Journal of Big Data

SURVEY Open Access

Hate speech detection in the Bengali language: a comprehensive survey

Abdullah Al Maruf[1†], Ahmad Jainul Abidin[2†], Md. Mahmudul Haque[1†], Zakaria Masud Jiyad[1†], Aditi Golder[3], Raaid Alubady[4] and Zeyar Aung[5*]

†Abdullah Al Maruf and Ahmad Jainul Abidin share the first authorship.

† Md. Mahmudul Haque and Zakaria Masud Jiyad share the second authorship.

*Correspondence: zeyar.aung@ku.ac.ae

[1] Department of Computer Science and Engineering, Bangladesh University of Business and Technology, Dhaka, Bangladesh
[2] Department of Information and Communication Technology, Bangladesh University of Professionals, Dhaka, Bangladesh
[3] Institute of Information Technology, Jahangirnagar University, Dhaka, Bangladesh
[4] Department of Information Technology, Engineering Technical College, Al-Ayen University, Thi-Qar, Iraq
[5] Center for Secure Cyber-Physical Systems (C2PS) and Department of Computer Science, Khalifa University of Science and Technology, Abu Dhabi, UAE

Abstract

The detection of hate speech (HS) in online platforms has become extremely important for maintaining a safe and inclusive environment. While significant progress has been made in English-language HS detection, methods for detecting HS in other languages, such as Bengali, have not been explored much like English. In this survey, we outlined the key challenges specific to HS detection in Bengali, including the scarcity of labeled datasets, linguistic nuances, and contextual variations. We also examined different approaches and methodologies employed by researchers to address these challenges, including classical machine learning techniques, ensemble approaches, and more recent deep learning advancements. Furthermore, we explored the performance metrics used for evaluation, including the accuracy, precision, recall, receiver operating characteristic (ROC) curve, area under the ROC curve (AUC), sensitivity, specificity, and F1 score, providing insights into the effectiveness of the proposed models. Additionally, we identified the limitations and future directions of research in Bengali HS detection, highlighting the need for larger annotated datasets, cross-lingual transfer learning techniques, and the incorporation of contextual information to improve the detection accuracy. This survey provides a comprehensive overview of the current state-of-the-art HS detection methods used in Bengali text and serves as a valuable resource for researchers and practitioners interested in understanding the advancements, challenges, and opportunities in addressing HS in the Bengali language, ultimately assisting in the creation of reliable and effective online platform detection systems.

Keywords: Hate speech, Bengali language, Natural language processing, Machine learning, Deep learning, Artificial intelligence

Introduction

Hate speech (HS) is a form of expression that spreads negativity and incites violence against people or groups based on their inborn traits, including religion, race, ethnicity, sexual preference, and gender. The increasing prevalence of social media platforms has facilitated the spread of HS by providing a medium for individuals to communicate with one another, regardless of their psychological traits and backgrounds. The global population is now estimated at almost 7.7 billion people, and approximately 3.484 billion people are active on social media [1].

Conflicts arise among individuals with diverse mindsets, as one may find it challenging to tolerate the thoughts of another. Social media has evolved into a platform for HS under the guise of freedom of speech. Text serves as one method of communication. When text is used to insult others, it is considered inaccurate and defamatory and could lead to harassment, religious conflicts, and eve-teasing on social media platforms [2]. In severe cases, HS can even threaten and incite people against society and the government. In addition to text, other media such as images, audio, videos, and image/graphical representations can all be used to propagate unpleasant or hateful content on social media. Moreover, the use of hashtags facilitates the transformation of text into clickable links, enabling information to be transmitted more quickly.

As there are no universal guidelines or comprehensive rules for defining HS, controlling and detecting hateful and offensive language pose major challenges in today's society. While social media offers a platform for all users to share their thoughts, instances of offensive and harmful content are not uncommon and can severely impact users' experience and even the civility of a community [3]. HS may also have significant cultural impacts depending on one's cultural background.

Automated HS identification is critical in addressing the spread of HS, especially on social media. Numerous methods have been developed for this task, including a recent increase in the development of deep learning (DL)-based approaches. Technology companies, scholars, and policymakers have been developing natural language processing (NLP) tools to identify HS and thus mitigate unlawful activities [4]. NLP is a large subdomain of artificial intelligence that helps computers understand, translate, and control human language [5].

To date, most hate speech research has focused on addressing the challenges posed by languages with abundant linguistic resources. As English is the most widely used language on the internet, extensive research has been conducted on English-language hate speech detection [6, 7]. To promote additional studies in this area, researchers created the Arabic datasets annotated for religious hate detection and the Arabic lexicon of religious hate terms [8, 9]. Some researchers have also worked on multilingual projects [10]. One researcher proposed a robust neural classifier for an HS binary classification task, with comparisons across different languages, namely, English, Italian and German, focusing on freely available Twitter datasets for HS detection [11].

According to statistics from 2018, Bengali (endonym "Bangla") is the sixth most widely spoken native language, with an estimated 300 million native speakers worldwide [12]. Bengali is Bangladesh's national, official, and most widely spoken language and the second-most commonly used language in India.

Social media usage has become increasingly prevalent in Bangladesh, with an estimated 50.3 million active users engaging with various online platforms in 2022. This figure represents approximately 29.7% of the population, a substantial proportion of the country's total population. There are several benefits to the use of online media and the internet, e.g., uniting people from all different backgrounds and allowing them to communicate and interact with one another [11]. As a result, people with diverse psychological and sociocultural backgrounds can share their thoughts, ideas, and opinions easily.

Among Bangladesh's social media users, there are nearly 46 million users on Facebook and around 29 million users on YouTube. According to UNICEF, approximately 32

percent of children in Bangladesh are vulnerable to cyberbullying. A study conducted by the Cyber Crime Awareness Foundation, an NGO, revealed that 73.71% of cybercrime victims are women [13]. The situation worsens when individuals resort to suicide [14]. Despite this large user base, the availability of comprehensive and linguistically diverse datasets for Bengali HS has been severely limited [15].

The use of vulgar language is influenced by sociocultural context and demographic factors [16], and exploring its prevalence in languages other than English [17] is important. As more people express their thoughts online via social media, HS has continued spread. Given that discriminatory speech may detrimentally impact society, the development of detection and prevention systems can benefit governments and social networking sites. In this study, we explore a solution to the problem of HS detection and prevention in Bengali text by providing a thorough summary of field research.

Collecting and preparing data, extracting features, constructing and deploying models, and obtaining results are all processes involved in detecting HS. The general steps for identifying HS in Bengali text are shown in Fig. 1 and explained in "Hate Speech Detection Methodologies" section. Note that there may be variations to these steps, and the framework may not always be the same.

- Data Collection: The first step in HS detection is data collection. Text, image/graphic, audio, and video datasets are the primary Bengali datasets used for HS detection and classification. Text datasets are mainly employed for HS detection because they are easily accessible through social media posts and comments, and social media users use text more often than other types of data.
- Data Preprocessing: The data preprocessing step includes labeling and cleaning data, as well as eliminating tokens, hashtags, emojis, and other irrelevant information from raw data. The quality and interpretability of the original data are enhanced, and the ability to detect relevant information is improved through data preprocessing.
- Feature Extraction: The large amount of raw data collected above is separated and reduced to more manageable categories through feature extraction processes, significantly simplifying the procedure. Several feature extraction approaches can be utilized in this phase, e.g., the TF-IDF, CountVectorizer, bag of words, n-grams, Word2Vec, FastText, word embedding, and part-of-speech tagging approaches.
- Model Training: Machine learning is used to train a model that accurately recognizes HS in Bengali. However, choosing an appropriate model can be arduous. Researchers have utilized several different models in HS detection studies. One can employ clas-

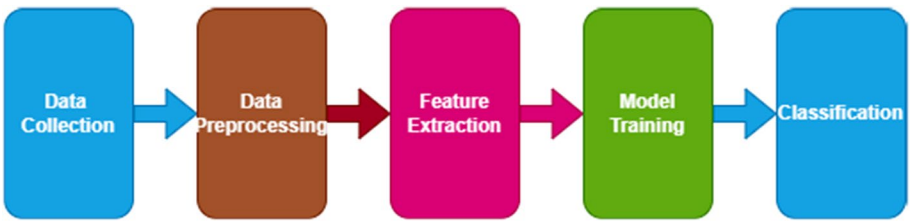

Fig. 1 General steps for HS detection in Bengali text

sical machine learning (ML), deep learning (DL), hybrid (a combination of classical ML and DL), and ensemble models to detect HS in Bengali text.

- HS Classification Using the Model: Classification involves partitioning a set of data into groups with related traits. After training the model, researchers can carry out a classification to detect HS in real Bengali text. In this step, text can be classified into labels, e.g., "HS", "Not HS", or "Neutral", and HS can be identified easily through automation.

In this study, we conduct a systematic review of the use of cutting-edge technology to identify HS in Bengali on online social networking sites. We examine the most popular and widely used natural language processing (NLP) methods that have helped to identify HS automatically. We also discuss the findings of various related research works and their limitations. Finally, we provide ideas and recommendations for additional research while examining the current issues. The overall contributions of our study are as follows:

1. To our best knowledge, this is the first comprehensive survey on hate speech detection in the Bengali language. While some surveys on HS detection research in English have been published [7, 18, 19], there are no existing surveys on HS detection in Bengali. In our survey, we carefully review the available research articles from academic journals/conferences, datasets, and other resources for the purpose of detecting hate speech in Bengali.
2. The aim of this survey is to investigate various machine learning (ML) and deep learning (DL) methods for the detection of HS in Bengali. We have developed a comprehensive taxonomy of the ML and DL methods employed for Bengali HS detection. We have also outlined the basic architectures that are used and performed a comparative study to clarify the advantages and disadvantages of various approaches.
3. We thoroughly review text preprocessing techniques, including a range of tools and packages to preprocess text for subsequent learning tasks. We also explore feature extraction techniques, explaining their models and architectures, and summarize the benefits and drawbacks of each.
4. We have carefully described the current restrictions and challenges in detecting hate speech in Bengali and outlined their possible solutions. In particular, we investigate the topic of aspect-based hate speech analysis, which could be transformative in the area of HS detection. However, this approach is not well-known to researchers in the context of Bengali text. We offer a brief explanation of aspect-based hate speech detection, highlighting its subtleties and suggesting potential approaches for its successful deployment.

The components of this study are visualized in Fig. 2.

This survey is designed for three primary groups: NLP researchers, computer scientists and engineers, and policymakers. NLP researchers are expected to provide insights into the theoretical and research challenges associated with detecting hate speech in Bengali text, including the nuances of language and the effectiveness of various models. Computer scientists and engineers are targeted for their expertise in the technical implementation, development, and optimization of algorithms and classifiers for hate speech

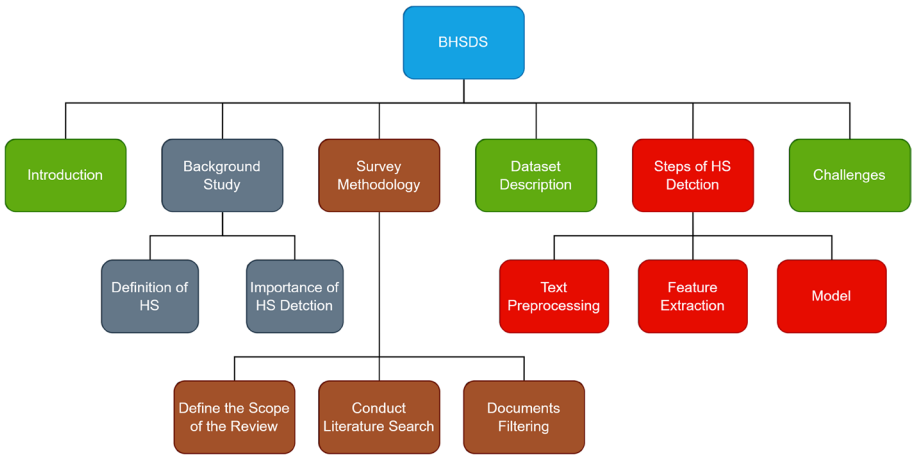

Fig. 2 Components of this study

detection, focusing on practical challenges and performance issues. Policymakers are included to address the broader societal impact, ethical considerations, and regulatory implications of deploying automated hate speech detection tools. Clearly defining these groups in the introduction helps tailor the survey questions to each audience's specific interests and expertise, ensuring relevant and actionable feedback that can guide future research, technical improvements, and policy development.

Background

In the realms of ML, NLP, and data science, HS detection is a promising area of study. While several surveys have been published on this topic, none have focused explicitly on the Bengali language. However, notable studies have been conducted in other languages, such as English and Arabic, providing valuable insights into HS detection techniques. Following are a few examples, by no means complete, of HS detection research in other languages.

- English: In a recent study, researchers provided a short and analytical survey of HS detection in NLP, specifically focusing on the English language [7]. They began by defining essential terms for studying HS and analyzed the commonly used features in this field. They also explored research on bullying and its applications, particularly in predicting social unrest. Finally, the authors addressed data and categorization issues and presented various approaches to address these issues.
  In another survey [19], the authors critically analyzed HS, mainly focusing on "cyber hate" in social media and on the internet (primarily for the English text). By examining the nature and impact of this phenomenon, the authors emphasized the need for proactive measures to combat HS online, urging the active involvement of social media platforms to create a safer and more inclusive digital space.
  In another article [18], the authors provided more precise definitions of HS and offered an overview of the field's development in recent years. They adopted a systematic method to analyze existing data collections that is more comprehensive than

previous methods. This study highlights the increasing awareness of the spread of HS through networks in Arabic regions and worldwide. Many countries are now actively working to regulate and counter such speech.

Some of the other recent surveys for HS detection in English are [20–23].

- Arabic: In the context of Arabic social media platforms, several studies have been conducted to detect offensive language and HS. One study utilized a multitask learning (MTL) model trained on diverse datasets representing different types of offensive and hateful text. The developed MTL model outperformed existing models and achieved superior performance in detecting Arabic offensive language and HS [24].

  In another innovative approach, a DL framework was proposed that combines a CNN and LSTM network to automatically detect cyber HS on Arabic Twitter. This approach, which employs word embeddings, yielded positive outcomes in terms of accurately classifying HS tweets using different evaluation metrics [25].

  Similarly, the effectiveness of a convolutional neural network (CNN), a CNN with long short-term memory (CNN-LSTM), and a bidirectional LSTM (BiLSTM)-CNN was explored in terms of automatically detecting hateful content on social media using the Arabic Hate Speech (ArHS) dataset, which consists of 9,833 annotated tweets. Different types of experiments were conducted, addressing binary, ternary, and multiclass classification for different categories of HS [26].

  Furthermore, another study focused on developing an automated system to detect HS in Arabic, aiming to monitor online behavior for threats to national security and cyberbullying. Deep recurrent neural networks (DRNNs) were employed for HS classification, utilizing a unique dataset of 4,203 comments categorized into seven HS classes [27].
- Danish: A Danish dataset capturing offensive language from platforms such as Reddit and Facebook was created, and four automatic classification systems were developed for both the English and Danish languages [28].
- Hindi: Similarly, the growing problem of online HS and the need for automatic detection methods were emphasized, explicitly focusing on code-mixed Hindi-English datasets. This study surveyed advancements in neural-based models designed to address HS in this context [29].

  Moreover, the issue of HS in user-generated social media content was addressed by focusing on code-mixed social media texts. A code-mixed dataset was used to assess two architectures for HS detection: a subword-level LSTM and a phonemic subword-level hierarchical LSTM with attention [30].
- Urdu: In another study, a vocabulary of hostile words and an annotated dataset named RUHSOLD were used for automatic HS and offensive language detection in Roman Urdu (RU). It was concluded that transfer learning is effective, and a CNN-gram model that exhibited greater robustness compared to that of baseline approaches was proposed [31].
- Bahasa Indonesia: In Indonesia, where limited research on HS detection exists, researchers aimed to create a new dataset encompassing various aspects of HS, targeting religion, race, ethnicity, and gender. A preliminary study was conducted to compare machine learning algorithms and features. It was found that word n-grams performed better than character n-grams in detecting HS [32].

Overall, these studies and surveys shed light on HS detection techniques in various languages and highlight the significance of addressing this issue to foster a more inclusive and respectful online environment. While research in HS detection in the Bengali language is still limited, the insights gained from the studies conducted in other languages can serve as a foundation for future research and development in this area.

Definition of hate speech

The definition of HS lacks a universal consensus, leading to disagreements among parties, organization, and individuals. Detecting HS is a challenging task. HS is characterized by language that uses stereotypes to express a hateful mindset. To annotate a corpus and create a consistent language model, addressing numerous difficulties related to defining HS is essential [33].

Some organizations and authors have defined HS as follows:

1. European Union: "Any public incitement to roughness or abomination against an individual or a member of a group of individuals characterized by their race, color, religion, ancestry, nationality, or ethnicity" [34].
2. International Minorities Association: "Any criminal activity that targets individuals because of their actual or perceived membership in a certain group is considered a hate crime. The crimes can take many different forms, including rape, property destruction, blackmail, physical and psychological intimidation, and hostility and violence. A language that disparages or insults a group on the basis of their race, ethnicity, religion, disability, gender, age, sexual orientation, or gender identity" [35].
3. Facebook: "We define hate speech as an outright attack on an individual based on one or more of the categories we refer to as protected characteristics, such as race, ethnicity, national origin, disability, religion, caste, sexual orientation, sex, and significant medical conditions. Attacks include what we define as aggressive or dehumanizing rhetoric, damaging stereotypes, claims of inferiority, expressions of scorn, disgust, or dismissal, profanity, and calls for segregation or exclusion. When used in conjunction with another protected trait, age is regarded as a protected characteristic. Although we permit opinion and criticism of immigration laws, we also shield refugees, migrants, immigrants, and asylum seekers from the harshest assaults. In a similar vein, we offer certain safeguards for traits like occupation when they are mentioned with a protected trait" [36].
4. Twitter (now X): "You aren't allowed to incite or harass others based on their race, ethnicity, national origin, caste, sexual preference, gender, or gender identity, their age, handicap, or a serious illness" [37].
5. YouTube: "We eliminate material that calls for harm to people or groups based on any of the following characteristics: Age, caste, handicap, ethnicity, gender expression, nationality, race, immigration status, religion, sex/gender, and sexual orientation, as well as veteran status and those who have experienced a significant act of violence and their family members, are also taken into consideration" [38].

After analyzing the terms used in the aforementioned definitions to comprehend the meaning of HS more clearly, we find that each definition emphasizes that HS targets a

specific entity, such as a person, group, or nationality. The majority of these definitions also address themes of violence, racism, and gender discrimination and issues related to race and ethnicity. Less common standards for measuring hate speech include the use of profane language and issues related to disability, property damage, advanced age, and significant illness. Table 1 presents common hate speech categories and a few examples of hate targets in each category.

Importance of HS detection

In recent years, HS has attracted the attention of researchers and has become popular in the NLP field. We specify why automatic HS detection is essential below.

- Social Media Safety: Detecting HS is essential to ensure the safety of social media users. HS can be defined as any form of speech that targets and dehumanizes a particular group based on factors such as race, ethnicity, religion, sexual orientation, and gender identity [24]. The consequences of HS can be severe, ranging from promoting violence and discrimination to potentially inciting genocide. In addition, HS can harm and threaten individuals and communities who are already marginalized and vulnerable [40]. An investigation was conducted in five Bangladeshi areas in 2020 by "Ain O Salish Kendra" (ASK), a rights and legal assistance NGO. They discovered that during the COVID-19 pandemic, many young students experienced online harassment. In particular, an alarming 30% of the 108 children, 61 girls and 47 boys, polled said they had experienced internet harassment [41].
  We can assist in defending these groups and fostering a safer online environment by identifying and preventing HS. HS has the potential to undermine civic dialogue, fostering prejudice, polarization, and division. Detecting and suppressing HS can play a crucial role in nurturing civil discourse and encouraging productive online discussions. HS often infringes upon human rights, including freedom of expression, non-discrimination, and equality. While information sharing has become more accessible, the rise of cyberbullying poses a growing concern. Moreover, the harmful effects of cyberbullying on children were discussed in one another, and it was demonstrated in another study that victims of such behavior are more likely to consider suicide than non-victims [42]. Upholding fundamental rights and preventing the spread of harmful and discriminatory content is crucial for creating a safe social media environment.

