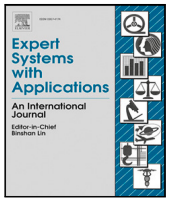
Expert Systems With Applications 236 (2024) 121228

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa



EnsMulHateCyb: Multilingual hate speech and cyberbully detection in online social media

Esshaan Mahajan [a], Hemaank Mahajan [b], Sanjay Kumar [c,*]

[a] University School of Information, Communication and Technology, Guru Gobind Singh Indraprastha University, Dwarka, New Delhi, India
[b] Department of Applied Mathematics, Delhi Technological University, Main Bawana Road, New Delhi 110042, India
[c] Department of Computer Science and Engineering, Delhi Technological University, Main Bawana Road, New Delhi 110042, India

ARTICLE INFO

Keywords:
Hate speech detection
Cyberbully detection
Ensemble deep learning
BiLSTM
Bi-GRU
Multilingual data streams

ABSTRACT

Nowadays, users across the globe interact with one another for information exchange, communication, and association on various online social media. However, some individuals exploit these venues for malicious practices like hate speech and cyberbully. In this paper, we present an improved multilingual hate speech and cyberbully detection model using bagging-stacking based hybrid ensemble deep learning techniques. The proposed model utilizes Bi-directional Long Short-Term Memory (BiLSTM), Bi-directional Gated Recurrent Unit (Bi-GRU), Convolutional Neural Network (CNN), and Long Short-Term Memory (LSTM) techniques to enhance the overall performance. We first preprocess the multilingual data streams followed by adoption of Global vectors for word Representation (GloVe) embeddings to convert words to a vector representation in parallel enabling the data streams for binary classification task. In order to construct an architecture for the detection of hate speech and cyberbully, we introduce a heterogeneous fusion of multiple effective models in a unique approach such that CNN-LSTM utilizes a stacking approach with stochastic gradient descent to achieve optimal weights, whereas all the base learners used bagging ensemble approach with cross-validation to reach optimal weights. The final output layer of the proposed ensemble deep learning architecture is achieved using a super learner approach on base learners. To show the efficacy of the proposed model, we conduct the simulation on a total of nine real-world social media datasets in different languages and compared the results with other contemporary hate speech and cyberbully detection methods. The collected findings show that the proposed model outperforms other models on considered datasets and shows an improvement of at least 4.44% in F1 scores.

1. Introduction

Online social media in today's world is becoming an inseparable component of how we communicate with others and express ourselves. It provides a network platform where massive information exchange occurs every day. One might use online social media to organize events, groups, and activities to showcase social issues and opinions and make a broad audience aware of them. It can be used efficiently to build a business and market it rapidly. The applications of enhanced communication have far-reaching effects which can prove to be helpful and of great value. Yet there is a chance that such unchecked information can be harmful. This information can be anything from misinformation creating panic, fake news with political agenda, or hate speech with personal invective. Social media is one of the primary sources where one can anonymously share any data and misuse his or her freedom of expression to create havoc and harm individuals. The rampant dissemination of hate speech, fake news, cyberbully, and other negative and abusive stances on online platforms poses a significant societal problem that necessitates platform recognition and removal of offensive content. The detection of such misinformation on online social media are among the most researched topic, and researchers from different communities have come up with numerous ideas to detect such content (Agarwal & Chowdary, 2021; Modha et al., 2020; Nascimento et al., 2022; Wullach et al., 2022). The youth is the most susceptible to such content, especially to cyberbully and hate speech. Cyberbully is defined as belittlement, manipulation, and targeted abuse using any form of deprecatory messages and posting (Kumar & Sachdeva, 2022; Pamungkas et al., 2021). Mean, hurtful, rude texts and spreading secrets or rumors about people all consist of cyberbully. Although an electronic form of bullying can have multiple aspects, like exclusion, outing, harassment, trickery, cyberstalking, dissing, frapping, masquerading, and faming (Kumar & Sachdeva, 2022), the intention to hurt and harm is common. On the other hand, hate speech is characterized as a direct

* Corresponding author.
E-mail addresses: esshaan.mahajan@gmail.com (E. Mahajan), hemaankmahajan101@gmail.com (H. Mahajan), sanjay.kumar@dtu.ac.in (S. Kumar).

https://doi.org/10.1016/j.eswa.2023.121228
Received 18 May 2023; Received in revised form 5 August 2023; Accepted 15 August 2023
Available online 19 August 2023

and serious attack on any safeguarded category of individuals based on their ethnicity, color, gender, religion, sexual identity, handicap, or illness (Cheng et al., 2021). For the bullied, these pose serious emotional, mental, and physical risks. The victims experience overwhelming feelings of helplessness, vulnerability, unease, humiliation, agitation, isolation, depression, embarrassment, retaliation, vengeance, and in some cases, suicidal thoughts. Not surprisingly, Hate speech has the potential to be even more damaging to the masses. Many cases of hate speech also instigate riots and cause irreparable property damage. Hate speech, at many times, also creates an environment of social discomfort among various communities and results in unjust discrimination and disparity. Such feelings in society indirectly give rise to stereotypes, fear, and misconceptions among people. This problem also affects stakeholders such as governments and social media platforms. Hence, social media companies are trying to reduce such horrifying incidents, and the government is also forced to intervene at periodic times to ensure that these platforms are used judiciously. For example, many big corporations, including Facebook, YouTube, Twitter, and Microsoft, recently signed a code of conduct (Corazza et al., 2020) proposed by the European Union, which makes removing unlawful hate speech by reviewing the majority of legitimate notifications within 24 h.

Recently, much emphasis has been given to detecting and eradicating such behavior online and thus preventing any serious damage from happening. For this, researchers are trying to make an efficient and reliable method to discern such content and categorize it. Yet, there exist certain challenges to accomplishing such tasks. Hate speech detection tasks cannot be solved simply by the presence of some keywords. Some background information about the user of a post can help in prediction as a feature. A habitual user to circulate hate speech content may post again. Such users can be traced to recognize hate speech and cyberbully content. The gender of the user may also help as a feature. Hence, the model needs to be trained on numerous features. In addition to all the inherent problems of detecting the main idea behind any textual data, synthesizing texts with more than one language is another emerging challenge in detecting and classifying hate speech and cyberbully content. With evolving character encoding formats, incorporating characters from different languages has made people more expressive and lucid in expressing their ideas through texts. Yet, it conjures one major obstacle for researchers hoping to understand the sentiments behind texts and create a process to automate them. Translation in itself may change the meaning of the sentence and give out a different meaning than the one intended, or it may fail to express the implicit meaning of the sentence.

Therefore, there is imperative to create models to tackle all these problems efficiently. But, the conventional methods majorly depend on only unilateral decisions and, as a result, rely on findings from a single model. Most of these techniques do not integrate the categorization power of alternative methods in a proper manner. Hence, this motivates us to propose our work.

In this paper, we propose a novel ensemble deep learning-based multilingual hate speech and cyberbully detection model. We utilize GloVe embeddings and distribute the results of these embeddings among different deep learning approaches, namely BiLSTM, BiGRU, and a CNN-LSTM model. Afterward, the trained neurons from the respective layers are accumulated and combined to obtain the classification. Furthermore, the approach used to combine different models is a heterogeneous stacking-bagging hybrid method that combines diverse base learners using an unconventional hyperparameter tuning method of stochastic gradient descent and super learners. We use lightweight heterogeneous fusion and bagging-stacking hybrid methods. The bagging approach reduces variance, avoids over-fitting, and solves the problem of noisy data, outliers, and unbalanced data. The stacking approach makes the model immune to bias. Also, as deep learning models are time and resource intensive to optimize numerous training bags. Therefore, we use the bagging approach and provide each base learner with the same bagging samples of original dimensions. In hyperparameter tuning, we utilize the concept of super learners using the weighted combination of the predictions of the base learners. To determine the best weights for integrating the learner predictions, a cross-validation method is used. We performed the simulation on a total of nine real-world social media datasets in different languages, including English, Spanish, Italian, Indonesian, and Bengali. The results obtained by the EnsMulHateCyb model are compared with several contemporary models on nine real-life datasets from different languages. In the majority of these diverse datasets, the proposed model performs fairly well. Finally, the primary contributions of this work are as follows.

(i) The proposed model for the identification of multilingual hate speech and cyberbullying in online forums introduces a lightweight heterogeneous fusion and bagging-stacking hybrid method leading to time efficiency.
(ii) We provide an approach to combine bagging, stacking, and super learning techniques in order to achieve a robust model for diverse multilingual datasets.
(iii) We present a technique that augments the model's capabilities to detect hate speech and cyberbullying by minimizing variance, preventing over-fitting, and producing a model that is bias-resistant.
(iv) The results obtained after comprehensive experimental analysis performed on various datasets of diversified nature and languages advocates the supremacy of the proposed model.

