Investigation of deep neural network architecture for cyberbullying detection over social media

Investigación de la arquitectura de redes neuronales profundas para la detección de ciberacoso en redes sociales

Investigação da arquitetura de redes neurais profundas para detecção de cyberbullying em mídias sociais

Subbaraju Pericharla[1]
Sivadi Balakrishna[2]

Received: April 15th, 2025
Accepted: July 19th, 2025
Available: September 5th, 2025

How to cite this article:
S. Pericharla and S. Balakrishna, "Investigation of Deep Neural Network Architecture for Cyberbullying Detection over Social Media," *Revista Ingeniería Solidaria,* vol. 21, no. 3, 2025.
doi: https://doi.org/10.16925/2357-6014.2025.03.01

Research article. https://doi.org/10.16925/2357-6014.2025.03.01

[1] Department of Information Technology, SRKR Engineering College, Bhimavaram, A.P., India.

Email: raju.pericherla74@gmail.com

ORCID: https://orcid.org/0000-0002-0701-6377

[2] Department of Advanced Computer Science and Engineering, Vignan's Foundation for Science, Technology & Research (Deemed to be University), Vadlamudi, Guntur, A.P., India.

Email: drsivadibalakrishna@gmail.com

ORCID: https://orcid.org/0000-0002-8939-9307

Abstract

Introduction: Nowadays, there has been a significant increase in cases of cyberbullying on digital devices and platforms such as Facebook, Instagram, Snapchat, and TikTok.

Problem: Many state-of-the-art approaches have been introduced for the detection of cyberbullying activities. However, the affordability of high-quality data resources, along with restrictions on their access, limits the applicability of these state-of-the-art approaches.

Objective: The detection of cyberbullying activities is of societal importance and has gained increasing prominence in research.

Methodology: In this paper, we explored convolutional neural networks for cyberbullying detection (CNN-CBD) architecture for the classification task and reported their performance on real-world databases such as Twitter, Wikipedia, and Formspring. We also compared the CNN-CBD performance with baseline machine learning (ML) models. Various issues regarding the handling of real-world databases and the selection of the most suitable deep neural network (DNN) model are reported and discussed in detail.

Results: Experiments showed that the proposed CNN-CBD model outperformed traditional ML algorithms in cyberbullying detection, achieving an accuracy of 97%.

Conclusions: We concluded that the proposed CNN-CBD model outperformed the existing baseline models.

Keywords: Deep learning, cyberbullying, social media, CNN-CBD, neural networks.

Resumen

Introducción: Actualmente, se ha observado un aumento significativo de casos de ciberacoso en dispositivos y plataformas digitales como Facebook, Instagram, Snapchat y TikTok.

Problema: Se han introducido numerosos enfoques de vanguardia para la detección de actividades de ciberacoso. Sin embargo, la asequibilidad de recursos de datos de alta calidad, junto con las restricciones de acceso, limita la aplicabilidad de estos enfoques.

Objetivo: La detección de actividades de ciberacoso reviste importancia social y ha adquirido una relevancia creciente en la investigación.

Metodología: En este artículo, exploramos la arquitectura de redes neuronales convolucionales para la detección de ciberacoso (CNN-CBD) para la tarea de clasificación e informamos sobre su rendimiento en bases de datos reales como Twitter, Wikipedia y Formspring. También comparamos el rendimiento de CNN-CBD con modelos de aprendizaje automático (ML) de referencia. Se informan y discuten en detalle diversos problemas relacionados con el manejo de bases de datos reales y la selección del modelo de red neuronal profunda (DNN) más adecuado.

Resultados: Los experimentos demostraron que el modelo CNN-CBD propuesto superó a los algoritmos de aprendizaje automático tradicionales en la detección del ciberacoso, alcanzando una precisión del 97 %.

Conclusiones: Concluimos que el modelo CNN-CBD propuesto superó a los modelos de referencia existentes.

Palabras clave: Aprendizaje profundo, ciberacoso, redes sociales, CNN-CBD, redes neuronales.

Resumo

Introdução: Atualmente, observa-se um aumento significativo nos casos de cyberbullying em dispositivos e plataformas digitais como Facebook, Instagram, Snapchat e TikTok.

Problema: Muitas abordagens de ponta foram introduzidas para a detecção de atividades de cyberbullying. No entanto, a acessibilidade a recursos de dados de alta qualidade, juntamente com as restrições de acesso, limita a aplicabilidade dessas abordagens de ponta.

Objetivo: A detecção de atividades de cyberbullying é de importância social e tem ganhado crescente destaque na pesquisa.

Metodologia: Neste artigo, exploramos a arquitetura de redes neurais convolucionais para detecção de cyberbullying (CNN-CBD) para a tarefa de classificação e relatamos seu desempenho em bancos de dados do mundo real, como Twitter, Wikipedia e Formspring. Também comparamos o desempenho da CNN-CBD com modelos de aprendizado de máquina (ML) de referência. Diversas questões relacionadas ao processamento de bancos de dados do mundo real e à seleção do modelo de rede neural profunda (DNN) mais adequado são relatadas e discutidas em detalhes.

Resultados: Os experimentos mostraram que o modelo CNN-CBD proposto superou os algoritmos tradicionais de aprendizado de máquina na detecção de cyberbullying, atingindo uma precisão de 97%.

Concluímos que o modelo CNN-CBD proposto superou os modelos de referência existentes.

Palavras-chave: Aprendizado profundo, cyberbullying, mídias sociais, CNN-CBD, redes neurais.

