Minimax Theory and its Applications

Volume 11 (2026), No. 4s, 179 to 193

CCBD-Net: A Cloud-Based Approach for Cyberbullying Detection on Twitter Using Character-Level Features withCBCNN Model

Padma P[1], Dr. G. Siva Nageswara Rao[2]

[1]Research scholar, Department of Computer Science and Engineering, , Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India And Degree Lecturer in Telangana Tribal Welfare Residential Educational Institutions (TTWREIS), Hyderabad. Email: rachaspadma@gmail.com
[2]Professor, Department of Computer Science and Engineering, Koneru Lakshmaiah Education Foundation, Vaddeswaram, Guntur, Andhra Pradesh, India, Email: sivanags@kluniversity.in

Abstract
Cyberbullying has become a pervasive issue on social media platforms like Twitter, posing significant societal challenges due to its harmful impact on users' emotional well-being and the vast volume of data involved, making manual monitoring unfeasible. This research proposes CCBD-Net, a systematic approach for automating cyberbullying detection using cloud computing and Twitter data. The study begins with the collection of Twitter data, followed by data preprocessing, character-level feature extraction, and the utilization of a Convolutional Neural Network (CNN) model for cyberbullying prediction.The character level N-gram (CLNG)feature extraction involves extracting character-level features, which delve into the nuances of language use by analyzing individual characters within the text. To predict cyberbullying instances, the CCBD-Net employs character basedbi-directional CNN (CBCNN) model, a deep learning architecture suitable for text classification tasks. The model leverages the CLNG features extracted to make binary predictions regarding the presence and intensity of cyberbullying in each Twitter post or interaction.The Proposed CCBD-Net demonstrates an average improvement of approximately 2.083% in accuracy, 2.034% in precision, 1.646% in recall, 0.790% in F1-score, and 2.788% in sensitivity when compared to existing AI-based methods, highlighting its effectiveness in enhancing various performance metrics for CBD classification.

Keywords: Cyberbullying, Twitter, social media, Cloud computing, Character-level features, CLNG feature extraction, CBCNN model, Online safety, Hate speech, Text classification, Ethical considerations.

1. Introduction

The motivation for conducting research on cyberbullying detection from Twitter is rooted in the growing significance of social media platforms in contemporary society [1]. Twitter, as one of the most popular and influential social media platforms, has become a significant space for communication, information dissemination, and social interaction. However, it has also become a breeding ground for cyberbullying, which poses serious threats to individuals' psychological well-being, emotional health, and even physical safety. The anonymity and ease of communication on Twitter can embolden individuals to engage in harmful behavior[2], including harassment, hate speech, and the spread of false information, often targeting vulnerable individuals based on their race, gender, sexual orientation, or other personal characteristics.This pervasive issue has spurred a critical need to develop effective cyberbullying detection mechanisms on Twitter. Researchers are motivated to address this problem for several reasons. First and foremost, cyberbullying can have devastating consequences, leading to depression, anxiety, and even suicide among victims. Detecting and mitigating cyberbullying on a platform as prominent as Twitter can potentially save lives and protect users from harm.Second, the sheer volume of data on Twitter makes manual monitoring and moderation nearly impossible[3]. Automation and machine learning algorithms are necessary to sift through the vast amount of content generated on the platform to identify and address instances of cyberbullying promptly.Furthermore, understanding the dynamics of cyberbullying on Twitter can provide insights into the broader issues of online harassment, the spread of misinformation, and the degradation of online discourse. Such research can lead to the development of more effective policies and interventions[4], not only on Twitter but also on other social media platforms facing similar challenges.

In recent years, the pervasive issue of cyberbullying on social media platforms, with a particular emphasis on Twitter, has grown into a pressing societal concern. The problem stems from the ease with which individuals

can engage in harmful behaviors such as harassment, hate speech, and the dissemination of false information on this widely used platform[5]. Cyberbullying not only threatens the emotional and psychological well-being of its victims but also poses significant challenges for Twitter as a platform. Despite the platform's policies and efforts to curb abusive behavior, the sheer volume of data and the rapidity of online interactions make it increasingly difficult to monitor and address instances of cyberbullying effectively.Cyberbullying is a widespread issue affecting users across various demographics[6]. The problem encompasses not only direct harassment but also the spread of harmful content, creating an environment that is hostile and intimidating for individuals.The impact of cyberbullying on victims is profound, with documented cases of depression, anxiety, and even self-harm. Therefore, the problem statement recognizes the critical need to protect individuals from harm and mitigate the potential consequences of online harassment[7]. Twitter, as a prominent social media platform, has a responsibility to ensure the safety of its users. The problem statement acknowledges that despite existing policies and moderation efforts, the dynamic nature of the platform presents unique challenges that require innovative solutions[8]. The problem also acknowledges the scale of data generated on Twitter and the necessity of automated solutions, such as machine learning algorithms, to effectively detect and respond to cyberbullying incidents in real-time. Beyond the immediate concerns of individual well-being, the problem statement highlights the broader societal implications of online harassment. This includes the spread of misinformation, erosion of online discourse, and potential impacts on offline behavior, underscoring the need for comprehensive research.

So, the problem statement encapsulates the urgency of addressing cyberbullying on Twitter, considering its wide-reaching consequences on users, the platform's responsibility, and the complex dynamics involved. Research in this area is crucial not only for improving the online experience but also for advancing our understanding of the challenges and solutions in the evolving landscape of social media[9].So, the novel contributions of this work are as follows:

- By mitigating cyberbullying, the research directly contributes to creating a safer cloud environment, protecting users from emotional and psychological harm, and fostering a more positive social media experience.
- The study proposes an automated approach, CCBD-Net, for detecting cyberbullying on Twitter, relieving the burden of manual monitoring and enhancing online safety for users.
- By employing rigorous data preprocessing techniques, the research ensures the utilization of high-quality, consistent data, thereby improving the accuracy of cyberbullying detection algorithms.
- Introducing character-level features through CLNG feature extraction techniques enables the identification of nuanced language patterns, enhancing the precision of cyberbullying prediction models.
- The research employs CBCNN models, a powerful deep learning tool, for cyberbullying prediction.

The rest of the article is organized as follows: Section 2 contains a literature survey with existing drawbacks. Section 3 gives the detailed analysis of proposed CCBD-Net with CLNG feature extraction, CBCNN classifier. Section 4 gives the detailed simulation results with performance comparison. Section 5 gives the conclusion with future scope, and ended with references.

2. Literature survey

Murshed et al. [10] developed FAEO-ECNN to address the issue of cyberbullying detection in social media platforms. TheFAEO-ECNN approach, which combined topic modeling and deep learning techniques. In their methodology, they first employed topic modeling to extract relevant topics from social media data. Then, they used deep learning algorithms, specifically CNNs, to classify these topics as cyberbullying or non-cyberbullying content. This two-step process allowed them to achieve more accurate detection of cyberbullying incidents in social media.Neha Minder Singh et al. 11] presented an efficient automated multi-modal cyberbullying detection system. Their methodology involved using multiple modalities, such as text and images, from social media platforms. They applied a decision fusion classifier, which combined the results from different modalities to improve the overall accuracy of cyberbullying detection. This approach ensured that various aspects of cyberbullying, including text-based and image-based content, were considered in the detection process.