The remainder of the paper is structured as follows. In Section 2, we discuss some of the existing work in the field of hate speech and cyberbully detection. Section 3 discusses the various data sets and the standard evaluation parameters for testing the results. Section 4 illustrates our suggested hybrid model EnsMulHateCyb for hate speech and cyberbully detection. The preprocessing, details about the models used, and detailed experimental results are described in Section 5. Finally, Section 6 concludes the paper.

2. Related work

A wide range of studies has been conducted to categorize user-generated textual content on online social media in terms of hate, abusive, or offensive language (Iwendi et al., 2020; Mozafari et al., 2020). Robust and excellent predictive models can be built by uncovering representative features utilizing deep learning models, which competently capture the word order arrangement and contextual semantics (Kumar & Sachdeva, 2022). Describing hate speech in one way across several languages and datasets may remove cultural nuances to the definition and therefore be able to generalize hate speech in datasets (Deshpande et al., 2022). The understanding of the sentiment behind a text and classifying it as hate speech requires learning from varied datasets (Hameed & Garcia-Zapirain, 2020). Most models used the sentiment as a feature to depend on a dictionary of expressive words to identify the polarity of sentiments (Cheng et al., 2021). Most classifications have used supervised classification for different sub-types of hate. The problems are concerned with using Single task learning. This paradigm updates the weights of neural networks utilizing one classification task, which involves one dataset.

Moreover, a variety of classification algorithms have been used for both deep neural network-based techniques and feature-based traditional machine learning methods that do not need handcrafting of features (Kapil & Ekbal, 2020). To analyze a text, the semantics, syntactic and spatial relationships are captured (Kumar & Sachdeva, 2022). In order to increase the effectiveness of cyberbully detection, sentiment-reliant characteristics are typically paired with content features (Cheng et al., 2021). Different perspectives of the data are captured via multi-view learning. This makes it easier to capture the various facets of the

classification process. Instead of combining all qualities into a single feature vector, each view classifier learns based on a specific type of feature. Integrating all features in one model by regularization risks the masking of relatively weak signals but key signals (MacAvaney et al., 2019). Also, due to over-fitting problems, the classifier cannot provide a limited convergent solution (Yuvaraj et al., 2021). Recent Advances in deep learning approaches achieve state-of-art results. These systems usually employ preprocessing of data and adding textual features and use of domain-specific embeddings. Due to a large number of external resources and a number of configurations, it becomes difficult to understand which classifier becomes robust for the task (Corazza et al., 2020).

With texts in social media being in numerous low-resource languages, we also focus on multilingual hate speech detection techniques. Djuric et al. (2015) suggested distributed representations of comments on Yahoo social media that convert paragraphs to vectors that can accurately identify hate speech. Using CNN and LSTM architecture in conjunction with Twitter data (Badjatiya et al., 2017) provided accurate cyberbully identification. For a variety of cross-lingual transfer tasks (Mossie & Wang, 2020) developed a hate speech detection method that uses Gated Recurrent Units (GRU) and a variety of Recurrent Neural Networks to find hate speech directed towards vulnerable minority groups on social media (RNNs). Paul and Saha (2022) showed how BERT could be used to identify cyberbullying content on three real-world corpora: Wikipedia, Twitter, and Formspring. Kumar and Sachdeva (2022) described how Bi-GRU with self-attention followed by CapsNet can be used to detect cyberbullying in the textual content of social media by utilizing learning from sequential semantic representations and spatial location data.

Pamungkas et al. (2021) proposed two joint-learning models that transfer information between language pairings utilizing various multilingual language representations. This offered instantaneous cross-lingual hate speech identification. Kocoń et al. (2021) provided a new perspective on analyzing offensive content by taking into account personalized details related to the content. Instead of using accepted offensiveness understanding, it allowed them to construct models that could be adjusted to individual user beliefs. Pronoza et al. (2021) developed a method to distinguish between attitudes toward various ethnic groups mentioned in the same text, which assisted them in identifying hate speech directed at particular ethnic groups in Russian social media writings. Sharma et al. (2022) used fine-tuned multilingual Bert and MuRIL language models to achieve hate speech detection in Hindi-English code-switched language. Wu et al. (2022) build a Gaussian Spatio-Temporal Mixture (GSTM) model for hate trend prediction based on the pre-analysis of Twitter datasets. Madhu et al. (2023) presented a dataset tailored for code-mixed conversations, encompassing both offensive and non-offensive instances. The authors collected the dataset from social media platforms and labeled offensive content to support the development and evaluation of several state-of-the-art models of offensive speech in code-mixed dialogue. del Valle-Cano et al. (2023) introduced a dichotomous approach, named SocialHaterBERT, that combines textual analysis and user profiles to detect hate speech on Twitter. By utilizing the BERT model and user-specific features, the approach shows promising results in the identification of hate speech.

3. Datasets and evaluation metrics

We outline the numerous datasets and evaluation metrics we used in this section. We used five real-life and latest hate speech/offensive textual datasets gathered from a variety of online available data sources and recent papers. Additionally, we employed a number of benchmarked evaluation measures to see how well our suggested approach performed compared with several conventional approaches as well as with recent practices. The datasets and performance metrics are described as follows:

3.1. Datasets

We used nine multilingual datasets for the experiment. Out of which five belong to the English language and four belong to Bengali, Indonesian, Italian and Spanish respectively. The textual data be accessed can through the hyperlinks provided. Basic text preprocessing like a purging of links, and special symbols were done on the datasets as per requirement. Detailed analysis and description of the datasets are provided as follows:

1. **HateCheck English Dataset**: Röttger et al. (2020) gave this dataset, which contains 3728 cases each having a gold standard label of hateful or non-hateful and a unique ID of the test cases. This dataset covers seven protected groups: black people, Muslims, immigrants, women, trans people, gay people, and disabled people. It does not contain real hate speech messages but rather synthetic texts aiming to evaluate the performance of methods based on a set of functionalities. The dataset can be accessed here: https://github.com/paul-rottger/hatecheck-data
2. **Ethos English Dataset**: The dataset is provided by Mollas et al. (2022). It has 998 comments with a label whose value ranges from 0 to 1, where 0 means absence of hate speech and 1 means presence of hate speech. 565 of them do not contain hate speech, while the rest 433 do. The dataset is available at https://github.com/intelligence-csd-auth-gr/Ethos-Hate-Speech-Dataset
3. **HateXplain English Dataset**: This dataset was compiled by Mathew et al. (2021). It has 20,148 entries and labels are classified as ‘normal’, ‘offensive’, and ‘hatespeech’. We have converged the ‘offensive’ and ‘hatespeech’ into one category of ‘hatespeech’ for binary classification. The dataset can be accessed from https://github.com/hate-alert/HateXplain/tree/master/Data
4. **Vico English Dataset**: From a White Supremacy Forum, De Gibert et al. (2018) created this dataset on hate speech. It contains Text data and the corresponding annotations. The labels are categorized as ‘hate’ and ‘nohate’ class. The dataset is present here: https://github.com/Vicomtech/hate-speech-dataset
5. **Multi-Off English Dataset**: This dataset was given by Suryawanshi et al. (2020). The text data is extracted from the OCR of multimodal meme images. The labels are categorized as ‘offensive’ and ‘non-offensive’. It contains 446 training images and 150 testing images with text.
6. **Bengali Dataset**: The dataset is made available by Romim et al. (2021). It includes user comments in Bengali from the YouTube and Facebook comment sections. Sports, entertainment, crime, religion, celebrities, TikTok, and meme are the seven categories that it is divided into. Out of these, we have used texts from categories sports and entertainment. Total hate speech and normal text instances were 5595. The dataset is available here: https://www.kaggle.com/datasets/naurosromim/bengali-hate-speech-dataset
7. **Indonesian Dataset**: Pratiwi et al. (2018) provided this hate speech dataset which contains in total 713 texts collected in indonasian language from instagram platform. Out of these 453 sentences are non-hate ones, while 260 are hate speech/cyberbully content. The dataset can be accessed from https://github.com/vidhur2k/Multilngual-Hate-Speech
8. **Italian Dataset**: The dataset provided by Sanguinetti et al. (2018) contains a corpus of italian hate speech against immigrants on the platform twitter. The data contains in total 5279 sentences. The dataset used for our experiment can be accessed from https://github.com/vidhur2k/Multilngual-Hate-Speech
9. **Spanish Dataset**: The dataset introduced by Basile et al. (2019) features spanish and english twitter texts which contain hate speech/cyberbully content against immigrants and women. It contain a total of 6600 texts. The dataset can be accessed from https://github.com/vidhur2k/Multilngual-Hate-Speech

3.2. Evaluation metrics

Precision, Recall, and F1 Score are three benchmark assessment measures that we used to test how well our model performed when applied to datasets for classification tasks. The description of these performance metrics that are used by us is given below:

1. Precision: It is a measure of how correct the algorithm is with respect to the data. It gives us a measure of the value of the positive class of the data, that model predicts correctly. Mathematically it is expressed as:

$$Precision = \frac{TP}{TP + FP} \tag{1}$$

2. Recall: The recall is the total of positive class predictions which come out of the positive instances in the relevant dataset. Lower expected false positives are indicated by a higher Recall value. It can be stated mathematically as follows:

$$Recall = \frac{TP}{TP + FN} \tag{2}$$

3. F1 Score: The F1 score is a quantity that combines Precision and Recall together. Mathematically, it can be expressed as follows:

$$F1Score = \frac{2 * Precision * Recall}{Precision + Recall} \tag{3}$$

In this case, TP (True Positive) represents that a message/comment falls under the category of hate speech and the model accurately recognized it as being positive. Similarly, FP (False Positive) portray that a message/comment is a non-hate message but the model has detected it as a hate message. Whereas TN (True Negative) suggests that a message is not in the category of hate speech and the model has predicted it to be having non-hate content. Similarly, FN (False Negative) implies that the message has hate content or cyberbully, but the model has detected him to be in the category of non-hate speech. The performance measures mentioned in this paper are calculated on the positive class only.

