the system Emoji2Vec, and sentence embeddings for hashtags. After that, three resulting representations were concatenated before classification. In the paper [36], they used the pre-trained multilingual BERT (m-BERT) model for computing the input embedding on the Hostility Detection Dataset (Hindi) later SVM, RF, MultiLayer Perceptron (MLP), LR models are used as classifiers. In coarse-grained evaluation, SVM reported the best weighted-F1 score of 84%, whereas they obtained 84%, 83%, and 80% weighted-F1 scores for LR, MLP, and RF. In fine-grained evaluation, SVM has the most excellent F1 score for evaluating three hostile dimensions, namely Hate (47%), Offensive (42%), and Defamation (43%). LR beats the others in the Fake dimension with an F1 score of 68%. Authors [28] prepare a Tamil dataset to identify Homophobia and Transphobia from comments. They experiment with baseline models, machine learning, deep learning, and transformer-based models. We found a paper[8], authors prepare a large-scale monolingual Indic to hate speech dataset in 5 languages: Hindi (Hi), Tamil (Ta), Telugu (Te), Malayalam (Ml) and Kannada (Kn). A multilingual, monolingual transformer comparison of hate speech dataset on HASOC-2019(Hindi), HASOC-2019(Marathi), and Bangla hate dataset [37].

3. NEIHS Dataset.

3.1. Dataset collection. Our primary goal while creating the dataset was to create it with different varieties of data, so we chose different kinds of ™Facebook pages like political, entertainment, etc. We first identify some controversial posts, including recent events, politicians, and actors who are more likely to contain hate speech. Then, we go through comments on such posts and find comments that are written in at least 80-90% monolingual. Then, we manually check whether these include hate and categorize them accordingly. Figure 3.1 shows both datasets' word clouds. Bodo shares the Devnagari script, and Assamese shares the Assamese script, likely the Bengali-Assamese script. Figure 3.2 shows the most used hate words in the NEIHS dataset.

We have collected all the data from ™Facebook and ™YouTube comments for the NEIHS dataset. We followed a few Bodo ™Facebook pages, such as news, entertainment, celebrity, politics, etc., and selected some uploads to manage related comments and retrieved over a thousand comments with non-hate and hate comments from various posts published between January 2022 and September 2022 using open source scrapper tools[9]. The same we did with some Bodo local news, celebrity, politics, etc. ™Youtube channels. Most of the comments we collected are in English transliteration, and the sentences include unwanted symbols like `-`, `'`, `(`, `)`, etc. We cleaned and preprocessed sentences. Removed unnecessary symbols, structured sentences translated and rewrote sentences, and Bodo Script. Finally, the sentences were annotated as either 'hate' or 'non-hate' by native speakers. Sentences with 'hate' that include hate words are considered hate-offensive statements, while sentences that convey formal information, suggestions, or questions are considered non-hate sentences. We have

[8]https://openreview.net/forum?id=HCnb1TByvx7 (Access on 02.10.2023)
[9]https://github.com/kevinzg/facebook-scraper (Access on 02.10.2023)

Fig. 3.1: Wordclouds of NEIHS - a) Assamese and b) Bodo datasets.

Hate words/ Assamese	Hate words/ Bodo
মাকচুদু (Motherfucker), বনৰি/বুনৰী/ৰেনদি (Prostitute), কেলা (It is a word that doesn't itself has a meaning but is used within sentences to express anger, irritation etc), চুদুৰভাই (Fucker), জহৰি (indicates women for tricking people), গেদা (indicates Bangladeshis mostly muslims)	फाग्ला (mad), गुनदा (Rapist), हांरोंने सुथुर (Enemy for comminity), हारि बेफारि (community seller), जामबा (mad man), बुरबक (Dump), जोलैनि हांगार (betrayer)

Fig. 3.2: Some examples of Hate words in NEIHS - a) Assamese and b) Bodo datasets

collected the Assamese dataset in the same manner.

3.2. Dataset annotation. Two native speakers, young adults in the 19 to 24 age range, annotated the data for each language. These annotators are students at the Central Institute of Technology in Kokrajhar, Assam, India. The annotation team's task was to manually categorize comments from NEIHS as either containing hateful content or not, using binary labels. In cases where there was a disagreement between the two student annotators regarding the label assignment, a third student with expertise in social media research was consulted to make the final call. Hate speech is a highly subjective issue. As a result, defining what constitutes hate speech is difficult. As a result, we've established specific strict guidelines. These regulations are based on the community standards of ™Facebook[10] and ™YouTube[11]. Comments with the following aims should be marked as hate. (a) *Profanity*: Comments that contain profanity, cussing, or swear words are marked as hate. (b) *Sexual orientation*: Sexual attraction (or a combination of these) to people of the opposite sex or gender, to people of the same sex or gender, to both sexes, or to people of more than one gender. (c) *Personal*: remarks on clothing sense, content selection, language selection, etc. (d) *Gender chauvinism*: Comments in which people are targeted because of gender. (e) *Religious*: Comments in which the person is criticized for their religious beliefs and practices. For example, comments challenging the use of a turban or a burkha (the veil), (f) *Political*: Comments criticizing a person's political beliefs. For instance, bullying people for supporting a political party. (g) *Violent intention*: Comments containing a threat or call to violence.

Examples as in figure 3.3 are '' which roughly translates to 'What kind of man is he?', which contains a word usually identified as the hate word '' but the meaning does not contain any hate. ''. Now the sentence roughly translates to 'Total AAP supporters– 80% Miya(50% of them are fake accounts and another 30% is their real account), after 80% Miya's. The remaining ones are Badan (These are the supporters of Congress and Akhil).' We need to know the context 'Miya' is used as a word to describe illegal Bangladeshi immigrants, usually 'Muslims', and 'Badan' is considered a traitor because he invited the rulers of Myanmar to attack

[10]https://web.facebook.com/communitystandards/ (Access on 02.10.2023)
[11]https://www.youtube.com/howyoutubeworks/policies/community-guidelines/ (Access on 02.10.2023)

Comments (Assamese/ English translation)	Hate category	Comments (Bodo/ English translation)	Hate category
চাল্লা কি মানুহ (What kind of man is he?)	non-hate	आंग सानो जाय बाराद्राय बखियो बे जेबो मावनो रोडा (Those who constantly abuse others will achieve nothing.)	non-hate
দেখাত জেহাদি জেহাদি লাগে (You look like a Jihadi.)	Personal Attack	AASU नि आसाम Accord नि खोथाखौ सोरबा बरफोरा मोनथिगौब्ला गोसो खांबानो जाबाय (Those Bodos who are familiar with the AASU and the Assam Accord are requested to keep this in mind.)	Personal Attack
মুঠ আপ সমর্থক--৮০% মিঞা(তাৰে ৫০% ফেক নামত কমেণ্ট দিয়ে বাকী ৩০% নিজ নামত) গ'ল ৮০% মিঞা। বাকী থাকিল ২০% বদন(এইকেইটা হ'ল কংগ্রেছ অখিল সমর্থক)" (Total AAP supporters-- 80% Miya(50% of them are fake accounts and the other 30% is their real accounts), after 80% Miya's. The remaining ones are Badan (These are the supporters of Congress and Akhil).)	Political Hate	हनै मालाय हारसा हिनजाबनि खिबु सुग्राफोरा मिनिसोदों (Look, Assamese women bum cleaner are laughing.)	Gender
99% গেদাই অসমত চোৰ ধর্ষন হত্যা কৰে ..। (99% of Rapes and Robberies are done by Gedas..)	Religious Hate	सोरबा मावा मोनसे खामानि मावनो थांनायाव मानि हेंथा गिखफोर ? (Why was there always a barieer when they were going to work for the good ?)	non-hate

Fig. 3.3: Examples from NEIHS dataset. Hate examples and their translations in English with descriptions (Hate category) are boxed in red, whereas non-hate examples are in green.