Table 1 Hate speech categories and examples of hate targets in each category [39]

Category	Examples of hate targets
Race	African American people, Black people, White people
Physical	Fat people, Ugly people, Beautiful people
Class	Ghetto people, Poor people, Rich people
Disability	Deaf or dumb people, Autistic people, Mentally ill people, Bipolar people
Religion	Jewish people, Atheist people
Sexual orientation	Gay people, Lesbian people, Homosexuality
Other	Annoying people, Drunk people, Drug-addicted people

Conversely, social media platforms that allow HS to thrive risk damaging their reputation and losing users. By actively detecting HS, platforms can demonstrate their commitment to fostering a safe and respectful online environment and thereby attract more users. One of the most significant challenges in detecting HS is that it exists in diverse forms, including subtle and implicit expressions [2]. Traditional methods for detecting offensive language, such as keyword-based filtering or manual moderation, often need to be revised to capture all instances of HS. Social media platforms can implement additional measures to enhance the safety. These measures include giving users more control over their feeds, creating reporting mechanisms that allow users to report HS more efficiently, and enforcing strict community guidelines.

- Mental Health: HS promotes violence and erodes social harmony and tolerance, and history is replete with examples of the destructive effects of hatred. Online platforms have quickly evolved into forums for divisive and hateful statements on a global scale, endangering peace and harmony. In today's digital world, HS can escalate, exerting a much more significant influence than ever before.
- Hate Crime Prevention: Hate crimes are those in which the victim is targeted because of their actual or perceived race, color, religion, disability, sexual orientation, or national origin [43]. To gather data, the United States Federal Bureau of Investigation (FBI) defines a hate crime as a "criminal offense against a person or property motivated in whole or in part by an offender's bias against a race, religion, disability, sexual orientation, ethnicity, gender, or gender identity" [44]. By decreasing the rate of HS, the rate of hate crimes could also be reduced.

Survey methodology

The method used to conduct this survey is the "systematic literature review" (SLR) method developed by Kitchenham and colleagues [45, 46], which involves the comprehensive identification, critical evaluation, and systematic interpretation of all existing research studies that are pertinent to the specific research question, topic area, or phenomenon under investigation. The processes involved in conducting an SLR are categorized into three distinct phases: defining the scope of the review, conducting the literature search, and filtering the documents. Each of these phases will be discussed in more detail in the following sections.

Defining the scope of the review

The first step in this study involved determining the extent and boundaries of the review, including the specific aspects of Bengali text hate speech detection that we wanted to cover, e.g., the types of hate speech, the methods used for detection, the datasets used for training and evaluation, and the performance metrics). After selecting the inclusion and exclusion criteria, we searched for the most relevant papers, articles, and studies that would be suitable for the survey. Our focus was primarily on the ACM, IEEE, Springer, Elsevier, and arXiv databases. Once we completed this step, we began the study.

Research questions

The fundamental research questions that we composed are as follows:

- What role does hate speech detection play in the digital sphere?
- How does its effectiveness impact the reduction of online toxicity and the promotion of welcoming online spaces?
- What are the linguistic indicators or criteria frequently used in the identification and detection of hate speech within diverse contexts?
- How do various organizations and councils define hate speech?
- What kinds of datasets are frequently used to identify HS in Bengali text?
- What linguistic resources and features are commonly employed in creating models that effectively identify HS in Bengali text?
- Which are the current approaches frequently used for feature extraction, classification, and preprocessing in text-based Bengali HS detection?
- How are HS detection systems generally assessed in light of the data's various linguistic and cultural contexts?
- How well do the models used for Bengali HS detection perform?
- What are the obstacles and constraints faced in Bengali HS detection today?
- In what creative ways can these problems be solved to overcome the challenges they face?
- What directions might future researchers explore to improve HS detection systems' efficacy?

The above research questions are addressed in "Background", "Dataset Description", "Hate Speech Detection Methodologies", and "Challenges" sections of this survey.

Conducting the literature search

To conduct the literature search, we posed a primary research query: "What are the current techniques and approaches utilized for recognizing hate speech in the Bengali language, as well as the important datasets utilized by the research community?" To answer these questions, we thoroughly explored the existing literature using multiple academic databases, such as the ACM Digital Library (which also indexes papers from Association for Computational Linguistics, ACL), IEEE Xplore, SpringerLink, ScienceDirect (Elsevier), arXiv, and Google Scholar. To collect papers for this study, we used keywords including "Bengali hate speech," "machine learning," "deep learning," "natural language processing," "Bengali text classification," "Bengali hate speech detection," "AI-based hate speech detection in Bengali" on Google Scholar. These keyword searches allowed us to identify some of the critical research articles and reviews on this topic.

An exact search was conducted on specific repositories such as IEEE, ACM, Springer, Elsevier, and arXiv. However, we obtained fewer articles due to their indexing methods. For example, our search on IEEE Xplore only returned a few papers, of which only a small fraction was relevant to this work. A similar outcome was observed on Springer, which also yielded only a few articles.

On the other hand, Google Scholar was found to be more inclusive and accessible, facilitating the discovery of all relevant content in one result. As a result, this platform was used for this study. Next, we used specific search terms to narrow the search results further. Finally, we searched for articles that cited the critical research articles identified during the literature search to identify additional relevant articles.

Research type

The research type specifies the category of written materials, which may include scholarly publications, conference or workshop papers, book chapters, or dissertations. The total number of journals and conference papers published in this field and used in our survey are shown graphically in Fig. 3.

Publication year and type

At the outset of this study, 150 papers were collected from various sources, and 38 papers were selected for the survey. Over 90% of these papers were published from 2010 to 2023. Consequently, we incorporated more recent articles to revise and augment this review.

Document filtering

Inclusion and exclusion criteria are typically used to determine whether participants should be included or eliminated in a study. These criteria aid researchers in ensuring that the participants used in their study are representative of the population under investigation and that they can provide relevant insights into the research topic. After applying inclusion and exclusion criteria to this study, we found that many of the research articles we retrieved were relevant to this study. The following inclusion criteria were employed:

- The paper is only from a peer-reviewed journal, conference proceeding, or arXiv preprint.
- The full text is available in a digital database.
- The paper includes the proposal of a model, various machine learning or deep learning techniques, or a framework.

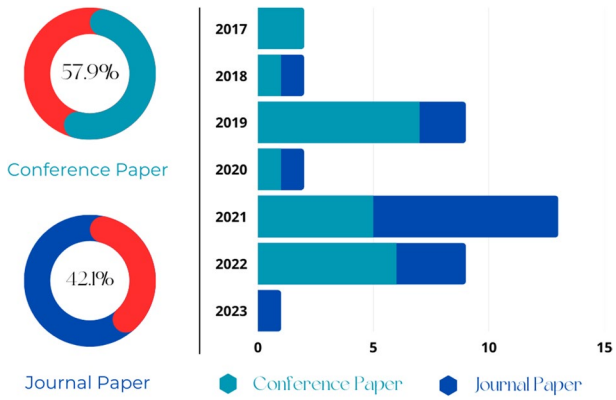

Fig. 3 Year-wise published articles

- The paper is written in English and focuses on Bengali hate speech detection.

In contrast, papers that were duplicates, appearing in multiple academic databases, reviews, book chapters, magazine articles, theses, and interview-based articles, not relevant to Bengali hate speech detection or did not have full-text access were excluded from our study.

We followed the "Preferred Reporting Items for Systematic Reviews and Meta-Analyses" (PRISMA) protocol [47], as depicted in Fig. 4.

Dataset description

In this section, we discuss the sources of data for Bengali HS detection. We found that each of the included papers focuses on specific or several related themes, such as politics, religion, or other relevant topics, when collecting data. Furthermore, we analyzed how data was extracted from these sources. Finally, we provide a table that summarizes the datasets used by various authors. Table 2 includes information on the dataset source, the data collection method employed, and the data size and availability.

Data sources

The effectiveness of HS detection models relies on the quality and diversity of the datasets utilized for training and evaluation. In this section, we present an overview of the data sources used in the studies reviewed in this survey. Most studies reviewed in our survey relied on social media platforms as their primary data source for detecting HS in the Bengali language. Facebook was the most frequently utilized among these platforms, followed by YouTube. A significant portion of the studies utilized Facebook [41, 48, 52, 57, 58, 60, 63, 64] as the main data source, while in a few works [5, 14, 40], Facebook comments were collected along with tweets.

Several studies utilized datasets of comments extracted from Bangladeshi news websites for training their HS detection models [2, 5, 14], in addition to comments from Facebook and YouTube. Users' comments on the Facebook pages of newspapers were collected in [65], and [15, 49, 53, 55] used comments from controversial YouTube videos uploaded on Facebook.

Publicly available datasets were used in other studies [50, 54, 59, 61, 66]. In contrast, some researchers created their own datasets by collecting data from existing datasets [15, 48, 49, 52, 58, 67]. In one study, two Bengali corpora consisting of 7,245 reviews

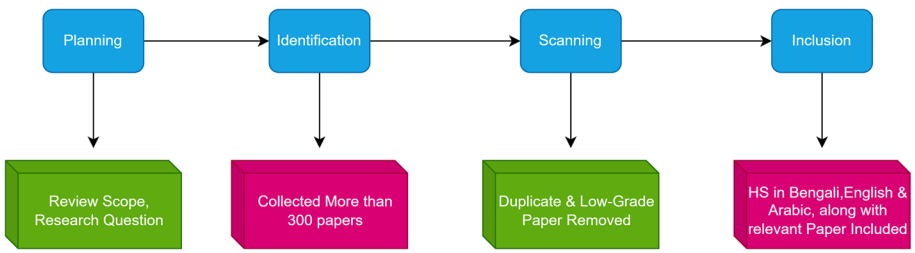

Fig. 4 PRISMA protocol for the survey methodology

Table 2 Dataset overview

Refs.	Source	Size	Collection method	Publicly available?
[48]	Facebook	1339	Web Scraper	Not Available
[49]	YouTube Facebook	30,000	FacePager3 (Open-Source Software)	Available
[50]	Facebook YouTube Newspapers	5000	Manual	Available
[51]	Facebook YouTube	15,000	Beautiful-soup1	Not Available
[52]	Facebook	44,001	Manual	Not Available
[53]	Facebook YouTube	50,314	FacePager3	Not Available
[54]	Facebook YouTube Newspapers	34,500	–	Not Available
[15]	Facebook YouTube	50,281	FacePager4	Available
[55]	Facebook YouTube	12,028	Graph API YouTube Scrapper	Not Available
[14]	YouTube Newspapers	4700	Language Detector	Not Available
[56]	Facebook YouTube	2000	exportcomments.com	Not Available
[57]	Facebook	5644	Manual	Available
[58]	Facebook	5126	FB Graph API	Not Available
[59]	GitHub	10,219	Existing Dataset	Available
[5]	Newspapers Facebook Twitter	1998	Manual	Not Available
[60]	Facebook	7425	Graph API	Available
[17]	YouTube	7245	Graph API Manual	Available
[61]	Facebook YouTube	30,000	Existing Dataset	Available
[62]	Twitter	10,000	Twitter API	Available
[63]	Facebook	7000	Instant Data Scraper Software	Available
[41]	Facebook	10,133	Manual	Available

and comments obtained from YouTube were created. These datasets were made available to the public for further use and analysis [17].

In other studies, diverse data collection methods were utilized. Some of these methods include keyword-based searches or targeted crawling of specific accounts [52, 58, 60, 65, 68]. In contrast, some studies focused on particular types of HS, such as religious, political, racist, and sexual HS [14, 41, 51, 61].

The widespread utilization of social media platforms as the primary data source highlights the need for developing methods to handle social media data's dynamic and noisy nature for HS detection in the Bengali language. The distribution of data sources for the collected datasets is presented in Fig. 5 in graphical form.

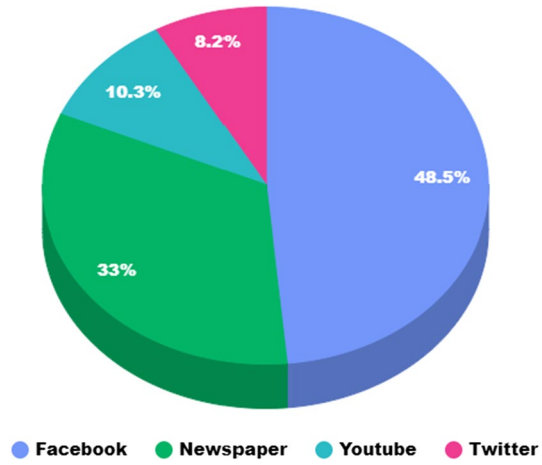

Fig. 5 Statistics of the platforms used for data collection

Data categories

The detection and classification of HS in the Bengali language is an essential area of research due to the limited availability of data compared to that of other languages, such as English. Most of the existing Bengali datasets were manually prepared, and data were grouped into multiple categories based on different criteria. Data collection for HS detection research has primarily been conducted through social media platforms such as Facebook, YouTube, and Twitter. Researchers have collected data based on various categories, including politics, religion, sports, entertainment, crime, memes, and TikTok videos [49].

The number of dataset categories varied across studies. However, use of five to seven categories was most common [15, 49, 53]. While the categories differed between authors, categories related to politics, religion, and celebrities were commonly included [50]. The results of these studies indicated that HS is more prevalent in specific categories than others. For example, HS related to religion and sports had higher occurrence rates than those of other categories. Categorizing data based on different classes provided valuable insight into the nature and prevalence of HS in Bengali.

- Politics: In the context of political HS, derogatory language is often used to attack political opponents based on their political ideology, party affiliation, or personal characteristics. The following is an example of HS in Bengali related to politics: "যারা শুধু আবার বাংলাদেশের প্রতিদুর্নীতিচিালানোর চেষ্টা করে তাদের হঠাৎ ধরে মার দিতে হবে‡" (Those who are attempting to defame Bangladesh should be suddenly caught and beaten up) [58].
- Religion: In the context of religious HS, people often use derogatory language to insult and discriminate against people of different faiths or sects. An example of HS in Bengali related to religion is as follows: "মুসলমানরা হৃদয়হীন ও জাহিল" (Muslims are heartless and ignorant).
- Sports: In the context of sports-related HS, fans of opposing teams often use derogatory language and insults to belittle each other. An example of HS in Bengali related

to sports is as follows:"ভারতীয়পাগলদে র বি রুদ্ধে জয় হবে ইনশাআল্লাহ" (Inshallah, we will win against the crazy Indians).
- Entertainment: HS related to entertainment can include derogatory remarks and personal attacks against actors, actresses, and other celebrities.
- Crime: HS related to a crime can include derogatory remarks and personal attacks against individuals accused of committing crimes.
- Memes: Memes in Bengali may contain derogatory remarks and HS against certain groups or individuals.

Data collection methods

There are various methods available to extract data, which refers to the process of collecting text data from social media sources. Different authors have used different approaches to extract data. Subsequently, the extracted data instances are annotated with different class labels (e.g., "HS", "Not HS", or "Neutral") in preparation for supervised learning.

Data extraction

The most commonly used data extraction methods are described below.

a. FacePager: FacePager is an open-source software tool designed to facilitate data collection from websites and social media platforms such as Facebook, YouTube, and Twitter by leveraging APIs and web scraping methods. This tool offers a user-friendly interface that enables researchers to retrieve data based on specific search criteria, such as hashtags, keywords, or user profiles. Researchers can use FacePager to define extraction projects focused on collecting Bengali text data linked to HS. By setting search parameters, individuals can extract posts, comments, and other pertinent data from social media sites. This application controls pagination, automates querying, and stores retrieved data in an SQLite database that can be exported to CSV format for further analysis.

For example, the authors in [15, 49, 53] utilized FacePager to collect public comments related to Bengali HS from Facebook and YouTube. Although FacePager facilitates data collection, it is important to remember that manual setup and management are still required. Users are accountable for defining search criteria and specifying the data they wish to extract.

b. Web Scraping: Web scraping tools are software programs or libraries designed to automate the process of extracting data from websites. This approach involves extracting data from web pages, social media platforms, and other online sources [69].

For example, a researcher may scrape data from social media sites such as Facebook, Twitter, or Instagram to build a dataset for HS detection in Bengali [48]. To scrape HS data from websites and social media platforms, researchers need to use web scraping tools that are designed to work with these specific platforms. Several web scraping tools, such as BeautifulSoup, Selenium or Scrapy, extract and filter text data based on predefined criteria such as hashtags, keywords, or user profiles. Researchers also can utilize APIs to obtain HS datasets from various social media platforms. There are a number of studies in which the authors used web scraping for HS data collection [70–72].

- BeautifulSoup: BeautifulSoup, a Python library, extracts data from HTML and XML documents and can be used to facilitate web scraping activities on various websites, including social media platforms such as Facebook, Twitter, and Instagram [2]. To perform data scraping with BeautifulSoup, the website(s) from which one wants to scrape HS data must be identified. Next, one must inspect the website(s) to understand the HTML structure of the web pages containing HS data. Then, BeautifulSoup it used to extract the relevant HTML elements containing HS data. The extracted data must be cleaned and preprocessed to remove unwanted characters, formatting, or duplicates and analyzed to identify trends, patterns, and insights. The HTML content of the website containing HS data is retrieved using the requests library, and then BeautifulSoup parses the HTML content and finds all HTML elements that include HS data. The text content of each HS element is then extracted and stored in a list for further analysis.
- Scrapy: The scrapy tool is a Python framework for web scraping that provides tools and libraries for building web scrapers. Scrapy can scrape data from websites, including social media platforms. First, the URLs or APIs that contain the HS data one wants to extract for each platform are identified, and then a new Scrapy project and spider are created for each platform [73].
- Selenium: Selenium is a tool that automates web browsers, allowing developers to simulate user interactions with websites, which helps to scrape data from websites and social media platforms [74].

It is important to ensure that the web scraping procedure conforms with the terms and conditions of the website and respects user privacy and rights when utilizing web scraping technologies to gather HS data from social media platforms. To eliminate false positives and ensure that the data are trustworthy and pertinent to the research issue, data preprocessing and filtering may be needed.

c. Social Media APIs: Application programming interfaces (APIs) of social media platforms such as Twitter, Facebook, and Instagram can be used to collect Bengali text data related to HS. Several Python libraries can extract HS data from social media platforms such as Facebook, Twitter, and Instagram. For instance, the official Facebook Graph API can extract data from Facebook pages, groups, and profiles.

An API provides access to various data, including posts, comments, likes, reactions, and user profiles, which can be filtered to extract data relevant to HS [60]. Libraries such as Facebook-SDK, Facebook-scraper, social media API, and PySocialWatcher can interact with the Facebook API. Facebook-SDK is a Python wrapper for the Facebook Graph API that allows the user to authenticate requests and retrieve data from Facebook's platform. One can use this library to extract posts, comments, and other text data from Facebook pages and groups and apply NLP techniques to analyze the text for HS.

PySocialWatcher is a Python library for monitoring social media platforms, including Facebook, for potentially harmful content, such as HS. This library uses machine learning algorithms to analyze text data and flag posts and comments that contain HS or other problematic content. In contrast, Facebook-scraper is a Python library for scraping Facebook pages and groups to extract data such as posts, comments, and reactions. Using NLP techniques, one can use this library to filter data using specific keywords or

hashtags and analyze text data for HS. Moreover, the official Twitter API can extract data from Twitter, including tweets, user profiles, and other metadata. This API provides access to various endpoints, such as search APIs and streaming APIs, which can be used to extract relevant data.