I. INTRODUCTION

With the increase in user-generated content on the internet, numerous ethical issues have emerged that must be addressed efficiently. Cyberbullying is one of the most widely acknowledged problems among individuals and communities. It is defined as violent and intentional actions conducted by an individual or group targeting another individual or particular groups through online platforms. Most of these actions are directed against victims who lack the ability to respond [1]. Such bullying activities are always considered a critical issue that requires serious attention. The extensive use of the internet on various social media platforms has made the issue of cyberbullying more pressing for human society. Hence, it is of utmost importance to detect cyberbullying activities and act accordingly. Research in this field is maturing and evolving every day, with existing studies spanning linguistics, psychology, and computer science.

Many psychologists have recognized that cyberbullying has negative psychological effects on victims, particularly when repeated over time [2]. They examined this phenomenon in a sample of 7,000 students and concluded that cyberbullying correlates with higher levels of loneliness and lower levels of social well-being for the majority of students. Consequently, psychologists emphasized the need for identifying cyberbullying activities for the betterment of society and supported automatic monitoring of such activities. Automatic monitoring of cyberbullying has become a common research interest in computer science. The main aim in this field is to develop

efficient and robust mechanisms for detecting cyberbullying incidents. Various machine learning techniques have been employed for the classification of text in messages containing bullying content.

The classification task in the literature is often treated as binary: either a bullying activity or a non-bullying activity [3]. This task is typically performed by extracting features from the textual content and feeding them into a classification algorithm. Existing studies have addressed the problem of cyberbullying detection from different perspectives. However, they generally fall into feature categories such as social-network-based, user-based, content-based, and emotion-based features.

There has been rapid growth in state-of-the-art approaches to cyberbullying detection, but certain aspects restrict the development of these systems. The first factor limiting progress in this field is data scarcity. Although there is abundant freely available content, only a very small number of standardized datasets exist. Twitter [4,7], Myspace [5], and Formspring [6] are among the few real-world datasets used for research purposes. Other factors that limit progress include evaluation criteria and reproducibility. Model selection for given input text features is another factor restricting performance. With the evolution of deep learning in machine learning algorithms, researchers have increasingly focused on building high-performance systems for classification tasks.

In this article, we present an investigative study using deep neural network architecture for cyberbullying detection. We also compare deep learning models with conventional machine learning algorithms. The comparison approach is shown in Figure 1.

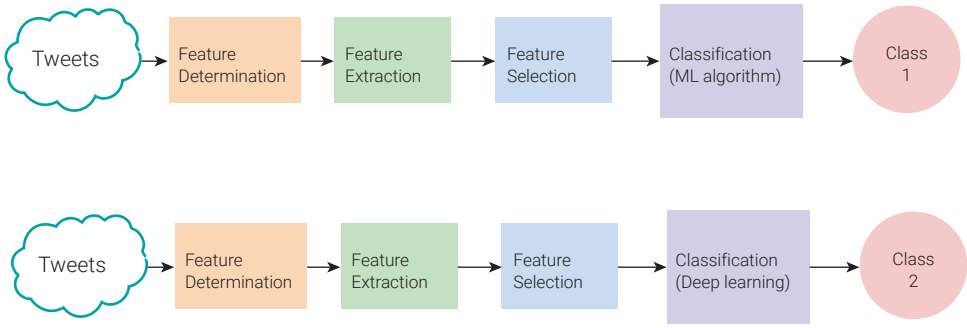

Figure 1. Classical Machine learning vs Deep learning algorithm process comparison.

The remainder of this paper is organized as follows. In Section 2, we discuss the related works and databases used in cyberbullying. In Section 3, we present the deep

learning architectures and traditional machine learning algorithms used for cyberbullying detection. The proposed CNN-CBD architectural model is described in Section 4. In Section 5, we provide the experiments and their results. Finally, a discussion of the results and the conclusions is given in Section 6.

2. RELATED WORK

In the literature, there has been significant focus on cyberbullying over the last decade. This field has generally been divided into three categories. Firstly, the task was considered a binary classification [8], where the relevant message was categorized as either a bullying activity or a non-bullying activity. This classification was considered at a broad level, limiting in-depth research. Later, finer-grained approaches were introduced to determine the roles of actors within bullying scenarios [9]. The main focus of both these finer- and broad-level approaches has been text-based features. Lastly, metadata-based approaches have been proposed for detection, which also consider the profile, image information, and network information of the user alongside the message content [10].

Many of the recent works on cyberbullying detection do not report their performance in terms of F1 scores due to the scarcity of databases. The classifiers used in the literature are discussed below. Research in this field of feature selection is divided into three categories: content-based, network-based, and user-based features. A detailed research summary of cyberbullying using these content, user-based, and network features is shown in Table 1.

Table 1. Summary of Cyberbullying Detection Approaches.

Paper	Data Mining Task	Algorithm Used	Features
[11]	Classification	multi-class multi-level (MCML)	Text profile user graph
[12]	Classification	ID-XCB	Swear words
[13]	Classification	BullyGen	Word Embeddings
[14]	Classification	Transfer Learning	Image features
[15]	Classification	CNN-SVM	Semantic features
[16]	Classification	SVM,DT,RF,LR	BoW,TF-IDF,Word2Vec
[17]	Classification	Fuzzy logic	Text word statistics
[18]	Classification Clustering	SVM J48	Text TFIDF
[19]	Classification	CNN-CB	Text Audio

Reference: own work

All the major studies in the literature have reported their classification techniques mostly on basic ML classifiers. There has been very little focus on deep learning architectures for classification, leaving a considerable research gap in this area. Hence, we focused our research on deep learning classifiers for the detection of cyberbullying activities. We also provide a comparison of DNNs with baseline ML classifiers.

3. BASIC MACHINE LEARNING ALGORITHMS EXPLORED FOR CYBERBULLYING DETECTION

Many studies in the literature have used basic ML algorithms for the detection of bullying activities. In this research study, we explored supervised and ensemble learning-based ML algorithms for bullying detection. Since the majority of practical machine learning studies use supervised learning, we opted for this approach. The rationale behind using supervised rather than unsupervised learning is that it relies on labeled data, which is associated with an output based on the algorithm applied. Supervised algorithms are generally simple, efficient, and highly accurate for handling input-labeled data. In contrast, unsupervised algorithms deal with unlabeled data, which makes their implementation more complex.