Alanoud Mohammed et al. [12] focused on detecting Arabic cyberbullying tweets using machine learning techniques. Their methodology included preprocessing the Arabic text data, which involved tokenization and stemming to prepare the text for analysis. They then applied machine learning algorithms, such as support vector machines (SVM) and decision trees, to classify the tweets as either cyberbullying or non-cyberbullying. This approach enabled them to address the unique challenges of detecting cyberbullying in the Arabic language.In 2023, Mohammed Hussein Obaid, et al. [13] developed a model for cyberbullying detection and severity determination. Their methodology involved using a combination of natural language processing (NLP)

techniques and machine learning algorithms. They first used NLP to extract features from social media content, including text and user interactions. Then, they employed machine learning models to classify the content as cyberbullying and determine its severity level. This comprehensive approach allowed them to not only detect cyberbullying but also assess its seriousness.

Malek Alzaqebahet al. [14] proposed a cyberbullying detection framework, specifically designed for short and imbalanced Arabic datasets. Their methodology included feature engineering and hybrid machine learning techniques. They leveraged feature expansion techniques, such as FastText, to extract meaningful features from the short Arabic text data. Additionally, they employed a combination of CNNs and bidirectional long short-term memory networks (BiLSTMs) for classification. This hybrid approach addressed the challenges posed by limited and imbalanced data in Arabic cyberbullying detection.Esshaan Mahajan, et al. [15] presented the EnsMulHateCyb approach for multilingual hate speech and cyberbullying detection in online social media. Their methodology incorporated the use of pretrained language models like BERT and GPT-3 to capture contextual information from the text. They also employed multilingual embeddings to handle diverse languages. Their approach demonstrated the ability to detect cyberbullying and hate speech across multiple languages, making it versatile for various social media platforms.Teng Hwai Teoh et al. [16] conducted a study, comparing machine learning and transfer learning approaches for cyberbullying detection in social networks. Their methodology involved implementing traditional machine learning algorithms, such as SVM and random forests, as well as transfer learning techniques, using pretrained deep learning models like BERT. By comparing these approaches, they aimed to identify the most effective method for cyberbullying detection in social networks.Amgad Muneer et al. [17] proposed a cyberbullying detection model that utilized stacking ensemble learning and enhanced BERT. Their methodology involved a two-stage approach. In the first stage, they employed various pretrained BERT models to extract contextual embeddings from social media text. In the second stage, they used stacking ensemble learning techniques to combine the predictions from multiple base models, enhancing the overall accuracy of cyberbullying detection.

Fan Wu et al. [18] introduced the FACapsnet, which was a fusion capsule network with congruent attention for cyberbullying detection. Their methodology included the use of capsule networks (CapsNets) and attention mechanisms. CapsNets are neural networks designed to capture hierarchical relationships in data, while the attention mechanism helped the model focus on relevant information in the text. This combination allowed for effective cyberbullying detection with attention to nuanced features in social media content.Muhammad Alfi et al. [19] enhanced cyberbullying detection on Indonesian Twitter by leveraging FastText for feature expansion and applying a hybrid approach using CNNs and BiLSTMs. Their methodology involved using FastText embeddings to enrich the feature space and then employing CNNs and BiLSTMs for classification. This hybrid approach was tailored to the Indonesian language and demonstrated improved performance in detecting cyberbullying on Twitter.

P. Nagaraj et al. [20] conducted a study on cyberbully detection on social media. Their methodology involved collecting and preprocessing social media data, including text and user interactions. They then applied machine learning models to classify the data as cyberbullying or non-cyberbullying content. Their research aimed to contribute to the development of effective cyberbullying detection systems for social media platforms.Nisha P. Shetty et al. [21] introduced FedBully, a cross-device federated approach for privacy-enabled cyberbullying detection using sentence encoders. Their methodology incorporated federated learning, which allowed models to be trained across multiple devices without sharing sensitive data. They used sentence encoders to represent social media text, ensuring privacy while still achieving accurate cyberbullying detection.Jianwei Zhang, et al. [22]constructed a Japanese bullying expression dictionary for automated cyberbullying detection on Twitter. Their methodology involved creating a dictionary of Japanese bullying expressions and then using this dictionary to identify cyberbullying content on Twitter. This approach aimed to improve the accuracy of cyberbullying detection in the Japanese language.Yang Jing et al. [23] explored soft computing techniques for detecting cyberbullying in social multimedia data. Their methodology involved the use of soft computing methods, including fuzzy logic and neural networks, to analyze social multimedia data for signs of cyberbullying. These techniques allowed for the consideration of complex and nuanced information in multimedia content, enhancing the detection of cyberbullying incidents.

Nitasha Dhingra et al. [24] proposed an improved method for detecting cyberbullying on social media using randomized sampling. Their methodology involved randomly selecting samples from the social media data and applying machine learning models to these samples. This randomized sampling approach aimed to reduce the computational complexity while maintaining the accuracy of cyberbullying detection.Darwin Samalo, et al. [25] presented an improved model for identifying cyberbullying based on tweets from Twitter. Their studyfocused on enhancing cyberbullying detection using an improved model. The methodology they employed included several key steps.First, they collected a dataset of tweets from Twitter, which served as their source of social media

data. They preprocessed this data by cleaning and tokenizing the text to prepare it for analysis. Next, they engineered features from the preprocessed text data, extracting relevant information that could be indicative of cyberbullying. These features likely included text-based attributes such as keywords sentiment analysis scores, and other linguistic characteristics.

3.Proposed Methodology

The research problem at hand concerns the pervasive issue of cyberbullying on Twitter, a platform where individuals readily engage in harmful behaviors like harassment and hate speech. This problem, with its far-reaching societal implications, stems from the platform's vast data volume, making manual monitoring infeasible, and the substantial impact on victims' emotional and psychological well-being. Effectively mitigating cyberbullying is imperative not only to safeguard users but also for Twitter's responsibility as a prominent platform. The multifaceted nature of this problem demands innovative, automated solutions, including machine learning algorithms, to detect and address cyberbullying instances in real-time. Research in this area is vital, not only for enhancing online safety but also for comprehending the broader dynamics of online harassment, misinformation, and the evolving landscape of social media.

Figure 1illustrates the proposed CCBD-Net block diagram, which represents cyberbullying detection using cloud computing and Twitter data. The proposed research work follows a systematic procedure, beginning with the collection of Twitter data and progressing through data preprocessing, character-level feature extraction, and ultimately using a CNN model for cyberbullying prediction. This structured approach ensures that the research is based on clean and relevant data and leverages advanced machine learning techniques to address the critical issue of cyberbullying detection on social media platforms like Twitter.Each step in this procedure plays a crucial role in the overall process:

Step 1: Twitter Data - In this initial stage, the research work begins by collecting Twitter data. This could involve gathering a substantial dataset of Twitter posts, tweets, or user interactions that are relevant to the study's objectives. This step is critical as the quality and quantity of the data collected will directly impact the accuracy and effectiveness of subsequent analysis.

Step 2: Data Preprocessing - After acquiring the Twitter data, it undergoes preprocessing to clean and prepare it for analysis. This may include tasks such as removing irrelevant or duplicate content, handling missing data, and standardizing the format of the text. Data preprocessing ensures that the subsequent analysis is conducted on high-quality, consistent data.