Libraries such as Tweepy, Twython, and Python-twitter can be used to interact with the Twitter API [75, 76]. Tweepy is a Python library that provides convenient access to the Twitter API, offering an easy-to-use interface for sending requests to the Twitter API, handling rate limits, and interacting with Twitter data. With Tweepy, one can easily authenticate an application, search for tweets, retrieve user information, post tweets, and more. Tweepy is widely used for social media analysis, sentiment analysis, and other data science applications involving Twitter data. Python-twitter is a collection of Python scripts and libraries for accessing the Twitter API, providing a simple and flexible interface for working with Twitter data.

Moreover, Instagram Graph API can extract data from Instagram accounts, hashtags, and locations. This API provides access to various data, including posts, comments, likes, and user profiles, which can be filtered to extract data relevant to HS [77]. Libraries such as Instaloader and Instagram API can be used to interact with the Instagram API.

Installer is a third-party Python library for downloading and processing data from Instagram. This library provides a user-friendly interface for effortlessly downloading images, videos, and metadata from Instagram. The collected data can be filtered based on relevant keywords and hashtags and then preprocessed and labeled for training an HS detection model [78].

Hate speech annotation

HS annotation is largely a manual process. It typically involves a team of human annotators who read text and make a determination regarding whether the text contains language that is meant to denigrate, intimidate, or incite violence against a particular individual or group based on their race, religion, gender, sexuality, or other characteristics. Manual labeling of Bengali text for HS detection was reported in [15].

One could hire human annotators who are proficient in the Bengali language to identify Bengali HS. The annotators would read through text and label it as HS or non-HS based on predetermined criteria [79]. A research team may select a set of Bengali text documents, such as social media posts, news articles, or online comments, and have a team of expert annotators manually label them. The criteria for labeling could include explicit HS terms, offensive language, discriminatory language, or threatening language [80].

Crowdsourcing is another manual data collection and labeling method. Crowdsourcing platforms have been utilized to annotate Bengali text data for HS [81, 82]. A researcher can post annotation tasks on a platform such as Amazon Mechanical Turk [83] or CrowdFlower [84] and receive annotations from many annotators. However, the annotation quality may be inconsistent, and it may be challenging to ensure the label accuracy.

Active learning Active Learning (AL) is a sophisticated method of data annotation that leverages model uncertainty to prioritize and select the most informative samples for labeling. Rather than passively annotating data, active learning actively involves the model in identifying which data points, when annotated, would most enhance its performance. It is a machine learning strategy that improves the process of collecting data by iteratively choosing the most useful examples to be annotated. It is also known as query learning. AL involves minimizing the quantity of labeled data needed to understand the target variable. To accomplish this, the user is asked to provide labels for the most valuable instances, resulting in a more efficient learning process with fewer examples [85].

Traditional data gathering approaches typically require substantial amounts of annotated data to train supervised ML models. Whereas AL focuses on querying samples that the model finds most uncertain or complex [86]. This method entails training an initial model using a limited amount of labeled data. The model is then used to identify data points where its predictions are least certain. Human annotators are then employed to provide labels for these data points [87]. By incorporating these newly labeled data points into the training set and retraining the model, learning becomes faster and performance increases.

Studies [88, 89] used the active learning approach for data annotation. Bengali is a low-resource language that lacks a substantial amount of data corpora. AL can be utilized to develop a domain-specific Bangla data corpus. Another study [90] adopted the AL technique to expand their labeled samples and automatically detect hate speech regarding Rohingyas (refugees in the border areas with Myanmar).

Hate speech detection methodologies

In "Introduction" section, we discussed some of the general steps in HS detection. In this section, we provide a detailed explanation of every step.

Text preprocessing

We discuss the most commonly used text preprocessing methods below.

Stemming: Stemming in Bengali text preprocessing involves reducing Bengali words to their basic form. This process involves eliminating common suffixes from Bengali words to derive their stem, a helpful technique to minimize the vocabulary size and improve the efficiency of NLP tasks such as text classification, sentiment analysis, and information retrieval. However, stemming in Bengali is a difficult task due to the complex morphology of the language.

For example, the Bengali word "করতে‡" can be stemmed to "কর" by removing the suffix "তে‡". Similarly, the word "পরে র" can be stemmed to "পর" by removing the suffix "এর". Stemming can also help to reduce dataset redundancy, making the dataset easier to analyze and interpret. In contrast, stemming may lead to the loss of information as it reduces words to their root form, resulting in the loss of nuance and context [54].

Lemmatization: Lemmatization is a technique used in Bengali text preprocessing to identify a word's base form (lemma) by considering its inflectional and derivational morphology. Unlike stemming, which involves reducing words to their root form by

removing common suffixes, lemmatization leads to more accurate results by considering the sentence's grammatical structure.

For example, the Bengali word "লিখেছিলেন" can be lemmatized to "লিখা" by identifying its base form and considering its inflectional morphology (the suffix "ছিলেন"). Similarly, the word "চলছে" can be lemmatized to "চলা" by identifying its base form and considering its inflectional morphology (the suffix "ছে"). There are different approaches to lemmatization in Bengali text preprocessing, including rule-based and ML-based approaches. Rule-based approaches rely on hand-crafted rules to identify the base form of a word, while ML-based approaches use statistical models trained on large corpora of Bengali text to achieve this task.

Normalization: Normalization in Bengali text preprocessing refers to converting text into a standard or canonical form that computer algorithms can easily process [91]. This preprocessing method involves several techniques, such as Unicode normalization, punctuation removal, digit normalization, and case folding. Normalization aims to ensure consistency and reduce the variability of the input text, thereby improving the accuracy of natural language processing tasks.

Tokenization: Tokenization in Bengali text preprocessing refers to breaking down a piece of text into smaller units, known as tokens, such as words, phrases, or symbols [55], that computer algorithms can efficiently process. The goal of tokenization is to simplify the text and enable more effective NLP tasks, such as information retrieval, text classification, and sentiment analysis.

For example, consider the following Bengali sentence: "আমি বাংলাদেশে থাকি।" (I live in Bangladesh). In word tokenization, this sentence is broken down into individual words: "আমি ", "বাংলাদেশে ","থাকি ".In phrase tokenization, this sentence is broken down into phrases: "আমি বাংলাদেশে ", "থাকি ". In symbol tokenization, any symbols in the sentence, such as punctuation marks or emojis, are identified and treated as separate tokens.

Stop Word Removal: Stop word removal is a common technique used in Bengali text preprocessing that involves removing commonly used words, known as stop words, from the input text. These words are usually short, function words that do not carry much meaning and are commonly used in natural language.

Examples of some stop words in Bengali include "একটি" (a), "একটা" (one), "একজন", "তার" (her/his),"সে " (he/she), and "এবং" (and), "যে " (which), "করে " (do), "করতে " (to do), "সব" (all), "বা" (or), "কিছু"(some), "না" (no/not), "আমরা" (we), and "তুমি " (you).

For example, in the sentence "আমি একজন শিক্ষক।" (I am a teacher), the stop words "একজন" (oneperson) and "এক" (one) would be removed during the stop word removal process, as they do not convey much information on their own. The goal of stop word removal in Bengali text preprocessing is to simplify the text and improve the accuracy of NLP tasks [57]. By removing commonly used stop words, the vocabulary size and noise are minimized, enabling a more effective analysis of the meaningful words in the text.

Stop word removal is typically performed after tokenization, as it involves identifying and removing specific tokens from the input text. Stop word lists are commonly used for Bengali text preprocessing. These lists include a predefined set of stop words that are commonly used in Bengali text. In addition, these lists can be customized

based on the specific context or domain of the text to ensure that relevant stop words are removed from the input text.

Emoticons and Punctuation Marks: Emoticons and punctuation marks can be essential in Bengali text preprocessing, as they convey important information about the text's tone, sentiment, and intent. Emoticons are graphical representations of facial expressions commonly used in electronic communications to convey emotions or attitudes.

For example, in the Bengali sentence "আমি খুশি ! :-)", the exclamation mark and emoticon indicatea positive emotional tone. Punctuation marks, such as commas (,), periods (|), and exclamation (!)points, also play an essential role in Bengali text preprocessing. For example, in the sentence "আমিবাসায় রয়ে ছি |", the period indicates the end of the sentence.

During Bengali text preprocessing, emoticons and punctuation marks may be retained or removed depending on the specific task and context [59]. For example, emoticons may be useful in sentiment analysis tasks to help identify the emotional tone of text. At the same time, punctuation marks may be removed during tokenization to simplify the text for further analysis.

Hashtag Removal: Hashtag removal is a common preprocessing technique in Bengali natural language processing to remove hashtags from text data. Hashtags are words or phrases that are preceded by the symbol '#', which is commonly used in social media platforms such as Twitter, Instagram, and Facebook to categorize and organize content [2].

For example, in the Bengali tweet "#আমার_ভাল বো_লাগা_গান সে ই সুরে র মাঝে মাঝে মন খুলে যায়|" (I love the song "#আমার_ভাল বো_লাগা_গান," it opens my heart), the hashtag "#আমার_ভাল বো_লাগা_গান" can be removed during preprocessing to simplify the text to "সে ই সুরে র মাঝে মাঝে মন খুলে যায়|" (It opens my heart among those tunes). This process can help to focus on the sentiment expressed in the text without the distraction of the hashtag.

Removing hashtags during preprocessing can help simplify text and make it easier to analyze certain tasks, such as text classification, sentiment analysis, or topic modeling, where hashtags may not be relevant.

Duplicate Removal and Padding: Duplicate removal and padding are two text preprocessing techniques used in Bengali language text data to improve the data quality and reduce noise. Duplicate removal involves identifying and removing exact duplicate copies of text data that may exist within a dataset. This operation is important because duplicate text can unnecessarily increase the size of the dataset and lead to skewed results.

For example, consider the following Bengali text: "আমি বাংলা ভাল বোবাসি। আমি বাংলা ভাল বোবাসি। আমি বাংলা ভাল বোবাসি।" (I love Bengali. I love Bengali. I love Bengali.). After removing the duplicate text, we have: "আমি বাংলা ভাল বোবাসি।" (I love Bengali.). Padding, conversely, is a technique used to ensure that all text data are of equal length. This operation is important when working with machine learning models that require input data of uniform length.

Padding involves adding extra characters or spaces to shorter text data to make it the same length as that of more comprehensive text data. If we want to pad all text to the same length, we could add extra spaces to the end of the shorter sentence to match the length of the longest sentence in the dataset.

For example: Original text: "আমি বাংলা ভাল বোবাসি।"; Padded text: "আমি বাংলা ভাল বোবাসি। ". Here, we added extra spaces to the end of the sentence to match the length of the longest sentence in the dataset. This process ensures that all the text data is of uniform length, which is required for some ML models.

Feature extraction

Now, we discuss feature extraction methods.

CountVectorizer: CountVectorizer is a feature extraction method that converts a collection of text documents into a matrix of token counts. This method considers each document as a vector of token counts, where each element represents the count of a particular word (token) in a document. The matrix contains one row per document and one column per unique token in the entire corpus. CountVectorizer, which is essentially a bag of word statistics across documents, is a simple and efficient method that can handle large datasets and is relatively easy to interpret. This method can identify Bengali terms that appear more frequently in HS documents than in non-HS documents [5]. In one study, text was tokenized using CountVectorizer, which builds a lexicon of recognized terms, and a new document was encoded using this lexicon [48].

Researchers used the machine learning-based algorithm CountVectorizer [14] to identify offensive Bengali texts. Moreover, two different vectorization methods were utilized for the experiments in one research work; one method was CountVectorizer, which considers the frequency of features. This study demonstrated that an SVM with a linear kernel performed best in terms of accuracy while an SVM with a sigmoid kernel performed the worst [42].

Term Frequency-Inverse Document Frequency (TF-IDF): TF-IDF is a powerful statistical metric for evaluating the importance of words within a corpus or document. This measure provides a score that accounts for a term's rarity across the entire corpus as well as its frequency in a specific text, providing a thorough assessment of its importance [55]. The term frequency (TF) is derived by Eq. 1 [28].

$$\mathrm{TF} = \frac{\text{Words that appear frequently across a document}}{\text{Total number of words in the document}} \tag{1}$$

The inverse document frequency (IDF), on the other hand, is computed using the logarithm of the ratio of the total number of texts in the sample to the number of documents where a certain word appears [48].

$$\mathrm{IDF} = \log_2\left(\frac{\text{Total number of documents}}{\text{Number of documents with a specific term}}\right) \tag{2}$$

Finally, the TF-IDF may be formed by multiplying the TF by the IDF, which will have normalized weights.

$$\mathrm{TF} - \mathrm{IDF} = \mathrm{TF} \cdot \mathrm{IDF} \tag{3}$$

The TF-IDF feature extraction approach removes irrelevant features, which helps minimize the corpus's dimensionality, leading to better classification results [55]. This approach transforms textual data into a numerical format, making it easier to feed it into

ML models for further analysis [67]. Moreover, one study utilized the sklearn TF-IDF vectorizer to represent communal hate towards a minority community [58].

The n-grams Method: The n-grams method refers to sequences of *n* words extracted from a text corpus and is a feature extraction method commonly used in NLP and ML. During an experiment [91], three types of string features, uni-gram, bi-gram, and tri-gram features, were extracted. Uni-gram features consider each word in a sentence independently, without considering the relationships between words. This type of feature does not consider the relevancy of words within a single sentence but can identify highly abusive words. On the other hand, bi-gram features consider the relationship between two consecutive words in a sentence. In the case of tri-gram features, the relationship between three successive words in a sentence is considered. This type of feature finds more of the context and meaning in a sentence, allowing it to better identify and flag abusive language [42].

A tri-gram model and word tokenization were used for text-based feature extraction [92], and a range of n-grams ($n = 1, 3$) was applied for extracting text [14]. Sazzed et al. [17] extracted unigrams and bigrams from text data and calculated the TF-IDF. The TF-IDF scores were then used as inputs for CML classifiers. In addition, Sarker et al. [56] incorporated n-grams modeling to detect antisocial comments using TF-IDF weights [58]; using this method, the authors more precisely captured the context and meaning of the text by looking at sequences of n words in comments. Another study [57] utilized an MNB classifier with TF-IDF weights for n-grams up to three.

Scholars have incorporated n-grams modeling to more accurately identify the presence of HS by considering combinations of words instead of individual words. While the word "তুই‡" may not be regarded as hateful or dehumanizing, it cannot be classified as hateful speech when analyzed using unigrams ($n = 1$). However, when combined with other words like "খানকি র বাচ্চা" to form bigrams or trigrams ($n = 2, 3$), this word can be classified as HS. Graphical visualizations of n-grams are shown in Fig. 6.

Word2Vec: When using a neural network-based method named Word2Vec, words are represented as dense vectors that capture both their semantic and syntactic meaning. This method can represent Bengali text as a numerical vector, where each element in the vector represents embedding a unique comment in the text corpus [61]. Word2Vec is an algorithm used to generate word embeddings, which are fixed-length vector representations of words in a corpus [59]. One of the fundamental equations in the Word2Vec model is the

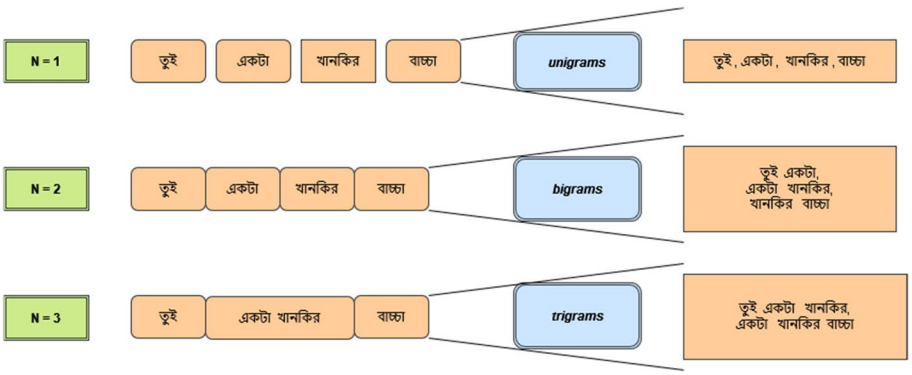

Fig. 6 A contiguous sequence of *n* tokens

skip-gram objective function, which is used to train a neural network. The skip-gram objective function is defined below.

$$L = \frac{1}{T}\sum_{t=1}^{T}\sum_{-m\le j\le m, j\neq 0} \log p(w_{t+j} \mid w_t) \tag{4}$$

where T is the total number of words in the corpus, m is the size of the context window, w_t is the center word, and w_{t+j} is the context word. The probability $p(w_{t+j}|w_t)$ is computed using the softmax function, which inputs the dot product of the vector representations of w_t and w_{t+j}.

The Word2Vec model can also be visualized using a diagram that shows how the neural network is structured and how the different layers are connected. This diagram typically includes input and output layers, one or more hidden layers, and various activation functions and weights that transform the input data into meaningful embeddings. One author used the Word2Vec model with the gensim module to train on a 30k dataset using the CBoW method, and the embedding dimension was set to 300 [49]. In another analysis, the embedding dimension was set to 16 to create the Word2Vec model, and 19,469 vocabulary words were accounted for. As a consequence, words mostly appear on two opposing edges [52]. Two Word2Vec models, referred to as Word2Vec skip-gram (W2V SG) and Word2Vec continuous bag of words (W2V CBOW) were trained using informal text and are hence referred to as informal embedding approaches. The training procedures employed 1.47 million Bengali comments from Facebook and YouTube from eight different categories [53]. The Word2Vec embedding architecture is shown in Fig. 7.

FastText: FastText is a variant of Word2Vec that can capture subword information. This approach is beneficial in languages where words can be constructed from multiple morphemes. The advantage of FastText is that it can understand the meaning of rare or misspelled words and can be used to represent out-of-vocabulary words [93]. In research work, researchers used the largest pretrained Bengali word embedding model based on FastText (BengFastTest) to test how the model performed on YouTube data [49]. Then, with the use of informal texts, two FastText embeddings, denoted as FastText skip-gram (FT SG) and FastText continuous bag of words (FT CBOW), were trained on YouTube comments [53]. Additionally, the FastText model was evaluated using lemmatization in

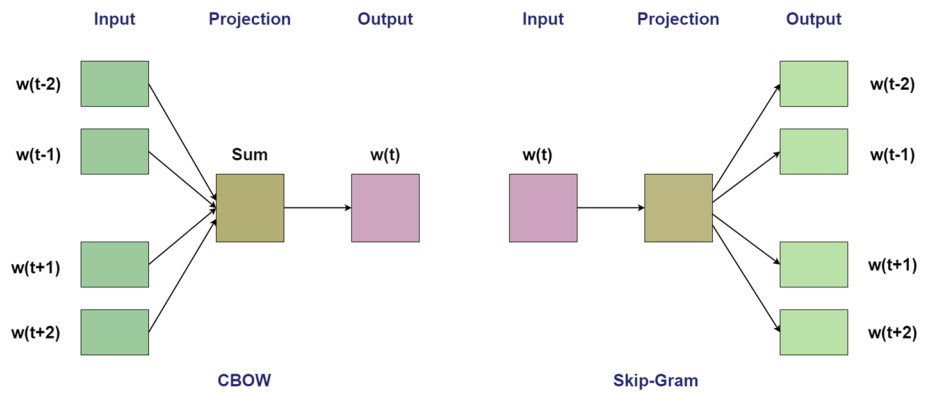

Fig. 7 Working principle of Word2Vec

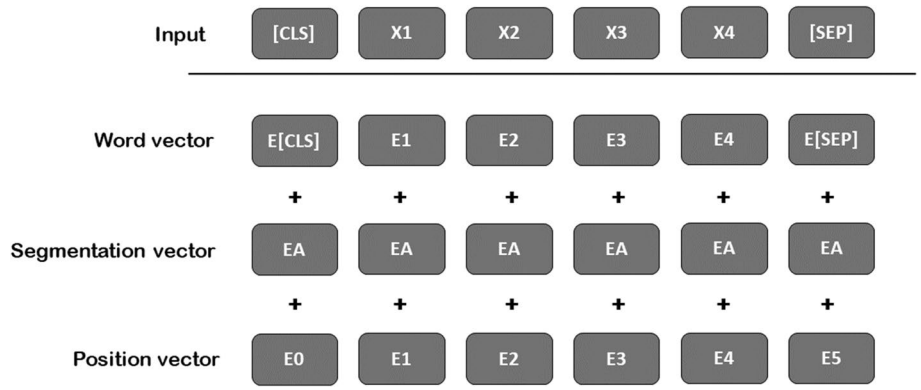

Fig. 8 Input sequence of BERT

one research project, where it was trained on Bengali articles for a classification benchmark study and showed somewhat greater accuracy. The sequences were limited to 100 words by truncating larger texts and padding shorter ones with zeros to prevent padding in convolutional layers with numerous blank vectors [50].