Supervised learning makes use of labeled training data and learns the mapping function from input variable (X) into output variable (Y), given by:

$$Y = f(X) \tag{1}$$

classification, regression, and ensemble are the different types of learning used in supervised algorithms

Classification is used to predict the outcome of a given sample when the output is in the form of categories, whereas regression is used to predict the outcome of a given sample when the output is in the form of real values. Ensembling refers to combining the predictions of ML models that are individually weak in order to produce a more accurate prediction on a new sample. The algorithms used for bullying detection are:

Logistic Regression [20]

It is a supervised machine learning algorithm primarily used for classification problems. It is often considered a predictive analysis algorithm based on the concept of

probability. Logistic Regression uses a complex cost function, namely the "sigmoid function," which is also known as the "logistic function." The hypothesis of logistic regression lies between 0 and 1, as shown in Figure 2. The range for the hypothesis is given as:

$$0 \leq \ h_\theta(x) \ \leq \ 1 \tag{2}$$

where $h_\theta(x)$ is logistic regression hypothesis expectation.

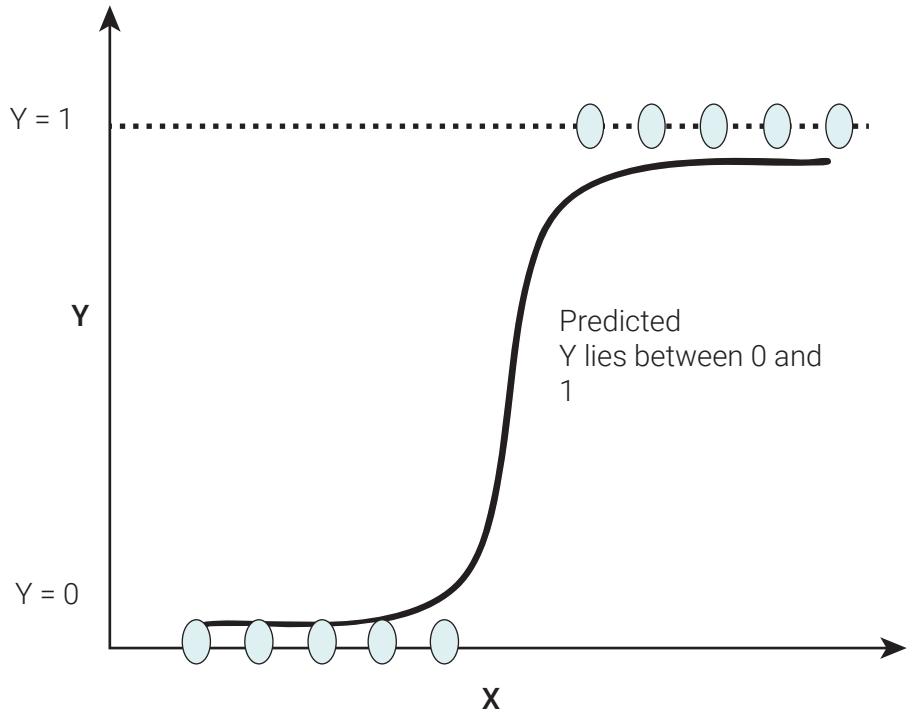

Figure 2. Logistic regression
Reference: own work

Linear Support vector machine (SVM) [21]:

SVM is a linear model used for classification and regression problems that formally separates the data into classes using a hyper-plane. This hyper-plane differentiates amongst tons of data, by creating a margin of separation. The linear classifier has the form,

$$f(x) = W^T \ X + b \tag{3}$$

e-ISSN 2357-6014 / Vol. 21, no. 3 / september-december 2025 / Bogotá D.C., Colombia
Universidad Cooperativa de Colombia

W is the weight vector and b represents the bias value. SVM is efficient for data-set with clear margin of separation. It is memory efficient and works well when dimensions are greater than the samples. SVMs for linearly separable data are shown in **Figure 3**.

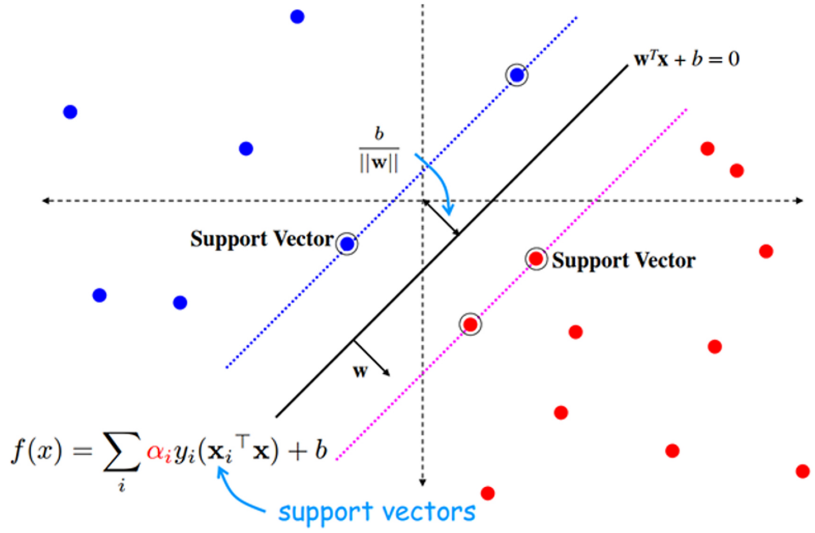

Figure 3. Linear SVM
Reference: own work

Decision Trees [22]

A Decision Tree is a supervised machine learning technique where the data is continuously split according to a certain parameter. Decision trees are used for both classification and regression trees. The decision variable can be either continuous or discrete as shown in Figure 4.

