Computers, Materials & Continua

Tech Science Press

https://doi.org/10.32604/cmc.2026.078997

ARTICLE

Semantic-Sentiment Fusion with Deep Learning: A Novel Framework for Hate Speech Detection

Choongwon Kang[1,2], **Haein Lee**[3,4] **and Jang Hyun Kim**[1,2,*]

[1]Department of Applied Artificial Intelligence, Sungkyunkwan University, Seoul, Republic of Korea
[2]Department of Human-Artificial Intelligence Interaction, Sungkyunkwan University, Seoul, Republic of Korea
[3]School of Interdisciplinary Studies, Dongguk University, Seoul, Republic of Korea
[4]Department of Computer Science and Artificial Intelligence, Dongguk University, Seoul, Republic of Korea

*Corresponding Author: Jang Hyun Kim. Email: alohakim@skku.edu

Received: 12 January 2026; Accepted: 18 March 2026; Published: 08 May 2026

ABSTRACT: With the rapid growth of social media and frequent anonymous interactions, hate speech has become widespread. As users express diverse opinions in digital spaces, the need for effective detection remains crucial. To address this, we propose a framework applicable to diverse hate speech types, combining sentence-level semantic representation vectors from the pre-trained Bidirectional Encoder Representations from Transformers (BERT) with sentiment score vectors from the Linguistic Inquiry and Word Count (LIWC) dictionary and the Valence Aware Dictionary for sEntiment Reasoning (VADER). This semantic-sentiment fusion integrates three deep learning models—Convolutional Neural Network (CNN), Long Short-Term Memory (LSTM), and Deep Neural Network (DNN) to enhance detection effectiveness. To verify generalizability, we used four datasets: two binary hate speech detection tasks, two multi-class tasks, and validation on another domain dataset. Results show that the proposed framework achieved the best performance, with accuracy up to 91.34%. This approach provides valuable direction for future research.

KEYWORDS: Hate speech detection; natural language processing (NLP); deep learning; sentiment analysis

1 Introduction

The social media space is a place in which people can freely communicate and interact with one another. It serves not just for entertainment and chatting with friends, but also for work and acquiring new knowledge and abilities [1]. As time passes and evolves, people spend more time on social media, which have integrated into their daily routines. By the year 2025, in excess of 5.4 billion people worldwide were actively engaging with social media, spending an average of 141 min daily on these platforms, and forecast that this figure will exceed 6.5 billion by 2030 [2,3]. Facebook maintained the largest monthly active users at approximately 3.07 billion, followed by Instagram and WhatsApp each with around 3 billion users, and YouTube with approximately 2.58 billion users [4]. As social media continues to expand, people are able to communicate openly and share their viewpoints. Nevertheless, this greater freedom of expression has also contributed to a surge in hate speech, online harassment, and inappropriate content, which can have serious repercussions for the broader society and users [5]. Hate speech on online platforms not only causes conflicts among users but also creates discrimination, intimidation in society, and criminal behavior in the real world [6,7]. Additionally, it sometimes has a negative psychological impact on users, which can lead to suicide in severe cases [8]. Therefore, the need to detect hate speech has increased.

Leading online platforms, like Facebook and Instagram, have implemented systems that allow users to flag harmful content. However, with this approach, user accounts that continually post hate speech may mistakenly be considered harmless, or overtly offensive content may remain unreported for extended periods. Considering the vast amount of content uploaded every day, human editing is not only inefficient but also impossible [9]. To confront the challenges associated with hate speech on these online platforms, various scholars have investigated automated methods for detecting such content. The process of identifying hate speech utilizes techniques from machine learning and deep learning, such as natural language processing (NLP), with the goal of identifying and eliminating harmful content from digital platforms to foster a more secure and tolerant atmosphere for all individuals [10]. Specifically, NLP tasks including feature engineering, lemmatization, and stemming are applied to analyze text as a foundation for detecting hateful content [11].

Over the years, researchers have recognized that hate speech is complex and demands a thorough examination of its linguistic representation, context structure, and user behavior [12]. Transformer-based language models have demonstrated outstanding performance in hate speech detection with reduced computational expenses and minimal requirements for intricate model engineering [13]. Recent studies have adapted transformer-based approaches to detect hate speech across diverse languages, such as low-resource Indian languages, Turkish, and various Arabic dialects [14–16]. Reflecting on this, we assumed that we needed information about the context to enhance understanding of the linguistic representation and contextual structure of each sentence in the hate speech dataset. To obtain this information, we extracted semantic representation vectors from the Bidirectional Encoder Representations from Transformers (BERT) [17], a transformer architecture that summarizes content and context using tokens. Additionally, numerous researchers have examined user psychology to understand behaviors related to hate speech. Various studies have considered the psychological facets of user behavior. These include a study that differentiates the sensitivity to hate speech by gender from a psychological viewpoint, a research work that explores hate speech against immigrants by integrating psychological perspectives, and a study on how online crowd psychology influences cyberbullying perpetrators so that they do not feel guilty about their actions [18–20].

Therefore, we utilized sentiment analysis tools to analyze the psychological features reflected in words within user-written sentences. These features are treated as sentence-level linguistic proxies rather than user-level traits. By identifying sentence-level affective and cognitive word usage patterns, we extracted sentiment recognition scores. The scores were then vectorized and combined with semantic representation vectors to further enhance information regarding linguistic representation. These combined vectors were subsequently used with deep learning models. In summary, this study created a new framework by combining semantic representation vectors generated through a transformer-based model and sentiment scores extracted by two sentiment analysis tools with deep learning models. To assess its generalizability, we employed four different types of hate speech datasets and validated the framework through cross-domain effectiveness testing on the sarcasm domain. The aim of this methodology is to improve performance over using a single model by combining BERT with deep learning techniques, chosen for their strength in obtaining text representations based on context and word positioning, thereby minimizing the loss of critical information [21]. Additionally, by incorporating sentiment analysis tools to gather information on psychological traits, behavior and emotion analysis, this approach demonstrates a more significant performance improvement than relying on only a single sentiment analysis tool or deep learning model [22]. The principal contributions are outlined as follows:

- A novel semantic–sentiment fusion framework is proposed, integrating BERT semantic representation vectors with sentiment score vectors derived from LIWC and VADER for hate speech detection.
- An empirical evaluation is performed to determine the most effective combination of three deep learning models applied to the integrated vectors for enhanced detection performance.

- The generalizability of the proposed framework is assessed through comprehensive experiments across four different categories of hate speech datasets encompassing both binary and multi-class classification tasks.
- Cross-domain effectiveness is further demonstrated by applying the proposed framework not only to hate speech datasets but also to the sarcasm domain, where strong performance is observed.

2 Related Work

Our study covers three related areas: the definition of hate speech, earlier studies employing sentiment analysis to detect hate speech, and prior research implementing machine learning and neural network methodologies to discern harmful contents.

2.1 Definition of Hate Speech

Hate speech such as racism remains present in online environments [23]. Detecting hate speech is crucial because it is closely linked to actual hate crimes [24]. The conceptualization of hate speech varies according to the particular perspective. Within academic study, hate speech is conceptualized as "language that undermines or provokes violence against communities based on attributes such as race, religion, or sexual orientation" [25]. Likewise, hate speech is described as "behavior that targets or disparages an individual or collective entities distinguished by traits including racial provenance, cultural heritage, religious creed, physical ability, gender, chronological age, or sexual predilections and gender self-identification" [26]. Online platforms such as Facebook, Instagram, and Threads, which are operated by Meta, define hate speech as direct attacks against individuals based on characteristics such as race, ethnicity, caste, religion, national origin, and gender, and strictly prohibit such content [27]. Consequently, what is considered hate speech varies based on the perspective, and there are many challenges in determining what constitutes hate speech.

2.2 Sentiment Analysis of Hate Speech

Detecting languages that include abusive, offensive, or insulting words is more challenging than expected because they can encompass a wide range of racial and minority insults, extend across sentence boundaries, and even be fluent and grammatical [26]. The personalities of those who post in these languages are often influenced by their characteristics [28]. Additionally, these languages are predominantly found in social networking societies, where the sharing of hateful content, topics, and comments often leads to stronger bonds among users [29,30]. To prevent these hateful expressions from spreading in social-network societies, researchers conduct sentiment analyses, considering factors such as personality, emotion, and psychology to detect and analyze them. Rodriguez et al. [31] conducted a study focused on the automatic identification of hate speech on Facebook by crawling posts and comments from the platform. They employed JAMMIN, which categorizes emotions in a text into eight types. Their research focused on filtering out negative vocabulary and used term frequency-inverse document frequency (TF-IDF) [32] to examine word frequency, clustering similar content together. Araque and Iglesias [33] utilized AffectiveSpace and SenticNet, which are methodologies designed to detect human emotions in text, to extract affective traits from hate speech data. They employed techniques like the TF-IDF and SIMON models in order to achieve word embedding and then combined all four approaches to classify hate speech. Their results showed that combining affect-aware features with text effectively enhanced performance. Zhou et al. [34] created a model called SKS that leverages both the sentiment features of an utterance and exogenous affective knowledge to enhance hate speech identification. This model employs a multi-task learning structure to capture the interactions between sentiment analysis and hate speech detection, while also acquiring tailored attributes for detecting hate speech. Sheth et al. [35] introduced a transformer-based causality framework named

PEACE, devised for identifying hateful content, that emphasizes extracting the overarching emotional tone and aggression embedded in the text. Consequently, researchers are continuously conducting studies to enhance hate speech detection through diverse approaches.

2.3 Machine Learning Approach for Hate Speech Detection

Numerous researchers have studied diverse detection techniques to efficiently discern hate speech. For instance, Omar et al. [36] utilized conventional algorithm-based approaches such as naïve bayes, support vector classification (SVC), logistic regression, convolutional neural network (CNN), and recurrent neural network (RNN) to recognize Arabic hateful contents on digital social platforms. Cao et al. [37] suggested a novel technique, HateGAN, which utilizes global vectors for word representation (GloVE) and Word2Vec embeddings, as well as long short-term memory (LSTM) and CNN as baseline models, to detect hate speech. Following the advent of the transformer framework by Vaswani et al. [38], researchers have conducted various studies using transformer-based architectures. Specifically, many studies have utilized BERT, a model introduced by Google that has made a substantial contribution to the field of NLP [17]. BERT can produce more intricate word features for comparing word semantics and exhibits impressive performance in advanced machine reading comprehension tests. Unlike traditional embedding methods, BERT offers both conceptual and practical advantages. Samghabadi et al. [39] propounded a comprehensive neural network model that employed attention mechanisms over BERT and adopted a multitask learning approach to tackle two tasks simultaneously. The first is to differentiate between benign texts, subtly hostile texts, and blatantly aggressive texts. The second is to classify texts as gendered, that is, related to sexuality or gender roles, or non-gendered. Parikh et al. [40] created a system for the multi-label task of gender discrimination that integrated sentence representations from BERT with word embeddings ELMo and GLoVE, incorporating recurrent elements and a convolutional hierarchical structure. Additionally, they supplied a dataset aimed at gender discrimination.