Step 3: CLNG Feature Extraction - In this step, the research work focuses on extracting features from the preprocessed Twitter data at the character level using CLNG feature extraction. Character-level features are derived from the individual characters within the text, which is useful for identifying patterns and nuances in language use. This might involve techniques such as character n-grams, where sequences of characters are analyzed for patterns that are indicative of cyberbullying.

Step 4: CBCNN Model Prediction - Finally, the research employs a CBCNN model for prediction. CBCNNs are a type of deep learning model commonly used for text classification tasks, including cyberbullying detection. The model leverages the character-level features extracted in the previous step to make predictions regarding whether a given Twitter post or interaction contains elements of cyberbullying. This prediction is binary (cyberbullying present or not) or multiclass (categorizing different forms or intensities of cyberbullying).

3.1. Dataset Preprocessing

Thepreprocessing operation is a crucial step in preparing the training data for a cyberbullying detection model. It involves data reading, cleaning, tokenization, label encoding, and data organization, ensuring that the data is suitable for training a machine learning model to detect cyberbullying in text messages. Figure 2 shows the proposed preprocessing block diagram.

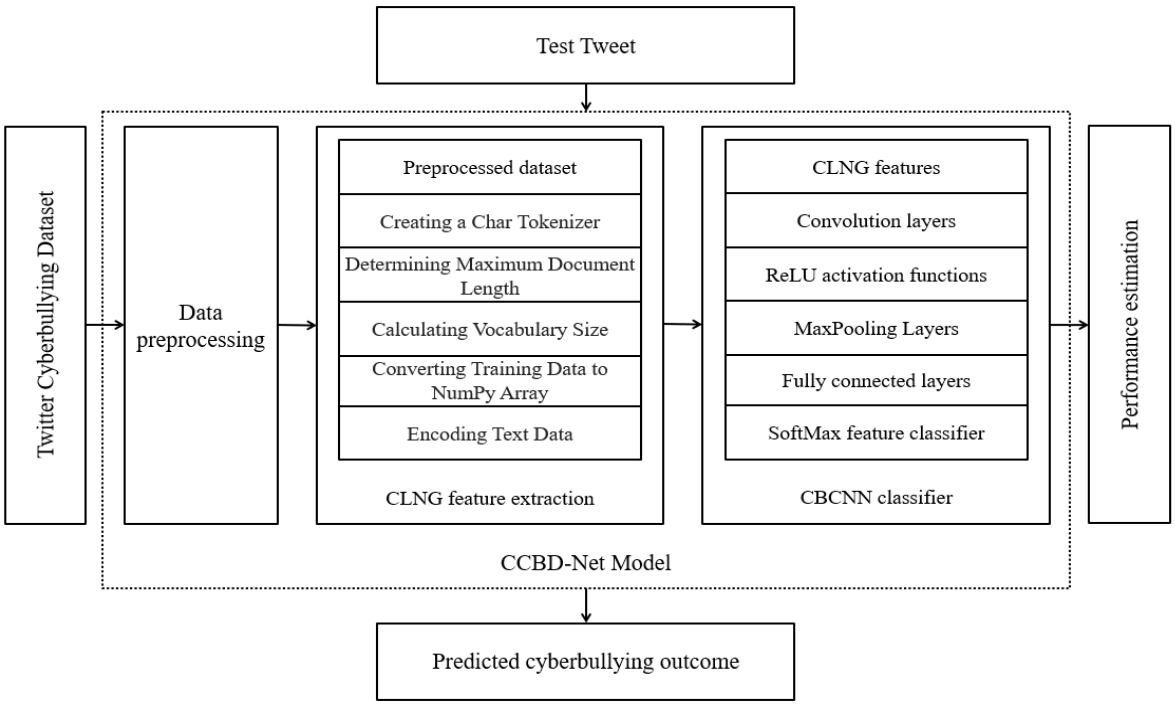


Figure 1. Proposed CCBD-Net block diagram.

Step 1: Initialization and Clearing Variables: At the beginning of the preprocessing operation, the code initializes and clears global variables. These global variables are used to store training data and count the instances of bullying and non-bullying content. Initializing and clearing these variables is essential to ensure that the preprocessing operation starts with a clean slate.

Step 2: Reading and Processing the Training Data: In this step, the code reads a CSV file containing the training data. The CSV file is specified by the "filename" variable. It's assumed that this file contains the training data needed to train the cyberbullying detection model. The data is encoded using the "iso-8859-1" encoding and separated by tabs ('\t'). A loop is then used to iterate over each row of the training data.

Step 3: Extracting Labels and Text Messages: Within the loop, the code retrieves two important pieces of information for each training instance: the label and the text message. The label is extracted from the 'label_bullying' column, and the corresponding text message is extracted from the 'text_message' column. Additionally, any leading or trailing whitespace in the text message is removed using the "strip ()" function. This step ensures that the label and text message are correctly associated and that there are no extraneous spaces.

Step 4: Tokenizing and Preprocessing the Text Messages: The code then processes the text messages further by passing them through a "clean_doc()" function. While the specific details of the "clean_doc()" function are not provided, it is typically responsible for tokenizing the text, removing stop characters, punctuation marks, and special characters, and performing other text preprocessing tasks. The resulting tokens (cleaned text) are stored in the "tokens" variable.

Step 5: Storing Character Data: If the number of tokens in a text message is greater than 0 (indicating that there are valid tokens present), the tokens are appended to the "char_trainX" list. This list is intended to represent the char-level training data. Simultaneously, the corresponding label is appended to the "char_trainy" list. This step effectively organizes the data into input features (char-level text) and labels (bullying or non-bullying).

Step 6: Counting Bullying and Non-Bullying Instances: This step involves incrementing counters to keep track of the number of bullying and non-bullying instances processed during the loop. This counting is important for later analysis and for understanding the distribution of classes in the training data.

Step 7: Label Encoding: The labels in the "char_trainy" list are converted into a categorical format using the "to_categorical()" function. This encoding likely transforms the labels into one-hot encoded vectors. One-hot encoding is a common practice in machine learning when dealing with categorical labels.

Step 8: Displaying Information: Finally, the code updates the user interface, specifically a text area, by displaying the lengths of the char-level and character-level training data arrays. This information provides insights into the size of the processed data, which can be helpful for monitoring the progress of the preprocessing operation.

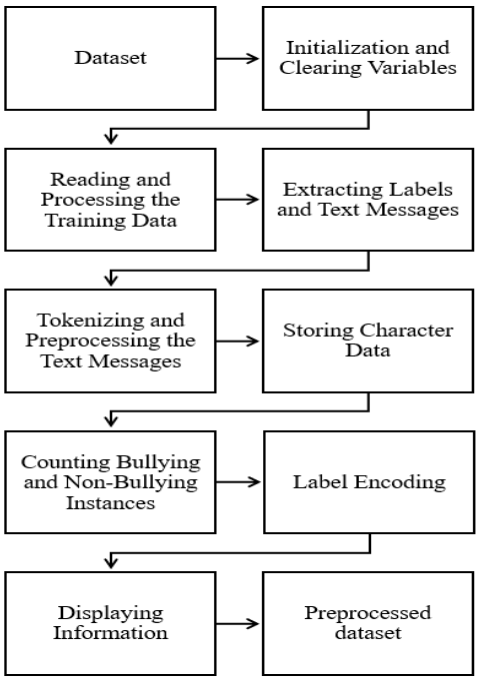


Figure 2. Proposed Preprocessing Steps.