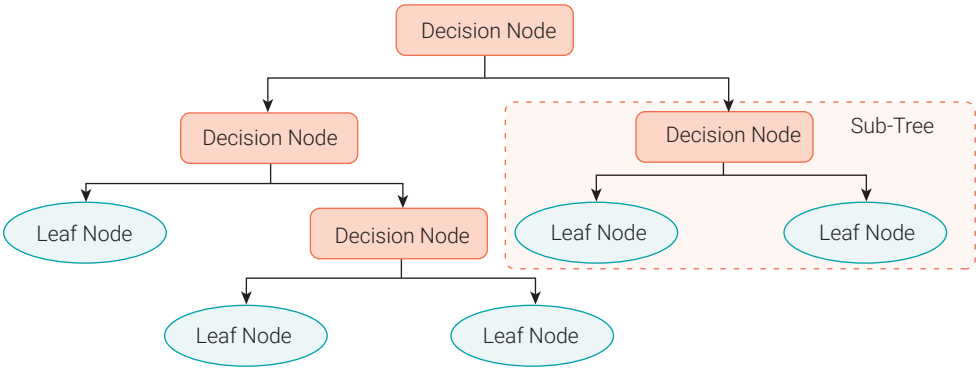

Figure 4. Decision Tree
Reference. own work

Random forest classifier [23]

A Random Forest Classifier is an ensemble tree-based learning algorithm with a set of decision trees from a randomly selected subset of the training set. The votes are aggregated from different decision trees to decide the final class of the test object shown in Figure 5. It is efficient on large databases and can handle thousands of input variables without their deletion.

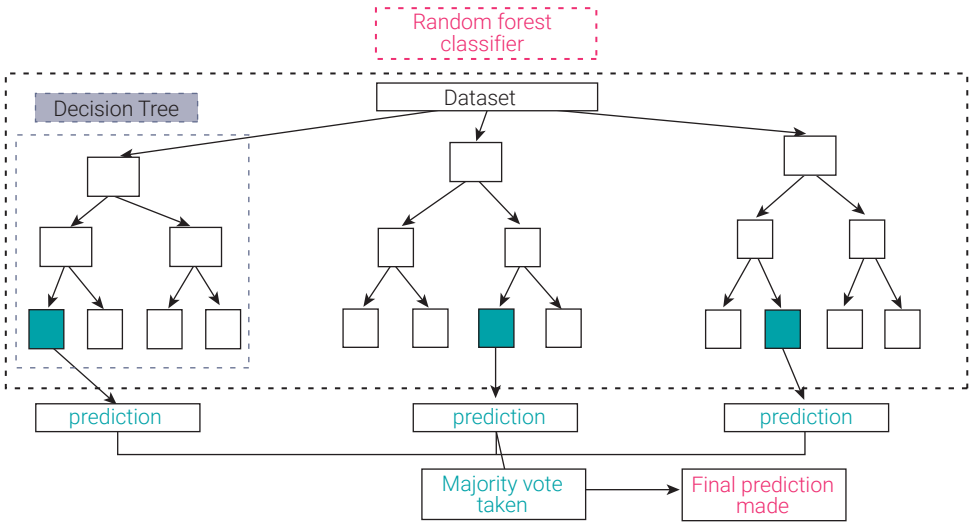

Figure 5. Random Forest classifier
Reference: own work

K-nearest neighbor (KNN) classifier [24]

KNN estimates the things that are similar based on the proximity of nature. The similar things are picked closer to each other, and the other things that are not similar deviate away. The main advantage of KNN is this learning is purely based on instances and is very easy to implement using the Euclidean distance.

Multinomial Naive Bayes (NB) classifier [25]

NB classifier is normally used for text classification. For a given document 'd' and class 'c' Bayes' rule is given by:

$$P(c|d) = (P(d|c)\,P(c)) / P(d) \tag{4}$$

NB classifier gives the maximum a posteriori (MAP) of the most likely class is given as:

e-ISSN 2357-6014 / Vol. 21, no. 3 / september-december 2025 / Bogotá D.C., Colombia
Universidad Cooperativa de Colombia

$$C_{MAP} = \arg_{c \in C} \max P(d|c)\, P(c) \quad (5)$$

when the document 'd' is represented as features ($x_1, x_2, \ldots\ldots x_n$),

$$C_{MAP} = \arg_{c \in C} \max P(x_1, x_2, \ldots\ldots x_n|c)\, P(c) \quad (6)$$

Multinomial NB (MNB) assumes the position of the bag of words does not matter and the feature probabilities $P(X_i|C_j)$ are independent for the given class C,

$$P(x_1, x_2, \ldots\ldots x_n|c) = P(x_1|c).P(x_2|c).P(x_3|c). \ldots\ldots P(x_n|c) \quad (7)$$

$$C_{MNB} = \arg_{c \in C} \max P(c_j) \prod_{x \in X} P(x|c) \quad (8)$$

MNB is a specialized and modified version of Naive Bayes which is designed to handle more text documents.

Whereas a simple NB would implicitly model a document based on the presence and absence of particular words, MNB explicitly models the word counts and adjusts the underlying calculations to account for them.

Bagging Classifier [26]

It is an ensemble meta-estimator that fits base classifiers on random subsets of the original dataset. This classifier aggregates their predictions, either by voting or averaging all the individual predictions, to form a final resultant prediction. This meta-estimator reduces the variance of a black-box decision tree estimator by randomizing the construction procedure and creating an ensemble from it. Each base classifier is trained in parallel using training subsets that are independent of each other. The problem of overfitting in bagging is reduced by averaging or voting, which, in turn, increases the bias. However, the reduction in variance compensates for the increased bias. It is similar to random forest classifiers, but it considers a bag of words instead of individual words.