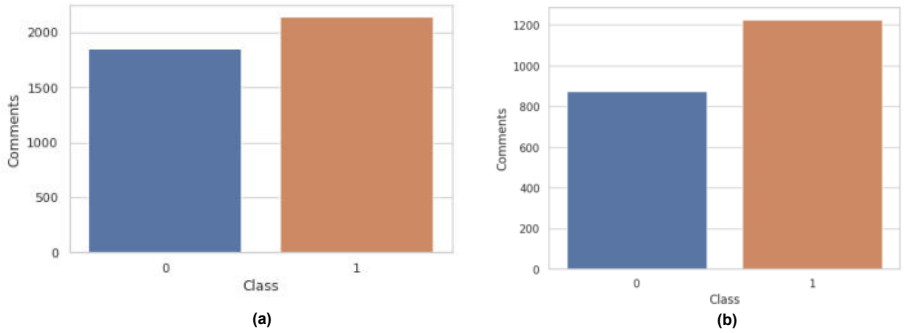

Fig. 3.4: Class distribution of NEIHS - a) Assamese and b) Bodo datasets where '0' is 'non-hate' and '1' is 'hate'.

Assam, and they did many unspeakable things and led the way to the end of Ahom rule and paved the way for the British.' '' . '' is a word used to describe illegal Muslim immigrants, kind of religious attack again. It is also used to mean criminally minded people. The sentence roughly translates to '99% of rapes and robberies are done by Gedas.'.

3.3. Dataset analysis. We summarize the key statistics of NEIHS in Table 3.1. For the Assamese dataset, 2,143 comments are hate out of 3,996. As a result, our data set is slightly skewed in favour of containing hate speech. Of 2,099 comments in our NEIHS - Bodo dataset, 1,225 contain hate speech. Figure 3.4 shows the details of class distribution. The length (number of words) of comments in each language is shown in figure 3.5. We split the dataset into a training set and a test set by 80:20. In NEIHS - Assamese, 1,705 comments are hate out of 3,196 in training data and 438 hate comments in test data out of 800. In the NEIHS - Bodo training set, 998 comments are hate out of 1,679 comments, and 227 are hate out of 420 in the test set.

4. Methods. Our input sample is x, consisting of m number of texts (each row) indicated as $x = \{r_1, r_2, r_3, .., r_i, ..., r_m\}$, where r_i is the i^{th} text or row, and m is equal to the total number of texts present in the set. Given a text or row r_i, the text has n sentences ($n <= 4$).

4.1. Preprocess. To improve performance, text data must be cleaned and noise-free before being used in the DL models. The languages spoken by those with few resources in India have several things in common. Researchers used identical preprocessing procedures for all datasets, even though they were written in various languages. Some tasks and datasets may call for slightly different text preparation processes. Raw comments

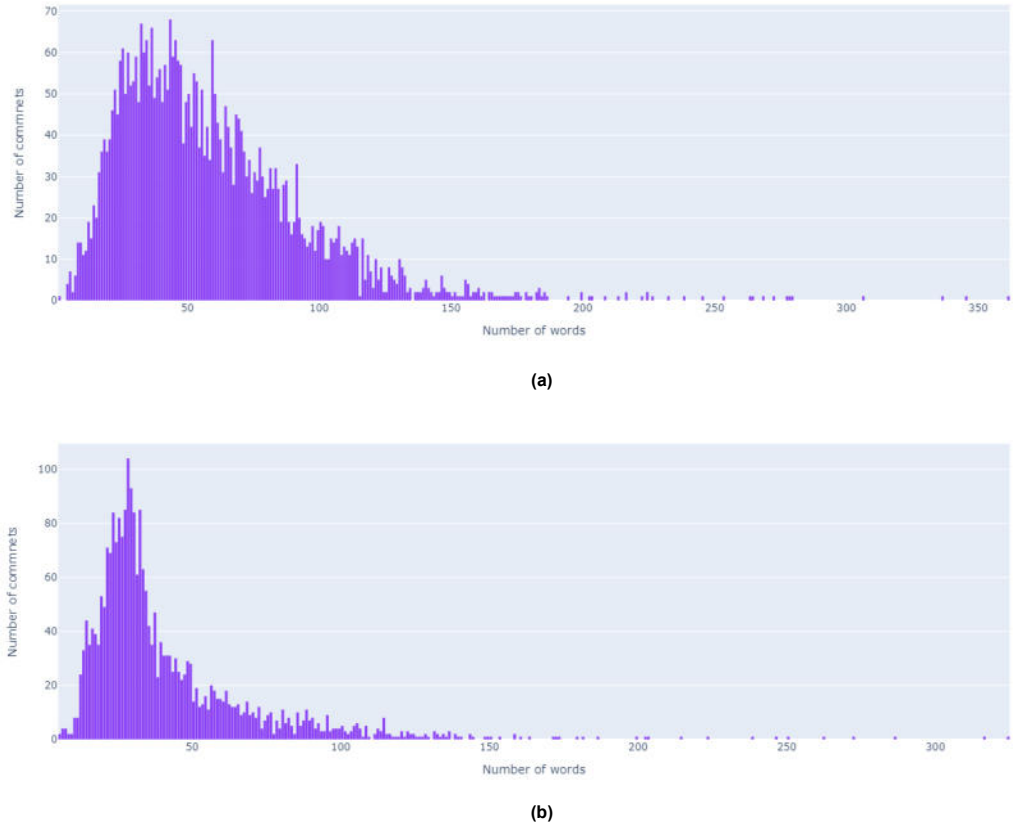

Fig. 3.5: The length (number of words) of comments in NEIHS a) Assamese and b) Bodo datasets.

Table 3.1: Class-wise distribution analysis for NEIHS dataset

Datasets	hate		non-hate		Total
	Train	Test	Train	Test	
NEIHS - Assamese	1,705	438	1,491	362	3,996
NEIHS - Bodo	998	227	681	193	2,099

with emojis, punctuation, and other undesired characters were present in some datasets. Commonly, the following procedures are used:

1. **Remove url, user**: We remove unwanted characters, url, URL occurrence with `xxurl`.