3.2. CLNG Feature Extraction

Figure 3 shows the proposed CLNG feature extraction block diagram.Initially, tokenization divides the text into individual characters while eliminating punctuation marks, non-alphabetic characters, and short tokens. These measures serve to reduce noise and ensure uniformity across the dataset. The removal of stop characters further enhances data cleanliness. Subsequently, the cleaned characters are rejoined into a single string, simplifying the text for subsequent analysis. Determining the maximum document length aids in understanding the data's variability, a crucial aspect when dealing with text of varying lengths, as seen in social media posts. Calculating the vocabulary size quantifies the unique characters in the dataset, fundamental for encoding and modeling tasks. Finally, the conversion of training data into a NumPy array and encoding text data as numerical tokens facilitates machine learning model compatibility, and displaying pertinent information helps researchers comprehend the dataset's characteristics, enabling informed decisions throughout the analysis.

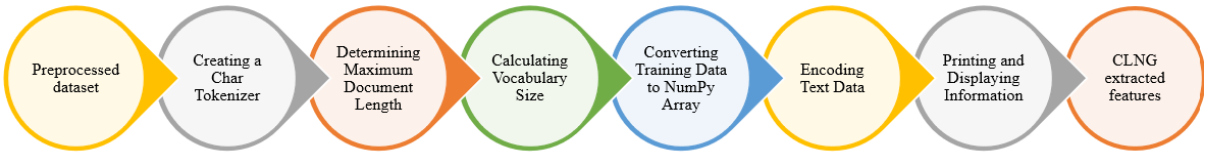


Figure 3. CLNG feature extraction block diagram.

Step 1: Creating a Char Tokenizer: Figure 4 shows theprocess of character tokenizer.Tokenization, the first step, involves splitting the input text into individual characters, providing granularity for subsequent analysis. The removal of punctuation marks streamlines the text, discarding symbols that may not contribute meaning, as seen in the transformation of "Hello, world!" into "Hello world." Eliminating non-alphabetic tokens ensures that only letters are considered during analysis, reducing noise and irrelevance in text data. The removal of stop characters, common non-informative elements like spaces or tabs, further refines the text, while the elimination of short tokens, typically single characters, or very brief words, helps purify the dataset by eliminating

distracting elements. Ultimately, these cleaning processes culminate in the reconstruction of the text into a coherent, cleaned string, facilitating subsequent analysis and model training.

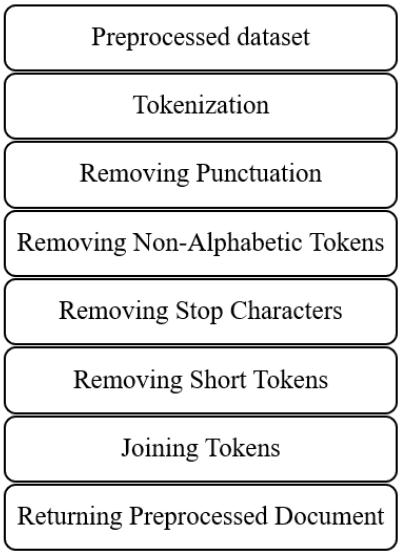


Figure 4. Process of character tokenizer.

- **Tokenization:** Tokenization is the process of breaking a text document into smaller units, such as words or, in this case, characters. This step involves splitting the input text into individual characters. For example, the word "hello" would be split into the characters "h," "e," "l," "l," "o."

- **Removing Punctuation:** Punctuation marks (e.g., '.', '!', '?') are often not relevant for text analysis and can be removed to simplify the text. For instance, "Hello, world!" would become "Hello world" after removing punctuation.

- **Removing Non-Alphabetic Tokens:** Sometimes, non-alphabetic characters like numbers or special symbols are not informative for the analysis. Removing these non-alphabetic characters ensures that only letters are considered. For example, "H3ll0 w0rld!" would become "Hllwrld" after this step.

- **Removing Stop Chars:** Stop characters are typically common, non-informative characters like spaces or tabs. Removing these characters helps clean the text further and standardize its format.

- **Removing Short Tokens:** Tokens with a length of one character, such as single letters or very short words, are often removed. This step helps eliminate noise from the data.

- **Joining Tokens:** After all these cleaning steps, the remaining characters are joined back together into a single string, separated by spaces. This reconstructed string is much cleaner and devoid of unwanted characters.

Step 2: Determining Maximum Document Length: This step involves finding the longest document or sequence of characters in the dataset. It's crucial for understanding the variability in document lengths. For example, tweets of varying lengths, and knowing the longest tweet helps with data preprocessing decisions like padding or truncating sequences to ensure they have consistent lengths for model training.

Step 3: Calculating Vocabulary Size: The vocabulary size represents the number of unique characters in the text data. It's essential for tasks like creating embeddings or one-hot encoding. For example, if the text contains characters like 'a,' 'b,' 'c,' 'd,' and 'e,' the vocabulary size would be 5.

Step 4: Converting Training Data to NumPy Array: This step involves converting the text data into a numerical format (NumPy array). Machine learning models require numerical input. Each character in the cleaned text is assigned a numerical value or index, making it suitable for further processing and analysis.

Step 5: Encoding Text Data: The encoded text data is obtained by converting the characters into numerical tokens, where each character is represented by a unique number. This step allows the machine learning model to

work with the text data as numerical input. It also often involves padding or truncating sequences to ensure they have the same length for model consistency.

Step 6: Printing and Displaying Information: Finally, this step provides essential information about the preprocessed data, such as the maximum document length, char vocabulary size, and the shape of the char training data. This information helps researchers understand the characteristics of their data and make informed decisions during model training and evaluation.

3.3. CBCNN Model

Like a traditional neural network, a CBCNN differs in its architecture from a CNN, ANN, and RNN. In a CBCNN, instead of separate inputs for each neuron, the network processes text data at the character level. Each character is treated as a feature, allowing the network to capture intricate patterns and nuances in the text.Figure 5 shows the architecture of CBCNN model with multiple layers. A CBCNN, like its counterpart, consists of several layers designed to extract meaningful features from text data. These layers include the character-level convolutional layer, the maximum pooling layer, and the fully connected layer. These layers work sequentially to process the input text and make predictions about the presence of cyberbullying.The character-level convolutional layer serves as the core of the network, responsible for learning feature representations from the input characters. It uses convolutional filters to scan the text and identify important patterns and structures related to cyberbullying. The convolutional layer is followed by a nonlinear activation function, often rectified linear unit (ReLU), which introduces non-linearity into the model and helps capture complex relationships in the data.Mathematical model of the whole process is found in Equation (1).

$$y(k) = f\left(\sum_1^N \tilde{x}j(k) \times W_p(k) + bp(k)\right) \quad (1)$$

Here, $W_p$represents the weights of convolution layer, $\tilde{x}$ represents the batch normalization of input features, $f(.)$represents the activation function ReLU,$p$represents the number of convolutional filters, $b_p$represents bias term value ranges from 0–1. The MaxPooling (maximum pooling) layer acts as a feature selector by reducing the dimensionality of the extracted features while retaining their most relevant information. This layer increases the model's robustness by focusing on the most salient aspects of the text data.The fully connected layer aggregates the features learned in the previous layers and connects them to the output layer. In the context of cyberbullying detection, the output layer typically employs a sigmoid or SoftMax activation function to produce binary or multi-class predictions, respectively.In the case of cyberbullying detection using a CBCNN, the input features consist of characters from the text data, and the output is a binary classification, where 1 represents the presence of cyberbullying and 0 signifies its absence. The CBCNN learns to detect patterns of cyberbullying directly from the character-level input, making it effective at handling diverse and context-rich online text data.To enhance the CBCNN's performance, batch normalization is applied to normalize input features within batches, reducing the risk of gradient saturation during training. Additionally, the ReLU activation function is utilized to introduce non-linearity, enabling the network to learn complex relationships between characters and cyberbullying patterns efficiently.