Previous studies used traditional embedding methods combined with machine learning or employed sentiment analysis tools for clustering to conduct hate speech analysis. More recently, some studies have adopted BERT-based models with ensemble approaches or have focused on extracting contextual and semantic features [15,16]. However, research that integrates sentiment analysis tools to simultaneously consider both the semantic and affective facets of hate speech remains scarce. In this study, we present a novel framework that combines BERT's semantic capabilities with the sentiment and psychological attributes from the newest version of the Linguistic Inquiry and Word Count (LIWC) dictionary, LIWC-22, along with the Valence Aware Dictionary for sEntiment Reasoning (VADER), a tool known for its exceptional performance in social media sentiment analysis. The LIWC tool analyzes both the psychological and emotional aspects of words in sentences and has proven effective for text analysis [41]. Because VADER is designed for affective appraisal of social media discourse, it exhibits superlative efficacy within the social media domain [42].

3 Methods

In this section, we propose our methods that utilizes a pre-trained BERT model together with diverse sentiment analysis tools and neural networks. As depicted in Fig. 1, the overall workflow of our model framework comprises three parts:

- **Semantic representation:** Extracting the BERT [CLS] tokens from a hate speech dataset
- **Sentiment recognition:** Extracting the LIWC and VADER sentiment recognition scores from the hate speech dataset
- **Deep learning techniques:** Using deep learning models for hate speech detection

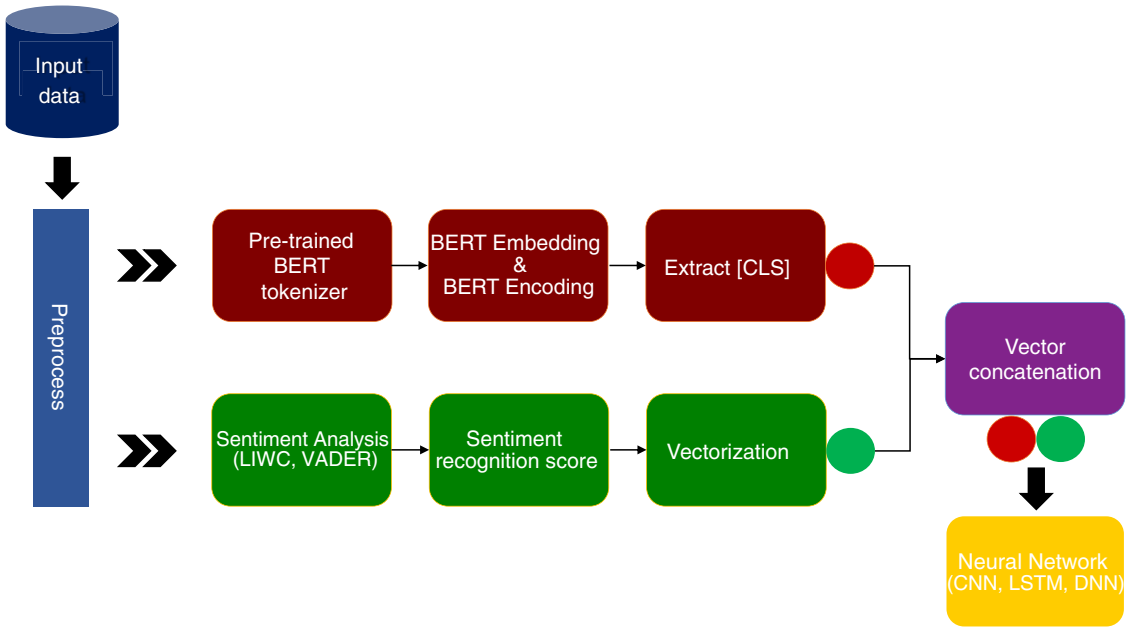

Figure 1: Overall workflow of the semantic–sentiment fusion approach model framework.

3.1 Semantic Representation

BERT is based on the transformer structure and uses only an encoder for training, achieving remarkable performance across multiple-sentence classification tasks [17]. BERT's tokenization method uses the WordPiece method, and it represents sentences employing token, segment and position embeddings. Token embeddings use two special tokens, [CLS] and [SEP], to distinguish between sentences. Segment embeddings are used with [SEP] tokens to help differentiate between sentences and are inserted at the end of each sentence. Position embeddings indicate the position of each token. We now describe how we applied the BERT model in our research.

In this study, we utilized a pre-trained BERT-base (uncased) model to derive semantic representations of the input texts for classification, without any fine-tuning. This model has 12 encoder layers, a hidden layer size of 768 dimensions, 12 multi-head attentions, and a total of 110M parameters. First, we utilized the hate speech dataset as the input to the pre-trained BERT model. We then gathered the [CLS] token from the model at the beginning of each sentence, which served as a representation of the entire sentence. During inference, a BERT tokenizer added a [CLS] token to the commencement of each sentence. This token was weighted by combining the entirety of token embeddings through the mechanism of self-attention. After token-level encoding, each sentence was represented as a 768-dimensional vector, denoted as $X \in \mathbb{R}^{768}$, where X is the sentence-level embedding extracted from the [CLS] token. Fig. 2 illustrates the workflow for extracting semantic representation vectors from BERT. With the semantic representation from the hate speech dataset extracted, we moved on to sentiment recognition.

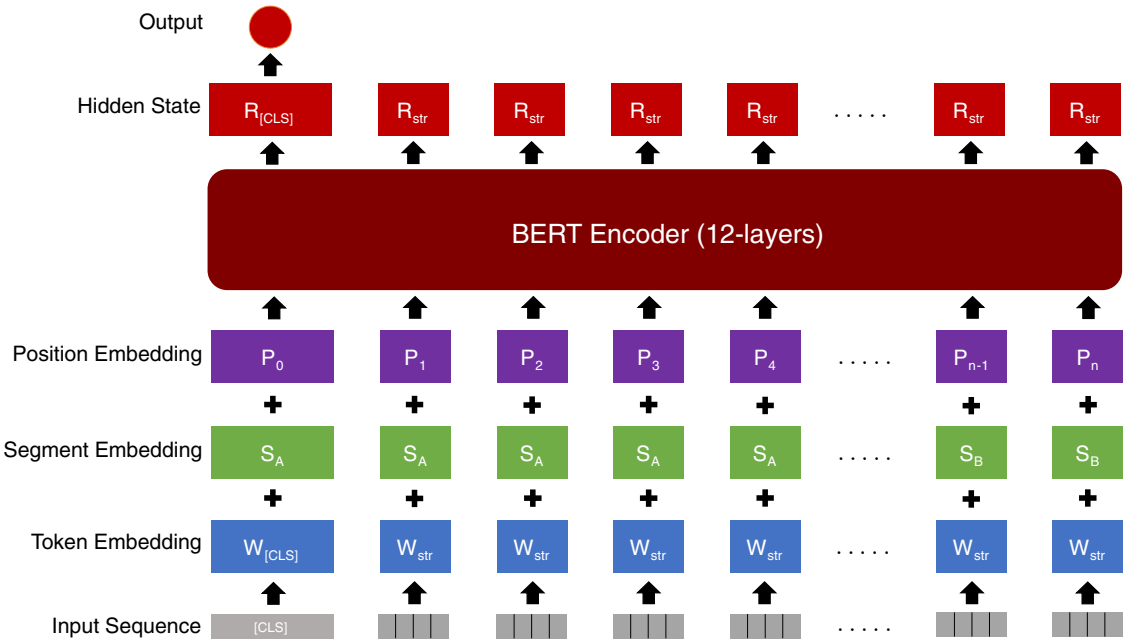

Figure 2: Process of extracting semantic representation vectors from the BERT model (Adapted from [17]).

3.2 Sentiment Recognition

Hate speech contains emotions and psychological states such as anger, disgust, and fear [43]. Although many sentiment analysis tools are available, LIWC is specifically designed to analyze a wide range of sentiments, including various emotional, cognitive, and process-related components in the text [44]. VADER integrates a lexical corpus conjoined with a prescriptive heuristic framework and is trained on online platform text, not only performs well in sentiment analysis but can also detect Internet vocabulary, emoticons, and emojis [42]. Therefore, we consider these tools the most suitable for this study.

3.2.1 Linguistic Inquiry and Word Count

LIWC-22, the latest version of LIWC, incorporates sophisticated analytical techniques along with a greater variety of linguistic examples [44]. Compared with previous versions, it has enabled the creation of more internally coherent lexicons with augmented psychometric and emotional attributes. In addition, an extensive text corpus was constructed, including both traditional and modern English samples from diverse contexts. LIWC-22 includes over 12,000 words, stems, and phrases. Every entry in the dictionary is part of multiple categories and sub-dictionaries designed to appraise a spectrum of psychosocial constructs.

Other representative lexicon-based sentiment analysis tools provide valence or polarity scores based on predefined lexicons [45,46]. These tools are lightweight and effective for capturing general affective orientation; however, they remain limited in capturing broader psychosocial constructs. In contrast, LIWC-22 consists of 101 variables categorized into linguistic dimensions, psychological processes, and an Expanded Dictionary. Linguistic dimensions include variables related to the structural components of a language, including elements like pronouns, prepositions, and verbs. Psychological processes include cognitive, affective, and social variables. The Expanded Dictionary includes a wide range of variables related to motives, perceptions, conversations, and punctuation. However, variables related to the structural components of language and punctuation were excluded because of a lack of emotional scores, resulting in 81 variables being used in the analysis. The scores for each variable indicate the proportion of the total terms within the text

belonging to that category. These were used as sentiment score vectors and were concatenated with a [CLS] token extracted from BERT.

3.2.2 VADER

VADER was introduced by Hutto and Gilbert [42], and is available in the Natural Language Toolkit (NLTK) Python library. It provides three scores: 'pos,' 'neg,' and 'neu,' which are aggregated to generate a final sentiment score, the 'compound' score, ranging between −1 and 1. A sentence becomes increasingly negative as its value nears −1, and increasingly positive as it approaches 1. This sentiment score was vectorized and combined with the [CLS] token. Fig. 3 illustrates the procedure for concatenating the [CLS] token with sentiment score vectors derived from LIWC and VADER. The VADER feature is a one-dimensional vector, whereas the LIWC feature is an 81-dimensional vector. Since both features were already bounded score values, no additional normalization was applied. These sentiment vectors were separately concatenated after the semantic representation vectors to evaluate their individual effects. Accordingly, the final input tensor shapes fed into the deep learning models were 768, 769, and 849, respectively. With the input vectors prepared, we now introduce the deep learning models used for detection.