4. PROPOSED CNN-CBD ARCHITECTURE

CNN-CBD is a deep learning architecture that has recently been applied in place of classical ML algorithms. There are only a limited number of studies on CNN

architectures for bullying detection. One such study proposed a CNN-CB-based architecture for cyberbullying detection [27,32]. In our study, we used the same architecture but increased the number of filters and kernels to observe improvements in detection performance.

The CNN-CBD model comprises four layers, each of which is described in the following subsections. The classification phases of the detection algorithm, including feature extraction, selection, and determination, are eliminated in this CNN-CBD architecture. Instead, word embeddings or numerical vectors are directly fed to the CNNs as input. The detailed architecture of CNN-CBD is shown in Figure 6. The algorithm for the proposed CNN-CBD architecture is explained in detail in Table 2.

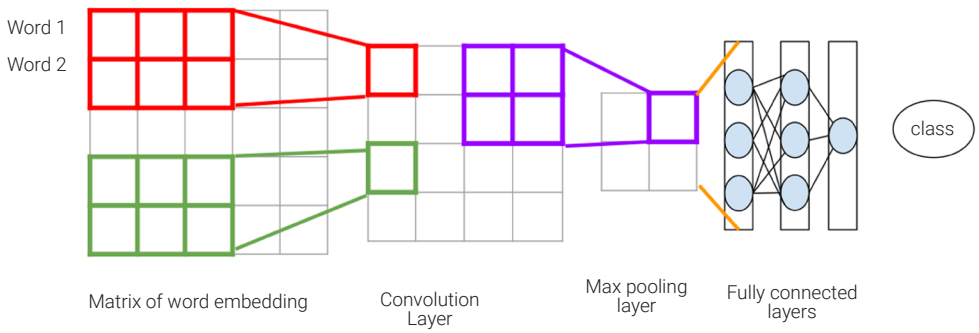

Figure 6. CNN-CBD architecture
Reference: own work

Word Embeddings

Word embeddings transform every word into a set of numbers represented as an N-dimensional vector. Each word is assigned a unique vector in the embedding space, and similar words end up having values closer to each other. Vectors with high similarity are located at shorter distances from each other when represented in vector space.

In the CNN-CBD architecture, the specific choice of embedding depends on the task specified. All the cleaned tweets are considered as text input, and the vocabulary vector space is generated. The use of word embeddings in CNN-CBD makes it more advanced compared to traditional detection approaches, as they incorporate semantics along with the features extracted from raw text [31]. Input in the form of word embeddings is provided in Keras format [28], which requires three parameters to be set before constructing the vector space: input dimension, input length, and output dimension.

Input dimension: Refers to the total number of words in the vocabulary of the entire corpus. Let the number of Tweets be T in the corpus which is derived as follows:

$$T=\{t_1,t_2,t_3.....t_n\}, \text{ where n is number of tweets} \quad (9)$$

Input dimension= length(tokens(T))

output dimension= The size of the output vector is known as it is supervised type.

Input length= The length of each vector i.e. the maximum number of words per tweet is not fixed it might vary over time. The length of the input is given by Input length = max (length for t in T).

Convolution layer:

It is the next layer after word embedding's. It convolves around the input vector to detect features. During this task, it compresses the original input vector alongside preserving the valuable features. The preservation of valuable features is done by creating a filter or weighted matrices. Each weighted matrix is convolved with input vectors resulting in feature maps through element-wise multiplication. For the input vector of words V and the filter of size h* w, the element-wise multiplication is given by

$$(V * F)_{xy}= \sum^{h}_{i=1} \sum^{w}_{j=1} V_{ij} . F_{x+i-1,y+j-1} \quad (10)$$

Max Pooling Layer

The major advantage of using CNNs is that they compress the input into smaller matrices. This can be observed in both the convolution and max pooling layers. The max pooling matrix in this layer slides across the output from the CNN layer and identifies the maximum value within the selected region. Hence, only the most meaningful information is captured at this stage.

Fully Connected or Dense Layer

The role of the first three layers is to compress the input features, but the actual task of classification occurs at this stage. The number of dense layers depends on the number of classes.

Table 2. CNN-CBD algorithm

Input: List out the number of tweets T={t_1,t_2,t_3,….t_n}
Output: Number C=1 or 0 (0=no bullying, 1= bullying)

1. Initialize the number of filters f
2. Initialize the size of kernels K
3. Initialize the pool size p
4. Initialize the count for the number of classes=2
5. Initialize the count for the number of neurons in dense layers

Begin

1. Tokenize all the tweets
2. Calculate the total vocabulary from the tokens vocab= length(tokens)
3. Calculate maximum length of tweet, length =max(length for t in T)
4. Split the tweets into training and test feeds
5. Encode training and testing tweets
6. Creation of embedding layer, embedding (vocab, length)
7. Creation of convolution layer, convolution(f,k)
8. Creation of max pooling layer, pooling (p)
9. Creation of dense layer, dense (n), C=dense(count)
10. End

Reference: own work

5. PERFORMANCE EVALUATION

The evaluation of the proposed CNN-CBD algorithm aims to experimentally investigate deep learning architecture for cyberbullying activity detection. Firstly, CNN-CBD achieves better results than traditional ML algorithms for detection. The evaluation metrics considered include loss, recall, and accuracy, implemented in comparison with basic ML algorithms. All experiments were run on an Ubuntu PC with 12 GB of RAM. All algorithms were programmed in a Python environment [29], where CNN-CBD was implemented using TensorFlow [30], and the models were built in Google Colab.