2. **Remove Punctuation**: Punctuation and numbers often don't add extra meaning to the text, hence being removed from the text.

3. **Remove stopwords**: Stopwords are the most commonly used words. Removed 15 Bodo stopwords from the NEIHS - Bodo dataset and 116 Assamese stopwords from the NEIHS - Assamese dataset.

4. **convert emojis to text**: We prepare an emoji dictionary to convert emojis to text. Only the Assamese dataset has emojis, and the Bodo dataset doesn't have emojis. So, examples are 😀 : '', 😂 : '

', 😊 : ''.

5. **Stemming**: Stemming is a technique that reduces any inflected word to its root form. Here, lightweight stemming is used, adapted from Sarmah et al. [38]. Some examples of Assamese stemming like, '', '', '' are the Assamese words, so we get the root words '', '', '' truncating '', '', ''. We didn't apply any stemming for the Bodo dataset, as any lightweight stemming does not exist.

6. **Tokenization**: We tokenize the sentences into words with the basic Keras' Tokenizer.

7. **Label encoding**: The class is tagged as 'non-hate' and 'hate' in both datasets. We encode them into a unique number. Like 'non-hate' to '0' and 'hate' to '1'.

After preprocessing we get $r_i = \{w_{i,1}, w_{i,2}, w_{i,3}, ..., w_{i,p}, .., w_{i,q}\}$, where $w_{i,p}$ denotes the p^{th} word in the i^{th} row and q is the number of words.

4.2. Machine Learning baseline models. Machine learning baselines such as NB, SVM are trained with TF-IDF-weighted character n-grams and word uni-grams.

4.3. Word embedding. A word's vector representation is necessary for any neural network model. Thomas Mikolov's Word2Vec models, described in detail at Google [39], are implemented to study word embeddings. Word2Vec's learned vector [40] allows for the inference of word relationships and similarities. Here, we use the Skip-gram model. To create a Word2Vec model, we used the gensim[12] module to train on the NEIHS dataset.

Now, the Word2Vec skip-gram model is trained using the training dataset used for this study. To train word embedding, we fit the parameters as embedding dimension = 300, window = 10, and saved the trained Word2Vec model for the next step. We embed each word $w_{i,p}$ to our pre-trained (trained on NEIHS dataset) word vector after loading the model into memory i.e. each word in the text is converted into a d-dimension embedding vector, where $w^v_{i,p} \in \mathbb{R}^d$ is d-dimension embedding vector of p^{th} word. The word level embedding as $r^v_i = \{w^v_{i,1}, w^v_{i,2}, w^v_{i,3}, ..., w^v_{i,p}, ..., w^v_{i,q}\}$ and $H = H_1, H_2, H_3, ..H_n$ is a hidden layer. We represent a text r_i with q words as a matrix $r_i \in \mathbb{R}^{d \times q}$.

4.4. Deep neural networks.

1. **LSTM**: It is an extension of Recurrent Neural Network (RNN) [41], capable of learning long dependencies. The LSTM neural networks contain three gates and a cell memory state. Here, $\{w^v_{i,1}, w^v_{i,2}, w^v_{i,3}, ..., w^v_{i,p}, ..., w^v_{i,q}\}$ denotes the word vector, q is the length of a text, $\{h_1, h_2, h_3, .., h_o, .., h_q\}$ represents the hidden vector. Now, we give r^v_i as a input to LSTM for feature extraction, namely F^{LSTM}_i, in equation

$$F^{LSTM}_i = LSTM(r^v_i) \tag{4.1}$$

   In the architecture $\{fh_1, fh_2, ..., fh_q\}$ is the forward hidden vector. The output layer is with the *sigmoid* activation function. Loss calculation is done by *binary_crossentropy*, and *adam* optimizer is used.

2. **BiLSTM**: Model works from both directions, where the equation will be

$$F^{BiLSTM}_i = BiLSTM(r^v_i) \tag{4.2}$$

   In Bidirectional LSTM, sequence data is processed in both directions with forward LSTM and backward LSTM layers, and these two hidden layers are connected to the same output layer. $\{fh_1, fh_2, ..., fh_q\}$ and $\{bh_1, bh_2, ..., bh_q\}$ represent the forward and the backward hidden vector respectively. $\{h_1, h_2, h_3, .., h_o, .., h_q\}$ represents final hidden layer. The final hidden vector h_o of the BiLSTM is shown as the following equation:

$$h_t = [fh_o, bh_o] \tag{4.3}$$

   Our model is enhanced with two BiLSTM layers. The fully connected layer with 256 neurons and *relu* activation is added. Because we have two classes, the output layer with *sigmoid* activation is introduced. Finally, use *binary_crossentropy* to compile the model.

[12]https://radimrehurek.com/gensim/models/word2vec.html (Access on 02.10.2023)

3. **CNN**: We transform a text r_i composed of q words into a matrix denoted as $r_i \in \mathbb{R}^{q \times d}$. Then, we perform a convolution operation on this matrix with a single stride. For each convolution operation, we utilize a filter denoted as $f_l \in \mathbb{R}^{a \times d}$, with a specific size of a, where d represents the dimension of the word vector. We apply 128 filters for $a \in \{3\}$, 256 filters for $a \in \{4\}$ and 512 filters for $a \in \{5\}$. $c(f_l, b) = relu(f_l \cdot b_{g:g+a-1})$ is the convolution function where f_l is the given filter, $b_{g:g+a-1}$ is the g^{th} vertical slice of the text matrix from g to $g : g + a - 1$ position. The result of this convolution is then passed through the *relu* activation function [42]. The function $c(f_l, b)$ generates a feature, denoted as c_g, which resembles n-grams for each slice g, resulting in a total of $q - a + 1$ features. To capture the most significant feature among these $q - a + 1$ features, we apply a max-pooling operation, i.e., $\hat{c}_l = max(c(f_l, b))$, which selects the maximum value. This max-pooling operation captures the most crucial aspect of each filter. In the proposed model, we have 896 filters (comprising 128+256+512), meaning we learn 896 of the most important features from the convolution layer. We pass these features to a *dense* layer with 256 perceptrons that use the *relu* activation function. Another *dense* layer with one perceptron is applied at the end with the *sigmoid* activation function.