BERT: Bidirectional encoder representations from transformers (BERT) is a type of deep learning model where each output unit is connected to each input unit, and the strengths of the connections are determined dynamically through calculations. BERT consists of three layers (Fig. 8). Collections of tokens, special symbols, words, etc., are input into the first layer. Following the input layer, the coding layer consists of diverse multilayer transformers [94], which are based on the attention mechanism [95]. These two-way transformers are employed to encrypt input text and generate the associated output matrices. BERT is another prominent pretrained unigram strategy used to construct token embeddings.

As shown in Fig. 8, BERT allows word vectors, segment embedding vectors, and several forms of position embedding vectors. BERT models include BERT-base and BERT-large. The BERT-base model uses 12 transformer encoder blocks with 768 hidden layers and 12 self-attention heads, for a total of approximately 110 million parameters. The BERT-large model, in contrast, consists of 24 transformer encoder blocks with 24 heads of self-attention layers, and generates over 340 million parameters. The accuracy of BERT is model-dependent, with BERT-large achieving higher accuracy than that of BERT-base. However, using BERT-large requires more extensive resources, making it more computationally expensive. BERT has several advantages, including its bidirectional ability to handle contextual information extraction. Additionally, this model trains faster than many other language models and has been successfully applied in various language modeling applications [96].

Data must undergo comprehensive preprocessing before Bengali texts containing political, personal, geopolitical, and religious HS can be classified. This preprocessing step involves cleaning and transforming collected raw text data to a standardized format that can be further analyzed and modeled. Some BERT architectures used for Bengali HS detection include the monolingual Bangla BERT-base [97], multilingual BERT-cased/uncased [98], and XLM-RoBERTa models [99].

The Bangla BERT-base model is a language model that is pretrained on a large corpus of Bengali text data using the BERT architecture. RoBERTa is an enhanced version of BERT that has been optimized by using larger batch sizes and dynamic masking and training on even larger datasets. XLM-RoBERTa can understand and generate text in multiple languages, including Bengali [50]. To detect aggression in text data written in the English, Hindi, and Bengali languages, the BERT and RoBERTa models were utilized [100]. Because XLM-RoBERTa and multilingual mBERT are pre-trained on data from multiple languages, including low-resource ones such as Bengali, they have a particularly large impact. They can transfer knowledge from high-resource languages to low-resource languages thanks to this multilingual training, which enhances performance on tasks like text translation and classification.

Meta-embedding: Combining different word embeddings to produce a cohesive representation that can use each embedding's unique advantages is known as meta-embedding. Vectors representing words in a continuous vector space-where similar words have similar representations-are called word embeddings. Combining various embedding techniques (e.g., Word2Vec, GloVe, FastText, BERT, etc.) can result in richer representations as they capture different aspects of word semantics. For languages with limited resources, meta-embedding combines multiple pre-trained embeddings from high-resource or multilingual embeddings to create a comprehensive word representation. This approach uses the semantic information encoded in pre-existing embeddings to improve word vector quality for a language with constrained linguistic resources. Let $\mathbf{E}_1, \mathbf{E}_2, \ldots, \mathbf{E}_n$ represent the embeddings from different sources. The meta-embedding $\mathbf{M}$ for a word w can be computed using techniques like concatenation $\mathbf{M}(w) = [\mathbf{E}_1(w); \mathbf{E}_2(w); \ldots; \mathbf{E}_n(w)]$, averaging $\mathbf{M}(w) = \frac{1}{n}\sum_{i=1}^{n} \mathbf{E}_i(w)$, or more advanced methods like canonical correlation analysis (CCA) which finds transformations $\mathbf{W}_i$ that maximize the correlation between pairs of embeddings: $\mathbf{M}(w) = \sum_{i=1}^{n} \mathbf{W}_i \mathbf{E}_i(w)$. This approach effectively transfers the richness of high-resource embeddings to the target language, enhancing its word representations and compensating for the lack of native linguistic data. Hossain et al. [101] used the average meta embedding for text classification in the resource-constrained language in terms of increased accuracy. For the domain-specific text classification meta embedding application is also promising [102].

Table 3 summarizes text preprocessing and feature engineering methods used in the papers on Bengali HS detection that we surveyed.

Detection models

We identified several models that can detect HS in the Bengali language. In this section, we provide additional information regarding these models. As mentioned in "Introduction" section, model training is an essential task in Bengali language HS detection. Furthermore, an in-depth discussion of the algorithms employed by these models will be presented. In addition, graphs will be used to demonstrate the detection and classification architectures of the models. The taxonomy of each model is provided in Fig. 9. In Table 5, we provide information on the models and algorithms used as well as their performance in relation to the evaluation criteria. In Table 4, we highlight the advantages and disadvantages of each model.

Table 3 Summary of Bengali text HS detection papers that applied text preprocessing and feature engineering

References	Text preprocessing	Feature engineering
[48]	Manual cleaning Negation	TF-IDF CountVectorizer
[49]	Cleaning	FastText, BengFast Word2Vec
[50]	Cleaning, Normalization, Stemming	FastText
[52]	Stop words removal, Tokenization	Word2Vec
[53]	Cleaning	TF-IDF, Word2Vec FastText
[54]	Cleaning, Stemming Normalization)	FastText
[15]	Duplicate removal	Word unigrams BengFastText multilingual FastText
[55]	Tokenization Stemming	TF-IDF
[14]	Cleaning Stemming	CountVectorizer TF-IDF vectorizer n-grams
[67]	Cleaning	TF-IDF
[65]	Cleaning	n-grams
[56]	Cleaning Tokenization	TF-IDF n-grams
[68]	Cleaning Tokenization Stemming	TF-IDF
[57]	Stemming, Cleaning	n-grams TF-IDF
[58]	Tokenization Stemming Stop words removal	n-grams TF-IDF vectorizer
[91]	Normalization Noise removal	n-grams
[59]	Cleaning Tokenization	Word2Vec
[5]	Tokenization, Cleaning	CountVectorizer
[60]	Cleaning, Tokenization Bengali Stemmer	TF-IDF Conv1D
[64]	Cleaning	CountVectorizer
[42]	Cleaning	n-grams CountVectorizer TF-IDF vectorizer
[92]	Cleaning, Tokenization Stemming, Porter stemming	TF-IDF Tri-grams
[61]	Tokenization, Bengali stemming Punctuation remove	FastText, Conv1D
[40]	Cleaning	TF-IDF
[2]	Tokenization, Normalization Stemming, Noise removal	TF-IDF, Word2Vec FastText, BERT
[63]	Cleaning	CountVectorizer, TF-IDF
[41]	Cleaning, Stemming Tokenization	FastText

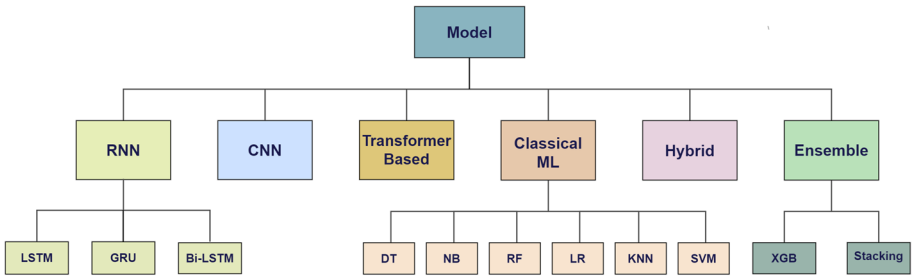

Fig. 9 Taxonomy of all models: Categorization of algorithms and techniques

Classical machine learning models (CMLMs)

In the past few years, there has been significant research in Bengali HS detection. Most studies in this area have utilized machine learning techniques to classify Bengali texts in a dataset [40, 42, 58, 63, 92]. The most commonly used approaches are the Naïve Bayes, SVM, k-nearest neighbors, decision tree, random forest, and combined techniques.

Naive Bayes (NB): NB is a widely used supervised ML algorithm based on the Bayesian theorem. A key characteristic of this classifier is its reliance on probability-based predictions, which determine the likelihood of an object being part of a particular category or class. The posterior probability P(A|B) is calculated using the Bayes theorem [103] and the equation is given as follows:

$$P(A|B) = \frac{P(B|A) \cdot P(A)}{P(B)} \tag{5}$$

The NB classification technique examines the relationship between each attribute and the class for each instance to determine the conditional probability for their interrelationship. This probability is then used to predict an instance's class based on its attributes' values. In several practical applications, such as text classification and spam detection, NB classifiers have demonstrated remarkable performance [104] and can work well with a small dataset.

Support Vector Machine (SVM): The SVM is one of the most commonly used supervised machine learning techniques for classification. SVMs examine data to find patterns and are utilized for performing classification and regression analysis [105]. This method aims to identify a linear separator (hyperplane) in multidimensional space that can effectively divide the data points of two different classes until a large minimum distance is discovered. This approach is well-suited for analyzing high-dimensional data, as an accurately constructed hyperplane can lead to superior performance [106]. For instance, Akhter et al. [92], used an SVM classifier for cyberbullying detection in Bengali text, and it performed the best among other classifiers.

k-Nearest Neighbors (KNN): The KNN approach is a lazy way of learning that may be applied to situations requiring classification as well as regression-based prediction. When using an instance-based learning approach, computation is performed until the classification task is completed. In this approach, local approximation is accounted for, and it is assumed that similar data are close together [107]. The KNN classifier records all the existing data in a given dataset and classifies a new data point based on its similarity

to the existing data. The classical KNN classifier utilizes a distance function such as the Euclidean distance to measure the distance between two data points [108]. The equation of the Euclidean distance between two data points A and B is given by:

$$dist(A, B) = \sqrt{\sum_{i=1}^{d} (A_i - B_i)^2} \tag{6}$$

where d is the number of dimensions of the data points. By calculating the distances between data points, we can obtain the nearest points, i.e., neighbors.

Logistic Regression (LR): LR is a statistical technique that uses a probabilistic approach in which the classifier uses a logistic function to evaluate the relationship between a dependent variable as a target class and one or more independent variables or features for a given dataset [109]. A logistic function is also referred to as a sigmoid function. This approach provides probabilistic values that lie between 0 and 1. LR is a more effective substitute for linear regression, which applies linear models to each class and makes forecasts for new instances based on majority voting [110]. Logistic regression is applied mainly to solve classification problems. In several studies [40, 59], LR was implemented along with other algorithms to identify toxic and abusive Bengali comments.

Decision Tree (DT): A DT is a supervised learning algorithm that employs a tree-like structure to represent decisions and their potential outcomes by consolidating a set of data-derived classification rules [111]. Internal nodes in a decision tree represent dataset features, branches represent decision rules, and each leaf node represents an outcome. To classify an instance, the process begins at the root of the decision tree using a top-down approach. Based on the outcome of the test, the algorithm moves down the tree to the appropriate child node, and the process is repeated until a leaf node is reached, which provides the final classification or prediction. To make sound decisions on each subset, DT algorithms recursively divide the training set by using superior feature values [112]. However, decision trees are prone to overfitting, which is a major problem. In addition, decision trees are not appropriate for continuous variables [113].

Ensemble models

Ensemble methods are a collection of techniques used to enhance the accuracy of models by combining multiple models rather than relying on a single model. By leveraging the collective predictions of these models, ensemble methods can yield significantly improved results. The use of ensemble models has contributed to the widespread adoption and popularity of ensemble methods in the field of machine learning. An ensemble method employs various modeling algorithms or uses different training datasets. The ensemble model then combines the predictions from each model to generate a final prediction for new, previously unseen data. The primary objective of ensemble modeling is to reduce the generalization error and enhance the prediction accuracy. Ensemble approaches effectively mitigate prediction errors by ensuring that the base models are diverse and independent. This approach leverages the collective wisdom of the models to arrive at more robust predictions. Despite comprising multiple base models, an ensemble model functions and performs as a unified entity. Consequently, ensemble modeling techniques are widely employed in practical data mining solutions.

Ensemble methods are widely regarded as the most advanced solution for numerous machine learning problems [114]. Ensemble models are commonly used in data science and machine learning applications because they can produce more accurate and reliable results than those of any single model alone. To enhance the accuracy of a multiclass classifier, researchers [52] have utilized ensemble model techniques that involve binary classifiers. Multiple models were trained, and their predicted results leveraged for all comments. Several supervised machine learning algorithms were employed, including the random forest, SVM, KNN, and Naïve Bayes algorithms. Notably, the SVM algorithm outperformed the other classifiers, achieving an improved accuracy of 85%.

Random Forest (RF): The random forest (RF) classifier is an ensemble of decision trees used for regression, classification, and other tasks. This classifier trains a large number of decision trees and then outputs the class that represents the mode of the classes (in the case of classification) or mean prediction of the individual trees (in the case of regression) [116]. By increasing the number of decision trees in a random forest, one can improve the model's accuracy and reduce the likelihood of overfitting. In text classification, RF classifiers are well suited for dealing with high-dimensional noisy data [117]. In addition, an RF classifier can assist in determining feature importance, which can aid in better understanding underlying data patterns.

Tuning RF hyperparameters, such as the maximum tree depth, is critical for attaining optimal performance for a specific task. However, hyperparameter tuning can be computationally expensive and time-consuming, particularly when working with large datasets or high-dimensional data. In a study by Islam et al. [63], an RF with a TF-IDF vectorizer was applied to detect abusive text and HS in Bengali text on social media platforms. There are several ensemble models. Some of these models are discussed here. Figure 10 shows the most popular ensemble methods graphically. The architecture used by [115] for Bangali HS detection is shown in Fig. 11.

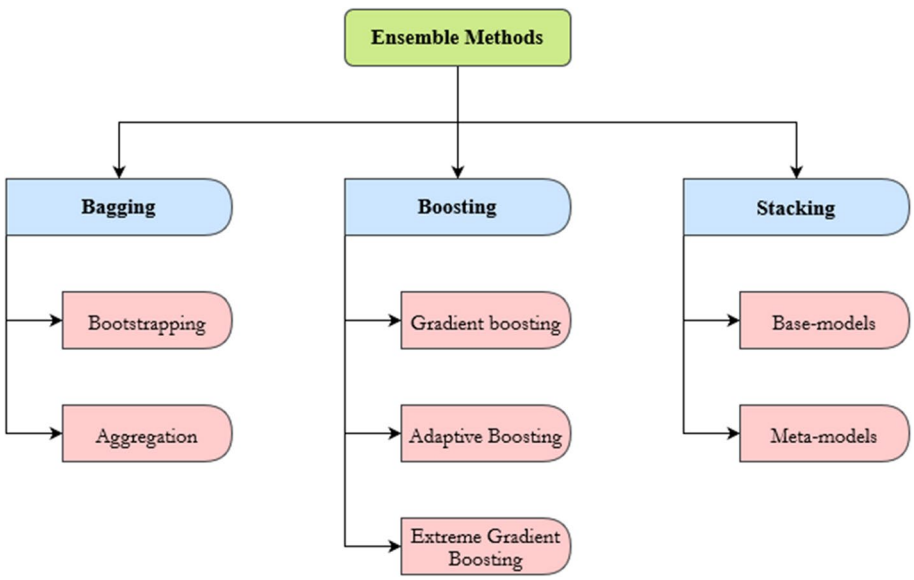

Fig. 10 Most popular ensemble methods

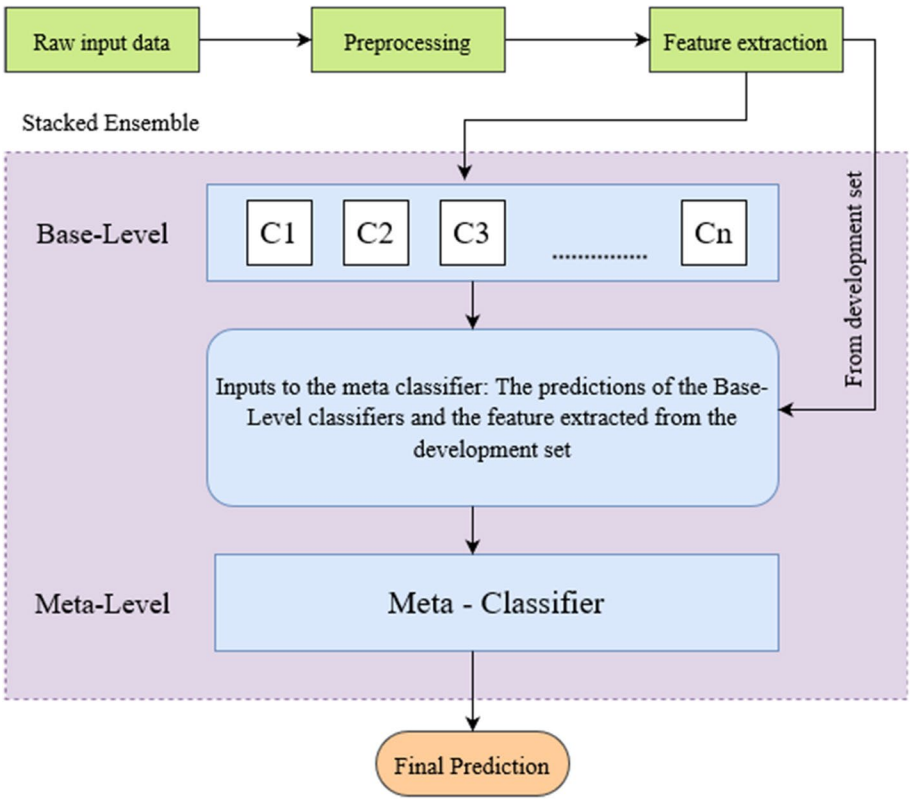

Fig. 11 Ensemble architecture with two levels of classifiers for HS detection [115]

Extreme Gradient Boosting (XGBoost): XGBoost is a tree-based ensemble method that utilizes a gradient-boosting machine learning framework [118] for solving regression and classification problems. Unlike the random forest (RF) approach, XGBoost employs distinct strategies for constructing and ordering of decision trees, handling feature importance, and combining the results of multiple trees. XGBoost follows a level-wise approach to constructing trees, beginning with a single root node and incrementally expanding the tree in a breadth-first manner. In contrast, an RF method builds trees independently and combines their predictions through voting or averaging.

XGBoost employs different algorithms to determine the optimal splits for tree nodes. For example, XGBoost incorporates a sparsity-aware algorithm to handle sparse vectorized textual data. When determining the split, the sparsity-aware algorithm identifies the features with zero values and excludes them from consideration. Instead, this algorithm focuses on nonzero features, which contain meaningful information for the learning process. This approach effectively reduces the computational overhead associated with zero-valued features. Furthermore, this algorithm intelligently allocates all the data instances with zero values to o to the side of the split that yields the most significant loss reduction. This approach ensures that the algorithm optimizes the learning process by effectively utilizing the available information while considering the sparsity patterns in the data [119].

In summary, XGBoost incorporates a sparsity-aware split algorithm that handles sparse vectorized textual data, ignoring zero-valued features during the split

computation and strategically assigning instances with zero importance to the side of the split that maximizes loss reduction. This approach optimizes the learning process while efficiently handling the sparsity of the data.