4. Proposed work

In this section, we describe our proposed ensemble deep learning-based multilingual hate speech and cyberbully detection model, named EnsMulHateCyb. The proposed model utilizes base learners such as BiLSTM, BiGRU, CNN, and stacked LSTM.

The proposed model uses an ensemble approach as a hybrid fusion of bagging and stacking heterogeneous methods. The combined learning from such different algorithms ensures that the shortcomings of one approach are balanced by the merits of another, resulting in a model that combines the greatest features of ensemble and deep models. The original data is transferred to each of the constituent models as bagging samples, like in the bagging ensemble approach. But, one of the original bagging samples is also provided at a later stage in the case of the CNN-LSTM layer in addition to the output of the previous module, similar to the stacking approach. This specific ensemble method yields better results in the overall F1 score of the complete model. The model takes advantage of parallel learning from the same data and combines the learning achieved through heterogeneous methods to obtain a generalized model which shows reliability and performance on datasets of entirely different nature and sources. The unique model requires unconventional tuning of hyperparameters. First, the CNN-LSTM layer is tuned by initializing the weights, and then fine-tuning is done via stochastic gradient descent. Then, using the concept of super learners, the base learners are collectively fine-tuned. This is how the model completes the binary classification task of categorizing whether the input string content is cyberbully or hate speech. The binary classification task can be expressed as follows,

$$y_c = h(x, \theta_c), y_c \in Z \tag{4}$$

where y_c is the category of the sample, x is the input feature vector, θ_c is the collection of learning parameters for h, and Z is the set of class labels, which can take two possible values, zero and one. Using Bi-GRU, BiLSTM, and CNN-LSTM as base classifiers in ensemble learning can have several advantages compared to other models.

Both Bi-GRU and BiLSTM are types of recurrent neural networks that are designed to capture long-term dependencies in sequential data. This makes them well-suited for tasks such as language modeling or sequence labeling, where the prediction at each time step depends on the entire input sequence. In addition, both Bi-GRU and BiLSTM are bidirectional, meaning that they process the input sequence in both forward and reverse directions. This allows them to capture both past and future context when making predictions, which can improve their performance on tasks such as language modeling or sequence labeling. Futhermore, the CNN-LSTM model combines a convolutional neural network with an LSTM network to extract local features from the input data and model long-term dependencies. The CNN layer is able to extract local features from the input data, such as patterns or motifs, while the LSTM layer is able to model long-term dependencies between these features.

The detailed phases of the proposed model are discussed below.

4.1. Conversion of input texts to word vectors through GloVe embedding layer

Word embeddings are mathematical vector representations of words distributed such that semantically similar meaning words (which generally appear in similar contexts) are located together. These can help find a correlation between different words through their vector representations. While training our model, we used GloVe embeddings to fulfill this task. These word embeddings are chosen due to their ease of parallel implementation. The embeddings were procured and given through three different models, whose results were combined to achieve the final classification result. The embedding was also provided again at a later stage in the CNN-LSTM model and combined with the activated neurons of the previous layer. The embeddings of the sentences indicate the presence of hate speech content by grouping certain words together in the vector space, which are frequently associated with instances of hate, personal offense, cyberbully, etc. These word text embeddings help find the relationships with the words used in texts, and through fine-tuning, in this supervised task, the models become equipped to classify the sentiments behind the texts. Only through this understanding the models are able to classify a certain sentence as hate speech or cyberbully content.

4.2. Bagging-stacking heterogeneous ensemble modeling

Heterogeneous ensembling is done on three different text models as base classifiers. Using various perspectives on data, model, and decision fusion, Heterogeneous Deep Network fusion (Tabik et al., 2020) demonstrates that complex heterogeneous fusion structures are more diversified and, as a result, exhibit greater generalization performance. The technique is applied to Bi-GRU, Bi-LSTM, and CNN-LSTM as base learners. The same data of original dimensions and nature is used as bagging samples for all the constituent models. Hence bagging approach is followed for the situation. Also, the original data is provided at later modules in addition to the output of the previous module in the CNN-LSTM base classifier. The bagging approach ensures reduced variance, avoids overfitting, and solves the problem of noisy data, outliers, and unbalanced data. Stacking done on one of the three base learners helps reduce bias. The bagging approach can be mathematically represented as follows,

$$\hat{g}_{bag} = \hat{g_1}(X) + \hat{g_2}(X) + \hat{g_3}(X) \tag{5}$$

where $\hat{g}_{bag}$ is the bagged prediction, and $\hat{g_1}(X)$, $\hat{g_2}(X)$ and $\hat{g_3}(X)$ are individual learners. Here, one of the individual learners is also following a stacked ensemble learning approach.

4.2.1. Using Bi-GRU as one of the base classifiers

A Bidirectional GRU or Bi-GRU is a sequence-processing recurrent neural network model which consists of two GRUs. One of them takes input in a forward direction and whereas the other one in a backward direction. It only contains input and forgets gates. It makes predictions in the current state using information from previous and later stages. Bi-GRU decodes the meaning and context of sentences more effectively than a simple GRU. In some cases, it can be less accurate than LSTM but works with less number of parameters.

In a GRU unit, the output y_t is calculated using current input x_t and previous state y_{t-1} under the control of two gates which is composed of a reset gate r_t and an update gate z_t. The outputs of the gates and the GRU unit are:

$$r_t = \sigma(W_r x_t + V_r y_{t-1} + b_r) \tag{6}$$

$$z_t = \sigma(W_z x_t + V_z y_{t-1} + b_z) \tag{7}$$

$$\overline{y_t} = tanh[W_y x_t + V_y(r_t \odot y_{t-1}) + b_y] \tag{8}$$

$$y_t = (1 - z_t) \odot y_{t-1} + z_t \odot \overline{y_t} \tag{9}$$

where σ is the logistic sigmoid function and V_z, W_r, V_r, W_z, W_y and V_y are the weight matrices. b_r, b_z and b_y the synthesis of bias vectors for input x_t and previous state y_{t-1}, tanh is the hyperbolic tangent activation function, $\odot$ denotes the Hadamard product.

In the proposed model, named EnsMulHateCyb, Bi-GRU plays one of the primary constituents of the entire framework. It takes word embeddings as the input from both forward as well as backward directions. Due to this, the model is able to utilize the complete sentence data and make better inferences based on the data available. This proportionally improves the training process, and the training takes place in a lesser number of parameters. After careful consideration and fine-tuning, the choice for output for the number of neurons was taken as 128. This parameter is chosen as its combination with other neurons from other models yielded better results. Thus, the BiGRU takes word embeddings as input and provides 128 neurons as output.

Fig. 1 shows the architecture of the Bi-GRU used. The architecture includes an input layer, an embedding layer, a Bi-GRU layer, and an output layer. The input layer is labeled w_i, where $i \in [1, n]$. This represents the input sequence. The input sequence is passed through the embedding layer. This layer is responsible for mapping the input sequence into a continuous vector space. The resulting vectors are then fed into the Bi-GRU layer. The Bi-GRU layer is labeled l_i and r_i, where $i \in [1, n]$. This layer consists of two GRU layers, one processing the input sequence in the forward direction and the other processing it in the reverse direction. The outputs of the Bi-GRU layer are then passed to the output layer. The output layer is represented as h_i, where $i \in [1, n]$. This layer produces the final output of the network.

4.2.2. Bi-LSTM as another base classifier

A Bidirectional LSTM or BiLSTM is a sequence-processing recurrent neural network model that consists of two LSTMs. One takes input in the foreword and the other in the back word. BiLSTMs increase the information available to the network (from previous and later states), thus consequently enhancing the contextual understanding of the sentences. The backward LSTM's hidden state at time t is

$$\overleftarrow{y_t} = LSTM(x_t, \overleftarrow{y_{t+1}}), \tag{10}$$

which processes the information from right to left direction. Here, x_t denote input state at time t and $\overleftarrow{y_{t+1}}$ output state at time $t+1$. Whereas the forward LSTM's hidden state is

$$\overrightarrow{y_t} = LSTM(x_t, \overrightarrow{y_{t-1}}) \tag{11}$$

which processes the information from the left to the right direction. Here, x_t denote input state at time t and $\overrightarrow{y_{t-1}}$ output state at time $t-1$.