4.5. Transfer learning. One challenge of the machine learning research known as 'transfer learning' is to find ways to generalize the insights obtained from addressing one problem to a new one that is conceptually similar. These two languages are so rare that no pre-trained transformer model is available. Google developed BERT, a transformer-based technique for NLP. BERT can generate contextualized embeddings. We insert [CLS] in the beginning and [SEP] between sentences or at the end. The sentence will then be as $r_i = \{[CLS]s_{i,1}[SEP]s_{i,2}[SEP]s_{i,3}....s_{i,j}[SEP]s_{i,j+1}....s_{i,n}[SEP]\}$, where $s_{i,j}$ is the j^{th}sentence of i^{th} row. Each sentence with p words. The text will need to be tokenized now. Tokenizing a text yields a dictionary containing the input ids, numerical representations of the tokens, and the attention mask, indicating whether the token is a [PAD]. $s_{i,j} = \{[CLS], t_{i,j,1}, t_{i,j,2}, ..., t_{i,j,k}, ..., t_{i,j,p}, [SEP]\}$, where $t_{i,j,k}$ is the k^{th} token in the j^{th} sentence of i^{th} row. The entire input looks like $r_i = \{[CLS], t_{i,1,1}, t_{i,1,2}, ..., t_{i,1,k}, ..., t_{i,1,r}, [SEP], t_{i,2,1}, t_{i,2,1}...., t_{i,2,s}, [SEP]\}$, where $t_{i,j,k}$ is k^{th} token of j^{th} sentence of i^{th} row. r and s are the number of tokens in sentences. Words and tokens should not be the same number as sentencepiece/ wordpiece tokenizers perform build-in stemming. Here, as transfer learning, we use pre-trained transformer-based BERT models.

1. **MuRILBERT**[13]: Multilingual Representations for Indian Languages (MuRIL) is a BERT model pre-trained on 17 Indian languages and their transliterated counterparts, i.e. monolingual segments, and parallel segments.
2. **mBERT (cased/ uncased)**[14][15]: It is pre-trained with the largest Wikipedia over 104 top languages worldwide, including Hindi, Bengali, and Marathi, using a masked language modelling (MLM) objective. Though Bodo and Assamese languages don't have capital letter issues, we applied both to observe the performance.
3. **RoBERTaHindi**[16]: This is a transformers model pre-trained on a large corpus of Hindi data (a combination of mc4, oscar and indic-nlp datasets).
4. **MahaRoBERTaMarathi**[17]: A Multilingual RoBERTa (xlm-roberta-base) model fine-tuned on publicly available Marathi monolingual datasets and L3Cube-MahaCorpus.
5. **BanglaBERT**[18]: Using mask language modeling, bangla-Bert-Base was pre-trained on data downloaded from OSCAR and Bengali Wikipedia Dump Dataset.

4.6. Experiments. Table 4.1 shows the preprocessing steps used in both languages before employing the data in the models. We kept 80% of the dataset as a train set and 20% as a test for both datasets. The architectural parameters used to train LSTM, BiLSTM, and CNN models are listed in Table 4.2, respectively. Due to memory and *GPU* issues, we did limited experiments with hyperparameters like batch size, epoch, learning rate, etc. In the first experiment, we use a TF-IDF vectorizer to transform words into features, and then we train the Multinomial NB model. We used the linear kernel and kept all other parameters at their default value. In the case of LSTM/BiLSTM/ CNN without Word2Vec, all the preprocessing steps are done

[13]https://huggingface.co/google/muril-base-cased (Access on 02.10.2023)
[14]https://huggingface.co/bert-base-multilingual-cased (Access on 02.10.2023)
[15]https://huggingface.co/bert-base-multilingual-uncased (Access on 02.10.2023)
[16]https://huggingface.co/flax-community/roberta-hindi (Access on 02.10.2023)
[17]https://huggingface.co/l3cube-pune/marathi-roberta (Access on 02.10.2023)
[18]https://huggingface.co/sagorsarker/bangla-bert-base (Access on 02.10.2023)

Table 4.1: Experiment wise preprocessing steps on NEIHS dataset

Preprocessings	Assamese		Bodo	
	Others	BERT	Others	BERT
Remove url, user	✓	✓	✓	✓
Remove Punctuation	✓	×	✓	×
Remove stopwords	✓	×	✓	×
Convert emojis to text	✓	✓	×	×
Stemming	✓	×	×	×
Tokenization	✓	✓	✓	✓
Label encoding	✓	✓	✓	✓

before employing the input to the first layer for the model; for embedding, the Keras embedding layer is applied. We set num_words to 5,142 for the Bodo dataset and 9,629 for the Assamese dataset. In transfer learning, we choose the advantage of MuRILBERT, which includes 17 Indian languages along with Assamese, and mBERT cased/uncased, which includes 104 top languages, BanglaBERT for only NEIHS-Assamese and RoBERTaHindi, MahaRoBERTaMarathi for the NEIHS-Bodo dataset, as Bodo shares Devnagari script and Assamese shares Assamese script likely Bengali-Assamese script. Besides this, we are using one pre-trained model, which is trained on the monolingual Hindi and Marathi language used for Bodo, and one Bangla pre-trained monolingual model for Assamese. However, we only performed lightweight preprocessing like removing URL, user name, emoji, and for the tokenization part, a sentencepiece/ wordpiece tokenizer is used, as previous studies have demonstrated that BERT-based models achieve higher classification accuracy on unclean texts. We use MuRILBERT, mBERT-uncased/cased for both NEIHS languages. Specifically, for transfer learning, we use $batchsize = 8$, $epochs = 20$, $learning\ rate = 2e-5$, and $epsilon = 1e-8$.

Below are all types of model combinations we tested on our dataset.

1. NB and SVM baseline evaluation
2. LSTM without Word2Vec
3. LSTM with Word2Vec
4. BiLSTM without Word2Vec
5. BiLSTM with Word2Vec
6. CNN without Word2Vec
7. CNN with Word2Vec
8. Transfer learning

5. Result. In this section, we discuss the precision, recall, weighted F1 score and accuracy obtained by training all the ML, DL, and transformer-based models on Assamese and Bodo datasets. Table 4.2 represents the results of all models trained on the datasets; we intentionally prefer a weighted F1 score over an accuracy score to evaluate the models because imbalanced class distribution exists in most classification problems. To evaluate our models, we use two class precisions ($P_{non-hate}$, P_{hate}), recalls ($R_{non-hate}$, R_{hate}), F1 scores ($F1_{non-hate}$, $F1_{hate}$) then calculate precision (W_P), recall (W_R), and F1 score (W_{F1}) here. At last, we calculate *Accuracy*.