5.1 Details of Datasets

Databases used for cyberbullying research are diverse, and some of the most widely used are listed below:

- **Formspring database [6]:** This is a large unlabeled dataset crawled from 50 IDs of Formspring.me during the summer of 2010. It contains approximately 20K posts and is well known across the research community. In this dataset, more than 84% of the samples are labeled. For our experiments, we considered this dataset.
- **Twitter [4,7]:** This database was collected from tweets gathered across two different sets. The annotations were performed manually by three individuals, with the final label determined by majority vote. In the first phase of collection, 220 positive and 5,162 negative examples were considered. The stream of messages largely contained words related to school, class, college, and campus. In the second set, the focus was primarily on traces of bullying activities. The combined dataset of approximately 30K posts was used for our research.
- **Myspace [5]:** This dataset consists of approximately 30K posts. It was originally used for information retrieval tasks, where posts were grouped into batches of 10 and assigned a single label for the entire batch, merged as one instance. In this batching process, the average number of tokens per instance is significantly higher than in other corpora. This dataset was also used for experimentation purposes.

The dataset used in our experiments was compiled from a combination of Twitter, Wikipedia, and Formspring databases. A total of 45,000 samples were considered for the experiments. A summary of the data collected for training and testing is presented in Table 3.

Table 3. Combined Dataset distribution

Class	Training samples	Testing samples
Bullying	10,000	3,000
No-bullying	25,000	7,000

Reference: own work

5.2 Performance Metrics

The cyberbullying detection for our study is normally a two-class problem whether it is a bullying activity or not. Hence the classification accuracy would be the obvious choice of metric for the classification task. This is an imbalanced class problem as

the count of bullying and non-bullying activity data is different. There is a need to consider other metrics of recall, F1 score, and precision along with accuracy metrics. The formula for the calculation of these metrics is given by:

$$\text{accuracy} = (\text{TP} + \text{TN})/(\text{TP}+\text{TN}+\text{FP}+\text{FN}) \tag{11}$$

$$\text{recall} = \text{TP}/(\text{TP} + \text{FN}) \tag{12}$$

$$\text{precision} = \text{TP}/(\text{TP} + \text{FP}) \tag{13}$$

F1 score is the harmonic mean of precision and recall.

$$\text{F1} = \text{TP}/(\text{TP} + 1/2(\text{FP}+\text{FN})) \tag{14}$$

where TP, FN, FP, and TN represent the number of true positives, false negatives, false positives, and true negatives respectively.

5.3 Results and Discussion

In this section, a comprehensive comparison between cyberbullying detection using traditional ML and CNN-CBD algorithms is presented. The aim of these experiments is to demonstrate that the proposed CNN-CBD algorithm outperforms traditional methods by providing better predictions across all metrics. The series of experiments begins by reporting the performance of baseline ML algorithms for both training and test cases across all metrics. In the subsequent experiments, CNN-CBD is tested with different values of filters, kernels, pooling, and neurons to show how these variations affect prediction quality.

In the first experiment, we report the performance metrics of baseline ML algorithms. Accuracy, precision, recall, F1 score, and training time are reported for the training samples in Table 4. The algorithms tested include bagging, logistic regression, decision tree, linear SVM, random forest, AdaBoost, multinomial NB, and k-nearest neighbors classifiers. For all experiments, 70% of the data was randomly selected for training the algorithms, while the remaining 30% was reserved for testing.

The best training accuracy was observed for the decision tree, linear SVM, and random forest classifiers, respectively. For precision, recall, and F1 metrics, a similar

trend was observed as in the accuracy results. These results demonstrate that the ML models were trained accurately. Among all ML algorithms, the bagging classifier required the longest training time, while the k-nearest neighbors classifier required the least.

Table 4. Performance metrics of baseline ML models for training cases

S.No	Algorithm	Accuracy: Train (%)	Precision: Train (%)	Recall: Train (%)	F1 score: Train (%)	Training Time (hrs.)
1	Bagging classifier	98.8376	99.6507	98.5780	99.1114	41.754
2	SGD classifier	98.2526	99.2377	98.0964	98.6638	0.0686
3	Logistic Regression	97.8763	99.0242	97.7338	98.3748	0.4678
4	Decision tree classifier	99.8845	99.9943	99.8300	99.9121	5.974688
5	Linear SVM classifier	99.7019	99.8298	99.7167	99.7733	0.681417
6	Random forest classifier	99.1990	99.7773	99.0029	99.3886	6.237272
7	AdaBoost classifier	90.9650	97.1744	88.8448	92.8231	1.964057
8	Multinomial NB	94.4560	95.6763	95.9039	95.7900	0.025667
9	K neighbors classifier	89.7727	92.7596	91.5982	92.1753	0.002175

Reference: own work

In the next series of experiments, we report the performance of all metrics for the test cases. In this experiment, we evaluate the models on the test data, as shown in Table 5. The highest detection accuracy was observed for logistic regression with 92%, while the lowest accuracy was observed for the k-nearest neighbors classifier. Prediction time was also the lowest for logistic regression and the highest for the k-nearest neighbors classifier.

Table 5. Performance metrics of baseline ML models for testing cases

S.no	Algorithm	Accuracy: Test (%)	Precision: Test (%)	Recall: Test (%)	F1 score: Test (%)	Prediction Time (hrs.)
1	Bagging classifier	92.7797	96.5493	92.3129	94.3836	0.419815
2	SGD classifier	92.7462	96.1044	92.7211	94.3824	0.001418
3	Logistic Regression	92.6344	96.4089	92.2279	94.2721	0.002266
4	Decision tree classifier	92.3438	95.2132	93.0272	94.1075	0.027815
5	Linear SVM classifier	91.6732	94.6599	92.5510	93.5936	0.001932

(continúa)

(viene)

S.no	Algorithm	Accuracy: Test (%)	Precision: Test (%)	Recall: Test (%)	F1 score: Test (%)	Prediction Time (hrs.)
6	Random forest classifier	91.0137	95.1120	91.0034	93.0123	0.347698
7	AdaBoost classifier	90.7567	97.2508	88.4354	92.6338	0.341812
8	Multinomial NB	89.3372	90.1663	94.0306	92.0579	0.004461
9	K neighbors classifier	85.7606	89.5161	88.7245	89.1186	32.5875

Reference: own work

All the ML algorithms performed well in the test cases, with an average 5% degradation in cyberbullying detection between the training and test results. In the next experiment, we report the performance of the proposed CNN-CBD algorithm across all metrics, as shown in Table 6. The performance of the CNN-CBD algorithm with varying filters, kernels, pooling layers, and neurons is presented in detail. From Table 6, it is evident that the proposed CNN-CBD architecture outperforms the traditional ML architectures reported in Table 5.