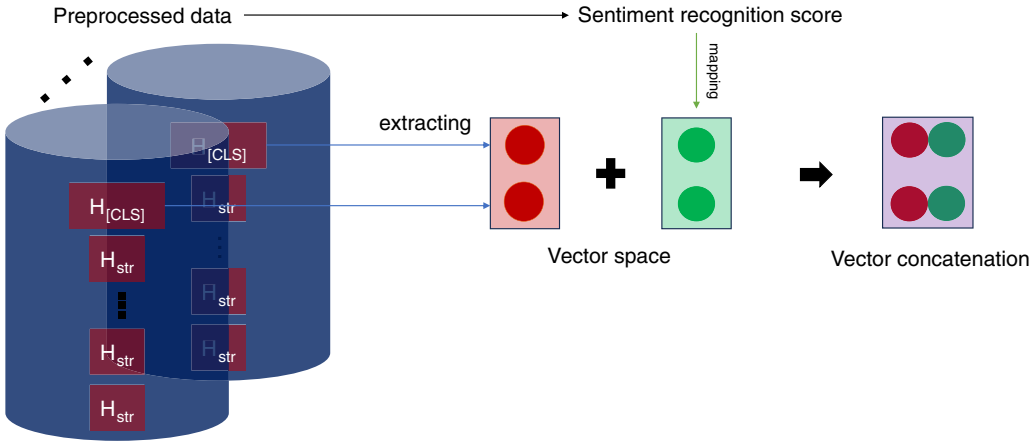

Figure 3: Procedure for concatenating the semantic and sentiment score vectors to form the input vector.

3.3 Deep Learning Techniques

Deep learning techniques are commonly used across various domains. They reveal intricate patterns in extensive datasets through the use of a backpropagation algorithm. These techniques have significantly advanced the cutting-edge of visual object recognition, speech recognition, and numerous other fields. In this study, we applied three deep learning models: CNN, LSTM, and Deep Neural Network (DNN). A comprehensive overview of the deep learning models utilized in this research is delineated in the ensuing subsection.

3.3.1 Convolutional Neural Network

CNN is designed to automatically and adaptively acquire feature representations via backpropagation. It leverages the localized receptive field of the preceding layer to create an intermediary representation for network training. By applying relevant filters, the CNN can better capture the corresponding input features, decrease the parameter count, and facilitate the reusability of weights. This process is accomplished through

various fundamental components, containing convolutional layers, pooling layers, and fully connected networks [47].

In this study, we convert the input data to a one-dimensional format and feed them into a single convolutional layer with a convolutional kernel size of 1 × 3. The input consisted of one channel, which was transformed into 64 output channels. To enhance model performance, batch normalization was applied to each output channel. The main role of the convolution layer is to capture characteristics from the input data, which are then passed through the subsequent network layers to enable accurate detection. The formula for the convolution operation is presented in Eq. (1).

$$f_i = f\left(W \cdot [X_{i:i+h-1}] + b\right) \tag{1}$$

Here, f_i represents the i-th output feature map value, W is the convolution kernel (filter), $X_{i:i+h-1}$ denotes the segment of input data from position i to $i + h - 1$, b is the bias term, and f is the activation function. Subsequently, a rectified linear unit (ReLU) activation function was utilized to prevent the vanishing gradient problem. The ReLU produces an output of 0 for negative input values and returns the input value itself for positive values, thereby preventing the gradient from vanishing. This formula is given in Eq. (2).

$$f(x) = \max(0, x) \tag{2}$$

Moreover, max pooling was applied to mitigate overfitting during model training. This reduces dimensionality of the input data, thereby lowering the quantity of model parameters and helps prevent overfitting. Max pooling follows Eq. (3).

$$f(m) = (f_1(m), f_2(m), \ldots, f_K(m))^{\top}, \quad f_K(m) = \max(x), \quad x \in X_K \tag{3}$$

Here, $f(m)$ is a vector representing the pooling result for the m-th window, $f_K(m)$ represents the K-th output value in the m-th window, and $\max(x)$ represents the maximum value within the window X_K.

After flattening the tensor containing 64 channels into a one-dimensional vector, the data were passed through a dense layer of 256 units, forming a hidden layer. Subsequently, a fully connected layer with the number of output units aligned to the number of classes was used with the softmax function, as shown in Eq. (4).

$$\sum_{j=1}^{N} e^{z_j} \mathrm{softmax}(z_i) = \frac{e^{z_i}}{\sum_{j=1}^{N} e^{z_j}} \tag{4}$$

The formula produces probabilities after considering the input value z_i and maps them to class probabilities within the range of 0 to 1, where i represents the class. Fig. 4 shows the overall CNN structure used in this study.

3.3.2 Long Short-Term Memory

LSTM is a type of RNN, designed to compensate for the problem of gradient vanishing [48]. It is useful for contextualizing sequence data as it retains past information for a longer period compared to RNN [49]. For this reason, LSTM was used in this study. The architecture of LSTM consists of four main components: forget gate, input gate, cell state, and output gate, which communicate with each other.

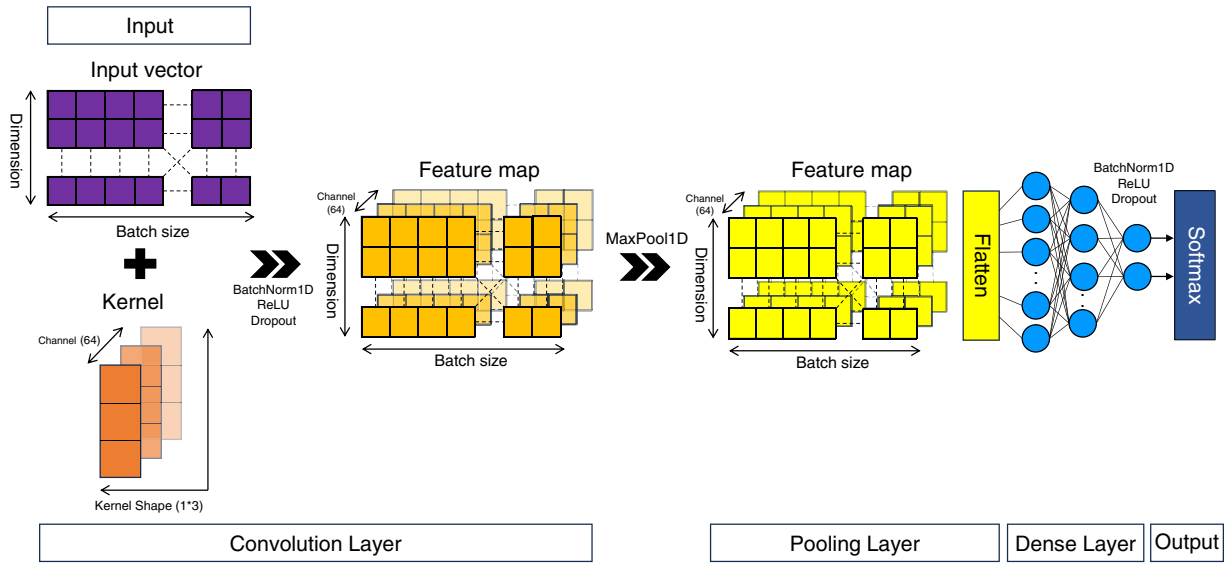

Figure 4: Structure of a CNN model that takes an input vector and passes it through convolution layers, pooling layers, and dense layers for hate speech detection (Adapted from [47]).

The forget gate separates useful information from useless information in the cell state, leaving only useful information. This gate employs a sigmoid function for activation. If the output is zero, all the values in the cell state are considered unnecessary. If the output is 1, the values are considered necessary information, and the information received from the cell state is retained. The formula representing the forget gate is given by Eq. (5).

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{5}$$

where f_t represents the output of the forget gate at time t, σ is the sigmoid function, W_f is the weight matrix of the forget gate, h_{t-1} specifies the hidden state in the previous time step, x_t is the input at the current time step, and b_f is the forget gate bias term.

As shown in Eq. (6), the input gate decides which parts of the incoming data to be retained in the cell state and to what extent.

$$i_t = \sigma\left(W_i \cdot [h_{t-1}, x_t] + b_i\right) \tag{6}$$

Here, i_t denotes the output of the input gate at time t, W_i denotes the weight matrix of the input gate, h_{t-1} is the hidden state at the previous time step, x_t denotes the input at the current time, b_i is the of the input gate, and the sigmoid function is employed to range the output between zero and one. Additionally, the new candidate values $\widetilde{C}_t$ that might be integrated to the cell state are calculated using the formula in Eq. (7).

$$\widetilde{C}_t = \tanh\left(W_c \cdot [h_{t-1}, x_t] + b_c\right) \tag{7}$$

where $\widetilde{C}_t$ is the candidate cell state with tanh signifying the hyperbolic tangent function, which ranges from −1 to 1, W_c is the weight matrix for the candidate cell state, and b_c is the bias value for the candidate cell state.

In the cell state, the updated information is merged with the information from the previous cell state after separating the necessary information from unnecessary information through the forget and input gates. The formula for updating the cell state is shown in Eq. (8).

$$C_t = f_t \cdot C_{t-1} + i_t \cdot \widetilde{C}_t \tag{8}$$

Here, C_t represents the updated cell state at time t, f_t is the output of the forget gate at time t and determines which information from the previous cell state C_{t-1} should be discarded. C_{t-1} denotes the cell state from the previous time step. i_t is the output of the input gate at time t, which determines the amount of new information to be incorporated into the cell state. $\widetilde{C}_t$ stands for the candidate cell state, computed using the tanh function, indicating the new information that could be added into the cell state.

Finally, the output gate determined the value to be produced. It outputs a filtered value based on the cell state, as expressed in Eq. (9).

$$\begin{aligned} o_t &= \sigma\left(W_o \cdot [h_{t-1}, x_t] + b_o\right) \\ h_t &= o_t \cdot \tanh(C_t) \end{aligned} \tag{9}$$

Here, o_t represents the value produced by the output gate at time t, and W_o represents the weight matrix for the output gate. h_{t-1} shows the hidden state of the preceding time step, and x_t is the input for the present time step. b_o denotes the bias term for the output gate, and outputs in the form of a sigmoid function. h_t represents the hidden state at time t, and C_t represents the current cell state. Fig. 5 shows the basic process of LSTM. We used an LSTM structure consisting of two hidden layers, each with 128 nodes. The outputs from these layers are passed to a fully connected layer, and the final prediction is carried out using the softmax function. Fig. 6 shows the overall structure of the LSTM framework used in our study.