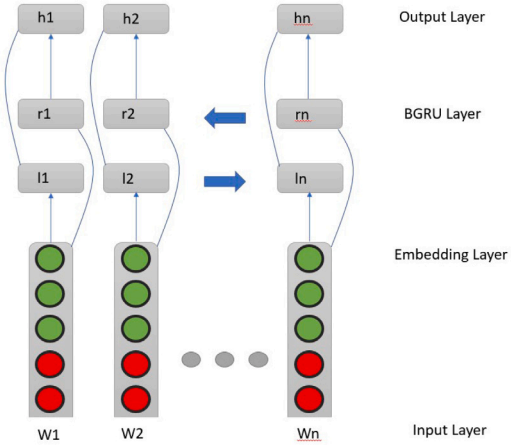


Fig. 1. Architecture of Bi-GRU.

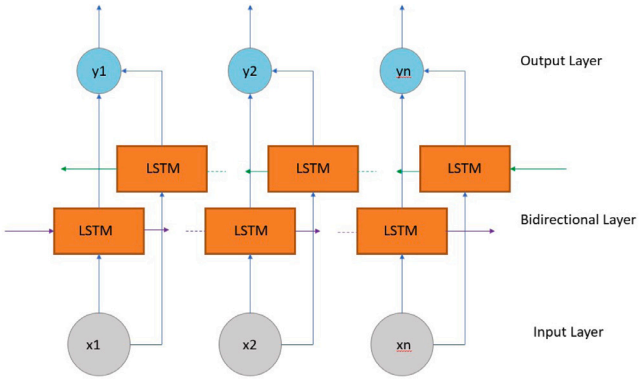


Fig. 2. Architecture of Bi-LSTM.

The output of BiLSTM can be summarized by concatenating the forward and backward states as

$$y_t = [\overrightarrow{y_t}, \overleftarrow{y_t}]. \tag{12}$$

In EnsMulHateCyb, Bi-LSTM is another essential constituent of the complete framework. It is chosen due to its ability to understand contextual meanings from data from the front and the back. It is proven to perform well on sequence-to-sequence tasks. The BiLSTM parameters are fixed, taking into account other models and itself. This leads us to choose the output neurons for this model as 64. This gives us better precision, recall, and f-scores. Thus, the BiLSTM takes in word vector embeddings as input and returns 64 neurons. These are further utilized by concatenating them with other layers. Fig. 2 shows the architecture of Bi-LSTM used. The diagram consists of three layers, an input layer, a bidirectional layer, and an output layer. The input layer is labeled x_i, where $i \in [1, n]$, representing the input sequence. The bidirectional layer is labeled LSTM, representing the bidirectional LSTM layer. The output layer consists of the output sequence labeled y_i, where $i \in [1, n]$. The input sequence is fed into the bidirectional LSTM layer, which processes it in both forward and backward directions. The outputs of the forward and backward LSTMs are then combined to produce the final output sequence.

4.2.3. Using CNN - LSTM as base classifier

Convolutional neural networks, or CNNs, are well renowned for their capacity to recognize distinct features. The convolution layer comprises of feature detectors or filters, where each convolutional layer in the network can detect more complex features. To get rid of linearity, RELU is used to the convolution layer. A pooled feature map

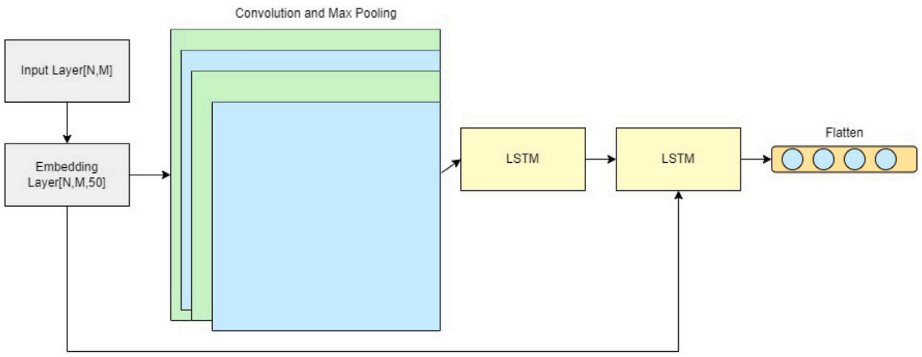


Fig. 3. Architecture of CNN-LSTM.

is produced by applying the pooling layer to the convolution layer. Thereafter, the flattening of layers is executed to create a single large vector. The neural network is given this vector as input. This fully connected neural network processes the features and generates output. Training of the network is done via front and backpropagation. We can consider text data as sequential data, like a one-dimensional matrix. We need to work with a one-dimensional convolution layer and a word embedding layer. The special word embeddings of the textual data are thus input into the convolution layer, which then performs convolution operations followed by max pooling to extract the features of textual data in vector space. These features are further passed onto LSTM layers to bring the contextual meanings into consideration. The input layer is of dimensions (NxM) where N is the total number of sentences in the dataset and M is the max length of the sentences (taken as 1000 characters for the experiment). After the sentences go through the word embeddings layer, it generates a 3-dimensional array (NxMx50), where 50 is the word embeddings value of each token that glove embeddings yield. The last layer, after convolution and max pooling, has (Nx7x128) dimensions. This layer is then passed onto LSTM layers.

Fig. 3 shows the architecture of CNN-LSTM used. The architecture consists of several layers, including an input layer, an embedding layer, a convolution and max pooling layer, two LSTM layers, and a flatten layer. The input layer is responsible for receiving the input data. The embedding layer is used for mapping the input data into a continuous vector space. The convolution and max pooling layer are responsible for extracting local features from the input data. The two LSTM layers are utilized for modeling long-term dependencies in the input data. These layers process the output of the convolution and max pooling layer using LSTM units. The flatten layer flattens the output of the LSTM layers into a single vector. This vector can then be fed into additional layers, such as fully connected layers or output layers, to produce the final classification.

Information can be stored in a long short-term memory network, which is an improved recurrent neural network (RNN). It can fix the vanishing gradient problem with the RNN. The LSTM layer takes CNN as input and then converts them from (7, 128) into dimensions (32, 32, 32). This is used because of the abilities of LSTM layers to understand contextual meanings from the data. At the next stage, original embeddings are concatenated with these and are passed through more LSTM layers, following the stacked ensemble learning approach. These are then flattened to 32 000 to obtain a single vector that represents the spacial and contextual learning from the data. This is then connected to a dense layer or a fully connected layer to lower the dimension of the neurons. The dropout technique was used to avoid overfitting. The chosen value for the activated neurons, in the end, is 256 taking into consideration the accuracy of other models as well.

Therefore, the base classifiers used were Bi-GRU, BILSTM and CNN-LSTM. These classifiers were big, yet the data available was more than sufficient to successfully train the proposed model. We verified this by carefully monitoring the validation performance of our model during training. In addition, we used cross-validation to estimate the model's performance on unseen data. This provided an estimate of how well the model is likely to perform on new data and whether the available data is sufficient for training. Also, we compared the amount of data used to train similar models in the literature which provided good results.

4.2.4. Hyperparameter tuning and ensembling of base classifiers

In this section, we explain the hyperparameter tuning of the base classifiers and their concatenation. To ensure that the network performs at its best, the hyperparameters must be improved. For this, we first initialized weights and fine-tuned them using stochastic gradient descent to achieve optimal performance on the CNN-LSTM base classifier, which used a stacked ensemble approach. It is not practical to optimize several deep models using various training bags since deep learning models need a lot of training time. That is why we use the bagging approach and provide each base learner with the same bagging samples of original dimensions. Unweighted averaging may not yield the best results when the ensemble has various base learners since it is influenced by the performance of the poor and overconfident learners, even though it is a reasonable option when the basis learners' performance is equivalent (Ju et al., 2018). For the task of combining the base learners and receive output, we use the concept of super learners, where we combine the predictions of the base learners with weights. Further, cross-validation is used to select the ideal weights for integrating the predictions of the base learners. It can be mathematically expressed as follows,

$$\hat{f}_{superlearner}(x) = \sum_{j=1}^{n} w_j f_j(x) \tag{13}$$

where $\hat{f}_{superlearner}(x)$ is a linear summation of the basis learners' projections $f_1, f_2, \ldots, f_n$, and w is the ideal weight vector that the meta learner determined.