Stacking: Stacking, also referred to as stacked generalization, is an ensemble method utilized in machine learning that combines the predictions of multiple learning algorithms to improve performance. This technique involves training a meta-model that learns to ensemble the predictions of several base models. Stacking has been successfully demonstrated in applications in various domains, such as regression, density estimation, distance learning, and classification. By leveraging multiple models' diverse perspectives and strengths, stacking can effectively enhance the overall predictive capability. Moreover, stacking can be utilized to estimate the error rate involved in bagging, another ensemble method. By observing the performance of different models within the stacking framework, evaluating and quantifying the error rate associated with the bagging process becomes possible.

The main idea behind stacking is to leverage the strengths of different models and to construct a more robust and accurate model by combining their predictions. The base models can be of different types, such as DT and SVM. Stacking can be performed in two ways: (i) level-0 models: Each base model makes its own independent prediction. These predictions are then combined with the meta-model to output the final prediction, and (ii) level-1 models: The outputs of the base models are used as input features to train another model, which outputs the final prediction. This second-level model is referred to as a meta-model. Level-0 classifiers ($C_1, C_2, C_3, \ldots, C_n$) at the initial stage of the model receive the input from the training set that has been created in this manner. SVM, LR, and XGBoost are used as base-level classifiers, and LR is used as a meta-level classifier in research work [115]. While stacking has not often been applied in Bengali HS detection, this approach is recommended for its powerful classification accuracy.

Convolutional neural network (CNN)

Neural network models have shown impressive performance in modeling sentences and documents. Convolutional neural networks (CNNs) and recurrent neural networks (RNNs) are two popular architectures for these tasks. These architectures employ distinct approaches to comprehending natural languages [120]. A CNN is a multilayer neural network consisting of input, convolutional, pooling, and fully connected layers. The architecture of a CNN enables it to learn the spatial hierarchies of features through a backpropagation algorithm in an automated and adaptive manner [121].

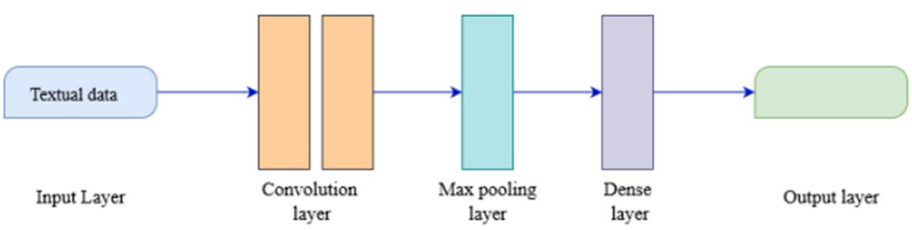

Fig. 12 Components of a CNN architecture [123]

CNNs use convolutional layers to extract local features from input data via filters, followed by pooling layers to minimize their dimensionality. Fully connected layers use the extracted features to classify the input into one or more output classes. A CNN also employs a dropout mechanism to address the issue of overfitting [122]. CNNs can identify sentences in the context of text classification by framing the input text as a sequence of one-dimensional vectors. For instance, in [41, 59, 60], a CNN-based technique was utilized to detect HS in Bengali using text data.

The entire architecture of a CNN, often referred to as a "convert," is a sequential arrangement of these layers. Each layer in the network applies a different function to transform one volume into another. To fully understand the advancements in the CNN architecture, it is crucial to comprehend the different components of CNNs and their applications. The components of CNNs are shown in Fig. 12. Familiarity with these components can provide insights into how CNNs have evolved and been utilized in various contexts [123].

- Input Layers: The input layer is the layer in which we give input to the model. In CNNs, the input typically consists of numerical data, such as images or numerical feature vectors. However, to use textual data as input for a CNN, we need to convert the text into a numerical representation, e.g., in the form of word embeddings or character embeddings. Each word or character is typically described as a vector. There are several ways to represent textual data numerically for input into a CNN. Two commonly used approaches are discussed below:

  a. Word Embeddings: Word embeddings represent words used in natural language processing (NLP). Typically, a representation is in the form of a real-valued vector representing a word's meaning. It is expected that words that are close to one another in the vector space have comparable implications. Popular word embedding techniques include Word2Vec, FastText, and GloVe (details on the first two word embedding techniques were discussed in "Feature extraction" section, i.e., "Feature Extraction"). A fixed-length vector represents each word, and the entire text is converted into a sequence of word vectors. This sequence can be considered as a 1D input in the CNN. Researchers can use pretrained word embeddings or obtain their own embeddings from a large corpus of text data.

  b. One-Hot Encoding: In machine learning, one-hot encoding involves transforming categorical data into a format that can be fed into machine learning algorithms to increase the prediction accuracy. Machine learning frequently uses one-hot encoding to handle absolute inputs. In this approach, researchers can create a vocabulary of unique words in their dataset. Each word is assigned a different index, and the place in the vector that corresponds to that index is set to 1, while all other positions are set to 0. The entire text is converted into a sequence of one-hot vectors, resulting in a 2D input for the CNN.

  After converting the textual data into numerical representations, they can be input into the CNN. The CNN typically consist of convolutional layers, pooling

layers, and fully connected layers, which will learn hierarchical representations of the text and make predictions based on the available features.

It should be noted that CNNs are most commonly used for image analysis tasks. Other architectures such as recurrent neural networks (RNNs) or transformer-based models (such as BERT or GPT) are often preferred for textual data. These models are designed to handle sequential data more effectively.

- Convolutional Layers: Convolutional layers in a CNN for textual data can be used to capture local patterns and dependencies within the text. Although CNNs are more commonly used for image processing, they can also be applied to sequential data such as text via treatment as a 1D signal. In the context of textual data, the input to the CNN is typically a sequence of word embeddings or one-hot encoded vectors representing the words in the text. Convolutional layers can be used in a CNN for textual data as described below:
- Pooling: Pooling layers downsample feature maps by summarizing the information in a local neighborhood. Max pooling is commonly used. In max pooling, the maximum value within a window is selected as the representative value for that region.
- Multiple Convolutional Layers: Stacking multiple convolutional layers in a CNN is a commonly used approach to capture higher-level features and more abstract text representations. The output of one convolutional layer becomes the input of the next, allowing the model to learn increasingly complex patterns.
- Flattening: After applying the convolutional and pooling layers, the resulting feature maps are flattened into a 1D vector. This vector is passed through one or more fully connected layers, which perform classification or regression tasks based on the learned features.

  It is important to note that the architecture and specific configuration of a CNN, including the number of filters, kernel size, pooling strategies, and number of layers, can vary depending on the particular task and dataset. These choices should be based on experimentation and performance evaluation.
- Fully Connected Layers: The input of the final layer is the output from the preceding layer. A fully connected layer performs appropriate classification or regression task computations. This layer utilizes the information, typically the previous layer's output, to generate the final predictions or outcomes. In the case of classification, the last layer often employs activation functions such as the softmax activation function to produce probabilities for different classes, whereas, in regression tasks, numerical values are generated directly. The neural network receives the final classification or regression results by processing the input data through this layer.
- Output Layer: For classification tasks, the output from the fully connected layer is further processed using a logistic function, such as a sigmoid or softmax function. This step is performed to convert the output values for each class into corresponding probability scores. The logistic function transforms the raw outputs into probabilities between 0 and 1. By applying the logistic function to the outcomes of the fully connected layer, the neural network outputs a probability scores that allows for interpreting and comparing different classes during the classification task.

  The activation function in the output layer ensures that the network's output is compatible with the task requirements. In contrast, the loss function measures the dis-

crepancy between the predicted outcome and the ground truth, guiding the learning process during training.

The architecture and configuration of the CNN, including the numbers of layers and neurons and the specific activation functions, may vary depending on the particular requirements of the task and dataset.

Recurrent neural network (RNN)

An RNN is a neural network architecture that is specially designed to handle sequence data modeling and analysis. The ability of an RNN to relearn from both previous and new datasets makes it ideal for analyzing time series data [124]. A typical RNN model comprises three layers: an input layer, a recurrent layer, and an output layer. The input layer transforms sensor output into a feature-conveying vector, which is then followed by a recurrent layer that offers feedback. Some recent models also consist of memory cells [125]. An RNN improves a neural network by incorporating recurrent layers. RNNs utilize a hidden state that allows them to process data in a sequential manner. The hidden state acts as a memory for the network, allowing it to retain information about prior inputs and utilize this information to inform future predictions. RNNs have been frequently employed for text categorization tasks in Bengali, such as HS identification. However, the vanishing gradient problem is a potential issue that can impede the ability of an RNN to learn and recall long-term dependencies.

Long Short-Term Memory (LSTM): LSTM is a special type of RNN architecture that is capable of handling long-time series data. A basic LSTM network consists of memory and three gates: an input gate, an output gate, and a forget gate. These three gates control the flow of information into and out of the cell; thus, the memory cell in an LSTM unit may store data for extended periods [126]. The input gate decides which information should be retained in the memory cell, whereas the forget gate indicates which information should be discarded. The output gate regulates the LSTM cell's output, allowing it to output information from the memory cell selectively. Overall, LSTM networks can effectively capture long-term dependencies in sequential data, allowing them to represent complicated interactions between input and output sequences while being less prone to overfitting than typical RNNs. In [49], LSTM was applied with several feature extraction techniques such as Word2Vec, FastText, and BengFastText for HS detection in the Bengali language.

Bidirectional LSTM (BiLSTM): Bidirectional long short-term memory (BiLSTM) is a more advanced form of LSTM that can learn from both past and future data. The flow of information in BiLSTM is bidirectional, with two hidden states facilitating communication in both the forward-to-backward and backward-to-forward directions [127]. This model comprises two LSTM layers that operate in contrasting directions, with one layer processing the input sequence from left to right and the other from right to left. By processing the input sequence in both directions, BiLSTM can learn and capture both forward and backward dependencies in the sequence data. Furthermore, BiLSTM can capture long-term dependencies in sequence data without storing redundant contextual information [128]. Hence, BiLSTM has been demonstrated to be extremely effective in natural language processing tasks, including text categorization and sentiment analysis. For instance, in [15, 53], a BiLSTM model was applied on a benchmark dataset to detect Bengali HS and yielded noteworthy results in terms of an F1 score evaluation.

Gated Recurrent Unit (GRU): A gated recurrent unit (GRU) is another type of recurrent neural network (RNN) architecture that is similar to LSTM in terms of its capacity to handle sequential data. The GRU was introduced by Cho et al. [129]. A GRU network comprises two gates, a reset gate and an update gate, and a hidden state that is updated at each time step. The reset gate controls how much of the prior hidden state should be forgotten, while the update gate specifies how much of the current hidden state should be updated. A GRU can use this gating mechanism to selectively retain or discard information from earlier time steps based on its relevance to the current input. GRUs have been employed to overcome the inherent difficulties of conventional RNNs in capturing long-term dependencies, specifically the issues of vanishing gradient and explosion [130]. GRU-based models were used in [56, 58] to detect hateful Bengali speech in social media.

Transformer-based models

A particular class of AI algorithm known as large language models (LLMs) [131] is capable of performing a wide variety of NLP tasks. Text generation, text analysis, sentiment analysis, question answering, hate speech detection, and other related tasks are among the most frequently performed tasks. LLM models are typically based on the transformer architecture [94].

Pretrained transformer-based LLMs [132], including GPT-3 and GPT-4 [133], LaMDA [134], BERT [135], BART [136], ALBERT [137], RoBERTa [138], XLNet [139], ELECTRA [140], T5 [141], and PEGAUSUS [142], are widely utilized. They have been trained on copious amounts of textual data. Pretrained LLM models achieve exceptional success in NLP owing to their excellent capacity to learn simple language representations by training on significant amounts of unstructured text input [143]. BERT and its family can be used in two ways (i) as a feature extraction tool (see "Feature extraction" section), which can be used together with any downstream classifier or (ii) as an integrated end-to-end classification tool, with a fully connected (dense) neural network typically serving as the final layer [144].

Architecturally speaking, LLM models exhibit differences in depth and size. For instance, GPT-3 produces 175 billion parameters spread over 96 levels, whereas Pam generates an even greater number of parameters, 540 billion, spread over 106 layers. Each of these models has a unique setup. The methods GPT-3 and PaLM use to generate output are different in their configurations. LLMs have assessed various datasets from social media, books, code repositories, Wikipedia, and question sets. They have proven they are capable of carrying out a variety of tasks effectively. As a result, LLMs have received much attention for their useful contributions to various fields, such as media marketing, healthcare, education, and other customer services. Some LLM programs perform better than others in a particular domain. For example, GPT-3 is well-known for its ability to generate text styles, while LaMDA performs better when answering factual questions. LLMs represent a new wave of technological innovation with the potential to revolutionize a number of industries.

Additionally, the LLM can be employed for augmentation. HateBERT is one such model [145]. HateBERT is a BERT [146] model that has been adjusted based on data on hate speech. The model can now recognize and understand hate speech with greater accuracy thanks to this fine-tuning. The language model can be used as an early fusion strategy. The early fusion involved combining text and vision before classification [147].

Kumar et al. [148] explored the effectiveness of SVM, BERT, ALBERT, and DistilBERT [149] in developing classifiers for detecting offensive and aggressive speech and conducted experiments to compare the efficacy of transformer models with SVMs and assess their potential usefulness in these specific tasks. In another study [54], the researchers employed various multimodal learning methods, combining early and late fusion techniques and utilizing transformer-based neural architectures such as Bangla BERT-base [97], multilingual BERT (mBERT) [98], and XLM-RoBERTa (XLM-R) [99].

Multilingual transformer models that have been pretrained on substantial corpora have undergone fine-tuning by researchers [66]. A monolingual Bengali model was not used because such a model is not readily available. Instead, model-specific tokenizers were employed to split the input text into a series of tokens. These models employ byte-pair encoding, which can split a word into several tokens. When utilizing the XLM-RoBERTa big tokenizer, the input sentence "তুই কু ত্তার বাচ্চা" (you are a son of a b*tch) is divided into four tokens. Compared to the performance of deep learning models such as CNNs and LSTMs trained on distributed word representations and conventional approaches that rely on hand-crafted features, the researchers showed that fine-tuning transformer models can lead to higher performance.

Researchers have introduced a retrained BERT model named BanglaHateBERT [51] to detect abusive language in Bengali text. To construct BanglaHateBERT, the pretrained BanglaBERT model was used. This model was retrained with 1.5 million objectionable posts from a vast corpus of offensive, abusive, and hateful Bengali postings

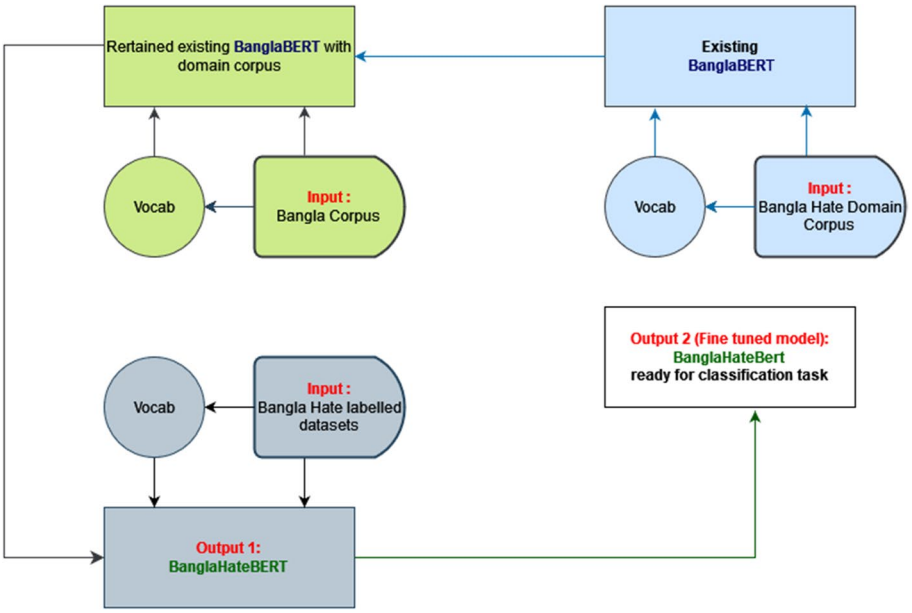

Fig. 13 Architecture of BanglaHateBERT [51]

collected from various sources and made public. The outcomes of a thorough comparison between a general pretrained language model and the model retrained using the HS corpus were given. The authors implemented the Huggingface transformers library to develop classifiers and performed fine-tuning on various transformer models with 70% of the training data. In every instance, BanglaHateBERT outperformed the mBERT, indicBERT, BanglaBERT, and CNN models in terms of identifying HS, demonstrating the superiority of domain-based contextual models with respect to performance. The BanglaHateBERT model outperformed competing models substantially, yielding an accuracy of 94.3% and an F1 score of 94.1% [51]. The general architecture of BanglaHateBERT is shown in Fig. 13.

Hybrid model

Hybrid models integrate distinct methods or algorithms to address issues or forecast future outcomes. Integrating multiple methodologies is a common strategy to leverage each approach's strengths while mitigating their shortcomings. A hybrid model's advantage lies in its ability to enhance its performance by combining multiple methodologies or algorithms without depending on any specific approach or technique. Additionally, hybrid models may exhibit greater robustness to variations in the input data or the environmental conditions. Hybrid models have been used in various applications, including NLP, computer vision, and prediction for several events.

In the context of Bengali text classification, a hybrid model may include a model incorporating a rule-based approach with either a DL or a classical ML-based method. The objective of this combination is to strengthen the accuracy of the classification process. While the DL methodology has the potential to extract patterns from textual data and subsequently make predictions based on those patterns, a rule-based technique is capable of detecting specific keywords or phrases. Hybrid models have been employed in numerous investigations for Bengali text classification. For instance, a neural ensemble approach was utilized to detect HS by incorporating transformer-based neural architectures, including monolingual Bangla BERT-base, multilingual BERT-cased/uncased, and XLM-RoBERTa, with layer-wise relevance propagation (LRP) [50]. The variability of the number of layers in a hybrid model for text categorization is contingent upon the particular architecture and design of the model.

A typical hybrid model for text categorization comprises multiple layers, namely an input layer, a word embedding layer, a convolutional layer, a recurrent layer, a fully connected layer, and an output layer. The general structure of a hybrid model is depicted in Fig. 14.

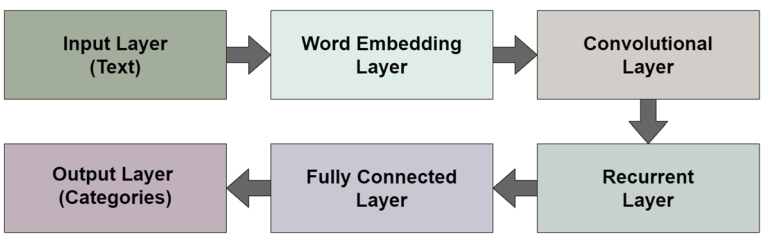

Fig. 14 General structure of a hybrid model for text categorization

The process of HS detection in the input layer of a hybrid model for text categorization involves the acquisition of textual data from diverse sources, such as social media, online forums, and other virtual platforms. The input layer receives both padded and embedded sequences, which are subsequently processed in the successive layers to identify and categorize instances of HS.

The word embedding layer then transforms the input text into a vectorized form through dense vector mappings. The input to the subsequent layer of the hybrid model comprises the vector representation for each term. The convolutional layer applies filters to the embedded words to extract relevant features. The convolutional layer in Bengali HS detection has the ability to learn and identify features such as hate words, derogatory terms, and offensive language. Filters are applied to the embedded words, which are then subjected to a nonlinear activation function and a pooling layer. Convolutional neural networks (CNNs) are a popular algorithm used in Bengali text categorization. Gated convolutional neural networks (GCNNs) and dilated convolutional neural networks (DCNNs) are also commonly used algorithms.

In addition, RNNs are a type of neural network that excels in processing sequential data, such as text. RNNs process an input sequence of embedded words by utilizing a series of recurrent units that retain a memory state of the previous inputs, allowing the model better to understand the context and meaning of the text. Recurrent layers in Bengali text categorization use LSTM and GRU algorithms to generate a sequence of hidden states associated with an embedded word in the input sequence. These algorithms address the issue of the vanishing gradient phenomenon, which can impede the ability of conventional RNNs to capture long-term dependencies. This feature is essential for tasks such as text classification.