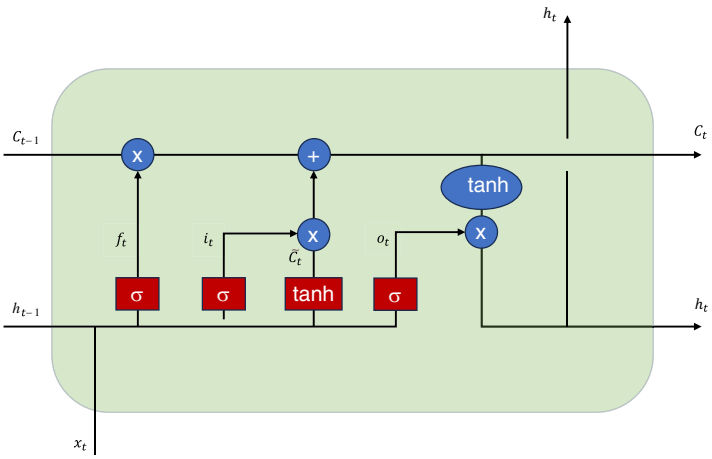

Figure 5: Basic structure of an LSTM model (Adapted from [48]).

3.3.3 Deep Neural Network

The DNN was inspired by the way the human brain handles complex processes [50]. DNNs are fundamental structures and key elements of deep learning and are extensively utilized in numerous AI applications. It responds equally to complex patterns by sequentially mapping the input space of each layer to the same output. The construction structure of these functions makes the computation frequently reusable, depending on the network depth. Although a DNN achieved high accuracy in many AI tasks, it requires significant computational resources and a long processing time. Therefore, in this study, we created a low-capacity neural network structure that achieves high accuracy without requiring significant computational power. The model was trained using three hidden layers, with ReLU applied as the activation function in each layer. The output layer then used the softmax function, with dimensions corresponding to the total number of

classes. To mitigate overfitting, a dropout value of 0.2 was utilized throughout the training process of all three deep learning models in our study. Dropout is essential for training neural networks, as it sets the output values of certain neurons to zero during training, helping to reduce overreliance on specific neurons. Fig. 7 illustrates the structure of the DNN used in this study.

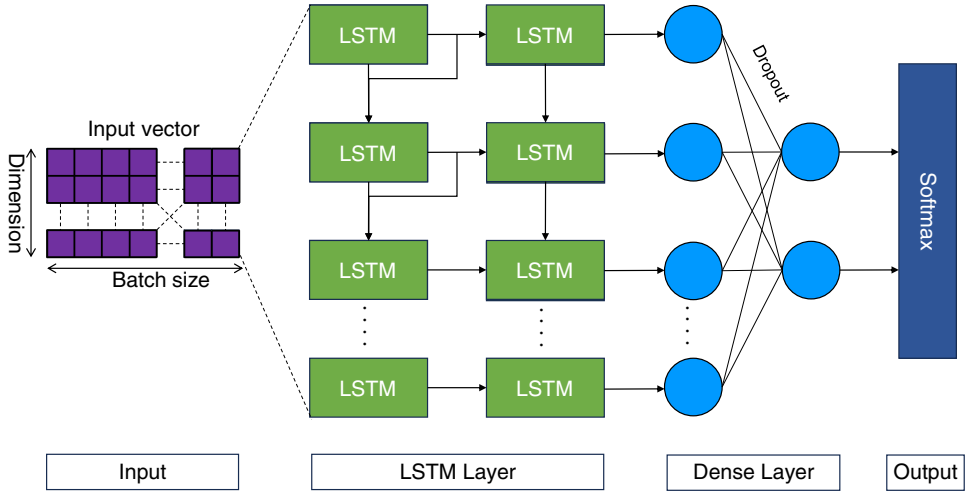

Figure 6: Structure of an LSTM model that takes an input vector and passes it through LSTM layers and dense layers for hate speech detection (Adapted from [48,49]).

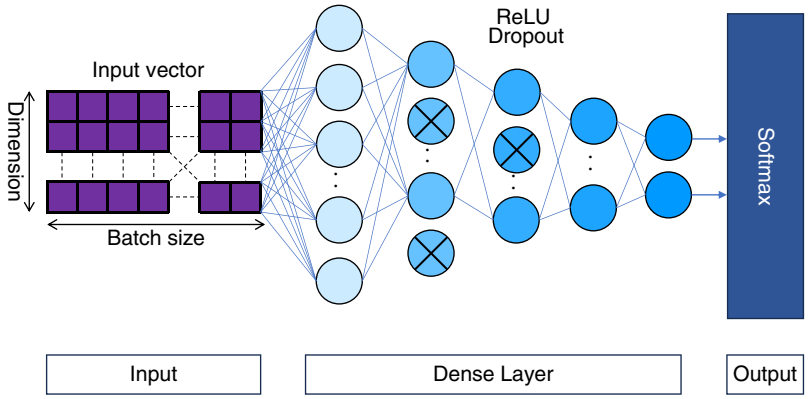

Figure 7: Structure of a DNN model that takes an input vector and passes it through dense layers for hate speech detection (Adapted from [50]).

4 Experiments

In this section, we explain the specific experimental setup and results. This section includes the details of the datasets, preprocessing steps, experimental design, and final results, along with a comparison to contemporary techniques in the field and an evaluation of the cross-domain effectiveness of the proposed framework.

4.1 Hate Speech Datasets

To date, many researchers have conducted studies on hate speech, leading to its widespread availability. In our study, we validate the proposed framework by addressing four types—sexism, supremacism,

aggression, and offensive language—under the broader category of hate speech. An overview of the datasets is presented in Table 1. The specific details of the datasets are as follows:

Table 1: Summary of the hate speech datasets.

Dataset	Platform	Class	Number of Data
CMSB	Twitter	Hate Speech	1809
		Not Hate Speech	11,822
		Total	13,631
SM	Stormfront	Hate Speech	1196
		Not Hate Speech	9507
		Total	10,703
TRAC	Twitter	Overtly Aggressive	548
		Covertly Aggressive	570
		Non-Aggressive	4211
		Total	5329
HSOL	Twitter	Hate Speech	1430
		Offensive Language	19,190
		Normal	4163
		Total	24,783

The 'Call me sexist but'[1] dataset was gathered by Samory et al. [51] by systematically searching for instances of the phrase 'call me sexist but' on Twitter. The idea is that when using this phrase on Twitter, it often introduces potentially sexist opinions as a preamble to their comments, which leaves the main part of the sentence following this phrase. In addition, the dataset includes hostile samples collected by Talat and Hovy [24], characterized by overtly hostile attacks on minority genders using negative stereotypes. Furthermore, the dataset included samples of evocative sexism collected by Jha and Mamidi [52], featuring various expressions that subtly undermine minority genders as less competent and more vulnerable through positive stereotypes. The data were labeled into two categories to determine whether a sentence was present. There were 1809 sexist sentences and 11,822 non-sexist sentences, for a total of 13,631 sentences. We called this dataset CMSB.

The 'Supremacist'[2] dataset was sourced from Stormfront, an online forum associated with white supremacist ideologies, and was published by De Gibert et al. [53]. These data were manually labelled by three experts according to specific annotation guidelines. The data were annotated with four labels: 'Hate,' 'NoHate,' 'Relation,' and 'Skip.' A sentence considered 'Hate' is aimed at a particular group of individuals and driven by elements of their identity. A sentence labelled 'Relation' is a sentence that does not encompass hate speech itself but is included in several sentence combinations. When a sentence was not written in English, it was labelled 'Skip.' Any sentences that did not meet these criteria were classified as 'NoHate.' Regarding the distribution of data, there were 10,944 sentences in total, including 11.29% labelled as 'Hate,' 86.09% as 'NoHate,' 1.69% as 'Relation,' and 0.93% as 'Skip.' Since our study aimed to classify the data as either 'Hate' or 'NoHate,' we excluded the 'Relation' and 'Skip' labels. We designated this dataset as SM.

[1]The dataset is available at https://search.gesis.org/research_data/SDN-10.7802-2251.
[2]The dataset is available at https://github.com/Vicomtech/hate-speech-dataset

The 'TRAC2020'[3] dataset was gathered from Twitter and used during the second workshop on Trolling, Aggression, and Cyberbullying held at LREC 2020. It comprises two distinct sub-tasks [54]. Task A is an Aggression Identification Shared Task that involves categorizing text into 'Overtly Aggressive,' 'Covertly Aggressive,' and 'Non-Aggressive.' Task B is a Misogynistic Aggression Identification Shared Task, which includes labeling text as either 'gendered' or 'non-gendered.' This dataset consists of texts in Bangla, Hindi, and English. Because our research aims to detect hate speech and continues past research, we used Task A and included only English data. A total of 5329 samples were used, with 548 labelled 'Overtly Aggressive,' 570 labelled 'Covertly Aggressive,' and 4211 labelled 'Non-Aggressive.'

The 'Hate speech and offensive language'[4] dataset from Twitter was collected by Davidson et al. [55]. These tweets were obtained by filtering out common abusive terms and retrieving all tweets posted by the selected users. The total dataset comprises 24,783 samples categorized into three classes: 4163 normal, 19,190 offensive, and 1430 hate. It was annotated by CrowdFlower workers and is named HSOL in this study.

4.2 Data Preprocessing

Data preprocessing is essential for improving model performance. We carried out preprocessing using Python's built-in 're' module for regular expressions. First, we removed all URL texts starting with Twitter:// or https://. Next, email addresses, special symbols, emoticons, and punctuations were eliminated. Numbers and non-English strings were also removed. Subsequently, all the words were converted to lowercase. Once the preprocessing was complete, we removed all rows with empty values. Additionally, after performing all preprocessing steps, TF-IDF [32] was utilized to identify significant words in each dataset. As presented in Table 2, the scores for the top ten words in each dataset were extracted.

Table 2: Top 10 words ranked by TF-IDF scores for each dataset.

CMSB		SM		TRAC		HSOL	
Words	**Scores**	**Words**	**Scores**	**Words**	**Scores**	**Words**	**Scores**
Man	247.7067	White	276.9113	Movie	147.5565	Bitch	1807.5605
Woman	246.4627	People	182.0405	Review	103.3184	Pussy	644.6584
People	225.2256	Black	123.9416	Video	98.4265	Hoe	630.4868
Sexist	133.4618	Race	77.8765	People	74.0682	Ass	457.7844
Girl	105.5734	Kids	75.8068	Sir	73.3672	Fuck	457.1239
Hate	104.6324	Hope	72.2415	India	64.1367	Trash	411.2587
Shit	102.3473	Thread	67.1128	Bollywood	49.9924	Shit	399.3763
Work	97.1580	Jews	51.3344	Man	49.7807	Nigga	381.9844
Female	89.8268	Children	50.9298	True	40.0867	Love	281.6739
Fuck	85.7827	Man	47.5873	Film	39.4600	Bad	236.4752

In the CMSB dataset, key terms such as 'woman,' 'man,' 'girl,' and 'sexist' were predominant, indicating the dataset's focus on gender-related topics. The SM dataset highlighted race-related terms, with words like 'white,' 'black,' and 'race' being frequent. The HSOL dataset revealed that the TF-IDF scores for offensive words such as 'bitch' were overwhelmingly high. Additionally, words like 'hoe,' 'pussy,' and 'ass' were prominent, indicating a significant prevalence of emotional or aggressive expressions. However, unlike

[3] The dataset is available at https://sites.google.com/view/trac2/shared-task.
[4] The dataset is available at https://github.com/t-davidson/hate-speech-and-offensive-language

the previous three datasets, the TRAC dataset's key terms, including 'movie,' 'review,' and 'video,' made it challenging to infer specific characteristics.