While hyper-tuning the respected weights of the base classifiers, the F1 score is maximized. The acquired number of neurons from CNN-LSTM, BiLSTM, and BiGRU are 256, 64, and 128, respectively. These are concatenated into 448 neurons. These are then passed through a fully connected or dense layer to achieve binary classification. Thus, a final classification is achieved. Fig. 4 expresses the complete architecture of EnsMulHateCyb. It illustrates how input is first converted into embeddings and then distributed to three chosen classifiers, identical to the bagging ensemble method. The CNN-LSTM uses a stacking ensemble method within it. The learning of all the classifiers is combined and ensembled using the approach of super learners and hence used for hate speech or cyberbullying detection in the end.

5. Experimental and result analysis

In this section, we present the experiment setup and analyze the results obtained. In this study, we utilized five latest and benchmark English data sets and four multilingual datasets of textual nature for

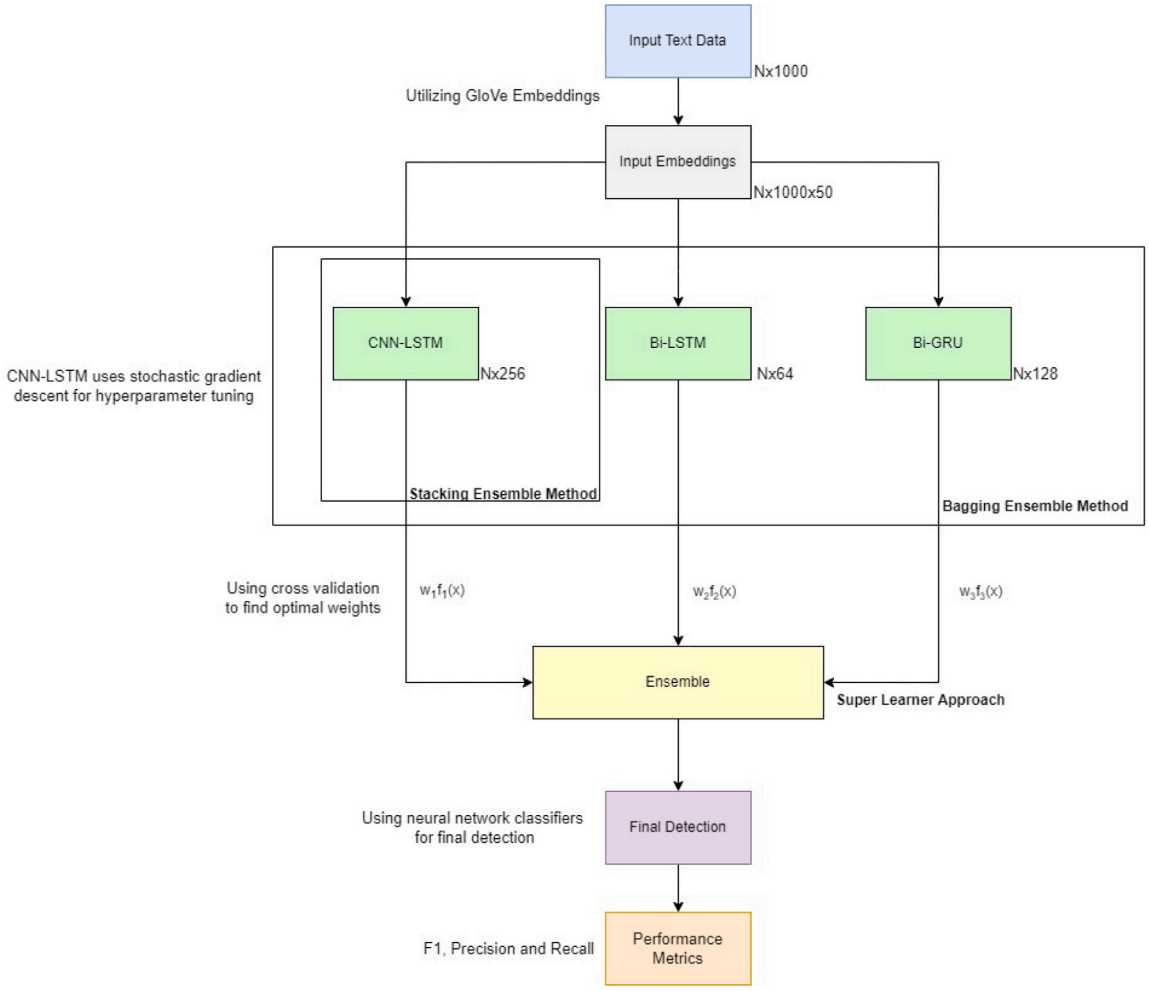


Fig. 4. Architecture of the proposed Ensemble Deep learning based Multilingual Hate speech and Cyberbullying Detection model, named EnsMulHateCyb.

cyberbullying and hate speech detection, as discussed in Section 3.1. For the need of the experiment, we turned these data sets into binary classification tasks. We performed appropriate preprocessing on each dataset to achieve the best possible results. The URL, punctuation, special symbols, links, and tags were removed. After the preprocessing, the tokens were created from the leftover text. These texts were then passed through six different models, and the results were noted in terms of precision, recall, and F-score. To prove the efficacy of the proposed EnsMulHateCyb model, we compared its performance with a total of six different methods where Bi-LSTM, Stacked LSTM, and Albert are baseline models, and CyberBERT (Paul & Saha, 2022), BiGRU with attention and CapsNet (Kumar & Sachdeva, 2022), and XLM-R (Conneau et al., 2019) as recent methods for the problem of cyberbullying and hate speech detection.

The code was developed in Python v3.10 using Google Colab and Jupyter Notebook. GPUs were utilized during the training of all the models. We primarily utilized Keras and TensorFlow frameworks for writing the majority of the model codes. Further various Python libraries are utilized like NLTK for stop words removal, Matplotlib and seaborn for visualization, pandas for data preparation, and pickle to store relevant variables. In cases where the data sets were not already divided into train and test, we have divided the complete data sets in the ratio of 70:30 for training and testing purposes. For some cases, the dataset was prepared on a local machine, which has the following configurations: Intel(R) Core(TM) i7-10510U CPU @ 1.80 GHz with 8 GB RAM Windows 11.

5.1. Details about various models considered including the proposed model

In BiLSTM, embeddings were passed onto a bi-directional LSTM with 32 as the parameter, and then the final classification result was obtained using a dense layer with activation sigmoid. The loss of binary cross entropy and the optimizer adam were used in the model's development. In the Stacked LSTM model, the embedding layer was passed on to an LSTM layer which in turn was connected to another LSTM layer. Thereafter, the original embeddings were concatenated with this final layer, and hence the resultant was flattened. Finally, a dense or fully connected layer provided the categorization outcome. The model uses adam as the optimizer and binary cross entropy as the loss. In Bi-GRU, the embedding layer was passed onto the bi-directional gated recurrent unit with 32 as the parameter. The final classification was reached after it had gone through a dense layer of sigmoid activation. Adam optimizer and binary cross entropy loss were utilized in the model. For the BERT model, we required a Bert tokenizer and TF Bert (TensorFlow model) for sequence classification. These were imported from the Transformers library. These were used to tokenize the inputs and obtain logits. TF.math.arg max function was applied to the logits to receive the predicted classes. In the ALBERT model, Transformers and sentence piece libraries were used to get the Albert tokenizer and TF Albert model for sequence classification. Similar to BERT, logits were obtained and argmax function of TensorFlow was applied, and predicted classes for each instance were procured. XLM model tokenizer and model for sequence classification were imported from the library Transformers were imported. Also, after tokenizing the

Table 1
The results obtained by considered models for various evaluation metrics on the Vico dataset.

Models	Precision	Recall	F1 score
BiLSTM	48.89	83.26	61.61
Stacked LSTM	51.04	81.59	62.80
XLM	49.12	82.01	61.44
Albert	41.50	43.93	42.68
Bert	38.77	23.85	29.53
BiGRU	49.40	84.72	62.41
EnsMulHateCyb	51.07	99.16	67.42

Table 2
The results obtained by considered models for various evaluation metrics on the HateXplain dataset.

Models	Precision	Recall	F1 score
BiLSTM	72.47	36.10	48.73
Stacked LSTM	72.11	99.33	83.56
XLM	71.84	87.69	78.98
Albert	62.79	28.05	38.73
Bert	69.37	56.58	62.32
BiGRU	71.23	35.81	47.66
EnsMulHateCyb	72.12	100.00	83.79

Table 3
The outcome of evaluation metrics for different models with Ethos dataset.

Models	Precision	Recall	F1 score
BiLSTM	41.13	94.31	57.28
Stacked LSTM	43.11	78.86	55.75
XLM	42.68	28.45	34.14
Albert	22.76	23.24	21.11
Bert	25.59	20.07	22.49
BiGRU	42.44	92.87	58.25
EnsMulHateCyb	44.15	27.64	34.99

Table 4
The outcome of evaluation metrics for different models with HateCheck dataset.