$$P_{non-hate} = \frac{True_{non-hate}}{True_{non-hate} + False_{hate}} \tag{5.1}$$

$$P_{hate} = \frac{True_{hate}}{True_{hate} + False_{hate}} \tag{5.2}$$

$$R_{non-hate} = \frac{True_{non-hate}}{True_{non-hate} + False_{non-hate}} \tag{5.3}$$

$$R_{hate} = \frac{True_{hate}}{True_{hate} + False_{non-hate}} \tag{5.4}$$

$$F1_{non-hate} = 2 * \frac{P_{non-hate} * R_{non-hate}}{P_{non-hate} + R_{non-hate}} \tag{5.5}$$

$$F1_{hate} = 2 * \frac{P_{hate} * R_{hate}}{P_{hate} + R_{hate}} \tag{5.6}$$

Table 4.2: Combination of parameters for LSTM, BiLSTM and CNN models

Model name	Parameter name	Parameter value
LSTM	Epochs	20
	Batch size	8
	Embedding dimension	200, 300
	LSTM layer 1	128, *relu* activation
	LSTM layer 2	128, *relu* activation
	Dropout	0.1
	Dense layer 1	32, *relu* activation
	Dropout	0.1
	Dense layer	1, *sigmoid* activation
	Learning-rate	0.001
BiLSTM	Epochs	20
	Batch size	8
	Embedding dimension	200, 300
	BiLSTM layer 1	64
	BiLSTM layer 1	64
	Dense layer 1	256, *relu* activation
	Dense layer 1	1, *sigmoid* activation
	Learning-rate	0.001
CNN	Epochs	20
	Batch size	8
	Embedding dimension	200, 300
	Conv1D layer 1	128 filters, 3 kernel_size, *relu* activation, 1 stride
	Global MaxPooling1D 1	-
	Conv1D layer 2	256 filters, 4 kernel_size, *relu* activation, 1 stride
	Global MaxPooling1D 2	-
	Conv1D layer 3	512 filters, 5 kernel_size, *relu* activation, 1 stride
	Global MaxPooling1D 3	-
	Concatenate	Global MaxPooling1D 1, Global MaxPooling1D 2, Global MaxPooling1D 3
	Dense layer 1	256, *relu* activation
	Dense layer 2	1, *sigmoid* activation
	Learning-rate	0.001

$$W_P = \frac{P_{non-hate} * T_{non-hate} + P_{hate} * T_{hate}}{T_{non-hate} + T_{hate}} \tag{5.7}$$

$$W_R = \frac{R_{non-hate} * T_{non-hate} + R_{hate} * T_{hate}}{T_{non-hate} + T_{hate}} \tag{5.8}$$

$$W_{F1} = \frac{F1_{non-hate} * T_{non-hate} + F1_{hate} * T_{hate}}{T_{non-hate} + T_{hate}} \tag{5.9}$$

$$Accuracy = \frac{True_{non-hate} + True_{hate}}{T_{non-hate} + T_{hate}} \tag{5.10}$$

where $True_{non-hate}$ = True-negative (model predicted the texts as non-hate, and the actual value of the same is also non-hate), $True_{hate}$ = True-positive (model predicted the texts as hate, and the actual value of the same is also hate), $False_{non-hate}$ = False-negative (model predicted the texts as non-hate, but the true value of the same is hate), $False_{hate}$ = False-positive (model predicted the texts as hate, but the true value of the same is non-hate), $P_{non-hate}$ = Precision of non-hate class, P_{hate} = Precision of hate class, $R_{non-hate}$ = Recall of non-hate class, R_{hate} = Recall of hate class, $F1_{non-hate}$ = F1 score of non-hate class, $F1_{hate}$ = F1 score of hate class, $T_{non-hate}$ = The total number of non-hate class text present in the test set, T_{hate} = The total number of hate class text present in the test set.

5.1. Analysis of hate speech detection. We evaluated two ML models, three DLs with or without Word2Vec, five transformer-based models on the NEIHS - Bodo dataset, and four variants on the NEIHS - Assamese dataset. mBERT-cased, mBERT-cased, and MahaRoBERTaMarathi are the best performing and

Table 5.1: Precision, Recall, F1 score, and Accuracy of all the models applied on NEIHS-Assamese and Bodo dataset

	Precision			Recall			F1 score			Accuracy
(NEIHS-Bodo)	**non-hate**	**hate**	**weighted**	**non-hate**	**hate**	**weighted**	**non-hate**	**hate**	**weighted**	
NB	0.79	0.64	0.71	0.41	0.91	0.68	0.54	0.75	0.65	0.68
SVM	0.43	0.54	0.49	0.07	**0.93**	0.53	0.12	0.68	0.42	0.53
LSTM with Word2Vec	0.77	0.84	0.80	0.83	0.78	0.80	0.80	0.81	0.81	0.80
LSTM without Word2Vec	0.78	0.85	0.82	**0.84**	0.79	0.81	0.81	0.82	0.81	0.81
BiLSTM with Word2Vec	0.82	0.83	0.83	0.80	0.85	0.84	0.81	0.84	0.83	0.83
BiLSTM without Word2Vec	0.81	0.82	0.82	0.79	0.84	0.82	0.80	0.83	0.82	0.82
CNN with Word2Vec	0.82	0.84	0.83	0.82	0.84	0.83	0.82	0.84	0.83	0.83
CNN without Word2Vec	0.80	0.83	0.81	0.80	0.83	0.81	0.80	0.83	0.81	0.81
mBERT-cased	**0.86**	0.84	**0.85**	0.80	0.89	**0.85**	**0.83**	**0.87**	**0.85**	**0.85**
mBERT-uncased	0.85	0.85	**0.85**	0.82	0.88	**0.85**	**0.83**	0.86	**0.85**	**0.85**
MuRILBERT	0.80	**0.86**	0.83	**0.84**	0.82	0.83	0.82	0.84	0.83	0.83
RoBERTaHindi	0.82	0.84	0.83	0.81	0.85	0.83	0.81	0.84	0.83	0.83
MahaRoBERTaMarathi	**0.86**	0.83	**0.85**	0.79	0.89	**0.85**	0.82	0.86	0.84	**0.85**
(NEIHS-Assamese)										
NB	0.60	0.63	0.61	0.47	0.74	0.61	0.52	0.68	0.61	0.61
SVM	0.50	0.55	0.53	0.00	**1.00**	0.55	0.01	**0.71**	0.39	0.55
LSTM with Word2Vec	0.65	0.69	0.67	0.60	0.73	0.67	0.62	**0.71**	0.67	0.67
LSTM without Word2Vec	0.61	0.70	0.66	0.66	0.66	0.66	0.64	0.68	0.66	0.66
BiLSTM with Word2Vec	0.61	0.69	0.66	0.65	0.66	0.65	0.63	0.67	0.65	0.65
BiLSTM without Word2Vec	0.64	0.70	0.67	0.63	0.71	0.67	0.63	0.70	0.67	0.67
CNN with Word2Vec	0.64	0.69	0.67	0.60	0.72	0.67	0.62	**0.71**	0.67	0.66
CNN without Word2Vec	0.62	0.73	0.68	0.72	0.63	0.67	0.66	0.68	0.67	0.67
mBERT-cased	0.63	**0.76**	0.70	**0.76**	0.63	**0.69**	**0.69**	0.69	**0.69**	**0.69**
mBERT-uncased	0.64	0.73	0.69	0.69	0.68	0.68	0.66	0.70	0.68	0.68
MuRILBERT	**0.70**	0.73	**0.72**	0.63	0.66	0.65	0.67	0.70	**0.69**	**0.69**
BanglaBERT	0.61	0.73	0.67	0.71	0.63	0.67	0.66	0.67	0.67	0.67