4.3 Experimental Environment and Hyperparameter Setup

In our experiments, all testing was carried out with Python version 3.11.5, and the core tasks for training the models were performed using PyTorch 2.1.2. Additionally, Pandas was used for data manipulation, and scikit-learn was used for model evaluation and preparation. The experiments were conducted on a system equipped with a 13th Gen Intel(R) Core(TM) i7-1360P CPU (2.20 GHz), 16 GB of RAM, and running Windows 64-bit Operating System. The entire dataset was partitioned into training, validation, and test subsets in a 6:2:2 distribution. Following settings commonly adopted in previous hate speech detection studies [15,16], the model was trained for 20 epochs using the training and validation subsets. AdamW optimization was employed, setting the learning rate at 2e-5 and applying a weight decay of 1e-2 to prevent overfitting. Batch sizes were adjusted to 16 and 20. The best-performing model was automatically saved, and we set an early stop to quit training if the model did not show any improvement. The same hyperparameter settings were applied to all models used in the experiments, as summarized in Table 3.

Table 3: Hyperparameter settings.

Hyperparameters	Values
Epoch	20
Optimizer	AdamW
Learning rate	2e-5
Weight decay	1e-2
Batch size	16, 20
Early stopping	3
Dropout rate	0.2
Activation function	ReLU

4.4 Performance Evaluation Metrics

The main concept behind evaluation metrics is to assess how accurately the data can be forecasted in relation to the target class. Because most studies analyze models using accuracy and F1-score, we adopted these metrics to assess each model's performance and compared the results to evaluate our approach, integrating both the fusion and single vector approaches with deep learning techniques. As shown in Eqs. (10)–(12), the F1-score is calculated using precision, which assesses the ratio of true positives (TPs) and false positives (FPs) that the model correctly identifies, and recall, which evaluates the proportion of TPs and false negatives (FNs) that the model correctly labels. The label distribution of the datasets is skewed toward one class; however, we did not apply undersampling or oversampling in order to preserve the original data distribution. Removing majority class samples or augmenting the minority class may lead to information loss or increase the risk of overfitting [56,57]. The macro F1-score treats all classes equally, thereby reducing bias toward majority classes, and was therefore adopted for binary-labeled datasets [58]. For multi-class labeled datasets, we used weighted F1-score, as it reflects class frequency proportions [59]. By adopting these evaluation metrics, model performance was assessed under the original class distribution.

$$\text{Precision} = \frac{TP}{TP + FP} \tag{10}$$

$$\text{Recall} = \frac{TP}{TP + FN} \tag{11}$$

$$F1 = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}} \tag{12}$$

Accuracy is calculated by considering the ratio of TPs and TNs to TP, FP, TN, and FN, as given by Eq. (13).

$$\text{Accuracy} = \frac{TP + TN}{TP + FP + TN + FN} \tag{13}$$

We also employed confusion matrices for each dataset's top-performing models to furnish a more nuanced perspective of their detection performance, transcending summary measures like accuracy and F1-score. Additionally, for the CMSB and SM datasets, we computed the AUROC values, which are widely used for binary detection tasks.

4.5 Results

We assessed the models' performance through the application of evaluation metrics. We begin by providing an overview of the models' performance, starting with BERT's semantic vectors, followed by semantic vectors with deep learning models excluding sentiment score vectors, and finally, deep learning models including both semantic and sentiment score vectors. Based on these results, it is apparent that the models show improved performance when sentiment score vectors are incorporated compared to the cases when they are omitted. Tables 4–7 present the performance outcomes of each model across the four datasets. Additionally, statistical significance was evaluated by comparing best-performing model against the other models using paired bootstrap resampling with 10,000 iterations. The findings showed that most of the performance differences were statistically significant.

Table 4: Performance over the CMSB dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
BERT	87.49	76.48	60.21	63.18***
BERT+CNN	89.80	80.84	71.77	75.14***
BERT+LSTM	90.49	82.24	74.27	77.41**
BERT+DNN	90.03	79.87	76.09	77.78*
BERT+CNN+LIWC	**91.34**	**83.06**	78.71	80.65
BERT+LSTM+LIWC	88.79	77.29	71.66	73.97***
BERT+DNN+LIWC	90.72	80.58	**80.79**	**80.68**
BERT+CNN+VADER	90.30	82.21	72.99	76.46***
BERT+LSTM+VADER	89.95	80.74	73.14	76.13***
BERT+DNN+VADER	90.07	80.13	75.65	77.61**

Note: *$p < 0.05$, **$p < 0.01$, and ***$p < 0.001$. The best results are highlighted in bold.

Table 5: Performance over the SM dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
BERT	88.00	73.10	52.10	51.10***
BERT+CNN	89.34	77.10	63.45	67.08
BERT+LSTM	89.08	76.00	62.57	66.01**
BERT+DNN	89.03	75.09	64.37	67.64
BERT+CNN+LIWC	**90.22**	80.82	**65.95**	**70.17**
BERT+LSTM+LIWC	89.34	79.29	60.71	64.24***
BERT+DNN+LIWC	89.50	**81.49**	60.25	63.82***
BERT+CNN+VADER	89.55	77.20	65.57	69.16
BERT+LSTM+VADER	88.93	75.29	62.12	65.44***
BERT+DNN+VADER	88.77	74.57	61.66	64.87**

Note: **$p < 0.01$, and ***$p < 0.001$. The best results are highlighted in bold.

Table 6: Performance over the TRAC dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
BERT	73.47	56.52	73.47	62.96***
BERT+CNN	73.84	70.90	73.84	71.08
BERT+LSTM	73.72	71.83	73.72	64.47***
BERT+DNN	73.59	71.03	73.59	63.71***
BERT+CNN+LIWC	**76.60**	71.75	**76.60**	**72.00**
BERT+LSTM+LIWC	74.22	**80.88**	74.22	63.48***
BERT+DNN+LIWC	74.22	77.86	74.22	63.70***
BERT+CNN+VADER	75.09	71.14	75.09	70.94
BERT+LSTM+VADER	74.09	66.15	74.09	65.03***
BERT+DNN+VADER	74.22	74.62	74.22	64.77***

Note: ***$p < 0.001$. The best results are highlighted in bold.

Table 7: Performance over the HSOL dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
BERT	83.96	78.47	83.96	80.51***
BERT+CNN	85.44	83.28	85.44	83.66***
BERT+LSTM	84.92	82.27	84.92	82.71***
BERT+DNN	85.30	86.12	85.30	82.81***
BERT+CNN+LIWC	**88.70**	86.83	**88.70**	**87.01**
BERT+LSTM+LIWC	88.58	**87.65**	88.58	86.19*
BERT+DNN+LIWC	88.29	86.58	88.29	85.99**
BERT+CNN+VADER	84.90	83.39	84.90	83.88***
BERT+LSTM+VADER	85.42	82.81	85.42	83.27***
BERT+DNN+VADER	85.75	86.44	85.75	83.18***

Note: *$p < 0.05$, **$p < 0.01$, and ***$p < 0.001$. The best results are highlighted in bold.

Table 4 displays the performance outcomes for each model for the CMSB dataset. The CNN model with combined semantic and LIWC sentiment score vectors showed the highest accuracy (91.34) and second-highest F1-score (80.65). The DNN model with the semantic and LIWC sentiment score vectors exhibited the highest F1-score achieved of 80.68. The LSTM model with semantic vectors showed the third-highest accuracy of 90.49, and the DNN model with semantic vectors showed the third-highest F1-score of 77.78.

Subsequently, Table 5 outlines the findings for the SM dataset. For the SM dataset, the CNN model combined with semantic vectors and LIWC demonstrated the best performance, recording an accuracy score of 90.22 and an F1-score of 70.17. Second, the CNN model utilizing semantic vectors combined with VADER sentiment score vectors demonstrated the second-best performance, achieving an accuracy rate of 89.55 and an F1-score of 69.16. The DNN model with the semantic and LIWC showed the third-highest accuracy of 89.50, whereas the DNN model with semantic vectors obtained an F1-score of 67.64.

Table 6 displays the outcomes for the TRAC dataset. For this dataset, the CNN model with combined semantic and LIWC sentiment score vectors recorded the highest accuracy and F1-score, with an accuracy score of 76.60 and an F1-score rate of 72.00. The CNN model with semantic and VADER sentiment scores demonstrated the second-highest accuracy of 75.09 and an F1-score of 70.94. Detection with the CNN model using semantic vectors registered the second-highest F1-score of 71.08. The third-highest accuracy, at 74.22, was achieved by three models: the fusion of semantics and LIWC with DNN, semantics and VADER with DNN, and semantics and LIWC with LSTM, all tied for third place.

Lastly, we discuss Table 7, which contains information about the HSOL dataset. The CNN model with semantic vectors and LIWC sentiment score vectors exhibited the highest performance, attaining an accuracy value of 88.70 and an F1-score of 87.01. The second-best model was the LSTM model with the integration of semantic vectors and LIWC, with an accuracy rate of 88.58 and an F1-score of 86.19. Finally, the DNN model with semantic vectors and LIWC was the third-best performing model, with an accuracy rate of 88.29 and an F1-score of 85.99. These results indicate that for the HSOL dataset, the LIWC sentiment score vectors significantly improved the model performance.

Based on the above findings, we can observe that, for the four datasets, the integration of BERT semantic vectors and LIWC sentiment score vectors with the CNN model represents superior performance. In addition, various deep learning models generally perform better when sentiment score vectors are included, highlighting the importance of incorporating diverse features depending on the characteristics of the dataset.

Fig. 8 presents the confusion matrices for the BERT+CNN+LIWC model, which achieved the best performance on each dataset. Fig. 9 illustrates the ROC curves for the CMSB and SM datasets. The value for CMSB was 94.16, and for SM was 86.78, indicating strong discriminative performance.