The fully connected layer is a neural network layer that performs a nonlinear transformation on the input features, resulting in a high-level representation of the text input. The dense layer establishes connections between all neurons in the current layer and those in the preceding layer. The output layer generates the final output of the model, which can be a probability distribution or a single predicted category. The binary and multiclass classification output layer usually comprises a solitary neuron with a sigmoid activation function. In multiclass classification, multiple neurons are used with a softmax activation function to transform the output of the preceding layer into a probability distribution. The loss function and gradients of the model are used to update the weights and biases of the output layer neurons.

Evaluation metrics

Evaluation metrics are quantitative measures used to assess the performance or effectiveness of a model, system, algorithm, or process. They provide numerical scores or statistics that help compare different models or methods, identify strengths and weaknesses, and make informed decisions about a system's or an approach's performance or success.

These metrics allow researchers, developers, and practitioners to objectively measure and evaluate the performance of their work against specific criteria or benchmarks [150]. Evaluation metrics in Bengali HS detection may vary depending on the specific research

and the goals of the study. However, commonly used evaluation metrics in HS detection tasks include the following:

Accuracy: By measuring the proportion of correct predictions to the total number of predictions, an HS detection model's overall accuracy can be evaluated. Although accuracy is a straightforward and popular statistic, an unbalanced dataset may prevent this metric from being sufficient.

$$\text{Accuracy} = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{7}$$

Precision: Precision is a metric that quantifies the accuracy of the HS classification by measuring the proportion of correctly predicted HS to the sum of true positives and false positives (non-HS instances incorrectly classified as HS). Precision measures the model's capability to identify HS instances and minimize false-positive instances correctly. The precision equation can be written as follows:

$$\text{Precision} = \frac{\text{True Positives}}{(\text{True Positives} + \text{False Positives})} \tag{8}$$

In this equation, "True Positives" represents the number of instances correctly classified as HS, and "False Positives" represents the number of instances incorrectly classified as HS when they are actually non-HS.

Recall: Recall measures the model's ability to identify all instances of HS correctly. This metric quantifies the ratio of true positives (correctly predicted HS) to the sum of true positive and false negatives (HS instances incorrectly classified as non-HS). The recall equation can be written as follows:

$$\text{Recall} = \frac{\text{True Positives}}{(\text{True Positives} + \text{False Negatives})} \tag{9}$$

Here, "True Positives" represents the number of instances correctly classified as HS, and "False Negatives" represents the number of instances of HS incorrectly classified as non-HS.

F1 Score: The F1 score is the harmonic mean of the precision and recall. This metric provides a balanced evaluation by considering both the precision and recall. The F1 score ranges between 0 and 1, where 1 indicates the best performance.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{10}$$

In this equation, "Precision" represents the precision value calculated using the true positives and false positives, and "Recall" represents the recall value calculated using the true positives and false negatives.

Specificity: This metric calculates the ratio of true negatives (correctly predicted non-HS) to the sum of true negatives and false positives (non-HS instances incorrectly classified as HS). Specificity measures the model's ability to identify non-HS instances correctly. The specificity equation can be written as follows:

$$\text{Specificity} = \frac{\text{True Negatives}}{(\text{True Negatives} + \text{False Positives})} \tag{11}$$

Here, "True Negatives" represents the number of instances correctly classified as non-HS, and "False Positives" represents the number of instances of non-HS incorrectly classified as HS.

Area Under the Receiver Operating Characteristic Curve (AUC-ROC): The AUC-ROC is a performance metric that is used to evaluate the discrimination ability of an HS detection model. The true positive rate (TPR) is plotted against the false positive rate (FPR) at various classification thresholds. The AUC-ROC ranges between 0.5 (random guessing) and 1 (perfect classification).

Matthews Correlation Coefficient (MCC): The MCC measures the quality of binary classification, accounting for true positives, true negatives, false positives, and false negatives. This metric provides a balanced evaluation even when the classes are imbalanced. The MCC equation can be written as follows:

$$\text{MCC} = \frac{\text{TP} \times \text{TN} - \text{FP} \times \text{FN}}{\sqrt{(\text{TP} + \text{FP}) \times (\text{TP} + \text{FN}) \times (\text{TN} + \text{FP}) \times (\text{TN} + \text{FN})}} \tag{12}$$

In this equation, TP refers to true positives (correctly predicted HS), TN refers to true negatives (correctly predicted non-HS), FP refers to false positives (non-HS instances incorrectly classified as HS), and FN refers to false negatives (HS instances incorrectly classified as non-HS).

Table 4 provides a comprehensive discussion of the strengths and weaknesses of various models; by analyzing these characteristics, we can gain insights into their applicability and limitations. Table 5 summarizes the models, algorithms, evaluation metrics, and performance statistics of the Bengali HS detection works surveyed in this paper.

Figures 15 and 16 show pie charts on the distribution of the models and the evaluation metrics used in Bengali HS detection covered in this survey.

Challenges

Bengali HS is a complex phenomenon, and detecting HS in Bengali is a difficult task that poses several challenges. In this section, we discuss some key challenges (summarized in Fig. 17) that researchers must address in detecting Bengali HS.

- Lack of Available Data: Bengali HS lacks an extensive availability of publicly accessible datasets. The absence of standardized Bengali datasets addressing HS or cyberbullying poses a notable challenge for research in this field [158]. Additionally, available datasets (such as those listed in Table 2) are limited in size and/or lack diversity in terms of classes. In [49], there was only one research project comprising 50k data, whereas research on hate speech detection in other languages typically involves datasets exceeding 85k data points [159, 160]. Moreover, collecting data from social media sites is a time-consuming and difficult task. Addressing these data challenges is crucial for the progress of research in the detection of HS in Bengali.
- Imbalanced Datasets: The development of the ML and DL models that effectively detect hate speech relies on accurately annotated, balanced, diversified, and substan-

Table 4 Strengths and weaknesses of all model categories

Model category	Strength	Weakness
CMLM [151]	• Simplicity, speed, and performance	• Limited ability to learn intricate representations from raw data
	• Effective handling of high-dimensional spaces	• Struggles with capturing complex nonlinear relationships
	• Generalizability on small- to medium-sized datasets	• Time-consuming and labor-intensive feature engineering
	• Proven effectiveness in process optimization, monitoring, and control applications	
RNN [49]	• Incorporation of past information and context	• Sequential processing leading to limited parallelization
	• Handling of variable length input outputs	• Computational expense for long sequences or large models
	• Performance on various sequential data tasks	• Difficulty in modeling long-term dependencies
	• Transferability of learned representations to related tasks	
Ensemble [114]	• Improved predictive accuracy through a combination of models	• Complexity in terms of implementation and maintenance
	• Reduction of bias, variance, and errors	• Management and training of multiple models
	• Handling of complex relationships in data	• Reduced interpretability compared to that of individual models
CNN [152]	• Effective handling of large-scale datasets and high-dimensional inputs	• Longer training times and increased resource consumption for large-scale models
	• Computational efficiency and memory-friendliness	• Limited receptive field capturing only local information
	• Suitability for data-driven challenges involving pattern recognition	
Transformer [143]	• Remarkable performance in various NLP tasks	• Computational and memory intensity for large-scale models
	• Ability to capture contextual information and leverage pre-trained representations	• Quadratic computation complexity regarding sequence length
	• Efficiency on longer sequences	• Requirement of substantial computational resources and lack of interpretability
Hybrid [50]	• Better performance through a combination of strengths	• Complexity in implementation, training, and maintenance
	• Ability to capture different aspects of data and collective knowledge	• Management and integration of multiple components
	• Flexibility in combining models, algorithms, or techniques	• Potential need for additional computational resources

tial datasets [161]. However, the scarcity of large-scale annotated balanced datasets is one of the major obstacles in detecting Bengali HS [17, 48, 58]. Suppose that there are a total of 4,000 samples in a dataset, and among them, there are 3,500 HS samples and 500 non-HS samples. This dataset is an example of an imbalanced dataset. A model may exhibit bias on this imbalanced dataset. The lack of balanced datasets can result in models being overfitted and biased towards non-hate speech, particularly when the training dataset is highly imbalanced [49]. This uneven distribution can impact the model's capability to identify instances of hate speech precisely and may lead to misclassification.

- Nature of the Language: Bengali is a classical and literary language with rich morphology and grammar, yet it is deemed still low-resource in digital space. There are

Table 5 Summary of all models applied in Bengali text HS detection

Refs.	Category	Model	Evaluation Metrics	Performance
[48]	CMLM	NB	Accuracy Precision Recall F1 Score	72% 75% 71% 73%
[49]	CMLM DL Model	SVM	Accuracy F1 Score	87.5% 91.1%
[59]	CMLM DL Model	CNN	Accuracy	95.30%
[41]	CMLM DL Model	SVM	Accuracy Precision Recall F1 Score	87.22% 87% 87% 87%
[50]	CMLM DNN Model Ensemble Model	Ensemble Model (Bangla BERT-base, XML-RoBERTa, mBERT-uncased)	Precision Recall F1 Score MCC	88% 88% 88% 82%
[63]	Ensemble Model	RF	Accuracy Precision Recall F1 Score	67% 68% 68% 67%
[66]	Transformer Model	XLM-RoBERTa	Accuracy F1 Score	93.8% 93.8%
[40]	CMLM Ensemble	SVM MNB KNN	Accuracy Precision Recall F1 Score	85.7% (SVM) 85% (MNB) 96% (KNN) 88% (SVM)
[61]	Hybrid Model	LSTM GRU CNN	Accuracy Sensitivity Specificity AUC	89.85% 84.50% 92.53% 0.89
[56]	CMLM RNN Model	SVM MNB GRU	Accuracy Precision Recall F1 Score	80.51% (MNB) 89.47% (SVM) 90% (GRU) 87.66% (GRU)
[68]	CMLM Ensemble Model DL Model	LR RF GRU LSTM	Accuracy Precision Recall F1 Score AUC	98.89% (GRU) 96.7% (LR) 96.7% (LR) 98.5% (RF) 1.00 (RF, LSTM)
[58]	CMLM DNN Model	GRU	Accuracy Precision Recall F1 Score	70% 68% 70% 69%
[57]	CMLM Hybrid Model	SVM CNN-LSTM	Accuracy Accuracy	78% 77.5%
[2]	CMLM Ensemble Model Hybrid Model	L-Boost (AdaBoost+LSTM)	Accuracy Precision Recall F1 Score	95.11% 94.51% 89.37% 91.85%
[55]	CMLM	SVM	Accuracy Precision Recall F1 Score	88% 88% 88% 88%
[53]	CMLM DL Model Transformer Model	Bi-LSTM	F1 Score MCC	86.85% 77.90%
[92]	CMLM	SVM	Accuracy Precision F1 Score ROC	95.40% 95% 94% 0.76

Table 5 (continued)

Refs.	Category	Model	Evaluation Metrics	Performance
[51]	Transformer Model	BanglaHateBERT	Accuracy F1 Score	94.3% 94.1%
[54]	Transformer Model DNN Model	XML-RoBERTa	Precision Recall F1 Score MCC	84% 83% 83% 67
[14]	CMLM Ensemble Model RNN Model	SVM, MNB, Logit, ANN RF LSTM	Accuracy	82.2%
[67]	CMLM DL Model	MNB CNN	Accuracy Precision Recall F1 Score	84% (MNB) 87% (MNB) 85% (CNN) 83% (MNB)
[60]	Hybrid Model	CNN + attention	Accuracy	77%
[17]	Lexicon CMLM DL Model	BengSwearLex Bi-LSTM	Precision Recall F1 Score	99.8% (BengSwearLex) 94% (Bi-LSTM) 91.7% (BengSwearLex)
[153]	Transformer Model	BERT GRU	Precision Recall F1 Score	91% 91% 90%
[154]	Hybrid Model	CNN+(Neural Network Tree-based)	Accuracy	78%
[155]	Hybrid Model	CNN+BERT	Accuracy Precision Recall F-score	95.67% 93.55% 92.67% 94.44%
[156]	Transformer Model	BERT	Weighted-F1	92.3%
[157]	Hybrid Model	CNN+BiLSTM	Accuracy	90%

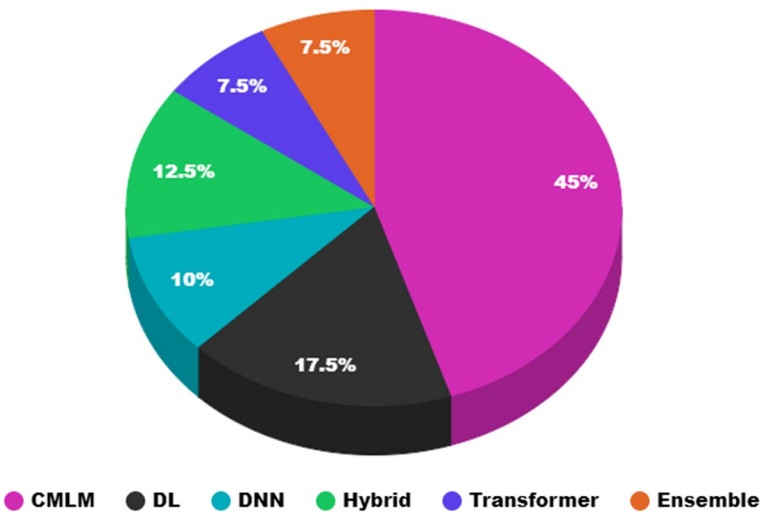

Fig. 15 Distribution of the models used in Bengali text HS detection: Percentage of all applied models

several dialects in Bengali, which makes the HS detection task even more challenging because people prefer to express their thoughts and views in their own dialect or by combining Bengali and English, known as "Banglish" [2, 61]. However, the

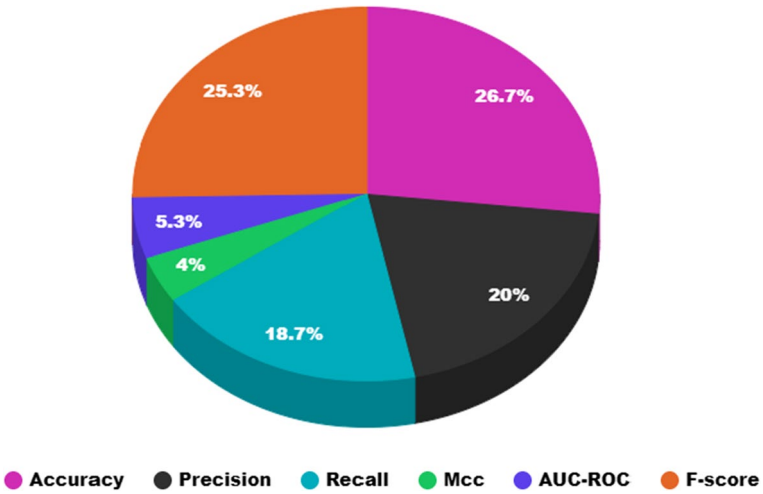

Fig. 16 Distribution of the evaluation metrics used in Bengali text HS detection: Percentage of all applied metrics

fluid nature of Bengali, with its user's frequent misspellings, grammatical inaccuracies as well as multiple words with similar meanings, makes it challenging to detect HS accurately. For example, [41] discovered the same term with several spellings are used to describe the same purpose, e.g., "নারি," "নারী," "নাড়ী," "নাড়ি" ((woman). Another challenge is the variation of Bengali verb forms. For instance, the Bengali verb "খাওয়া" (to eat) can be found in many different forms such as"খাই", "খে য়ে ছি", "খে য়ে ছ", "খাবে ", "খাচ্ছে". Therefore, it can be difficult to recognize these distinct forms as being the same verb without stemming or lemmatization.

Table 6 Example of Bengali HS in different regional Bengali dialects

Region	HS example
Chittagong	মাগীর প নেয়া
Barishal	খানকি র প নেলা
Rajshahi	বে শ্যার ব্যাটা
Pabna	নটি র ছাওয়াল
Mymensingh	মাগীর পুৎ

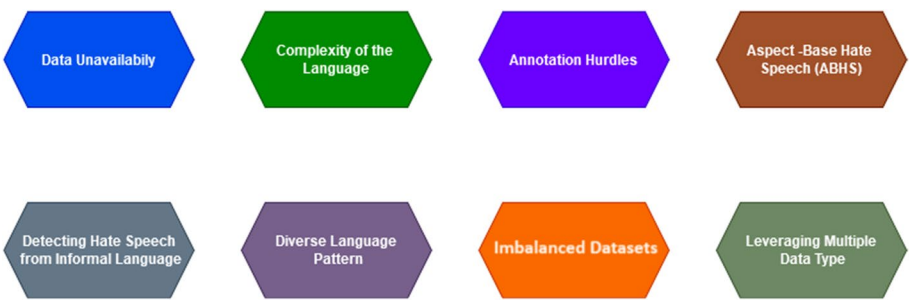

Fig. 17 Challenges and limitations in Bengali HS detection

- Diverse Language Patterns: Bengali, endonym Bangla, is the sixth most spoken language in the world and is also well-known around the world [12]. People in Bangladesh speak two distinct dialects of Bengali, and some speak a local dialect. In Bangladesh, individuals communicate with each other across 64 districts using 55 distinct regional dialects. Bengali is spoken by a diverse population with a range of cultural and social backgrounds. Because of the sociocultural distinctions between Bengali native speakers, it is highly unlikely that the corresponding Bengali words are used in a similar context within the Bengali language [17, 52]. The existence of this diversity across various contexts makes it difficult to develop models that can capture the nuances of HS accurately. Table 6 provides examples of offensive language in Bengali originating from different regional Bengali dialects, with the corresponding English translation being "son of a sex worker."
- Leveraging Multiple Data Types: The use of hateful language in Bengali is not restricted to written text only but can also contain a wide range of multimedia formats such as images, videos, emojis, and several other types of media. For instance, in [68], the authors collected a dataset of videos containing HS and then extracted the audio and converted it to a textual representation to apply several ML models. However, integrating several modalities creates considerable hurdles for HS detection models. Another challenge is the lack of suitable multimodal analysis models to address HS in Bengali beyond the written text effectively.
- Annotation Hurdles: The difficulty in annotating hate speech primarily stems from the ambiguity surrounding the definition of hate speech. HS is a subset of a broader category of offensive language, and as a result, these terms are frequently used interchangeably [162]. Additionally, recognizing hate speech can be subjective, as it frequently relies on the perspective of the interpreter [163]. For Bengali HS, there is no standardized annotation guideline [53]. In the pursuit of Malay hate speech detection, the authors of [164] encountered similar challenges. As a result, researchers often develop annotation guidelines based on their criteria, leading to inconsistencies in the identification and labeling of HS. Moreover, most scholars rely on comments and posts from platforms such as Twitter, Facebook, or YouTube [165] to collect data for hate speech identification. However, the criteria employed by YouTube for detecting hate speech may not align with the content that Twitter or Facebook has labeled as such. In addition, identifying hurtful language necessitates a thorough comprehension of the context in which the statements were expressed. Without sufficient context, annotators may find it difficult to classify content, which could result in incorrect classifications. These challenges make it difficult to compare results across different studies and datasets.

  Another challenge in annotation is the diversity of standards. The vocabulary, jargon, and language styles related to HS vary across different domains [166]. For example, Facebook, Twitter (now known as X), YouTube, and the Code of Conduct of the European Union have distinct definitions of hate speech [18]. The variety of language usage and subtleties in context present a significant obstacle to the identification of hate speech across domains. Detection models need to be flexible and context-aware since different platforms and institutions may classify particular expressions or phrases in different ways. Let us consider the following statement as an example.

"গাজা যুদ্ধরে জন্য ইসরায়লে দায়ী" ("Israel is responsible for Gaza war."). While Facebook sometimes considers that statement as a hate speech and removes the content, Twitter (now X) regards it as freedom of speech.