best-fitted models on the Bodo dataset, giving the highest weighted F1-score of 85%, which is about 1% to 5% better than other transformer models. At the same time, BiLSTM with Word2Vec, CNN with Word2V, MuRILBERT, and RoBERTaHindi also performed moderately well, i.e., weighted F1-score of 83%. LSTM with without Word2Vec performs only 1% less. We notice that DL and transformer-based models perform almost the same. Both of the ML results are very poor compared to other models. mBERT-cased and MuRILBERT give the best result on the Assamese dataset, giving the highest weighted F1-score of 69%, which is about 1% to 8% better than other transformer models. mBERT-uncased achieved 68% weighted F1-score. At the same time, CNN without Word2Vec, LSTM with Word2Vec, BiLSTM without Word2Vec, and BanglaBERT scored 67%. CNN with Word2Vec, LSTM without Word2Vec and BiLSTM with Word2Vec also performed moderately well. NB and SVM perform very poorly. Figure 5.1 shows the confusion matrix of the best models on two NEIHS datasets separately. As best models predict almost the same, we provide only one confusion matrix for each dataset.

5.2. Error analysis. The results show that the model's performance is not up to the mark in the Assamese dataset but good in the Bodo dataset. However, as a pioneering work in the Assamese and Bodo, it needs an error analysis to identify the system's weakness for further improvement. Here, the error analysis is not performed at a granular level; rather, we have tried to identify the major source of error.

Errors in the preprocessing. A DL model performs very well on the Bodo dataset, but we need to experiment with this model by removing stopwords. We found only 15 stop words from native speakers (Students) but need to gather more if any others are left. Stop words are the most commonly used words. Commonly used words don't mean frequent words in the datasets, as both concepts are slightly different and depend on a particular dataset. Wordpiece tokenizer is checked on the Assamese dataset but found shocking results, and resultant tokens are not meaningful. This sentence ‘’ (English translation: He/ She looks like a Jihadi.) is giving accurate result like [‘’, ‘’, ‘’, ‘’, ‘’, ‘’] but at the same time ‘’ (English translation: They think life is theirs only) is giving [‘’, ‘##’, ‘##’, ‘##’, ‘[UNK]’, ‘’, ‘##’, ‘’, ‘##’, ‘[UNK]’, ‘’, ‘[UNK]’, ‘’], which is an error. This can be one reason for the worst result despite having more Assamese data than Bodo data. So,

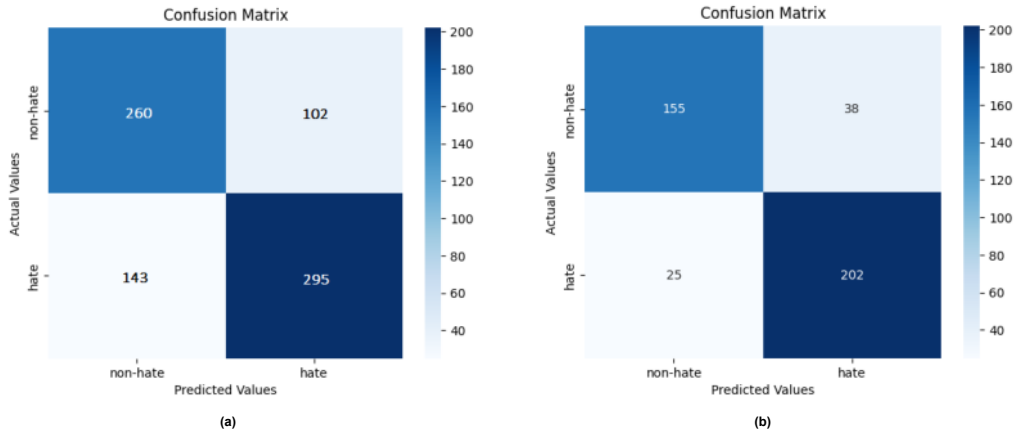

Fig. 5.1: Confusion matrix of the best models on two NEIHS datasets separately, (a) NEIHS-Assamese (best model: MuRILBERT) and (b) NEIHS-Bodo (best model: mBERT-cased)

tokenizer training for Assamese data is required. For both languages, an emoji-to-text Python library would help researchers. We noticed in other languages that converting emojis to text gives 2%-4% better results than the complete removal of emojis from the text. We need a proper lightweight stemming list if the lemmatization of words is complex or expensive. It could give more accurate results while training with DL.

Errors in the model. We perform fine-tuning on transformer-based models that have been pre-trained on either multilingual or monolingual datasets. However, these pre-trained models do not currently incorporate Bodo data. At the same time, Google Muril's dataset includes Assamese among its 17 languages. Due to insufficient raw datasets, we rely on word embedding for the same dataset rather than training a neural word embedding model from scratch.

Errors in the data set. In some instances, confusion surrounding annotation can adversely affect performance. To ensure optimal performance, annotators may need more contextual information. Moreover, more data is required.

Errors in the language phenomenon. Some inherent language problems are difficult to address at the computation level. In the context of hate speech, sometimes the text does not contain any hate-related words but still conveys a piece of hate information. On the other hand, though the text contains hate words, the text is not a hateful sentence. Sometimes, it needed the word sense disambiguation to identify the hateful information.

Errors in prediction. For the NEIHS-Assamese, mBERT-cased and MuRILBERT is the best model based on our analysis. We divide the error cases into the following categories[19]:

1. Implicit hate: Situations where there are no openly abusive words but express a complex thought. ‘’ (English translation: Come, we Assamese people should come together to beat up miyas.). There are no swear words here, but the message is meant to inspire intolerance against a particular religion.
2. Annotator confusion: Annotators get confused about whether the sentence is complement or slang. ‘ ’ (English translation: Your figure is sexy.), this comment is ambiguous. Here, the model prediction cannot be considered incorrect as the comment could be analysed in both ways, relying on cultural perceptiveness.
3. Contextual abuse: Some words are used more often in the abuse samples. Depending on the context, it changes the meaning. ‘’ (English translation: Godi media is buying all the news channels. They ended the freedom of speech in this country.). Here, ‘’ (Godi media) indicates media is supporting a particular person's opinion.

[19]https://openreview.net/forum?id=HCnb1TByvx7 (Access on 02.10.2023)

4. Spelling mistakes: It's common for profanity to have an incorrect spelling on social media, whether on purpose to get through the moderation system or accidentally owing to the relaxed nature of the medium. '' (English translation: We will throw you miya's out of assam.). '' is actually misspelled of ''.