Models	Precision	Recall	F1 score
BiLSTM	68.13	100.00	81.04
Stacked LSTM	67.13	100.00	80.33
XLM	66.82	56.90	61.47
Albert	51.64	21.87	30.72
Bert	55.56	23.75	33.27
BiGRU	64.71	92.65	76.19
EnsMulHateCyb	67.31	90.52	77.21

inputs, logits were obtained through the model, then TF.math.argmax function was used on logits to obtain the predicted class. Hence, the labels were predicted.

In the EnsMulHateCyb model, we applied word embeddings to a Bi-GRU layer consisting of 64 units and a BiLSTM layer with 32 units. Additionally, it passes through three convolutional layers with max-pooling to reduce neuron count before being processed by an LSTM. Then the original embedded layer is merged with this LSTM, which afterward gets flattened (a dropout and dense layer is used). These layers are finally concatenated with BiGRU and BiLSTM. Then there is a dense layer fully connected layer for final classification. The model was built using binary cross entropy loss and Adam optimizer, while the final dense layer employed sigmoid activation.

Now, we present the discussion on the performance of the proposed EnsMulHateCyb model compared against several contemporary and classical methods for hate speech and cyberbully detection. The experiments are performed on nine datasets of various sizes, natures, and languages and the values of various performance metrics are computed as described in Section 3.2.

5.2. Vico English dataset

In Table 1, we obtained the evaluation metrics for the various algorithms. From the results, we can deduce that our model outperformed all the other models and gave the highest precision of 51.07% and the highest recall of 99.16%. We examined that our model considerably outperformed other models by a substantial margin. Stacked LSTM performed best among the other contemporary models giving an F1 score of 62.8% with the highest precision of 51.04% among them. BiGRU scored the second highest recall of 84.72% among them. The collected data further demonstrate that our suggested strategy produced consistent outcomes across the performance indicators. The F1 score obtained by the proposed model,EnsMulHateCyb, is 67.43%.

5.3. HateXplain English dataset

Table 2 provides us with the results of performance metrics of various algorithms on the dataset. Our model again outperformed all the other models and gave the near-highest precision of 72.12% and the highest recall of 100.00%. Thus, the proposed model gave the highest F1 score of 83.79%. The highest precision score was by BiLSTM which was 72.47% followed by our model and Stacked LSTM, which was 72.11%. The highest recall next to our model was by Stacked LSTM, which was 99.33%, and an F1 score of 83.56% which was just next to our model. In comparison to other models, our model was able to capture more details and produce higher performance since it was able to optimally blend the findings of all the pre-trained models that were selected.

5.4. Ethos English dataset

Table 3 lists the results of the performance metrics of various algorithms on the dataset. From the results, it is inferred that BiLSTM performed better than other algorithms by giving an F1 score of 57.28% with the highest recall of 94.31% and precision of 41.13%. Further, it is deduced that our proposed method scored the highest precision of 44.15%. Furthermore, this dataset proved to be better for BiLSTM training followed by Stacked LSTM and then by XLM and EnsMulHateCyb. However, EnsMulHateCyb lacked in performance and produced an F1 score of 34.99% The possible reason for the slightly low recall and F1 score for EnsMulHateCyb could be the incompetence of the CNN-LSTM component of the model to analyze this particular nature of data as other constituent models seem to have performed well individually.

5.5. HateCheck English dataset

In the Table 4, we list the results of the performance metrics of all the algorithms on the HateCheck English Dataset dataset. From the results, it is evident that the BiLSTM model performed better than other models by having an F1 score of 81.04% which was slightly above the F1 score of Stacked LSTM, which was 80.33%. Our proposed model, EnsMulHateCyb on the other hand, produced an F1 score of 77.21% which was just below the stacked LSTM. However, the highest precision was given by Stacked LSTM and BiLSTM, which was 100% and the next highest was given by our model with a value of 90.52%. According to the result, Bi LSTM and stacked LSTM performed pretty well and EnsMulHateCyb adapted to the nature of this data and prevailed the learning from BiGRU and BiLSTM components of the model and provided a much more universal model.

5.6. Multi-off English dataset

The performance metrics of dataset Multi-off are presented in Table 5 for various algorithms considered in this study. The XLM produced the highest F1 score and recall of 50.25% and 68.96% respectively. Furthermore, the proposed model attained the highest precision of 46.32% and gave the F1 score of 42.59% which was just

Table 5
The outcome of evaluation metrics for different models with Multi-Off dataset.

Models	Precision	Recall	F1 score
BiLSTM	34.69	29.31	31.77
Stacked LSTM	35.80	50.65	41.72
XLM	39.21	68.96	50.25
Albert	38.70	62.07	47.68
Bert	35.08	34.48	34.78
BiGRU	37.21	32.45	34.67
EnsMulHateCyb	46.32	39.65	42.59

Table 6
The performance comparison of the different models on the Bengali dataset.

Models	Precision	Recall	F1 score
BiLSTM	48.46	48.69	48.57
Stacked LSTM	49.36	36.90	42.34
XLM	54.23	31.31	39.69
Albert	45.30	67.53	54.22
Bert	52.99	51.66	52.32
BiGRU	50.77	49.12	49.93
EnsMulHateCyb	83.87	59.40	69.55

Table 7
Results obtained by various models for all evaluation metrics on the Indonesian dataset.

Models	Precision	Recall	F1 score
BiLSTM	23.52	11.94	15.84
Stacked LSTM	27.78	7.46	11.76
XLM	33.96	80.59	47.78
Albert	31.21	50.57	38.59
Bert	30.43	41.79	35.22
BiGRU	26.69	14.55	18.83
EnsMulHateCyb	77.78	31.34	44.68

below XLM and Albert. The trends are as such because XLM and Albert used the concept of transfer learning and were pre-trained on large quantities of similar kinds of datasets. For this reason, XLM and Albert were able to perform quite well on a small dataset like Multi-Off English while other ones could not. However, if EnsMulHateCyb is trained on huge amounts of similar kinds of datasets, its performance can be improved as well.

5.7. Bengali dataset

Performance metrics for the Bengali language are given in Table 6 for all the algorithms. After analyzing the results, we deduce that EnsMulHateCyb outperformed all the other algorithms by scoring the highest F1 score of 69.55% . Also, our model gave the highest precision of 83.87%. However, the highest recall was scored by the Albert model at 67.53%. Furthermore, the BiLSTM model performed better than the stacked LSTM. Yet, in terms of F1 score, EnsMulHateCyb beat even the present state-of-the-art models by a good margin. This indicated the ability of the EnsMulHateCyb model to successfully understand a raw translation of Bengali text and perform decently with respect to all evaluation metrics.

5.8. Indonesian dataset

In Table 7, we obtained the performance metrics of the algorithms applied to the Indonesian language dataset. For this dataset, we observe that XLM portrayed the highest F1 score of 47.78% with the highest recall of 80.59%. On the other hand, BiLSTM and stacked LSTM did not perform well on this dataset by scoring a low F1 score of 15.84% and 11.76%, respectively. Also, we notice that Albert's model performed better than the BERT model. Moreover, the highest precision of 77.78% was scored by our proposed model EnsMulHateCyb and with an F1 score of 44.68%. This F1 score was just next to XLM. The recall of

Table 8
The performance comparison of the different models on the Italian dataset.

Models	Precision	Recall	F1 score
BiLSTM	2.13	11.24	3.58
Stacked LSTM	3.85	10.12	5.58
XLM	17.48	38.76	24.09
Albert	13.03	59.69	21.38
Bert	13.85	31.40	19.21
BiGRU	5.21	16.73	7.95
EnsMulHateCyb	50.75	13.18	20.92

Table 9
The performance comparison of the different models on the Spanish dataset.

Models	Precision	Recall	F1 score
BiLSTM	45.45	1.88	3.62
Stacked LSTM	41.46	2.14	4.06
XLM	40.44	34.80	37.41
Albert	41.17	63.57	49.97
Bert	38.43	34.42	36.32
BiGRU	46.95	3.96	7.30
EnsMulHateCyb	76.19	22.11	34.27

EnsMulHateCyb indicated a large number of false negatives (wrongly predicted hate speech). Thus, there were some instances of hate speech that the model predicted incorrectly, yet due to its high precision. It could be concluded that it predicted a true positive, which is quite the reverse of XLM. It could be inferred that these two models performed equally well in the F1-score, yet their perception of the dataset was quite different.

5.9. Italian dataset

Table 8 depicts the results of the performance metrics of all the algorithms on the Italian corpus of hate speech against immigrants on Twitter. Overall, this data proved to be a challenge for all the models with a pretty low F1 score. Yet, XLM managed to score the highest F1 score of 24.09%. Furthermore, Albert, EnsMulHateCyb, and Bert performed equally well. We can observe that EnsMulHateCyb scored the highest precision and Albert scored the highest recall of 50.75% and 59.69%, respectively, in this dataset. Here, our proposed model, EnsMulHateCyb gave an F1 score of 20.92%. However, BiLSTM and stacked LSTM did not perform relatively well. Despite the fact, BiLSTM, EnsMulHateCyb, and Stacked LSTM all used the same word embeddings (Glove), the results were drastically good for EnsMulHateCyb. This showed the competence of the proposed novel Multilingual framework. Here, EnsMulHateCyb demonstrated its near equal proficiency compared to XLM, Bert, and Albert.