- Detecting Hate Speech From Informal Language: On social sites, individuals express their opinions and feelings through slang, sarcasm, humor, irony, and other means. In some circumstances, it might be challenging to distinguish between HS, hilarious speech, and sarcasm in Bengali [65]. Humor and sarcasm can be confused with HS in some instances, while HS can sometimes be disguised as hilarious or sarcastic language in others. On the other hand, code-mixed languages such as Bengali-Hindi and Romanized-Bengali hate speech detection are complicated tasks [62]. In communities speaking Bengali and Hindi, code-switching is a frequent linguistic occurrence that makes HS detection algorithms more difficult to use. Expressions in Hindi or Bengali often seamlessly integrate English or other regional languages [167]. For code-mixed languages, accuracy is a major disadvantage of machine translation approaches [168].
- Aspect-Based Hate Speech (ABHS): Hate speech sometimes depends on different aspect. A deep understanding of societal norms is necessary because hate speech is context-dependent and influenced by cultural factors. The intricacy of language is increased by its dynamic evolution over time, necessitating frequent algorithm updates to stay up to date with new linguistic trends and expressions. Therefore, it can be difficult to recognize and understand the complex ways in which hate speech targets different facets of a person's identity. These challenges are ethical as well as technical. Let us look at the statement "সাকিব ভাল খেলে ", which translates to "Shakib plays well." Shakib is a well-known name in Bangladesh. He is one of the most well-known cricket players and the highest-ranking all-rounder. When this sentence is considered from a cricket perspective, it is not hate speech. However, if we consider this sentence in the movie industry context, it is hate speech. The real intended meaning is that "Shakib (hero) can f**k well." Now, let us look at the statement "শোরুম সাকিবের শোরুম উদ্ভোদন সফল হোক", which translates to "May Shakib's showroom launch be a success." This statement could be considered offensive to the cricketer Shakib on the pitch. However, in business, this statement is intended as a positive and supportive wish and, thus, is not hate speech. The above examples highlight the significance of fully understanding the context-specific nuances in language to build an aspect or domain-based dataset. Unfortunately, no substantial research has been conducted in this area for Bengali, and no aspect-based Bengali HS dataset is available. Future researchers could adapt aspect-based sentiment analysis methods developed for other languages, such as English [169, 170], or develop novel methods dedicated to Bengali.

Conclusion

In the field of NLP research, there has been an increase in HS detection in Bengali. This paper provides a comprehensive analysis of the available literature concerning the detection of HS in Bengali. We have thoroughly explored various aspects, including the main datasets used, textual features employed, machine learning models utilized, evaluation criteria adopted, and the challenges encountered in this field. Our study observed that

Facebook and YouTube are prominent data sources for studying Bengali HS. However, many datasets are small and lack reliability due to the data collection and processing techniques employed. Most implementations used classical machine learning models (CMLMs), such as SVM and NB. Nonetheless, it is worth noting that deep learning-based techniques, such as CNNs, RNNs, and transformers, were also employed and demonstrated promising results. Common measures, including accuracy, precision, F1 score, and recall, were frequently used for assessing the efficacy of Bengali HS detection models.

Despite the progress made in detecting HS in the Bengali language, difficulties remain due to linguistic nuances, slang, and context-specific expressions. To overcome these challenges, future research should focus on developing more sophisticated models, incorporating domain-specific knowledge, and leveraging the latest advancements in NLP. Promoting collaboration among researchers, developing standardized annotation guidelines, addressing the specific linguistic challenges of the Bengali language, and implementing suitable evaluation benchmarks will also contribute to the advancement of Bengali HS detection.

Acknowledgements
The authors thank their respective institutions for all provided support.

Author contributions
Abdullah Al Maruf, Ahmad Jainul Abidin, Md. Mahmudul Haque, Zakaria Masud Jiyad, and Aditi Golder collected the literature survey data, carried out data analysis, and wrote the initial draft of the manuscript. Raaid Alubady conceptualized the survey work and guided throughout the process. Zeyar Aung mentored the survey work and completed the final revision of the manuscript. All authors reviewed the manuscript.

Funding
This work was funded in part by Khalifa University of Science and Technology, Abu Dhabi, UAE.

Availability of data and materials
The availability of each dataset is listed in Table 2 of "Dataset Description" section.

Declarations

Ethics approval and consent to participate
Not applicable.

Consent for publication
All the authors have consented to the submission of this manuscript to the journal.

Competing interests
None declared.

Received: 3 July 2023 Accepted: 11 July 2024
Published online: 23 July 2024

References

1. Dhar S, Bose I. Empirical study of social capital factors formed through digital social networking, in Proceedings of the 2019 International Conference on Information Systems (ICIS), 2019:2983.
2. Mridha MF, Wadud MAH, Hamid MA, Monowar MM, Abdullah-Al-Wadud M, Alamri A. L-Boost: identifying offensive texts from social media post in Bengali. IEEE Access. 2021;9:164681–99.
3. Nobata C, Tetreault J, Thomas A, Mehdad Y, Chang Y. Abusive language detection in online user content, in Proceedings of the 25th International Conference on World Wide Web (WWW), 2016:145–153.
4. Sharif O, Hoque MM. Identification and classification of textual aggression in social media: Resource creation and evaluation, in Combating Online Hostile Posts in Regional Languages during Emergency Situation: First International Workshop, CONSTRAINT 2021, Collocated with AAAI 2021, Virtual Event, February 8, 2021, Revised Selected Papers 1, pp. 9–20, Springer, 2021.
5. Lucky EAE, Sany MMH, Keya M, Khushbu SA, Noori SRH. An attention on sentiment analysis of child abusive public comments towards Bangla text and ML, in Proceedings of the 2021 12th International Conference on Computing Communication and Networking Technologies (ICCCNT), pp. 1–6, IEEE, 2021.

6. MacAvaney S, Yao H-R, Yang E, Russell K, Goharian N, Frieder O. Hate speech detection: challenges and solutions. PLOS One. 2019;14(8): e0221152.
7. Kearns C, Sinclair G, Black J, Doidge M, Fletcher T, Kilvington D, Liston K, Lynn T, Rosati P. A scoping review of research on online hate and sport. Commun Sport. 2022;11(2):21674795221132730.
8. Albadi N, Kurdi M, Mishra S. Are they our brothers? analysis and detection of religious hate speech in the Arabic Twittersphere, in Proceedings of the 2018 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining (ASONAM), pp. 69–76, IEEE, 2018.
9. Chowdhury AG, Didolkar A, Sawhney R, Shah R. ARHNet-leveraging community interaction for detection of religious hate speech in Arabic, in Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL): Student Research Workshop, 2019:273–280.
10. Corazza M, Menini S, Cabrio E, Tonelli S, Villata S. A multilingual evaluation for online hate speech detection. ACM Trans Int Technol. 2020;20(2):1–22.
11. Shahadat AB, Rony M, Rahman M, Anwar M, Joy EA. et al., Hate speech detection from social networking posts using CNN and XGBoost, b.sc. thesis, Brac University, Bangladesh, 2019.
12. Central Intelligence Agency, USA, The world factbook. https://www.cia.gov/the-world-factbook/. Accessed 21 Feb 2018.
13. A. Al-Hassan and H. Al-Dossari, Detection of hate speech in social networks: A survey on multilingual corpus, in *Proceedings of the 6th International Conference on Computer Science and Information Technology (ICCSIT)*, vol. 10, pp. 10–5121, 2019.
14. Emon EA, Rahman S, Banarjee J, Das AK, Mittra T. A deep learning approach to detect abusive Bengali text, in 2019 7th International Conference on Smart Computing & Communications (ICSCC), pp. 1–5, IEEE, 2019.
15. Romim N, Ahmed M, Islam M, Sharma AS, Talukder H, Amin MR. et al., BD-SHS: A benchmark dataset for learning to detect online Bangla hate speech in different social contexts, arXiv preprint arXiv:2206.00372, 2022.
16. Holgate E, Cachola I, Preoţiuc-Pietro D, Li JJ. Why swear? analyzing and inferring the intentions of vulgar expressions, in Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2018:4405–4414.
17. Sazzed S. Identifying vulgarity in Bengali social media textual content. Peer J Comput Sci. 2021;7: e665.
18. Fortuna P, Nunes S. A survey on automatic detection of hate speech in text. ACM Comput Surv. 2018;51(4):1–30.
19. Schmidt A, Wiegand M. A survey on hate speech detection using natural language processing, in Proceedings of the 5th International Workshop on Natural Language Processing for Social Media (SocialNLP), 2017:1–10.
20. Mullah NS, Zainon WMNW. Advances in machine learning algorithms for hate speech detection in social media: a review. IEEE Access. 2021;9:88364–76.
21. Alkomah F, Ma X. A literature review of textual hate speech detection methods and datasets. Information. 2022;13(6):273.
22. Subramanian M, Sathiskumar VE, Deepalakshmi G, Cho J, Manikandan G. A survey on hate speech detection and sentiment analysis using machine learning and deep learning models. Alexandria Eng J. 2023;80:110–21.
23. Gandhi A, Ahir P, Adhvaryu K, Shah P, Lohiya R, Cambria E, Poria S, Hussain A. Hate speech detection: a comprehensive review of recent works, Expert Systems, 2024:e13562.
24. Aldjanabi W, Dahou A, Al-qaness MAA, Elaziz MA, Helmi AM, Damaševičius R. Arabic offensive and hate speech detection using a cross-corpora multi-task learning model. Informatics. 2021;8(4):69.
25. Faris H, Aljarah I, Habib M, Castillo PA. Hate speech detection using word embedding and deep learning in the Arabic language context, in Proceedings of the 2020 International Conference on Pattern Recognition Applications and Methods (ICPRAM), 2020:453–460.
26. Duwairi R, Hayajneh A, Quwaider M. A deep learning framework for automatic detection of hate speech embedded in Arabic tweets. Arab J Sci Eng. 2021;46:4001–14.
27. Anezi FYA. Arabic hate speech detection using deep recurrent neural networks. Appl Sci. 2022;12(12):6010.
28. Sigurbergsson GI, Derczynski L. Offensive language and hate speech detection for Danish, in Proceedings of the 12th Language Resources and Evaluation Conference (LREC), 2020:3498–3508.
29. Dowlagar S, Mamidi R. A survey of recent neural network models on code-mixed Indian hate speech data, in Forum for Information Retrieval Evaluation, 2021:67–74.
30. Santosh TYSS, Aravind KVS. Hate speech detection in Hindi-English code-mixed social media text, in Proceedings of the ACM India Joint International Conference on Data Science and Management of Data, 2019:310–313.
31. Rizwan H, Shakeel MH, Karim A. Hate-speech and offensive language detection in Roman Urdu, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020:2512–2522.
32. Alfina I, Mulia R, Fanany MI, Ekanata Y. Hate speech detection in the indonesian language: A dataset and preliminary study, in Proceedings of the 2017 International Conference on Advanced Computer Science and Information Systems (ICACSIS), pp. 233–238, IEEE, 2017.
33. Warner W, Hirschberg J. Detecting hate speech on the world wide web, in Proceedings of the 2nd Workshop on Language in Social Media (LSM), 2012:19–26.
34. European Commission, Countering illegal hate speech online-commission initiative shows continued improvement, further platforms join. https://ec.europa.eu/commission/presscorner/detail/en/IP_18_261, 2018. Accessed: 2023-06-26.
35. ILGA-Europe, Anti-LGBTI attacks in your country: Our A-to-Z of hate-crime across Europe and Central Asia. https://www.ilga-europe.org/blog/anti-lgbti-attacks-your-country/. Accessed 16 Apr 2023.
36. Facebook, Community standards. https://transparency.fb.com/en-gb/policies/community-standards/hate-speech/. Accessed 16 Apr 2023.
37. Twitter, Rules and policies. https://help.twitter.com/en/rules-and-policies/hateful-conduct-policy. Accessed 16 Apr 2023.

38. YouTube, "YouTube policy." https://transparencyreport.google.com/youtube-policy/featured-policies/hate-speech?hl=en. Accessed 16 Apr 2023.
39. Silva L, Mondal M, Correa D, Benevenuto F, Weber I. Analyzing the targets of hate in online social media, in Proceedings of the 2016 International AAAI Conference on Web and Social Media (ICWSM), 2016;10:687–690.
40. Sultana S, Redoy MOF, Al Nahian J, Masum AKM, Abujar S. Detection of abusive Bengali comments for mixed social media data using machine learning, Research Square preprint, 2023.
41. Remon NI, Tuli NH, Akash RD. Bengali hate speech detection in public Facebook pages, in Proceedings of the 2022 International Conference on Innovations in Science, Engineering and Technology (ICISET), pp. 169–173, IEEE, 2022.
42. Eshan SC, Hasan MS. An application of machine learning to detect abusive Bengali text, in Proceedings of the 2017 20th International conference of computer and information technology (ICCIT), pp. 1–6, IEEE, 2017.
43. Sullaway M. Psychological perspectives on hate crime laws. Psychol Public Pol Law. 2004;10(3):250.
44. Poynting S. Hate crime, in The Routledge companion to criminological theory and concepts, pp. 301–305, Routledge, 2018.
45. Kitchenham B, Charters S. Guidelines for performing systematic literature reviews in software engineering, tech. rep., Keele University, UK and Durham University, UK, 2007. version 2.3.
46. Kitchenham B. Procedures for performing systematic reviews, Tech. Rep. TR/SE-0401, Keele University, UK, 2004.
47. Liberati A, Altman DG, Tetzlaff J, Mulrow C, Gøtzsche PC, Ioannidis JPA, Clarke M, Devereaux PJ, Kleijnen J, Moher D. The PRISMA statement for reporting systematic reviews and meta-analyses of studies that evaluate health care interventions: explanation and elaboration, Annals of Internal Medicine, 2009;151(4):W–65.
48. Ahammed S, Rahman M, Niloy MH, Chowdhury SMH. Implementation of machine learning to detect hate speech in Bangla language, in Proceedings of the 2019 8th International Conference System Modeling and Advancement in Research Trends (SMART), pp. 317–320, IEEE, 2019.
49. Romim N, Ahmed M, Talukder H, Saiful Islam M. 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, 2021.
50. Rezaul Karim M, Kanti Dey S, Islam T, Sarker S, Hasan Menon M, Hossain K, Raja Chakravarthi B, Hossain MA, Decker S. DeepHateExplainer: Explainable hate speech detection in under-resourced Bengali language, in Proceedings of the 2021 IEEE 8th International Conference on Data Science and Advanced Analytics (DSAA), 2021:1–10.
51. Jahan MS, Haque M, Arhab N, Oussalah M. BanglaHateBERT: BERT for abusive language detection in Bengali, in Proceedings of the 2nd International Workshop on Resources and Techniques for User Information in Abusive Language Analysis, 2022:8–15.
52. Faisal Ahmed M, Mahmud Z, Biash ZT, Ryen AAN, Hossain A, Ashraf FB. Cyberbullying detection using deep neural network from social media comments in Bangla language, arXiv preprint arXiv:2106.04506, 2021.
53. Romim N, Ahmed M, Islam MS, Sharma AS, Talukder H, Amin MR. BD-SHS: A benchmark dataset for learning to detect online Bangla hate speech in different social contexts, in Proceedings of the 13th Language Resources and Evaluation Conference (LREC), 2022:5153–5162.
54. Rezaul Karim M, Kanti Dey S, Islam T, Raja Chakravarthi B Multimodal hate speech detection from Bengali memes and texts, in SPELLL: International Conference on Speech and Language Technologies for Low-resource Languages, 2023:293–308.
55. Islam T, Ahmed N, Latif S. An evolutionary approach to comparative analysis of detecting Bangla abusive text. Bull Elect Eng Inf. 2021;10(4):2163–9.
56. Sarker M, Hossain MF, Liza FR, Sakib SN, Al Farooq A. A machine learning approach to classify anti-social Bengali comments on social media, in Proceedings of the 2022 International Conference on Advancement in Electrical and Electronic Engineering (ICAEEE), pp. 1–6, IEEE, 2022.
57. Chakraborty P, Seddiqui MH. Threat and abusive language detection on social media in Bengali language, in Proceedings of the 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), pp. 1–6, IEEE, 2019.
58. Ishmam AM, Sharmin S. Hateful speech detection in public Facebook pages for the Bengali language, in Proceedings of the 2019 18th IEEE International Conference on Machine Learning and Applications (ICMLA), pp. 555–560, IEEE, 2019.
59. Banik N, Rahman MHH. Toxicity detection on Bengali social media comments using supervised models, in Proceedings of the 2019 2nd International Conference on Innovation in Engineering and Technology (ICIET), pp. 1–5, IEEE, 2019
60. A. K. Das, A. Al Asif, A. Paul, and M. N. Hossain, Bangla hate speech detection on social media using attention-based recurrent neural network, *Journal of Intelligent Systems*, vol. 30, no. 1, pp. 578–591, 2021.
61. Ghosh T, Chowdhury AAK, Banna MHA, Nahian MJA, Kaiser MS, Mahmud M. A hybrid deep learning approach to detect Bangla social media hate speech, in Proceedings of International Conference on Fourth Industrial Revolution and Beyond: IC4IR 2021, pp. 711–722, Springer, 2022.
62. Das M, Banerjee S, Saha P, Mukherjee A. Hate speech and offensive language detection in Bengali, in Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (IJCNLP) Volume 1: Long Papers, pp. 286–296, 2022.
63. Islam M, Hossain MS, Akhter N. Hate speech detection using machine learning in Bengali languages, in Proceedings of the 2022 6th International Conference on Intelligent Computing and Control Systems (ICICCS), pp. 1349–1354, IEEE, 2022.
64. Jubaer ANM, Sayem A, Rahman MA. Bangla toxic comment classification (machine learning and deep learning approach), in Proceedings of the 2019 8th international conference system modeling and advancement in research trends (SMART), pp. 62–66, IEEE, 2019.
65. Hussain MG, Al Mahmud T, Akthar W. An approach to detect abusive Bangla text, in Proceedings of the 2018 International Conference on Innovation in Engineering and Technology (ICIET), pp. 1–5, IEEE, 2018.
66. Alam T, Khan A, Alam F. Bangla text classification using transformers, arXiv preprint arXiv:2011.04446, 2020.