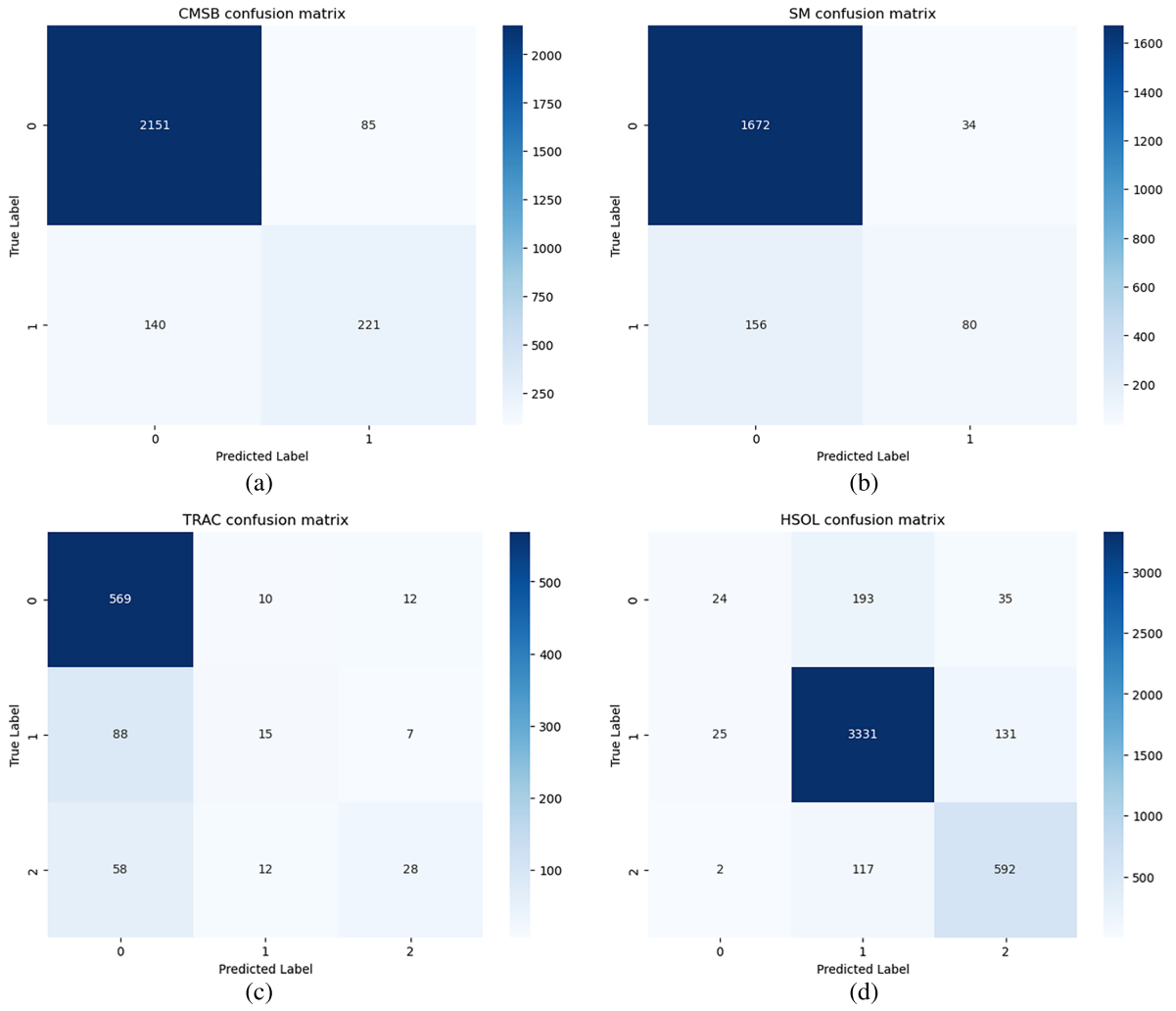

Figure 8: Confusion matrices of BERT+CNN+LIWC on four datasets: (**a**) CMSB; (**b**) SM; (**c**) TRAC; (**d**) HSOL.

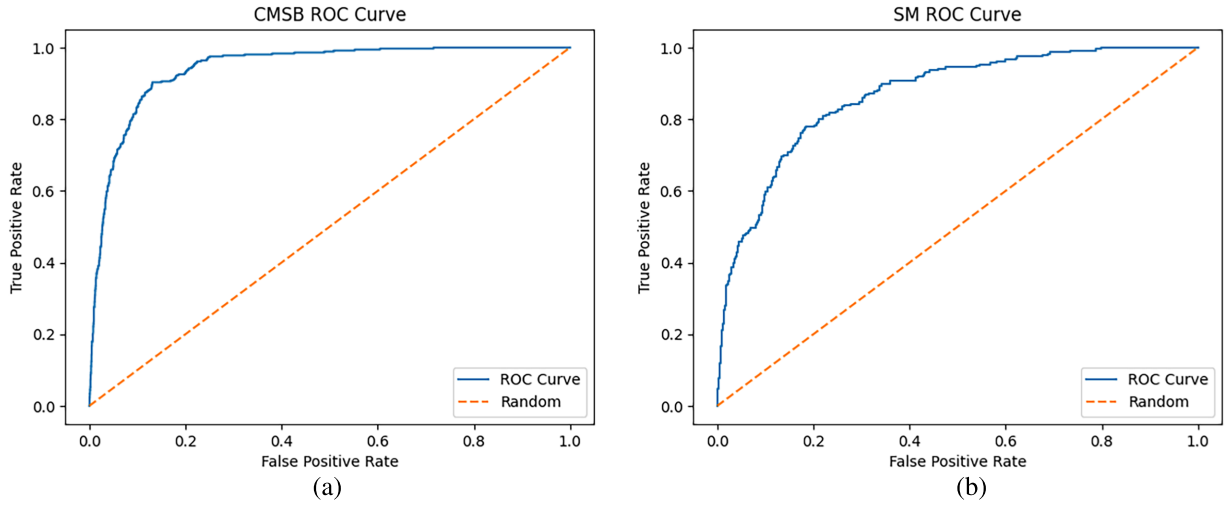

Figure 9: ROC curves of BERT+CNN+LIWC on binary-labeled datasets: (**a**) CMSB; (**b**) SM.

4.6 Comparison with Contemporary Techniques

Now that we have observed the performance of our proposed semantic-sentiment fusion approach on each dataset, we compare their performances on the same dataset against contemporary techniques. To examine the CMSB dataset, Muti et al. [60] applied data augmentation using the RoBERTa-hate model, which was previously fine-tuned on hate speech data, and Antypas and Camacho-Collados [61] fine-tuned the TimeLMs model with a hate speech dataset. For the SM dataset, random embedding with LSTM was proposed by De Gibert et al. [53], and BERT with data augmentation was used by Wullach et al. [62]. For the TRAC dataset, a BERT with attention and a fully connected layer was introduced by Samghabadi et al. [39], whereas Kumar et al. [63] used a pre-trained BERT model with English data. Finally, for the HSOL dataset, SBERT combined with HateNet was proposed by Duong et al. [64], and BiLSTM with CNN and hierarchical attention, referred to as BiCHAT, was proposed by Khan et al. [65], whereas Wullach et al. [62] suggested BERT with data augmentation. Tables 8–11 present performance comparisons with contemporary techniques for each dataset. We marked a '-' where the existing research was not explicit.

Table 8: Performance comparison with contemporary techniques on the CMSB dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
RoBERTa-hate+data augmentation [60]	–	–	–	**81.00**
TimeLMs [61]	–	–	–	79.70
BERT+CNN+LIWC (Our study)	**91.34**	**83.06**	**78.71**	80.65

Note: The best results are highlighted in bold.

Table 9: Performance comparison with contemporary techniques on the SM dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
LSTM+Random embedding [53]	78.00	–	–	–
BERT+data augmentation [62]	87.20	60.00	58.20	59.10
BERT+CNN+LIWC (Our study)	**90.22**	**80.82**	**65.95**	**70.17**

Note: The best results are highlighted in bold.

Table 10: Performance comparison with contemporary techniques on the TRAC dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
BERT+fully connected layer [39]	73.10	–	–	71.40
Pre-trained BERT [63]	–	69.00	70.00	67.00
BERT+CNN+LIWC (Our study)	**76.60**	**71.75**	**76.60**	**72.00**

Note: The best results are highlighted in bold.

Table 11: Performance comparison with contemporary techniques on the HSOL dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
SBERT+HateNet [64]	84.50	–	–	84.30
BiCHAT [65]	76.00	74.00	75.00	75.00

(Continued)

Table 11 (continued)

Models	Accuracy	Precision	Recall	F1-Score
BERT+data augmentation [62]	**93.50**	**92.30**	81.40	86.50
BERT+CNN+LIWC (Our study)	88.70	86.83	**88.70**	**87.01**

Note: The best results are highlighted in bold.

4.7 Cross-Domain Effectiveness of the Proposed Framework

To verify the effectiveness and generalizability of the semantic-sentiment fusion approach, we applied it to the sarcasm dataset. The dataset consisted of sarcastic news headlines, comprising a total of 28,619 samples, including 13,634 sarcastic headlines and 14,985 non-sarcastic headlines [66]. The preprocessing steps and hyperparameters used in the model remained consistent with those used in the hate speech tasks. The detection results are presented in Table 12.

Table 12: Performance over the sarcasm dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
BERT	83.30	83.68	80.67	82.15
BERT+CNN	87.30	87.29	85.85	86.56
BERT+LSTM	87.05	87.14	85.44	86.28
BERT+DNN	87.39	87.06	86.36	86.71
BERT+CNN+LIWC	86.97	88.02	84.09	86.01
BERT+LSTM+LIWC	84.36	83.98	83.02	83.50
BERT+DNN+LIWC	86.32	85.81	85.41	85.61
BERT+CNN+VADER	87.68	85.98	**88.60**	**87.27**
BERT+LSTM+VADER	87.32	**88.47**	84.38	86.37
BERT+DNN+VADER	**87.93**	88.33	86.03	87.16

Note: The best results are highlighted in bold.

Table 13 presents a performance comparison with prior studies on sarcasm detection. Mai et al. [67] utilized LLaMA3 to evaluate the capability of Large Language Models (LLMs) in sarcasm detection. Thambi et al. [68] analyzed optimal detection models by experimenting with deep learning architectures such as CNN, Bi-LSTM, and RNN. Chy et al. [69] compared LSTM and GRU models for sentiment analysis on sarcastic news headlines. Singh and Jaiswal [70] proposed a novel framework by combining a hybrid Naïve Bayes–Random Forest technique with Bag-of-Words embeddings.

Table 13: Performance comparison with contemporary techniques on the sarcasm dataset (%).

Models	Accuracy	Precision	Recall	F1-Score
Llama3 [67]	–	67.00	66.00	66.00
GRU [69]	78.00	–	–	–
CNN+Bi-LSTM [68]	78.53	76.00	77.00	77.00
Bi-LSTM [68]	79.34	77.00	77.00	77.00
CNN [68]	79.49	77.00	78.00	78.00

(Continued)

Table 13 (continued)

Models	Accuracy	Precision	Recall	F1-Score
RNN [68]	79.88	83.00	70.00	76.48
LSTM [69]	82.00	–	–	–
Hybrid Technique [70]	85.37	–	–	77.58
BERT+CNN+VADER (Our study)	**87.68**	**85.98**	**88.60**	**87.27**

Note: The best results are highlighted in bold.