5.10. Spanish dataset

It featured Spanish and English Twitter texts against women and immigrants. Performance metrics for the Spanish dataset are given in Table 9 for all the models. From the result, we infer that Albert outperformed all the other algorithms by scoring the highest F1 score of 49.97% and the highest recall score of 63.57%. At the same time, our proposed model EnsMulHateCyb scored the highest precision of 76.19% in the dataset. Moreover, XLM and Bert performed equally well, followed by EnsMulHateCyb. However, we noticed that BiLSTM and stacked LSTM did not perform well on this dataset. After our analysis, we draw the inference that perceptions about the dataset of EnsMulHateCyb and Albert are quite different. Also, EnsMulHateCyb showed its versatility in giving a decent score on a hybrid dataset of Spanish and English combined and achieve an F1 score of 34.27%. It could successfully figure out the embedded meanings behind the texts.

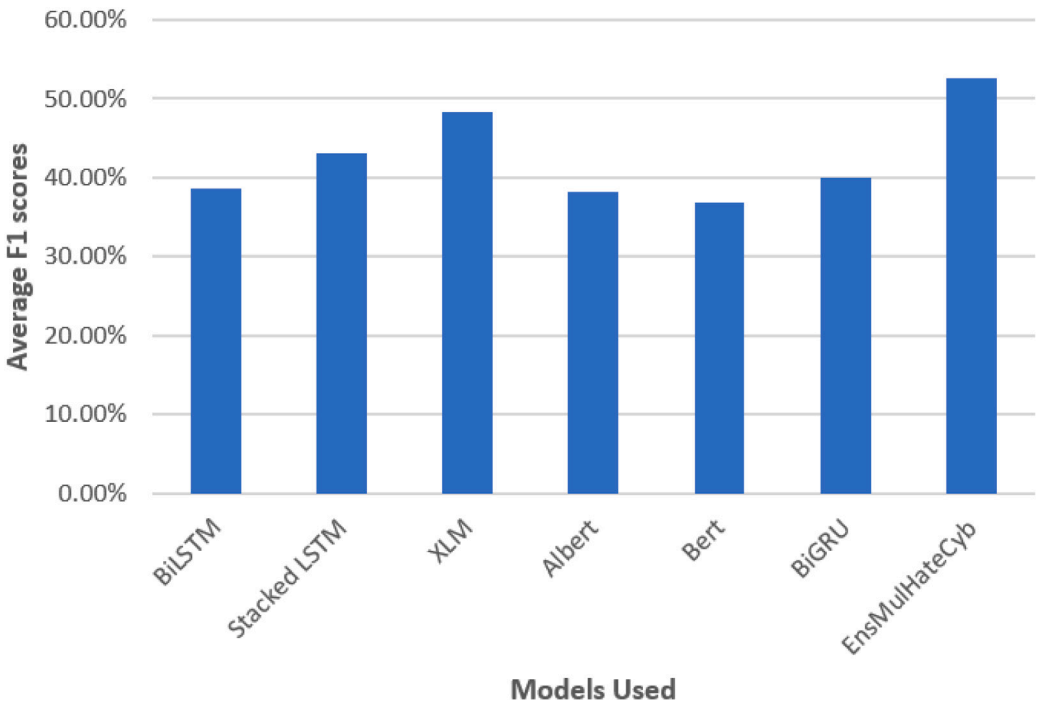


Fig. 5. Average F1 score attained by various comparing models including the proposed model on all nine datasets.

5.11. Findings and implications

The above analysis of results from subsection 5.2 to 7.10 advocate that the proposed model EnsMulHateCyb is capable of reliable and efficient detection and categorization of texts into hatespeech and cyberbully content. In our analysis of comparing the performance of the proposed model against six different contemporary models, namely, BiLSTM, Stacked LSTM, XLM, Albert, Bert and BiGRU, we have taken the arithmetic mean of F1 scores of six comparing models and compared it with the average of F1 score of our method across all the datasets. The average F1 scores of the methods used are as follows: EnsMulHateCyb(52.7%), XLM(48.26%), Stacked LSTM(43.03%), BiGRU(40.07%), BiLSTM(38.56%), Albert(38.25%), Bert(36.88%). This allows the proposed model to be clearly contrasted against other techniques, and hence allowed us to understand the effectiveness of our model on different datasets.

It is evident that the proposed method demonstrated enhanced detection capability with an increase of at least 4.44% in the F1 scores compared to six comparing hate speech and cyberbully detection models across all the nine datasets. This comparison is illustrated in Fig. 5. The pictorial results depicts the average F1 score of all the models used on nine datasets. It is evident that the proposed model outperforms all other models. However, XLM model performed second best on average and also outperforms our proposed model in four datasets on F1 score factor. This is due to the following reasons. XLM is pre-trained on large amounts of text data in multiple languages, allowing it to learn shared representations across languages. This can improve its performance on cross-lingual tasks, where the goal is to transfer knowledge from one language to another. Additionally, XLM uses advanced pre-training objectives, such as masked language modeling and translation language modeling, to learn cross-lingual representations. These objectives are designed to encourage the model to learn shared representations across languages, which can improve its performance on cross-lingual tasks. Also, XLM is based on transformer architecture, which has been shown to be effective for a wide range of NLP tasks. The transformer architecture uses self-attention mechanisms to capture long-range dependencies within the input data, which can improve its performance on tasks such as language modeling or machine translation. In the future, the above-mentioned attributes may be included in our proposed method to improve its efficiency. Moreover, our method to use a hybrid stacking-bagging ensemble approach along with a unique hyperparameter tuning technique can be used to improve the existing methods of hate speech and cyberbully detection. In addition, our model can be scaled and used to augment the process of automating hate speech and cyberbully content detection and eradication of the same from social media. The model's ability to understand implicit meanings of multilingual content makes it relevant to social media contents of varied low-resource based languages as well.

Absolute runtime comparison

We present a comparison of the absolute runtime of the proposed model against other models considered in this study and Table 10 depicts the same. Here, absolute runtime includes the training and testing time required by a model on a particular dataset. From the results, it is evident that the proposed model required intermediate absolute runtime, in contrast with other methods used. We can infer that XLM, Albert, and Bert required comparatively larger training and testing time as compared to models like BiLSTM, Stacked LSTM, BiGRU, and EnsMulHateCyb. This is due to the complex nature of their architecture. The parameters in XLM, Albert, and Bert required a higher amount of time to train. However, much simpler models with comparatively lower parameters were trained much faster, like BiLSTM, Stacked LSTM, and BiGRU. The proposed model, EnsMulHateCyb, completed training and testing in an intermediate time. Hence, the proposed model shows a balance of complexity and performance by providing good results in reasonable training and testing time.

6. Conclusion

Online social media fosters fast and efficient information sharing and exchange among connected users. However, few users manipulate these platforms for hostile practices like hate speech and cyberbullying. In this paper, we presented an ensemble deep learning-based multilingual novel cyberbully and hate speech detection model. With

Table 10
The absolute runtime (in seconds) comparison of the different models on all datasets used.

Datasets	BiLSTM	Stacked LSTM	XLM	Albert	Bert	BiGRU	EnsMulHateCyb
Vico	14.21	25.14	406.29	336.13	349.57	18.04	38.76
HateXplain	20.78	50.42	553.55	476.22	478.98	31.67	94.97
Ethos	6.98	12.34	401.11	332.60	338.56	7.89	24.23
HateCheck	8.35	15.77	418.23	356.81	345.92	12.78	30.55
Multi-Off	5.22	45.52	390.25	320.79	317.33	6.54	22.35
Bengali	14.04	23.14	456.29	376.13	379.57	15.47	32.88
Indonesian	5.02	44.44	378.54	314.60	320.62	7.13	23.45
Italian	14.87	26.89	416.08	338.43	345.90	19.99	41.18
Spanish	15.55	28.75	423.12	342.50	350.27	20.59	45.09

texts in social media being in numerous low-resource languages, in this study, we also focused on multilingual datasets. The proposed ensemble model utilized base learners using Bi-directional long short-term memory (BiLSTM), Bi-directional gated recurrent unit (Bi-GRU), Convolutional neural network (CNN), and long short-term memory (LSTM). We incorporated different models in a heterogeneous stacking-bagging hybrid method that combines diverse base learners using an unconventional hyperparameter tuning method of stochastic gradient descent and super learners. We utilized the concept of super learners using the weighted combination of what the basic learners have predicted for hyperparameter tuning. A number of modern models are compared to the outcomes produced by the suggested model on nine real-life datasets from different languages. The proposed model produces commendable results in most of these varied datasets.