67. Ahmed MT, Rahman M, Nur S, Islam A, Das D. Deployment of machine learning and deep learning algorithms in detecting cyberbullying in Bangla and Romanized Bangla text: A comparative study, in Proceedings of the 2021 International Conference on Advances in Electrical, Computing, Communication and Sustainable Technologies (ICAECT), pp. 1–10, IEEE, 2021.
68. Junaid MIH, Hossain F, Rahman RM. Bangla hate speech detection in videos using machine learning, in Proceedings of the 2021 IEEE 12th Annual Ubiquitous Computing, Electronics & Mobile Communication Conference (UEMCON), pp. 0347–0351, IEEE, 2021.
69. Karim MR, Chakravarthi BR, McCrae JP, Cochez M. Classification benchmarks for under-resourced Bengali language based on multichannel convolutional-LSTM network, in Proceedings of the 2020 IEEE 7th International Conference on Data Science and Advanced Analytics (DSAA), pp. 390–399, IEEE, 2020.
70. Köffer S, Riehle DM, Höhenberger S, Becker J. Discussing the value of automatic hate speech detection in online debates, Multikonferenz Wirtschaftsinformatik (MKWI 2018): Data Driven X-Turning Data in Value. Germany: Leuphana; 2018.
71. Vrysis L, Vryzas N, Kotsakis R, Saridou T, Matsiola M, Veglis A, Arcila-Calderón C, Dimoulas C. A web interface for analyzing hate speech. Fut Int. 2021;13:80.
72. Saleem HM, Dillon KP, Benesch S, Ruths D. A web of hate: Tackling hateful speech in online social spaces, arXiv preprint arXiv:1709.10159, 2017.
73. Sharma AS, Mridul MA, Islam MS. Automatic detection of satire in Bangla documents: A CNN approach based on hybrid feature extraction model, in Proceedings of the 2019 International Conference on Bangla Speech and Language Processing (ICBSLP), pp. 1–5, IEEE, 2019.
74. Shibli GMS, Shawon MTR, Nibir AH, Miandad MZ, Mandal NC. Automatic back transliteration of Romanized Bengali (Banglish) to Bengali, Iran Journal of Computer Science, 2022:1–12.
75. Defersha NB, Tune KK. Detection of hate speech text in Afan Oromo social media using machine learning approach. Ind J Sci Technol. 2021;14(31):2567–78.
76. Maruf AA, Biplob MNH, Khanam F. Covid-19 vaccine sentiment detection and analysis using machine learning technique and NLP, in Proceedings of the 2022 International Conference on Machine Intelligence and Emerging Technologies (MIET), pp. 401–414, Springer, 2022.
77. Kulai A, Sankhe M, Anglekar S, Halbe A. Emotion analysis of Covid tweets using FastText supervised classifier model, in Proceedings of the 2021 International Conference on Communication Information and Computing Technology (ICCICT), pp. 1–6, IEEE, 2021.
78. Dadgar S, Neshat M. A novel hybrid multi-modal deep learning for detecting hashtag incongruity on social media. Sensors. 2022;22(24):9870.
79. Hossain E, Sharif O, Hoque MM. MUTE: A multimodal dataset for detecting hateful memes, in Proceedings of the 2nd Conference of the Asia-Pacific Chapter of the Association for Computational Linguistics and the 12th International Joint Conference on Natural Language Processing (AACL-IJCNLP): Student Research Workshop, 2022:32–39.
80. Sadiq S, Mehmood A, Ullah S, Ahmad M, Choi GS, On B-W. Aggression detection through deep neural model on twitter. Fut Gen Comput Syst. 2021;114:120–9.
81. Kocoń J, Figas A, Gruza M, Puchalska D, Kajdanowicz T, Kazienko P. Offensive, aggressive, and hate speech analysis: from data-centric to human-centered approach. Inf Process Manag. 2021;58(5): 102643.
82. Sazzed S. Abusive content detection in transliterated Bengali-English social media corpus, in Proceedings of the 5th Workshop on Computational Approaches to Linguistic Code-Switching (CALCS), 2021:125–130.
83. S. Ghosh, M. Suri, P. Chiniya, U. Tyagi, S. Kumar, and D. Manocha, CoSyn: Detecting implicit hate speech in online conversations using a context synergized hyperbolic network, in *Proceedings of the 2023 Conference on Empirical Methods in Natural Language Processing (EMNLP)*, pp. 6159–6173, 2023.
84. Khan MU, Abbas A, Rehman A, Nawaz R. Hateclassify: a service framework for hate speech identification on social media. IEEE Int Comput. 2020;25(1):40–9.
85. S. Arora and S. Agarwal, Active learning for natural language processing, tech. rep., Language Technologies Institute, School of Computer Science, Carnegie Mellon University, USA, 2007.
86. Tharwat A, Schenck W. A survey on active learning: state-of-the-art, practical challenges and research directions. Mathematics. 2023;11(4):820.
87. B. Settles, Active learning literature survey, tech. rep., Department of Computer Science, University of Wisconsin-Madison, USA, 2009.
88. H. Cañizares-Díaz, A. Piad-Morffis, S. Estevez-Velarde, Y. Gutiérrez, Y. A. Cruz, A. Montoyo, and R. Muñoz, Active learning for assisted corpus construction: A case study in knowledge discovery from biomedical text, in *Proceedings of the International Conference on Recent Advances in Natural Language Processing (RANLP)*, pp. 216–225, 2021.
89. M. A. U. Haque, A. Rahman, and M. A. Hashem, Sentiment analysis in low-resource Bangla text using active learning, in *Proceedings of the 2021 5th International Conference on Electrical Information and Communication Technology (EICT)*, pp. 1–6, IEEE, 2021.
90. Palakodety S, KhudaBukhsh AR, Carbonell JG. Voice for the voiceless: active sampling to detect comments supporting the Rohingyas. Proc AAAI Conf Artif Intell. 2020;34:454–62.
91. M. G. Hussain and T. Al Mahmud, A technique for perceiving abusive Bangla comments, *Green University of Bangladesh Journal of Science and Engineering*, vol. 4, no. 1, pp. 11–18, 2019.
92. S. Akhter *et al.*, Social media bullying detection using machine learning on Bangla text, in *Proceedings of the 2018 10th International Conference on Electrical and Computer Engineering (ICECE)*, pp. 385–388, IEEE, 2018.
93. T. Yao, Z. Zhai, and B. Gao, Text classification model based on fastText, in *Proceedings of the 2020 IEEE International Conference on Artificial Intelligence and Information Systems (ICAIIS)*, pp. 154–157, IEEE, 2020.
94. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, Kaiser Ł, Polosukhin I. Attention is all you need. Adv Neural Inf Process Syst. 2017;30:6000–10.
95. Niu Z, Zhong G, Yu H. A review on the attention mechanism of deep learning. Neurocomputing. 2021;452:48–62.
96. F. A. Acheampong, H. Nunoo-Mensah, and W. Chen, Transformer models for text-based emotion detection: A review of BERT-based approaches, *Artificial Intelligence Review*, pp. 1–41, 2021.

97. Bhattacharjee A, Hasan T, Ahmad WU, Samin K, Islam MS, Iqbal A, Rahman MS, Shahriyar R. BanglaBERT: language model pretraining and benchmarks for low-resource language understanding evaluation in Bangla. Find Assoc Comput Linguistics NAACL. 2022;2022:1318–27.
98. T. Pires, E. Schlinger, and D. Garrette, How multilingual is multilingual BERT?, in *Proceedings of the 57th Annual Meeting of the Association for Computational Linguistics (ACL)*, pp. 4996–5001, 2019.
99. 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, *arXiv preprint*arXiv:1911.02116*, 2019*.
100. Baruah A, Das K, Barbhuiya F, Dey K, Aggression identification in English, Hindi and Bangla text using BERT, RoBERTa and SVM," in *Proceedings of the 2nd Workshop on Trolling, Aggression and Cyberbullying (TRAC)*, pp. 76–82, 2020.
101. Hossain MR, Hoque MM, Siddique N. Leveraging the meta-embedding for text classification in a resource-constrained language. Eng Appl Artif Intell. 2023;124: 106586.
102. Wu X, Cai Y, Kai Y, Wang T, Li Q. Task-oriented domain-specific meta-embedding for text classification, in Proceedings of the 2020 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2020:3508–3513.
103. Kamath CN, Bukhari SS, Dengel A. Comparative study between traditional machine learning and deep learning approaches for text classification, in Proceedings of the 2018 ACM Symposium on Document Engineering (DocEng), 2018:1–11.
104. Das K, Behera RN. A survey on machine learning: concept, algorithms and applications. Int J Innov Res Comput Commun Eng. 2017;5(2):1301–9.
105. O. L. Mangasarian and D. R. Musicant, Lagrangian support vector machines, *Journal of Machine Learning Research*, vol. 1, no. Mar, pp. 161–177, 2001.
106. Cervantes J, Garcia-Lamont F, Rodríguez-Mazahua L, Lopez A. A comprehensive survey on support vector machine classification: applications, challenges and trends. Neurocomputing. 2020;408:189–215.
107. Sarker IH, Kayes ASM, Watters P. Effectiveness analysis of machine learning classification models for predicting personalized context-aware smartphone usage. J Big Data. 2019;6(1):1–28.
108. Kadhim AI. Survey on supervised machine learning techniques for automatic text classification. Artif Intell Rev. 2019;52(1):273–92.
109. LaValley MP. Logistic regression. Circulation. 2008;117(18):2395–9.
110. Kumar GR, Ramachandra GA, Nagamani K. An efficient prediction of breast cancer data using data mining techniques. Int J Innov Eng Technol. 2013;2(4):139.
111. Safavian SR, Landgrebe D. A survey of decision tree classifier methodology. IEEE Trans Syst Man Cybernet. 1991;21(3):660–74.
112. Yang L, Shami A. On hyperparameter optimization of machine learning algorithms: theory and practice. Neurocomputing. 2020;415:295–316.
113. Balaji TK, Annavarapu CSR, Bablani A. Machine learning algorithms for social media analysis: a survey. Comput Sci Rev. 2021;40: 100395.
114. Sagi O, Rokach L. Ensemble learning: a survey. Wiley Interdiscip Rev Data Mining Knowl Discov. 2018;8(4): e1249.
115. Aljero MKA, Dimililer N. A novel stacked ensemble for hate speech recognition. Appl Sci. 2021;11(24):11684.
116. Breiman L. Random forests. Mach Learn. 2001;45:5–32.
117. Islam MZ, Liu J, Li J, Liu L, Kang W. A semantics aware random forest for text classification, in Proceedings of the 28th ACM International Conference on Information and Knowledge Management (CIKM), 2019:1061–1070.
118. Friedman JH. Greedy function approximation: a gradient boosting machine. Ann Stat. 2001;29(5):1189–232.
119. Alzamzami F, Hoda M, El Saddik A. Light gradient boosting machine for general sentiment classification on short texts: a comparative evaluation. IEEE Access. 2020;8:101840–58.
120. Zhou C, Sun C, Liu Z, Lau F. A C-LSTM neural network for text classification, arXiv preprint arXiv:1511.08630, 2015.
121. Yamashita R, Nishio M, Do RKG, Togashi K. Convolutional neural networks: an overview and application in radiology. Insights Imaging. 2018;9:611–29.
122. A. Géron, *Hands-on machine learning with Scikit-Learn and Tensorflow: concepts. tools, and techniques to build intelligent systems*, O'Reilly Media, 2017.
123. Bhatt D, Patel C, Talsania H, Patel J, Vaghela R, Pandya S, Modi K, Ghayvat H. CNN variants for computer vision: history, architecture, application, challenges and future scope. Electronics. 2021;10(20):2470.
124. H. Abdel-Jaber, D. Devassy, A. Al Salam, L. Hidaytallah, and M. El-Amir, A review of deep learning algorithms and their applications in healthcare, *Algorithms*, vol. 15, no. 2, p. 71, 2022.
125. Rezk NM, Purnaprajna M, Nordström T, Ul-Abdin Z. Recurrent neural networks: an embedded computing perspective. IEEE Access. 2020;8:57967–96.
126. Sarker IH. Deep learning: a comprehensive overview on techniques, taxonomy, applications and research directions. SN Comput Sci. 2021;2(6):420.
127. Sharfuddin AA, Tihami MN, Islam MS. A deep recurrent neural network with BiLSTM model for sentiment classification, in Proceedings of the 2018 International conference on Bangla speech and language processing (ICBSLP), pp. 1–4, IEEE, 2018.
128. Liang D, Zhang Y. AC-BLSTM: Asymmetric convolutional bidirectional LSTM networks for text classification, arXiv preprint arXiv:1611.01884, 2016.
129. Cho K, Van Merriënboer B, Gulcehre C, Bahdanau D, Bougares F, Schwenk H, Bengio Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation, in Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), 2014:1724–1734.
130. Zulqarnain M, Ghazali R, Hassim YM, Rehan M. Text classification based on gated recurrent unit combines with support vector machine. Int J Elect Comput Eng. 2020;10(4):3734.
131. Chang Y, Wang X, Wang J, Wu Y, Yang L, Zhu K, Chen H, Yi X, Wang C, Wang Y, et al. A survey on evaluation of large language models. ACM Trans Intell Syst Technol. 2024;15(3):1–45.
132. Wang H, Li J, Wu H, Hovy E, Sun Y. Pre-trained language models and their applications. Engineering. 2022;25:51–65.
133. Kalyan KS. A survey of GPT-3 family large language models including ChatGPT and GPT-4, Natural Language Processing Journal, 2023:100048.

134. Thoppilan R, De Freitas D, Hall J, Shazeer N, Kulshreshtha A, Cheng H-T, Jin A, Bos T, Baker L, Du Y. et al., Lamda: Language models for dialog applications, arXiv preprint arXiv:2201.08239, 2022.
135. Devlin J, Chang M-W, Lee K, Toutanova K. BERT: Pre-training of deep bidirectional transformers for language understanding, in Human Language Technology: Conference of the North American Chapter of the Association of Computational Linguistics (HLT-NAACL), 2019:4171–4186.
136. Lewis M, Liu Y, Goyal N, Ghazvininejad M, Mohamed A, Levy O, Stoyanov V, Zettlemoyer L. BART: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension, in Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (ACL), 2020:7871–7880.
137. Lan Z, Chen M, Goodman S, Gimpel K, Sharma P, Soricut R. ALBERT: A lite BERT for self-supervised learning of language representations, in Proceedings of the 8th International Conference on Learning Representations (ICLR), 2020.
138. Liu Y, Ott M, Goyal N, Du J, Joshi M, Chen D, Levy O, Lewis M, Zettlemoyer L, Stoyanov V. RoBERTa: A robustly optimized BERT pretraining approach, arXiv preprint arXiv:1907.11692, 2019.
139. Yang Z, Dai Z, Yang Y, Carbonell J, Salakhutdinov RR, Le QV. XLNet: generalized autoregressive pretraining for language understanding. Adv Neural Inf Process Syst. 2019;32:5753–63.
140. Clark K, Luong M-T, Le QV, Manning CD. ELECTRA: Pre-training text encoders as discriminators rather than generators, in Proceedings of the 8th International Conference on Learning Representations (ICLR), 2020:1–18.
141. Ni J, Ábrego GH, Constant N, Ma J, Hall KB, Cer D, Yang Y. Sentence-T5: scalable sentence encoders from pretrained text-to-text models. Findings Assoc Comput Linguistics ACL. 2022;2022:1864–74.
142. Zhang J, Zhao Y, Saleh M, Liu P. Pegasus: Pre-training with extracted gap-sentences for abstractive summarization, in Proceedings of the 2020 International Conference on Machine Learning (ICML), pp. 11328–11339, PMLR, 2020.
143. Kalyan KS, Rajasekharan A, Sangeetha S. AMMU: a survey of transformer-based biomedical pretrained language models. J Biomed Inf. 2022;126: 103982.
144. Ayik F. Mastering text classification with BERT: a comprehensive guide," 2023. https://medium.com/@ayikfurkan1/mastering-text-classification-with-bert-a-comprehensive-guide-194ddb2aa2e5.
145. Kennedy CJ, Bacon G, Sahn A, von Vacano C. Constructing interval variables via faceted rasch measurement and multitask deep learning: A hate speech application, arXiv preprint arXiv:2009.10277, 2020.
146. Devlin J, Chang M-W, Lee K, Toutanova K. Bert: Pre-training of deep bidirectional transformers for language understanding, arXiv preprint arXiv:1810.04805, 2018.
147. Abdullakutty F, Naseem U. Decoding memes: a comprehensive analysis of late and early fusion models for explainable meme analysis. Companion Proc ACM Web Conf. 2024;2024:1681–9.
148. Kumar R, Lahiri B, Ojha AK. Aggressive and offensive language identification in Hindi, Bangla, and English: a comparative study. SN Comput Sci. 2021;2(1):26.
149. Sanh V, Debut L, Chaumond J, Wolf T. DistilBERT, a distilled version of BERT: smaller, faster, cheaper and lighter, arXiv preprint arXiv:1910.01108, 2019.
150. Opitz J. From bias and prevalence to macro F1, kappa, and MCC: a structured overview of metrics for multi-class evaluation, tech. rep., Heidelberg University, Germany, 2022.
151. Wuest T, Weimer D, Irgens C, Thoben K-D. Machine learning in manufacturing: advantages, challenges, and applications. Prod Manuf Res. 2016;4(1):23–45.
152. Li H. Deep learning for natural language processing: advantages and challenges. Natl Sci Rev. 2018;5(1):24–6.
153. Keya AJ, Kabir MM, Shammey NJ, Mridha MF, Islam MR, Watanobe Y. G-BERT: an efficient method for identifying hate speech in Bengali texts on social media, IEEE Access, 2023.
154. Aporna AA, Azad I, Amlan NS, Mehedi MHK, Mahbub MJA, Rasel AA. Classifying offensive speech of Bangla text and analysis using explainable AI, in Proceedings of the 6th International Conference on Advances in Computing and Data Sciences (ICACDS), pp. 133–144, Springer, 2022.
155. Saha SK, Mim AA, Akter S, Hosen MM, Shihab AH, Mehedi MHK. BengaliHateCB: A hybrid deep learning model to identify Bengali hate speech detection from online platform, in Proceedings of the 2024 6th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT), pp. 439–444, IEEE, 2024.
156. Nandi A, Sarkar K, Mallick A, De A. Combining multiple pre-trained models for hate speech detection in Bengali, Marathi, and Hindi, Multimedia Tools and Applications, 2024:1–25.
157. Islam MH, Farzana K, Khalil I, Ara S, Shazid MRA, Mehedi MHK. Unmasking toxicity: A comprehensive analysis of hate speech detection in Banglish, in Proceedings of the 2024 6th International Conference on Electrical Engineering and Information & Communication Technology (ICEEICT), pp. 963–968, IEEE, 2024.
158. Nobo TM, Galib M, Rabib HK. A model agnostic explainable approach for detecting cyber bullying in Bangla language using transformer based models, bachelor's thesis. Bangladesh: Islamic University of Technology; 2022.
159. Van Hee C, Lefever E, Verhoeven B, Mennes J, Desmet B, De Pauw G, Daelemans W, Hoste V. Detection and fine-grained classification of cyberbullying events, in Proceedings of the 2015 International Conference Recent Advances in Natural Language Processing (RANLP), 2015:672–680.
160. Founta A, Djouvas C, Chatzakou D, Leontiadis I, Blackburn J, Stringhini G, Vakali A, Sirivianos M, Kourtellis N. Large scale crowdsourcing and characterization of Twitter abusive behavior, in Proceedings of the 2018 International AAAI Conference on Web and Social Media (ICWSM), 2018;12:491–500.
161. E. Omran, E. Al Tararwah, and J. Al Qundus, A comparative analysis of machine learning algorithms for hate speech detection in social media, *Online Journal of Communication and Media Technologies*, vol. 13, no. 4, p. e202348, 2023.
162. Davidson T, Warmsley D, Macy M, Weber I. Automated hate speech detection and the problem of offensive language, in Proceedings of the 2017 International AAAI Conference on Web and Social Media (ICWSM), 2017;11:512–515.
163. Waseem Z. Are you a racist or am I seeing things? annotator influence on hate speech detection on Twitter," in Proceedings of the First Workshop on NLP and Computational Social Science, pp. 138–142, Association for Computational Linguistics, 2016.
164. Maity K, Bhattacharya S, Saha S, Seera M. A deep learning framework for the detection of malay hate speech, IEEE Access, 2023.

165. Vrysis L, Vryzas N, Kotsakis R, Saridou T, Matsiola M, Veglis A, Arcila-Calderón C, Dimoulas C. A web interface for analyzing hate speech. Fut Int. 2021;13(3):80.
166. Salminen J, Hopf M, Chowdhury SA, Jung S-G, Almerekhi H, Jansen BJ. Developing an online hate classifier for multiple social media platforms. Human-centric Comput Inf Sci. 2020;10:1–34.
167. Jahan M, Ahamed I, Bishwas MR, Shatabda S. Abusive comments detection in Bangla-English code-mixed and transliterated text, in Proccedings of the 2019 2nd International Conference on Innovation in Engineering and Technology (ICIET), 2019:1–6.
168. Saadany H, Orasan C, Quintana RC, Carmo Fd, Zilio L. Challenges in translation of emotions in multilingual user-generated content: Twitter as a case study, arXiv preprint arXiv:2106.10719, 2021.
169. Nazir A, Rao Y, Wu L, Sun L. Issues and challenges of aspect-based sentiment analysis: a comprehensive survey. IEEE Trans Affect Comput. 2020;13(2):845–63.
170. Do HH, Prasad PW, Maag A, Alsadoon A. Deep learning for aspect-based sentiment analysis: a comparative review. Expert Syst Appl. 2019;118:272–99.

Publisher's Note