5 Discussion

The performance analysis showed that the semantic-sentiment fusion with CNN achieved the best results across the four hate speech datasets. This outstanding performance led to several key discussions. Utilizing semantic vectors with CNN through the integration of convolutional and dense layers improves the detection performance more effectively than using only fully connected layers in a DNN. This underscores the importance of leveraging the structural advantages of CNN in processing semantic information to improve detection outcomes [68]. Furthermore, the LSTM does not provide a significant improvement in hate speech detection. This may be because the semantic vectors have already been extracted through the BERT self-attention mechanism, which inherently captures contextual information [17,71]. Therefore, the LSTM, which reflects previous time steps, does not contribute significantly to improving model performance.

Overall, our proposed fusion framework with CNN exhibited superior performance across all datasets, regardless of whether the task was binary or multi-class detection. This finding suggests that the 81 sentiment elements provided by LIWC, which quantify emotional state, psychological condition, and positive or negative sentiments inherent in a sentence, offer significant advantages in hate speech detection. Specifically, in the multi-class task, the inclusion of LIWC results in an approximate three-point increase in accuracy. VADER, which represents the sentiment state of a sentence as a one-dimensional vector ranging from −1 to 1, contributed to enhanced model performance in the sarcasm detection task. For sentences with more implicit than explicit features, such as those found in sarcasm data, VADER performed better than LIWC. This difference arises from the nature of sarcasm, which involves an inversion of literal meaning within the sentence [72]. Since LIWC processes sentiment based on predefined words and phrases [41], it may struggle to capture the sentiment of sarcastic sentences when explicit indicators are absent. In contrast, VADER better captures contextual sentiment [42], making it more effective at detecting implicit meanings in context.

These findings strengthen the validation of the proposed semantic-sentiment fusion approach, as integrating sentiment score vectors also yielded the highest performance in the sarcasm detection task. This indicates that while the multidimensional sentiment analysis provided by LIWC is advantageous for sentences with clearly expressed emotional cues, VADER may be more effective for those mainly composed of implicit sentiment. In terms of practical implications, the proposed framework can be effectively applied to social media platforms, enabling more robust detection across diverse linguistic patterns. We confirmed that our framework shows the best detection performance when identifying both explicit and implicit expressions in text, which may contribute to enhancing platform safety and supporting regulatory compliance requirements.

6 Conclusion and Future Work

As social media becomes more active and users spend more time on it, the frequency of hate speech increases because of the greater expression of opinions. Therefore, the identification and detection of hate speech remain a crucial area of ongoing study and enhancement.

In this paper, we present two insights that have not been mentioned in existing research. To enhance the hate speech detection performance, we employed a fusion approach. First, we went beyond merely considering the semantic aspects of words in hate speech data by incorporating the various sentiment factors implied by words. We extracted BERT semantic representation vectors, which encapsulate the semantic content of sentences. Additionally, we obtained sentiment recognition scores using the LIWC-22 dictionary, and VADER. These scores were then mapped onto the vector space. Finally, we combined the semantic representation vectors with the sentiment score vectors to devise a framework for hate speech detection, utilizing CNN, LSTM, and DNN as deep learning models. Through this framework, we observed a clear and notable enhancement when sentiment score vectors were included compared to detection using only semantic representation vectors and deep learning models. Notably, it achieved over 90% accuracy on binary detection tasks, such as those on the CMSB and SM datasets.

Second, existing research often requires high computational resources and significant time to model hate speech detection. However, our proposed framework demonstrates that by using semantic representation vectors and fewer than 5M trainable parameters—compared to approximately 110M parameters in BERT-base (uncased)—it is possible to achieve outstanding performance in detection tasks without embedding all words in the dataset. This allows for effective detection even with less computational power, requiring on average less than 10 min for training and less than 5 s for inference in a CPU-only setting. Furthermore, through evaluations against cutting-edge models, we established that the effectiveness of models using our proposed framework was superior and did not lag behind that of other models. However, this study had several limitations. The dataset used in this study was relatively small and class-imbalanced, resulting in lower performance for smaller classes compared to the overall dataset performance. It also raises concerns regarding its applicability to future big data analyses. Second, only three deep learning models were considered in this study; therefore, further validation using a wider range of models is necessary. Lastly, as the proposed framework solely focuses on text data, future research should explore multimodal approaches for hate speech detection.

Acknowledgement: Not applicable.

Funding Statement: This research was supported by the "Regional Innovation System & Education (RISE)" through the Seoul RISE Center, funded by the Ministry of Education (MOE) and the Seoul Metropolitan Government (2026-RISE-01-018-04).

Author Contributions: Choongwon Kang: conceptualization, data curation, formal analysis, investigation, methodology, validation, visualization, writing—original draft, writing—review & editing. Haein Lee: formal analysis, investigation, project administration, supervision, validation, writing—review & editing. Jang Hyun Kim: funding acquisition, project administration, resources, software, supervision. All authors reviewed and approved the final version of the manuscript.

Availability of Data and Materials: The code and data used in this study can be found at https://github.com/ChoongwonKang/Hate_speech_detection.

Ethics Approval: This work did not require ethical approval from a human subject or animal welfare committee.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Vujičić Stanković S, Mladenović M. An approach to automatic classification of hate speech in sports domain on social media. J Big Data. 2023;10(1):109. doi:10.1186/s40537-023-00766-9.
2. Statista. Daily time spent on social networking by internet users worldwide from 2012 to 2025 [Internet]. 2025 [cited 2026 Mar 10]. Available from: https://www.statista.com/statistics/433871/daily-social-media-usage-worldwide/.
3. Statista. Number of social media users worldwide from 2017 to 2030 [Internet]. 2025 [cited 2026 Mar 10]. Available from: https://www.statista.com/statistics/278414/number-of-worldwide-social-network-users/.
4. Statista. Most popular social networks worldwide as of October 2025, by number of monthly active users [Internet]. 2025 [cited 2026 Mar 10]. Available from: https://www.statista.com/statistics/272014/global-social-networks-ranked-by-number-of-users.
5. 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. Alex Eng J. 2023;80(2):110–21. doi:10.1016/j.aej.2023.08.038.
6. Zueva N, Kabirova M, Kalaidin P. Reducing unintended identity bias in Russian hate speech detection. arXiv:2010.11666. 2020.
7. Mullah NS, Zainon WMNW. Advances in machine learning algorithms for hate speech detection in social media: a review. IEEE Access. 2021;9:88364–76. doi:10.1109/access.2021.3089515.
8. Hinduja S, Patchin JW. Bullying, cyberbullying, and suicide. Arch Suicide Res. 2010;14(3):206–21. doi:10.1080/13811118.2010.494133.
9. Arango A, Pérez J, Poblete B. Hate speech detection is not as easy as you may think: a closer look at model validation (extended version). Inf Syst. 2022;105(4):101584. doi:10.1016/j.is.2020.101584.
10. De la Peña Sarracén GL, Rosso P. Systematic keyword and bias analyses in hate speech detection. Inf Process Manag. 2023;60(5):103433. doi:10.1016/j.ipm.2023.103433.
11. Narula R, Chaudhary P. A comprehensive review on detection of hate speech for multi-lingual data. Soc Netw Anal Min. 2024;14(1):244. doi:10.1007/s13278-024-01401-y.
12. Masud S, Pinkesh P, Das A, Gupta M, Nakov P, Chakraborty T. Half-day tutorial on combating online hate speech: the role of content, networks, psychology, user behavior, etc. In: Proceedings of the Fifteenth ACM International Conference on Web Search and Data Mining. New York, NY, USA: ACM; 2022. p. 1629–31.
13. Mukherjee S, Das S. Application of transformer-based language models to detect hate speech in social media. J Comput Cogn Eng. 2023;2(4):278–86. doi:10.47852/bonviewjcce2022010102.
14. Pannerselvam K, Rajiakodi S. Systematic literature review on hate speech detection in Indian low-resource languages. J Comput Soc Sci. 2026;9(1):5. doi:10.1007/s42001-025-00432-5.
15. Aksoy Ç, Demirezen MU, Sağıroğlu Ş. Hate speech detection in Turkish: an ensemble transformer-based deep learning approach. Eng Appl Artif Intell. 2026;164:113147.
16. Abdelsamie MM, Azab SS, Hefny HA. The dialects gap: a multi-task learning approach for enhancing hate speech detection in Arabic dialects. Expert Syst Appl. 2026;295:128584.
17. Devlin J, Chang MW, Lee K, Toutanova K. Bert: pre-training of deep bidirectional transformers for language understanding. In: Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Stroudsburg, PA, USA: ACL; 2019. p. 4171–86.
18. Zhang J. Sexual disgust sensitivity mediates the sex difference in support of censoring hate speech. Pers Individ Dif. 2019;145:89–96.
19. Chiang SY. Well, I'm a lot of things, but I'm sure not a bigot': positive self-presentation in confrontational discourse on racism. Discourse Soc. 2010;21(3):273–94. doi:10.1177/0957926509360653.
20. Bankov K. Cyberbullying and hate speech in the debate around the ratification of the Istanbul convention in Bulgaria: a semiotic analysis of the communication dynamics. Soc Semiot. 2020;30(3):344–64. doi:10.1080/10350330.2020.1731175.
21. Murfi H, Syamsyuriani, Gowandi T, Ardaneswari G, Nurrohmah S. BERT-based combination of convolutional and recurrent neural network for Indonesian sentiment analysis. Appl Soft Comput. 2024;151(6):111112. doi:10.1016/j.asoc.2023.111112.