The best performance of the CNN-CBD architecture was observed when the number of kernels was 12, with 32 neurons and 4 pooling layers. We then compared the performance of the proposed algorithm with that of the traditional ML algorithms, as described next. The performance of CNN-CBD during each epoch consistently improved due to incremental learning at every stage. The model started with an accuracy of 68% but increased to 91% after 12 epochs.

Table 6. CNN-CBD performance metrics for varying filters, kernels, size of pooling, and number of neurons.

No.of filters	No.of kernels	Size of Pooling	No. of neurons	Accuracy (%)	Precision (%)	Recall (%)
11.6	12.8	13.1	14.6	15.68	16.99	17.73
18.6	19.8	20.1	21.24	22.96	23.79	24.88
25.6	26.12	27.4	28.32	29.97	30.95	31.75
32.6	33.12	34.4	35.47	36.97	37.89	38.79
39.6	40.12	41.4	42.58	43.96	44.82	45.83
46.12	47.25	48.4	49.6	50.67	51.99	52.69
53.12	54.25	55.6	56.24	57.95	58.84	59.76
60.12	61.25	62.6	63.32	64.95	65.89	66.76

(continúa)

(viene)

No.of filters	No.of kernels	Size of Pooling	No. of neurons	Accuracy (%)	Precision (%)	Recall (%)
67.12	68.30	69.6	70.47	71.97	72.80	73.77
74.12	75.30	76.6	77.58	78.95	79.82	80.83

Reference: own work

When CNN-CBD was compared to traditional ML approaches, it achieved better results in the three metrics of accuracy, precision, and recall, as shown in Figures 7, 8, and 9, respectively. There was a noticeable improvement of around 5% in accuracy. Among the three metrics studied, accuracy showed the most significant difference.