6. Conclusion and future work. This paper describes our contribution, i.e., the NEIHS dataset, which includes Assamese and Bodo datasets, a binary classification task. We performed training NB, SVM, LSTM, BILSTM, and CNN on the NEIHS datasets. Transfer learning is also employed for the task. The best-performing model we found for NEIHS-Assamese is MuRILBERT and mBERT-cased, with a 69% weighted F1-score, where mBERT-cased and mBERT-uncased got 86% on NEIHS-Bodo dataset. Transformer-based models perform sufficiently but are burdened by longer computation times than traditional DL models, where DL models perform a little less in this aspect. As mentioned earlier, due to the lack of Bodo and Assamese tools in the NLP field, we failed to perform experiments like stemming and emoji to text in the Bodo language. In the case of transfer learning, we cannot use a monolingual transformer-based model. Nevertheless, the shortage of extensive datasets poses a significant obstacle to advancing automatic hate speech detection systems for Indic languages, particularly for North East Indian languages. To address this issue, we have contributed the human-annotated NEIHS dataset, which covers multiple languages from a popular social media platform. However, we intend to expand this dataset by collecting additional data with diverse categorical annotations, focusing on multiclass classification. In the future, we aim to incorporate more data from the same languages and other North-Eastern languages such as Meiteilon (spoken by 2 million native speakers), Manipuri (spoken by 1.8 million native speakers), and Mizo (spoken by 830,846 native speakers). Additionally, recognizing the complexity of code-mixed comments, where Indic languages are expressed using Roman characters, we also plan to enhance the NEIHS dataset by including such instances.

7. Acknowledgement. We thank Debarshi Sonowal, Abhilash Basumatary, and Bidisha Gogoi for their help in collecting and tagging the Assamese hate dataset.

REFERENCES

[1] G. W. Giumetti and R. M. Kowalski, *Cyberbullying via social media and well-being, Current Opinion in Psychology*, Volume 45, 2022, 101314, ISSN 2352-250X, https://doi.org/10.1016/j.copsyc.2022.101314.
[2] P. Sarkar, *Media And Politics. Pratishthaa Publishing House*, 2019, https://books.google.co.in/books?id=8HSPDwAAQBAJ.
[3] Z. Laub, *Hate speech on social media: Global comparisons. Council on foreign relations 7* (2019).
[4] M. Alhajji, S. Bass, and Dai, T, *Cyberbullying, Mental Health, and Violence in Adolescents and Associations With Sex and Race: Data From the 2015 Youth Risk Behavior Survey. Global Pediatric Health*, 2019, https://doi.org/10.1177/2333794X19868887.
[5] A. Guterres et al, *United Nations strategy and plan of action on hate speech.* Taken from: https://www. un. org/en/genocideprevention/documents/U 20Strategy (2019).
[6] R. Ramdev, S.D. Nambiar, and D. Bhattacharya, *Sentiment, Politics, Censorship: The State of Hurt. SAGE Publications*, 2015, https://books.google.co.in/books?id=i4GTCwAAQBAJ.
[7] T. Mandl, S. Modha, P. Majumder, D. Patel, M. Dave, C. Mandlia, and A. Patel, *Overview of the HASOC track at FIRE 2019: Hate Speech and Offensive Content Identification in Indo-European Languages. In Proceedings of the 11th Annual Meeting of the Forum for Information Retrieval Evaluation (FIRE '19). Association for Computing Machinery, New York, NY, USA*, 14–17. https://doi.org/10.1145/3368567.3368584.
[8] T. Mandl, S. Modha, A. K. Jaiswal, and B. R. Chakravarthi, *Overview of the HASOC Track at FIRE 2020: Hate Speech and Offensive Language Identification in Tamil, Malayalam, Hindi, English and German. In Forum for Information Retrieval Evaluation (Hyderabad, India) (FIRE 2020). Association for Computing Machinery, New York, NY, USA*, 29–32. https://doi.org/10.1145/3441501.3441517.
[9] S. Modha, T. Mandl, G. K. Shahi, H. Madhu, S. Satapara, T. Ranasinghe, and M. Zampieri, *Overview of the HASOC Subtrack at FIRE 2021: Hate Speech and Offensive Content Identification in English and Indo-Aryan Languages and Conversational Hate Speech. In Proceedings of the 13th Annual Meeting of the Forum for Information Retrieval Evaluation (FIRE '21). Association for Computing Machinery, New York, NY, USA*, 1–3, 2022. https://doi.org/10.1145/3503162.3503176.
[10] R. Kalra and A. K. Dutt, *Exploring Linguistic Diversity in India: A Spatial Analysis. Handbook of the Changing World Language Map* (2019).
[11] M. Narzary, G. Muchahary, M. Brahma, S. Narzary, P. K. Singh, and A. Senapati, *Bodo Resources for NLP-An Overview of Existing Primary Resources for Bodo. AIJR Proceedings* (2021), 96–101.
[12] S. Narzary, M. Brahma, M. Narzary, G. Muchahary, P. Kumar Singh, A. Senapati, S. Nandi, and B. Som, *Generating Monolingual Dataset for Low Resource Language Bodo from old books using Google Keep. In Proceedings of the Thirteenth*

Language Resources and Evaluation Conference. European Language Resources Association, Marseille, France, 6563–6570, 2022. https://aclanthology.org/2022.lrec-1.705.

[13] M. ElSherief, V. Kulkarni, D. Nguyen, W. Y. Wang, and E. Belding, *Hate Lingo: A Target-Based Linguistic Analysis of Hate Speech in Social Media. Proceedings of the International AAAI Conference on Web and Social Media 12, 1 (Jun. 2018)*. https://doi.org/10.1609/icwsm.v12i1.15041.

[14] R. Izsák, *Hate speech and incitement to hatred against minorities in the media. UN Humans Rights Council. Technical Report*, 2015, A/HRC/28/64.

[15] J. Salminen, H. Almerekhi, M. Milenković, S. G. Jung, J. An, H. Kwak, and B. J. Jansen, *Anatomy of online hate: developing a taxonomy and machine learning models for identifying and classifying hate in online news media. In Twelfth international AAAI conference on web and social media.*

[16] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, *BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding*. CoRR abs/1810.04805 (2018). arXiv:1810.04805 http://arxiv.org/abs/1810.04805.

[17] Y. Liu, M. Ott, N. Goyal, J. Du, M. Joshi, D. Chen, O. Levy, M. Lewis, L. Zettlemoyer, and V. Stoyanov, *RoBERTa: A Robustly Optimized BERT Pretraining Approach*. CoRR abs/1907.11692 (2019). arXiv:1907.11692 http://arxiv.org/abs/1907.11692.

[18] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, *ALBERT: A Lite BERT for Self-supervised Learning of Language Representations*. CoRR abs/1909.11942 (2019). arXiv:1909.11942 http://arxiv.org/abs/1909.11942.

[19] V. Sanh, L. Debut, J. Chaumond, and T. Wolf, *DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter*. CoRR abs/1910.01108 (2019). arXiv:1910.01108 http://arxiv.org/abs/1910.01108.