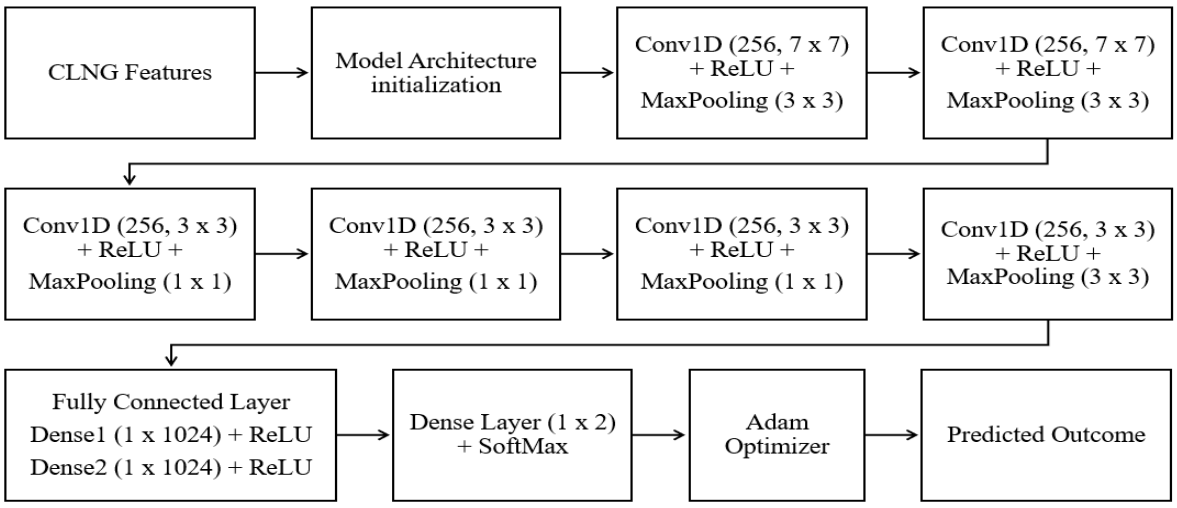


Figure 5. CBCNN architecture.

4. Results and discussion

In this research, the Twitter dataset was selected as the primary dataset for conducting simulations and evaluating the proposed model's performance. To assess the effectiveness of the proposed model, a comparative analysis was carried out, pitting it against an existing model using the same Twitter dataset. It's worth noting that the simulation environment remained consistent across all models under evaluation. This standardization of the simulation environment is a crucial factor in ensuring that the performance comparison is conducted under fair and valid conditions. By utilizing the same dataset and simulation setup for both models, researchers can draw meaningful conclusions about the proposed model's relative performance and effectiveness when compared to the existing model, thus contributing to the advancement of cyberbullying detection methods in a reliable and consistent manner.

4.1 Dataset

This work used the Twitter API's streaming functionality and around 32 distinct cyber-bullying keychars to compile the input dataset of tweets. Some examples of chars and phrases that are discouraged by the psychological literature include: idiot, ni**er, LGBTQ (le***an, g*y, bisexual, transgender, and queer), whore, pussy, faggot, crap, sucker, slut, donkey, live, frightened, moron, poser, rape, fuck, fucking, ugly, bitch, ass, whale, etc. Nonetheless, other chars like "ban," "kill," "die," "evil," "hate," "attack," "terrorist," "threat," "racism," "black," and "Muslim," "Islam," and "Islamic" were also proposed as suitable keychars. Char-based terms including racism, insult, sexism, and profanity make up over 130000 of the original dataset's 435764 tweets. There are many out-of-the-ordinary tweets in this data collection. Tweets published in other languages that include terms from English are removed, and retweets are checked for improper material since only tweets written in English are allowed. After all the types of irrelevant tweets were deleted, the remaining tweets were randomly selected to form the final dataset of about 10,000 tweets. As part of the procedure's preliminary processing, all these activities are carried out automatically.

4.2 Training environment

The procedure outlines the steps for training a neural network model using a Twitter dataset. The data is split into training and validation sets, and the model is trained over multiple epochs, with batch processing, using the 'adam' optimizer and 'categorical_crossentropy' loss function.

Data Splitting: The research work begins by splitting the dataset into training and validation sets. In this case, 20% of the data is allocated to the validation set (specified by **validation_split=0.2**). This division allows for model training on one portion of the data while validating its performance on another, which helps prevent overfitting and assesses the model's generalization ability.

Batch Processing: During model training, data is processed in batches rather than all at once. In this procedure, a batch size of 128 (**batch_size=128**) is chosen. This means that the training dataset is divided into 128-sample batches, and the model's weights are updated after processing each batch. Batch processing helps optimize memory usage and can lead to faster convergence during training.

Training Epochs: The model is trained over a specified number of epochs. In this case, the model will undergo training for 10 epochs (**epochs=10**). An epoch represents one complete pass through the entire training dataset. During each epoch, the model iteratively learns from the training data and updates its weights.

Optimizer: The choice of optimizer for updating the model's weights is 'adam'. Adam is a popular optimization algorithm that adapts the learning rate during training, making it suitable for a wide range of machine learning tasks. It is often used in deep learning models for its efficiency and effectiveness in gradient descent optimization.

Loss Function: The loss function used for this research work is 'categorical_crossentropy'. This loss function is commonly employed in multi-class classification tasks. It measures the dissimilarity between the predicted class probabilities and the true class labels, providing a quantifiable measure of how well the model is performing. The goal during training is to minimize this loss.

Model Training: The actual model training occurs within the **char_model.fit()** function. During training, the model learns to make predictions based on the input data (x_train) and adjusts its internal parameters (weights) to minimize the specified loss function. The verbose mode is set to 1, which means that training progress and performance metrics are displayed in the output, enabling researchers to monitor the model's training process.

Table 1 and Figure 6 presents a comprehensive performance comparison of the proposed CCBD-Net method across various batch sizes: 128, 64, and 32. These metrics provide valuable insights into the effectiveness of the model under different training configurations.

Accuracy (%): Accuracy measures the overall correctness of the model's predictions, indicating the percentage of correctly classified instances. It is evident that as the batch size decreases from 128 to 32, there is a slight decrease in accuracy. This suggests that larger batch sizes lead to marginally better overall classification performance.

Precision (%): Precision quantifies the model's ability to make accurate positive predictions, minimizing false positives. The results indicate that as the batch size decreases, precision remains relatively high, demonstrating the model's consistent ability to make precise cyberbullying predictions.