22. Kilic IY, Pan S. Incorporating LIWC in neural networks to improve human trait and behavior analysis in low resource scenarios. In: Proceedings of the Thirteenth Language Resources and Evaluation Conference. Paris, France: ELRA; 2022. p. 4532–9.
23. Kettrey HH, Laster WN. Staking territory in the "World White Web" an exploration of the roles of overt and color-blind racism in maintaining racial boundaries on a popular web site. Soc Curr. 2014;1(3):257–74. doi:10.1177/2329496514540134.
24. Talat Z, Hovy D. Hateful symbols or hateful people? Predictive features for hate speech detection on twitter. In: Proceedings of the NAACL Student Research Workshop. Stroudsburg, PA, USA: ACL; 2016. p. 88–93.
25. Fortuna P, Nunes S. A survey on automatic detection of hate speech in text. ACM Comput Surv (CSUR). 2018;51(4):1–30. doi:10.1145/3232676.
26. 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. New York, NY, USA: ACM; 2016. p. 145–53.
27. Meta. Hateful conduct. 2025 [cited 2026 Mar 10]. Available from: https://transparency.meta.com/policies/community-standards/hateful-conduct/.
28. Suler J. The online disinhibition effect. Cyberpsychol Behav. 2004;7(3):321–6. doi:10.1089/1094931041291295.
29. Bagavathi A, Bashiri P, Reid S, Phillips M, Krishnan S. Examining untempered social media: analyzing cascades of polarized conversations. In: Proceedings of the 2019 IEEE/ACM International Conference on Advances in Social Networks Analysis and Mining. New York, NY, USA: ACM; 2019. p. 625–32.
30. Mathew B, Dutt R, Goyal P, Mukherjee A. Spread of hate speech in online social media. In: Proceedings of the 10th ACM Conference on Web Science. New York, NY, USA: ACM; 2019. p. 173–82.
31. Rodriguez A, Argueta C, Chen YL. Automatic detection of hate speech on Facebook using sentiment and emotion analysis. In: 2019 International Conference on Artificial Intelligence in Information and Communication (ICAIIC). Piscataway, NJ, USA: IEEE; 2019. p. 169–74.
32. Salton G, Buckley C. Term-weighting approaches in automatic text retrieval. Inf Process Manag. 1988;24(5):513–23. doi:10.1016/0306-4573(88)90021-0.
33. Araque O, Iglesias CA. An ensemble method for radicalization and hate speech detection online empowered by sentic computing. Cogn Comput. 2022;14(1):48–61. doi:10.1007/s12559-021-09845-6.
34. Zhou X, Yong Y, Fan X, Ren G, Song Y, Diao Y, et al. Hate speech detection based on sentiment knowledge sharing. In: Proceedings of the 59th Annual Meeting of the Association for Computational Linguistics and the 11th International Joint Conference on Natural Language Processing. Stroudsburg, PA, USA: ACL; 2021. p. 7158–66.
35. Sheth P, Kumarage T, Moraffah R, Chadha A, Liu H. Peace: cross-platform hate speech detection—a causality-guided framework. In: Joint European Conference on Machine Learning and Knowledge Discovery in Databases. Cham, Switzerland: Springer; 2023. p. 559–75.
36. Omar A, Mahmoud TM, Abd-El-Hafeez T. Comparative performance of machine learning and deep learning algorithms for Arabic hate speech detection in OSNs. In: The International Conference on Artificial Intelligence and Computer Vision. Cham, Switzerland: Springer; 2020. p. 247–57.
37. Cao R, Lee RKW, Hoang TA. DeepHate: hate speech detection via multi-faceted text representations. In: Proceedings of the 12th ACM Conference on Web Science. New York, NY, USA: ACM; 2020. p. 11–20.
38. Vaswani A, Shazeer N, Parmar N, Uszkoreit J, Jones L, Gomez AN, et al. Attention is all you need. arXiv:1706.03762. 2017.
39. Samghabadi NS, Patwa P, Pykl S, Mukherjee P, Das A, Solorio T. Aggression and misogyny detection using BERT: a multi-task approach. In: Proceedings of the Second Workshop on Trolling, Aggression and Cyberbullying. Stroudsburg, PA, USA: ACL; 2020. p. 126–31.
40. Parikh P, Abburi H, Badjatiya P, Krishnan R, Chhaya N, Gupta M, et al. Multi-label categorization of accounts of sexism using a neural framework. arXiv:1910.04602. 2019.
41. Pennebaker JW, Francis ME, Booth RJ. Linguistic inquiry and word count: LIWC 2001. Vol. 71. Mahway: Lawrence Erlbaum Associates.

42. Hutto C, Gilbert E. Vader: a parsimonious rule-based model for sentiment analysis of social media text. In: Proceedings of the International AAAI Conference on Web and Social Media. Palo Alto, CA, USA: AAAI Press; 2014. p. 216–25.
43. Mnassri K, Rajapaksha P, Farahbakhsh R, Crespi N. Hate speech and offensive language detection using an emotion-aware shared encoder. In: ICC 2023-IEEE International Conference on Communications. Piscataway, NJ, USA: IEEE; 2023. p. 2852–7.
44. Boyd RL, Ashokkumar A, Seraj S, Pennebaker JW. The development and psychometric properties of LIWC-22. Austin, TX, USA: University of Texas at Austin; 2022.
45. Loria S. TextBlob documentation [Internet]. [cited 2026 Mar 10]. Available from: https://app.readthedocs.org/projects/textblob/downloads/pdf/dev/.
46. Nielsen FÅ. A new ANEW: evaluation of a word list for sentiment analysis in microblogs. arXiv:1103.2903. 2011.
47. Yamashita R, Nishio M, Do RKG, Togashi K. Convolutional neural networks: an overview and application in radiology. Insights Imaging. 2018;9(4):611–29. doi:10.1007/s13244-018-0639-9.
48. Hochreiter S, Schmidhuber J. Long short-term memory. Neural Comput. 1997;9(8):1735–80. doi:10.1162/neco.1997.9.8.1735.
49. Sherstinsky A. Fundamentals of recurrent neural network (RNN) and long short-term memory (LSTM) network. Phys D Nonlinear Phenom. 2020;404(8):132306. doi:10.1016/j.physd.2019.132306.
50. Montesinos López OA, Montesinos López A, Crossa J. Fundamentals of artificial neural networks and deep learning. In: Multivariate statistical machine learning methods for genomic prediction. Cham, Switzerland: Springer; 2022. p. 379–425. doi: 10.1007/978-3-030-89010-0_10.
51. Samory M, Sen I, Kohne J, Flöck F, Wagner C. "Call me sexist, but...": revisiting sexism detection using psychological scales and adversarial samples. In: Proceedings of the International AAAI Conference on Web and Social Media. Palo Alto, CA, USA: AAAI Press; 2021. p. 573–84.
52. Jha A, Mamidi R. When does a compliment become sexist? Analysis and classification of ambivalent sexism using twitter data. In: Proceedings of the Second Workshop on NLP and Computational Social Science. Stroudsburg, PA, USA: ACL; 2017. p. 7–16.
53. De Gibert O, Perez N, García-Pablos A, Cuadros M. Hate speech dataset from a white supremacy forum. arXiv:1809.04444. 2018.
54. Bhattacharya S, Singh S, Kumar R, Bansal A, Bhagat A, Dawer Y, et al. Developing a multilingual annotated corpus of misogyny and aggression. arXiv:2003.07428. 2020.
55. Davidson T, Warmsley D, Macy M, Weber I. Automated hate speech detection and the problem of offensive language. In: Proceedings of the International AAAI Conference on Web and Social Media. Palo Alto, CA, USA: AAAI Press; 2017. p. 512–5.
56. Mohammed R, Rawashdeh J, Abdullah M. Machine learning with oversampling and undersampling techniques: overview study and experimental results. In: 2020 11th International Conference on Information and Communication Systems (ICICS). Piscataway, NJ, USA: IEEE; 2020. p. 243–8.
57. Chawla NV, Bowyer KW, Hall LO, Kegelmeyer WP. SMOTE: synthetic minority over-sampling technique. J Artif Intell Res. 2002;16:321–57. doi:10.1613/jair.953.
58. Mandl T, Modha S, Majumder P, Patel D, Dave M, Mandlia C, et al. Overview of the HASOC track at fire 2019: hate speech and offensive content identification in Indo-European languages. In: Proceedings of the 11th Annual Meeting of the Forum for Information Retrieval Evaluation. New York, NY, USA: ACM; 2019. p. 14–7.
59. Mullah NS, Zainon WMNW. Improving detection accuracy of politically motivated cyber-hate using heterogeneous stacked ensemble (HSE) approach. J Ambient Intell Humaniz Comput. 2023;14(9):12179–90. doi:10.1007/s12652-022-03763-7.
60. Muti A, Fernicola F, Barrón-Cedeño A. UniBoe's at SemEval-2023 task 10: model-agnostic strategies for the improvement of hate-tuned and generative models in the classification of sexist posts. In: Proceedings of the 17th International Workshop on Semantic Evaluation (SemEval-2023). Stroudsburg, PA, USA: ACL; 2023. p. 1138–47.
61. Antypas D, Camacho-Collados J. Robust hate speech detection in social media: a cross-dataset empirical evaluation. arXiv:2307.01680. 2023.

62. Wullach T, Adler A, Minkov E. Towards hate speech detection at large via deep generative modeling. IEEE Internet Comput. 2020;25(2):48–57.
63. 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.
64. Duong C, Zhang L, Lu CT. HateNet: a graph convolutional network approach to hate speech detection. In: 2022 IEEE International Conference on Big Data (Big Data). Piscataway, NJ, USA: IEEE; 2022. p. 5698–707.
65. Khan S, Fazil M, Sejwal VK, Alshara MA, Alotaibi RM, Kamal A, et al. BiCHAT: BiLSTM with deep CNN and hierarchical attention for hate speech detection. J King Saud Univ Comput Inf Sci. 2022;34(7):4335–44.
66. Misra R, Arora P. Sarcasm detection using news headlines dataset. AI Open. 2023;4(5):13–8. doi:10.1016/j.aiopen.2023.01.001.
67. Mai Z, Zhang J, Xu Z, Xiao Z. Is LLAMA 3 good at sarcasm detection? A comprehensive study. In: Proceedings of the 2024 7th International Conference on Machine Learning and Machine Intelligence (MLMI). New York, NY, USA: ACM; 2024. p. 141–5.
68. Thambi J, Samudrala SSH, Vadluri SR, Nair PC, Venugopalan M. Sarcasm detection in news headlines using ML and DL models. In: 2024 International Conference on Advances in Computing, Communication and Applied Informatics (ACCAI). Piscataway, NJ, USA: IEEE; 2024. p. 1–8.
69. Chy MSR, Chy MSR, Mahin MRH, Rahman MM, Hossain MS, Rasel AA. Sarcasm detection in news headlines using evidential deep learning-based LSTM and GRU. In: Asian Conference on Pattern Recognition. Cham, Switzerland: Springer; 2023. p. 194–202.
70. Singh N, Jaiswal UC. Sarcasm text detection on news headlines using novel hybrid machine learning techniques. ADCAIJ. 2024;13:e31601. doi:10.14201/adcaij.31601.
71. de Lima Santos DB, de Carvalho Dutra FG, Parreiras FS, Brandão WC. Assessing the effectiveness of multilingual transformer-based text embeddings for named entity recognition in Portuguese. In: International Conference on Enterprise Information Systems (ICEIS). Setúbal, Portugal: SCITEPRESS; 2021. p. 473–83.
72. Camp E. Sarcasm, pretense, and the semantics/pragmatics distinction. Noûs. 2012;46(4):587–634. doi:10.1111/j.1468-0068.2010.00822.x.