CRediT authorship contribution statement

Esshaan Mahajan: Conceptualization, Methodology, Software, Writing – original draft, Writing – review & editing. **Hemaank Mahajan:** Conceptualization, Methodology, Software, Writing – original draft. **Sanjay Kumar:** Conceptualization, Methodology, Visualization, Investigation, Writing – review & editing.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

Data will be made available on request.

References

Agarwal, S., & Chowdary, C. R. (2021). Combating hate speech using an adaptive ensemble learning model with a case study on COVID-19. *Expert Systems with Applications*, *185*, Article 115632.

Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017). Deep learning for hate speech detection in tweets. In *Proceedings of the 26th international conference on world wide web companion* (pp. 759–760).

Basile, V., Bosco, C., Fersini, E., Nozza, D., Patti, V., Pardo, F. M. R., Rosso, P., & Sanguinetti, M. (2019). Semeval-2019 task 5: Multilingual detection of hate speech against immigrants and women in twitter. In *Proceedings of the 13th international workshop on semantic evaluation* (pp. 54–63).

Cheng, L., Guo, R., Silva, Y. N., Hall, D., & Liu, H. (2021). Modeling temporal patterns of cyberbullying detection with hierarchical attention networks. *ACM/IMS Transactions on Data Science*, *2*(2), 1–23.

Conneau, A., Khandelwal, K., Goyal, N., Chaudhary, V., Wenzek, G., Guzmán, F., Grave, E., Ott, M., Zettlemoyer, L., & Stoyanov, V. (2019). Unsupervised cross-lingual representation learning at scale. arXiv preprint arXiv:1911.02116.

Corazza, M., Menini, S., Cabrio, E., Tonelli, S., & Villata, S. (2020). A multilingual evaluation for online hate speech detection. *ACM Transactions on Internet Technology (TOIT)*, *20*(2), 1–22.

De Gibert, O., Perez, N., García-Pablos, A., & Cuadros, M. (2018). Hate speech dataset from a white supremacy forum. arXiv preprint arXiv:1809.04444.

Deshpande, N., Farris, N., & Kumar, V. (2022). Highly generalizable models for multilingual hate speech detection. arXiv preprint arXiv:2201.11294.

Djuric, N., Zhou, J., Morris, R., Grbovic, M., Radosavljevic, V., & Bhamidipati, N. (2015). Hate speech detection with comment embeddings. In *Proceedings of the 24th international conference on world wide web* (pp. 29–30).

Hameed, Z., & Garcia-Zapirain, B. (2020). Sentiment classification using a single-layered BiLSTM model. *IEEE Access*, *8*, 73992–74001.

Iwendi, C., Srivastava, G., Khan, S., & Maddikunta, P. K. R. (2020). Cyberbullying detection solutions based on deep learning architectures. *Multimedia Systems*, 1–14.

Ju, C., Bibaut, A., & van der Laan, M. (2018). The relative performance of ensemble methods with deep convolutional neural networks for image classification. *Journal of Applied Statistics*, *45*(15), 2800–2818.

Kapil, P., & Ekbal, A. (2020). A deep neural network based multi-task learning approach to hate speech detection. *Knowledge-Based Systems*, *210*, Article 106458.

Kocoń, J., Figas, A., Gruza, M., Puchalska, D., Kajdanowicz, T., & Kazienko, P. (2021). Offensive, aggressive, and hate speech analysis: From data-centric to human-centered approach. *Information Processing & Management*, *58*(5), Article 102643.

Kumar, A., & Sachdeva, N. (2022). A bi-GRU with attention and CapsNet hybrid model for cyberbullying detection on social media. *World Wide Web*, *25*(4), 1537–1550.

MacAvaney, S., Yao, H.-R., Yang, E., Russell, K., Goharian, N., & Frieder, O. (2019). Hate speech detection: Challenges and solutions. *PLoS One*, *14*(8), Article e0221152.

Madhu, H., Satapara, S., Modha, S., Mandl, T., & Majumder, P. (2023). Detecting offensive speech in conversational code-mixed dialogue on social media: A contextual dataset and benchmark experiments. *Expert Systems with Applications*, *215*, Article 119342.

Mathew, B., Saha, P., Yimam, S. M., Biemann, C., Goyal, P., & Mukherjee, A. (2021). Hatexplain: A benchmark dataset for explainable hate speech detection. *vol. 35*, In *Proceedings of the AAAI conference on artificial intelligence* (pp. 14867–14875).

Modha, S., Majumder, P., Mandl, T., & Mandalia, C. (2020). Detecting and visualizing hate speech in social media: A cyber watchdog for surveillance. *Expert Systems with Applications*, *161*, Article 113725.

Mollas, I., Chrysopoulou, Z., Karlos, S., & Tsoumakas, G. (2022). ETHOS: a multi-label hate speech detection dataset. *Complex & Intelligent Systems*, 1–16.

Mossie, Z., & Wang, J.-H. (2020). Vulnerable community identification using hate speech detection on social media. *Information Processing & Management*, *57*(3), Article 102087.

Mozafari, M., Farahbakhsh, R., & Crespi, N. (2020). Hate speech detection and racial bias mitigation in social media based on BERT model. *PLoS One*, *15*(8), Article e0237861.

Nascimento, F. R., Cavalcanti, G. D., & Da Costa-Abreu, M. (2022). Unintended bias evaluation: An analysis of hate speech detection and gender bias mitigation on social media using ensemble learning. *Expert Systems with Applications*, *201*, Article 117032.

Pamungkas, E. W., Basile, V., & Patti, V. (2021). A joint learning approach with knowledge injection for zero-shot cross-lingual hate speech detection. *Information Processing & Management*, *58*(4), Article 102544.

Paul, S., & Saha, S. (2022). CyberBERT: BERT for cyberbullying identification: BERT for cyberbullying identification. *Multimedia Systems*, *28*(6), 1897–1904.

Pratiwi, N. I., Budi, I., & Alfina, I. (2018). Hate speech detection on indonesian instagram comments using FastText approach. In *2018 international conference on advanced computer science and information systems* (pp. 447–450). IEEE.

Pronoza, E., Panicheva, P., Koltsova, O., & Rosso, P. (2021). Detecting ethnicity-targeted hate speech in Russian social media texts. *Information Processing & Management*, *58*(6), Article 102674.

Romim, N., Ahmed, M., Talukder, H., & Saiful Islam, M. (2021). Hate speech detection in the bengali language: A dataset and its baseline evaluation. In *Proceedings of international joint conference on advances in computational intelligence: IJCACI 2020* (pp. 457–468). Springer.

Röttger, P., Vidgen, B., Nguyen, D., Waseem, Z., Margetts, H., & Pierrehumbert, J. B. (2020). Hatecheck: Functional tests for hate speech detection models. arXiv preprint arXiv:2012.15606.

Sanguinetti, M., Poletto, F., Bosco, C., Patti, V., & Stranisci, M. (2018). An italian twitter corpus of hate speech against immigrants. In *Proceedings of the eleventh international conference on language resources and evaluation (LREC 2018)*.

Sharma, A., Kabra, A., & Jain, M. (2022). Ceasing hate with moh: Hate speech detection in hindi–english code-switched language. *Information Processing & Management*, *59*(1), Article 102760.

Suryawanshi, S., Chakravarthi, B. R., Arcan, M., & Buitelaar, P. (2020). Multimodal meme dataset (multioff) for identifying offensive content in image and text. In *Proceedings of the second workshop on trolling, aggression and cyberbullying* (pp. 32–41).

Tabik, S., Alvear-Sandoval, R. F., Ruiz, M. M., Sancho-Gómez, J.-L., Figueiras-Vidal, A. R., & Herrera, F. (2020). MNIST-NET10: A heterogeneous deep networks fusion based on the degree of certainty to reach 0.1% error rate. Ensembles overview and proposal. *Information Fusion*, *62*, 73–80.

del Valle-Cano, G., Quijano-Sánchez, L., Liberatore, F., & Gómez, J. (2023). Social-HaterBERT: A dichotomous approach for automatically detecting hate speech on Twitter through textual analysis and user profiles. *Expert Systems with Applications*, *216*, Article 119446.

Wu, X.-K., Zhao, T.-F., Lu, L., & Chen, W.-N. (2022). Predicting the hate: A gstm model based on Covid-19 hate speech datasets. *Information Processing & Management*, *59*(4), Article 102998.

Wullach, T., Adler, A., & Minkov, E. (2022). Character-level hypernetworks for hate speech detection. *Expert Systems with Applications*, *205*, Article 117571.

Yuvaraj, N., Chang, V., Gobinathan, B., Pinagapani, A., Kannan, S., Dhiman, G., & Rajan, A. R. (2021). Automatic detection of cyberbullying using multi-feature based artificial intelligence with deep decision tree classification. *Computers & Electrical Engineering*, *92*, Article 107186.