Recall (%): Recall gauges the model's capability to correctly identify all actual positive instances, reducing false negatives. Similar to precision, recall also maintains a high level of performance across different batch sizes, with only a slight reduction as the batch size decreases.

F1-score (%): The F1-score is the harmonic mean of precision and recall, providing a balanced measure of a model's performance. Here, we observe a similar pattern to precision and recall, with a slight decrease in F1-score as the batch size decreases. This suggests that the model's balance between precision and recall remains consistent.

Sensitivity (%): Sensitivity is synonymous with recall and represents the model's ability to identify true positives. In this case, sensitivity exhibits a high level of performance across various batch sizes, emphasizing the model's consistent capability to correctly detect instances of cyberbullying.

Table 1. Performance comparison of proposedCCBD-Net method for various batch sizes.

Metric	Batch Size 128	Batch Size 64	Batch Size 32
Accuracy (%)	98.75	96.170	95.095
Precision (%)	98.98	97.386	95.720
Recall (%)	97.58	96.091	95.450
F1-score (%)	97.34	96.395	95.959
Sensitivity (%)	99.15	97.499	95.982

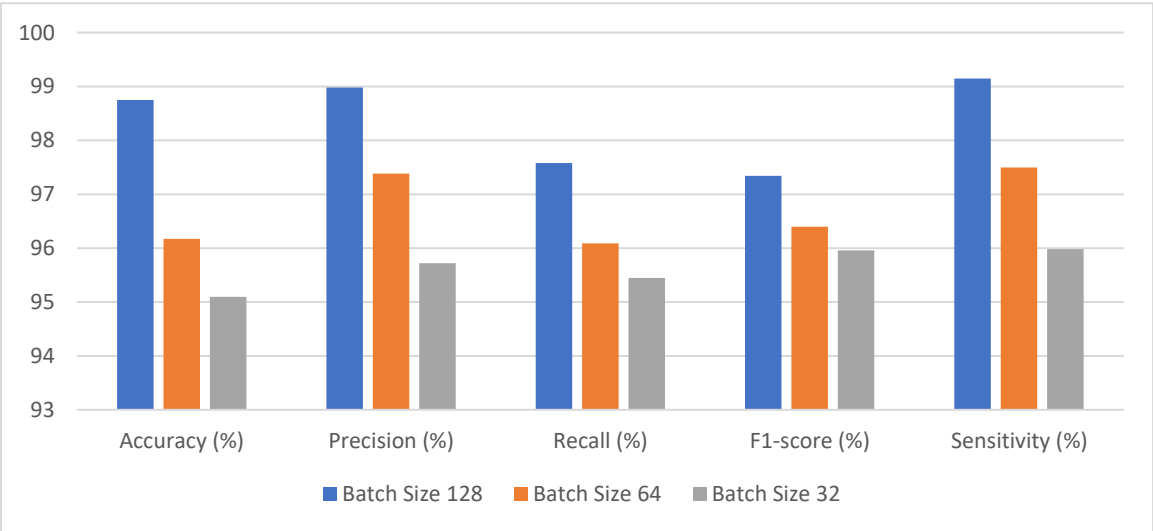


Figure 6. Graphical representation of proposed CCBD-Net method performance for various batch sizes.

Table 2 presents a detailed performance comparison of the proposed CCBD-Net method across various numbers of training epochs, ranging from Epoch 1 to Epoch 10. Each row represents a different epoch, and the corresponding evaluation metrics provide insights into how the model's performance evolves during training. Figure 7 shows the graphical representation of proposed CCBD-Net method performance for various epochs.The accuracy metric ranges from a minimum of approximately 96.056% in Epoch 3 to a maximum of 98.75% in Epoch 10, showcasing an improvement in the model's overall correctness of predictions over the training epochs.Precision values fluctuate within the range of approximately 96.148% (in Epoch 9) as the minimum and 98.98% (in Epoch 10) as the maximum.Recall values span from a minimum of around 96.051% (in Epoch 7) to a maximum of 97.872% (in Epoch 3).The F1-score metric varies between approximately 96.215% (in Epoch 2) as the minimum and 97.34% (in Epoch 10) as the maximum.Sensitivity values range from a minimum of approximately 96.011% (in Epoch 4) to a maximum of 99.15% (in Epoch 10).

Table 2. Performance comparison of proposed CCBD-Net method for various epochs.

Epochs	Accuracy (%)	Precision (%)	Recall (%)	F1-score (%)	Sensitivity (%)
Epoch 1	96.371	97.911	97.331	96.875	97.995
Epoch 2	96.476	96.673	96.738	96.215	96.064
Epoch 3	96.056	96.648	97.872	96.981	97.969
Epoch 4	96.077	97.113	96.449	97.255	96.011
Epoch 5	97.686	96.321	96.459	96.844	97.450
Epoch 6	96.986	96.354	96.124	96.313	97.930
Epoch 7	96.864	96.498	96.051	97.975	96.394
Epoch 8	97.125	96.160	97.451	97.842	97.609
Epoch 9	96.876	96.148	96.476	97.760	96.324
Epoch 10	98.75	98.98	97.58	97.34	99.15

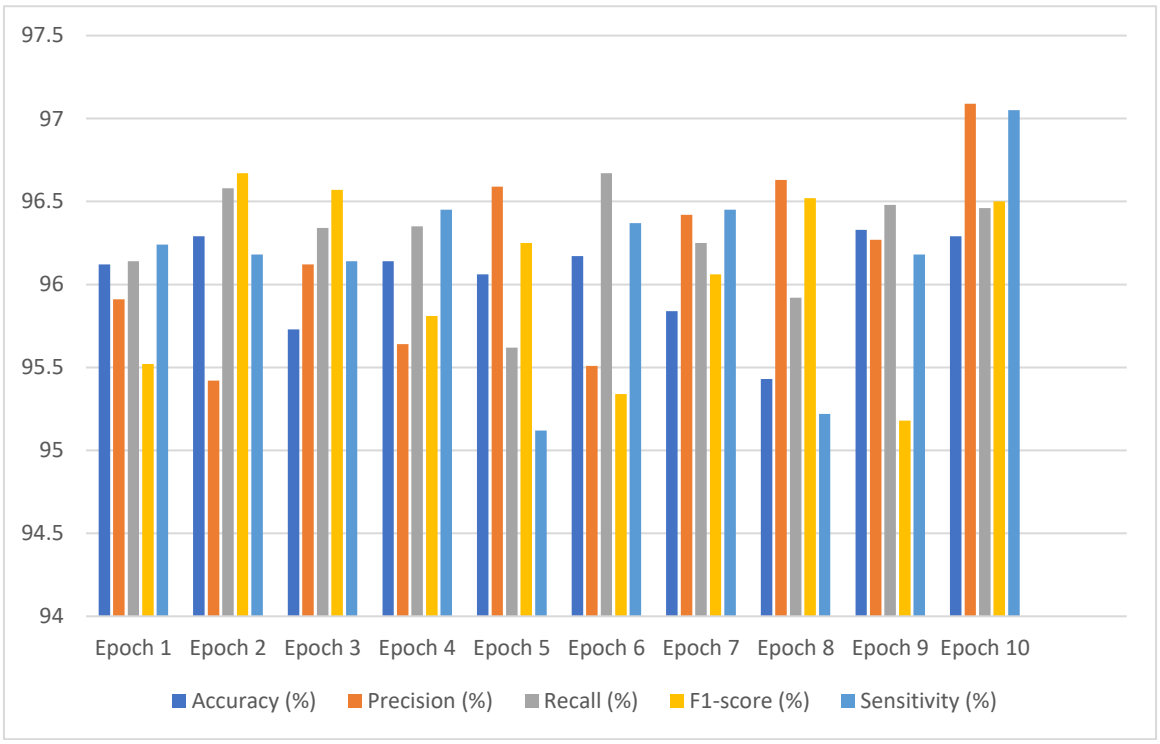


Figure 7. Graphical representation of proposed CCBD-Net method performance for various epochs.