[20] S. Khanuja, D. Bansal, S. Mehtani, S. Khosla, A. Dey, B. Gopalan, D. K. Margam, P. Aggarwal, R. T. Nagipogu, S. Dave, S. Gupta, S. C. B. Gali, V. Subramanian, and P. P. Talukdar, *MuRIL: Multilingual Representations for Indian Languages*. CoRR abs/2103.10730 (2021). arXiv:2103.10730 https://arxiv.org/abs/2103.10730.

[21] K. Jain, A. Deshpande, K. Shridhar, F. Laumann, and A. Dash, *Indic-Transformers: An Analysis of Transformer Language Models for Indian Languages*, 2020, arXiv:2011.02323 [cs.CL].

[22] D. Kakwani, A. Kunchukuttan, S. G. Gokul N.C., A. Bhattacharyya, M. M. Khapra, and P. Kumar, *IndicNLPSuite: Monolingual Corpora, Evaluation Benchmarks and Pre-trained Multilingual Language Models for Indian Languages. In Findings of the Association for Computational Linguistics: EMNLP 2020, pages 4948–4961, Online. Association for Computational Linguistics. Online*, 4948–4961. https://doi.org/10.18653/v1/2020.findings-emnlp.445.

[23] R. Joshi, *L3Cube-MahaCorpus and MahaBERT: Marathi Monolingual Corpus, Marathi BERT Language Models, and Resources*. CoRR abs/2202.01159 (2022). arXiv:2202.01159 https://arxiv.org/abs/2202.01159.

[24] R. Joshi, *L3Cube-MahaCorpus and MahaBERT: Marathi Monolingual Corpus, Marathi BERT Language Models, and Resources. In Proceedings of The WILDRE-6 Workshop within the 13th Language Resources and Evaluation Conference. European Language Resources Association, Marseille, France*, 97–101, 2022.

[25] A. Conneau, K. Khandelwal, N. Goyal, V. Chaudhary, G. Wenzek, F. Guzmán, E. Grave, M. Ott, L. Zettlemoyer, and V. Stoyanov, *Unsupervised Cross-lingual Representation Learning at Scale*. CoRR abs/1911.02116 (2019). arXiv:1911.02116 http://arxiv.org/abs/1911.02116

[26] S. Sarker, *BanglaBERT: Bengali Mask Language Model for Bengali Language Understading*, 2020, https://github.com/sagorbrur/bangla-bert

[27] A. Sharma, A. Kabra, and M. Jain, *Ceasing hate with MoH: Hate Speech Detection in Hindi–English code-switched language. Information Processing Management* 59, 1 (2022), 102760. https://doi.org/10.1016/j.ipm.2021.102760.

[28] B. R. Chakravarthi, R. Priyadharshini, R. Ponnusamy, P. K. Kumaresan, K. Sampath, D. Thenmozhi, S. Thangasamy, R. Nallathambi, and J. P. McCrae, *Dataset for Identification of Homophobia and Transophobia in Multilingual YouTube Comments*. CoRR abs/2109.00227 (2021). arXiv:2109.00227 https://arxiv.org/abs/2109.00227.

[29] Md. A. Bashar and R. Nayak, *QutNocturnal@HASOC'19: CNN for Hate Speech and Offensive Content Identification in Hindi Language. CoRR abs/2008.12448 (2020)*. arXiv:2008.12448 https://arxiv.org/abs/2008.12448.

[30] J. C. Mensonides, P. A. Jean, A. Tchechmedjiev, and S. Harispe, *IMT mines ales at HASOC 2019: automatic hate speech detection. In FIRE 2019-11th Forum for Information Retrieval Evaluation, Vol.* 2517, p–279, 2019.

[31] S. Mishra and S. Mishra,*3Idiots at HASOC 2019: Fine-tuning Transformer Neural Networks for Hate Speech Identification in Indo-European Languages. In FIRE (Working Notes)*. 208–213, 2019.

[32] V. Mujadia, P. Mishra, and D. M. Sharma, *IIIT-Hyderabad at HASOC 2019: Hate Speech Detection.. In FIRE (Working Notes)*, 271–278, 2019.

[33] Y. Freund and R. E. Schapire, *A decision-theoretic generalization of on-line learning and an application to boosting. Journal of computer and system sciences* 55, 1 (1997), 119–139.

[34] R. Raj, S. Srivastava, and S. Saumya. *"NSIT & IIITDWD@ HASOC 2020: Deep learning model for hate-speech identification in Indo-European languages." In FIRE (Working Notes), pp.* 161-167. 2020.

[35] M. Bhatia, T. S. Bhotia, A. Agarwal, P. Ramesh, S. Gupta, K. Shridhar, F. Laumann, and A. Dash, *One to rule them all: Towards Joint Indic Language Hate Speech Detection*. CoRR abs/2109.13711 (2021). arXiv:2109.13711 https://arxiv.org/abs/2109.13711.

[36] M. Bhardwaj, Md S. Akhtar, A. Ekbal, A. Das, and T. Chakraborty, *Hostility detection dataset in Hindi*. arXiv preprint arXiv:2011.03588 (2020).

[37] K. Ghosh and Dr. A. Senapati, *Hate speech detection: a comparison of mono and multilingual transformer model with cross-language evaluation. In Proceedings of the 36th Pacific Asia Conference on Language, Information and Computation, pages 853–865, 2022, Manila, Philippines. De La Salle University*.https://www.paclic2022.net/papers.html.

[38] J. Sarmah, S. K. Sarma, and A. K. Barman, *Development of Assamese Rule based Stemmer using WordNet. In Proceedings of the 10th Global Wordnet Conference. Global Wordnet Association, Wroclaw, Poland*, 135–139, 2019.

https://aclanthology.org/2019.gwc-1.17.

[39] T. Mikolov, I. Sutskever, K. Chen, G. S. Corrado, and J. Dean, *Distributed Representations of Words and Phrases and their Compositionality. In Advances in Neural Information Processing Systems, C.J. Burges, L. Bottou, M. Welling, Z. Ghahramani, and K.Q. Weinberger (Eds.), Vol. 26. Curran Associates*, Inc.https://proceedings.neurips.cc/paper/2013/file/9aa42b31882ec039965f3c4923ce901b-Paper.pdf.

[40] A. Khatua, and E. Cambria,*A tale of two epidemics: Contextual Word2Vec for classifying twitter streams during outbreaks. Information Processing Management*, 56, 1 (2019), 247–257. https://doi.org/10.1016/j.ipm.2018.10.010

[41] S. Hochreiter, and J. Schmidhuber; *Long Short-Term Memory. Neural Computation* 1997; 9 (8): 1735–1780. doi: https://doi.org/10.1162/neco.1997.9.8.1735.

[42] V. Nair, and G.E. Hinton; *Rectified linear units improve restricted boltzmann machines. In: Proceedings of the 27th international conference on machine learning (ICML-10). pp.* 807–814 (2010)

Edited by: Rajni Mohana
Special issue on: Sentiment Analysis and Affective computing in Multimedia Data on Social Network
Received: Jul 31, 2023
Accepted: Oct 5, 2023