4.3 Simulation results

Table 3 compares the performance of various AI-based CBD classification methods. Here, proposed CCBD-Net resulted in superior cyberbullying detection performance as compared to SVM [12], BiLSTM [14],and FAEO-ECNN [10]. Figure 8 shows thegraphical representation of various AI based CBD methods performance.It achieves a 3.35% higher accuracy than SVM, a 3.30% higher accuracy than BiLSTM, and a 1.60% higher accuracy than FAEO-ECNN. Additionally, in terms of precision, the Proposed CCBD-Net outperforms SVM by 2.64%, BiLSTM by 2.32%, and FAEO-ECNN by 2.63%. In recall, it showcases improvements of 2.34% over SVM, 2.07% over BiLSTM, and 0.73% over FAEO-ECNN. The F1-score is also enhanced, with the Proposed CCBD-Net achieving a 1.40% improvement over SVM, a 1.08% improvement over BiLSTM, and a 0.74% improvement over FAEO-ECNN. Furthermore, its sensitivity is improved by 2.10% compared to SVM, 2.08% compared to BiLSTM, and 1.47% compared to FAEO-ECNN. These improvements underscore the effectiveness of the proposed CCBD-Net in enhancing the accuracy and precision of CBD classification, making it a promising advancement in AI-based CBD analysis.

Table 3. Performance estimation of various AI based CBDmethods.

Metric	SVM [12]	BiLSTM [14]	FAEO-ECNN [10]	Proposed CCBD-Net
Accuracy (%)	95.312	95.446	97.151	98.75
Precision (%)	95.344	96.658	96.287	98.98
Recall (%)	95.076	95.552	97.806	97.58
F1-score (%)	95.718	95.391	96.570	97.34
Sensitivity (%)	96.235	96.048	97.759	99.15

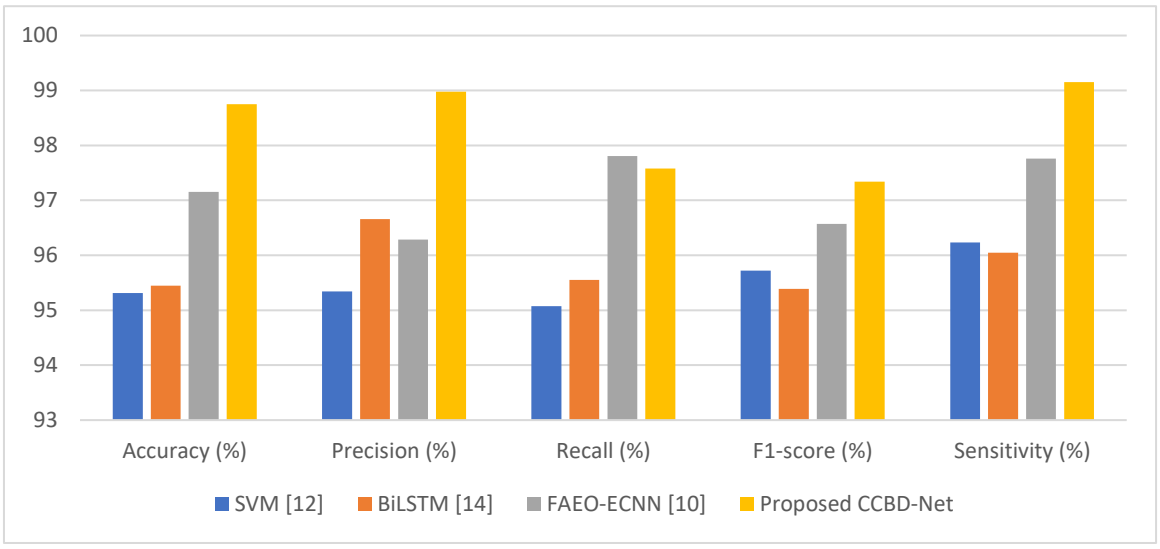


Figure 8. Graphical representation of various AI based CBD methods performance.

Table 4 compares the performance of various DL-based CBD classification methods. Here, proposed CCBD-Net resulted in superior cyberbullying detection performance as compared to EnsMulHateCyb [15], FedBully [21], and FACapsnet [18]. Figure 9 shows the graphical representation of various DL based CBD methods performance.It achieves a substantial 3.405% higher accuracy than EnsMulHateCyb, a 1.008% higher accuracy than FedBully, and an impressive 0.841% higher accuracy than FACapsnet. In terms of precision, the Proposed CCBD-Net outperforms EnsMulHateCyb by 2.490%, FedBully by 2.463%, and FACapsnet by 1.149%. Furthermore, in recall, it demonstrates a 1.996% improvement over EnsMulHateCyb, a 0.469% improvement over FedBully, and a 1.895% improvement over FACapsnet. The F1-score is also significantly enhanced, with the Proposed CCBD-Net achieving a 0.532% improvement over EnsMulHateCyb, a 0.174% improvement over FedBully, and a 0.818% improvement over FACapsnet. Additionally, its sensitivity is greatly improved, with a

substantial 4.870% improvement compared to EnsMulHateCyb, a notable 2.171% improvement compared to FedBully, and an impressive 0.757% improvement compared to FACapsnet. These substantial improvements highlight the efficacy of the Proposed CCBD-Net in augmenting the accuracy, precision, and overall performance of CBD classification models, positioning it as a promising advancement in AI-based CBD analysis.

Table 4. Performance estimation of various DL based CBD methods.

Metric	EnsMulHateCyb [15]	FedBully [21]	FACapsnet [18]	Proposed CCBD-Net
Accuracy (%)	95.345	97.742	97.909	98.75
Precision (%)	96.490	96.517	97.831	98.98
Recall (%)	95.584	97.111	98.685	97.58
F1-score (%)	96.703	97.166	96.522	97.34
Sensitivity (%)	94.925	96.979	98.393	99.15

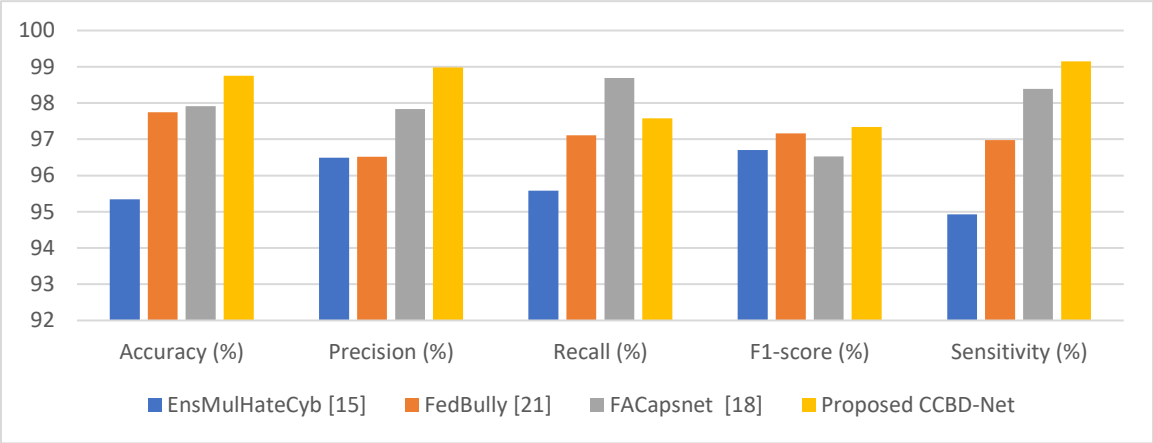


Figure 9. Graphical representation of various DL based CBD methods performance.

5. Conclusion

In an era where social media has become an integral part of our lives, the pervasive issue of cyberbullying on platforms like Twitter demands innovative and automated solutions. This research presented CCBD-Net, a structured approach for cyberbullying detection, rooted in the understanding that safeguarding users and fostering a responsible online environment are paramount.The collection of Twitter data, the first step in our methodology, serves as the foundation for subsequent analysis. Quality and quantity considerations ensure the reliability of our findings while respecting ethical data usage guidelines.Data preprocessing, the second step, optimizes the data for analysis by eliminating noise and inconsistencies. Clean data enhances the robustness of our approach, making it more effective in identifying cyberbullying instances.The introduction of character-level feature extraction, the third step, brings nuance to our analysis. By scrutinizing individual characters within text, we identify subtle patterns and nuances in language use that are indicative of cyberbullying. This finer-grained analysis is crucial for improving detection accuracy.Finally, our utilization of a CBCNN model for prediction underscores the power of deep learning in tackling complex text classification tasks. Leveraging the character-level features, this model delivers binary or multiclass predictions, allowing for a nuanced understanding of cyberbullying intensity. Further, this work can be extended to synonym-based CBD with advanced transfer learning, extreme learning models.

References

1. Balakrisnan, Vimala, and Mohammed Kaity. "Cyberbullying detection and machine learning: a systematic literature review." *Artificial Intelligence Review* (2023): 1-42.

2. Pokhun, Leevesh, and Yasser M. Chuttur. "Can Machine Learning Really Detect Cyberbullying?." *International Journal of Bullying Prevention* (2023): 1-20.

3. Gupta, Anamika, et al. "Machine Learning Methods for detection of bystanders: A Survey." *International Journal of Advanced Computer Technology* 12.4 (2023): 06-14.

4. Yi, Peiling, and Arkaitz Zubiaga. "Session-based cyberbullying detection in social media: A survey." *Online Social Networks and Media* 36 (2023): 100250.

5. Hasan, Md Tarek, et al. "A Review on Deep-Learning-Based Cyberbullying Detection." *Future Internet* 15.5 (2023): 179.

6. Mahmud, Tanjim, et al. "Cyberbullying detection for low-resource languages and dialects: Review of the state of the art." *Information Processing & Management* 60.5 (2023): 103454.

7. Akhter, Arnisha, et al. "A robust hybrid machine learning model for Bengali cyber bullying detection in social media." *Natural Language Processing Journal* 4 (2023): 100027.

8. Akinyemi, Joseph D., et al. "Cyberbullying Detection and Classification in Social Media Texts Using Machine Learning Techniques." *International Conference on Computer Science, Engineering and Education Applications*. Cham: Springer Nature Switzerland, 2023.

9. Fati, Suliman Mohamed, et al. "Cyberbullying Detection on Twitter Using Deep Learning-Based Attention Mechanisms and Continuous Bag of Words Feature Extraction." *Mathematics* 11.16 (2023): 3567.

10. Murshed, Belal Abdullah Hezam, et al. "FAEO-ECNN: cyberbullying detection in social media platforms using topic modelling and deep learning." *Multimedia Tools and Applications* (2023): 1-40.

11. Singh, Neha Minder, and Sanjay Kumar Sharma. "An efficient automated multi-modal cyberbullying detection using decision fusion classifier on social media platforms." *Multimedia Tools and Applications* (2023): 1-29.

12. Alduailaj, Alanoud Mohammed, and Aymen Belghith. "Detecting arabic cyberbullying tweets using machine learning." *Machine Learning and Knowledge Extraction* 5.1 (2023): 29-42.

13. Obaid, Mohammed Hussein, Shawkat K. Guirguis, and Saleh M. Elkaffas. "Cyberbullying detection and severity determination model." *IEEE Access* (2023).

14. Alzaqebah, Malek, et al. "Cyberbullying detection framework for short and imbalanced Arabic datasets." *Journal of King Saud University-Computer and Information Sciences* 35.8 (2023): 101652.

15. Mahajan, Esshaan, Hemaank Mahajan, and Sanjay Kumar. "EnsMulHateCyb: Multilingual hate speech and cyberbully detection in online social media." *Expert Systems with Applications* 236 (2024): 121228.

16. Teng, Teoh Hwai, and Kasturi Dewi Varathan. "Cyberbullying Detection in Social Networks: A Comparison between Machine Learning and Transfer Learning Approaches." *IEEE Access* (2023).

17. Muneer, Amgad, et al. "Cyberbullying Detection on Social Media Using Stacking Ensemble Learning and Enhanced BERT." *Information* 14.8 (2023): 467.

18. Wu, Fan, et al. "FACapsnet: A fusion capsule network with congruent attention for cyberbullying detection." *Neurocomputing* 542 (2023): 126253.

19. Syahri Nasution, Muhammad Alfi, and Erwin Budi Setiawan. "Enhancing Cyberbullying Detection on Indonesian Twitter: Leveraging FastText for Feature Expansion and Hybrid Approach Applying CNN and BiLSTM." *Revue d'IntelligenceArtificielle* 37.4 (2023).

20. Nagaraj, P., et al. "Cyberbully Detection On Social Media." *Journal of Survey in Fisheries Sciences* (2023): 2311-2316.

21. Shetty, Nisha P., et al. "FedBully: A Cross-Device Federated Approach for Privacy Enabled Cyber Bullying Detection using Sentence Encoders." *Journal of Cyber Security and Mobility* (2023): 465-496.

22. Zhang, Jianwei, Lin Li, and Shinsuke Nakajima. "Constructing Japanese Bullying Expression Dictionary for Automated Cyberbullying Detection on Twitter." *Vietnam Journal of Computer Science* 10.02 (2023): 135-158.

23. Jing, Yang, et al. "Soft Computing Techniques for Detecting Cyberbullying in Social Multimedia Data." *ACM Journal of Data and Information Quality* (2023).

24. Dhingra, Nitasha, et al. "An Improved Detection of Cyberbullying on Social Media Using Randomized Sampling." *International Journal of Bullying Prevention* (2023): 1-13.

25. Samalo, Darwin, Rizky Martin, and DitditNugeraha Utama. "Improved Model for Identifying the Cyberbullying based on Tweets of Twitter." *Informatica* 47.6 (2023).