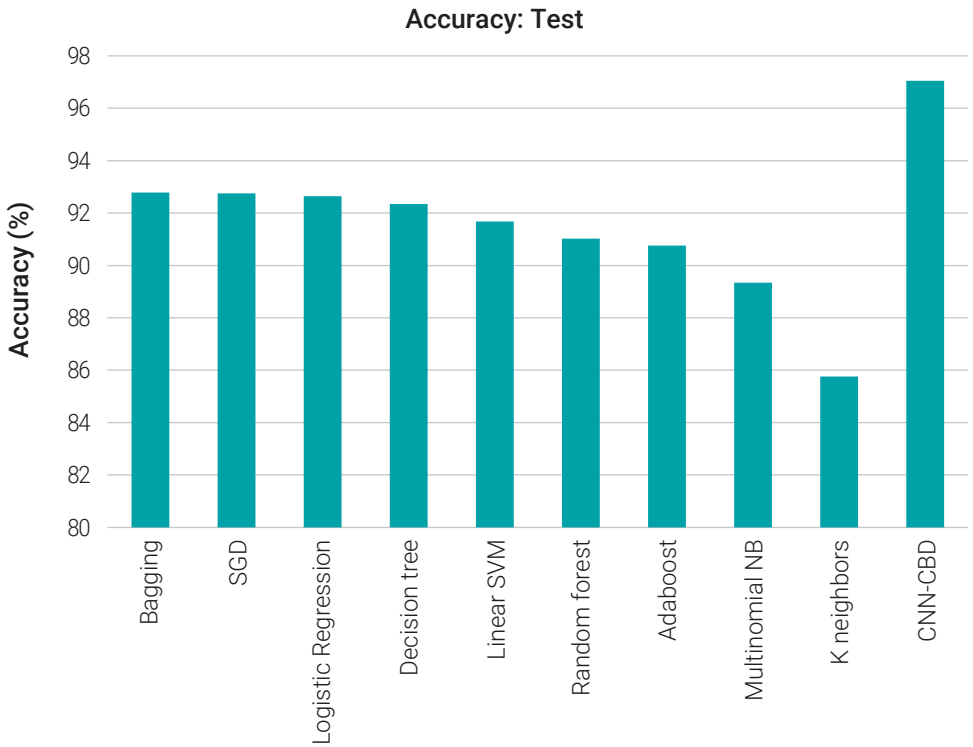

Figure 7. CNN-CBD vs ML (accuracy)
Reference: own work

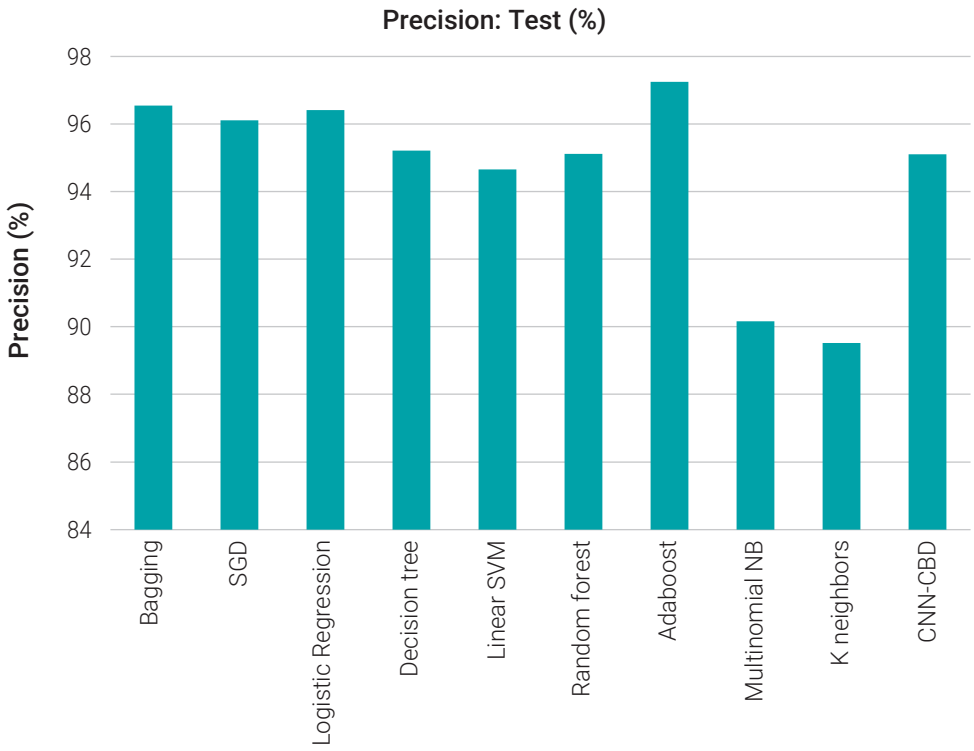

Figure 8. CNN-CBD vs ML (precision)
Reference: own work

The best accuracy is observed for the proposed CNN-CBD algorithm shown in Figure 7. The lowest is observed for the K neighbor's classifier. Coming to the precision metrics the proposed CNN-CBD has shown a lesser precision of 2% compared to other ML algorithms. The highest precision is observed for AdaBoost shown in Figure 8 and the lowest for multinomial NB and classifier. In Figure 9**,** CNN-CBD has shown the lowest recall rate, and the highest was observed for multinomial NB.

e-ISSN 2357-6014 / Vol. 21, no. 3 / september-december 2025 / Bogotá D.C., Colombia
Universidad Cooperativa de Colombia

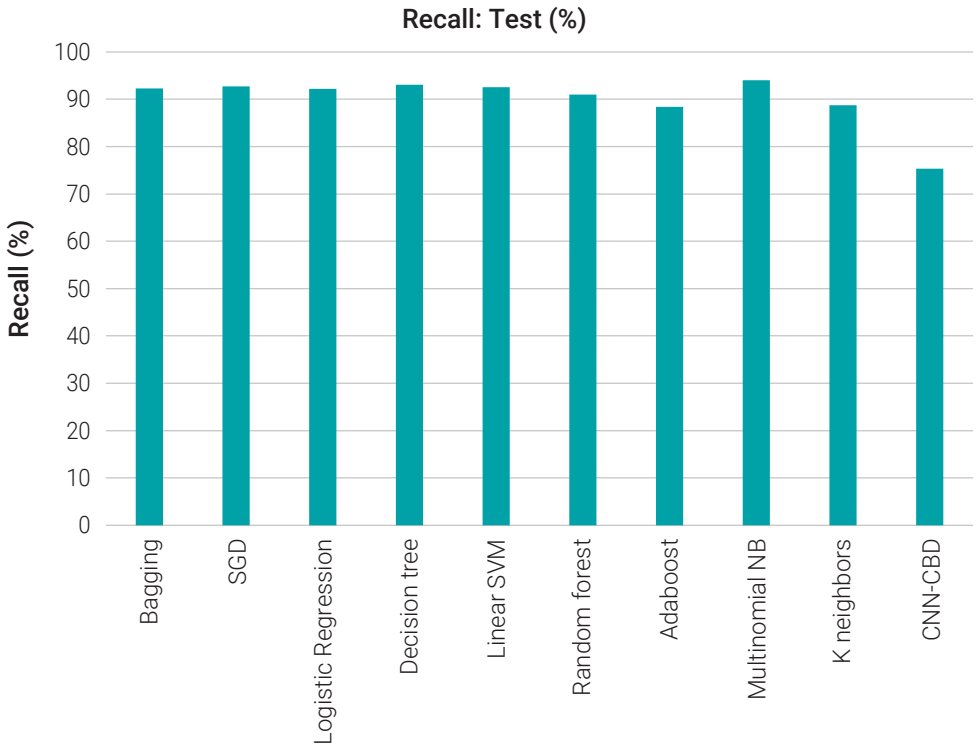

Figure 9. CNN-CBD vs ML (Recall)
Reference: own work

From Table 6, it is observed that modifying the CNN structure resulted in an improvement in the accuracy metric. The detection accuracy increased from 68% to 97% for 12 kernels. However, when the number of kernels was increased beyond 12, performance began to degrade. The reason for this degradation is that the network requires more training data as the kernel size increases.

6. CONCLUSION AND FUTURE WORK

The issue of cyberbullying detection on the combined three databases has been handled efficiently. To the best of our knowledge, there have been very few studies on CNN architectures for cyberbullying detection and on automating the CNN process. The proposed CNN-CBD provides a robust and effective solution for detecting bullying activities. Comprehensive experiments demonstrated that deep learning models outperformed traditional ML models in the cyberbullying problem. For future work, we plan to adapt transfer learning in deep learning models to further improve performance across all evaluation metrics.

References

[1] R. Shetgiri, "Bullying and victimization among children," *Advances in Pediatrics*, vol. 60, no. 1, pp. 33–51, Jul. 2013.

[2] D. Halpern, M. Piña, and J. Vásquez, "Loneliness, personal and social well-being: Towards a conceptualization of the effects of cyberbullying / Soledad, bienestar social e individual: hacia una conceptualización de los efectos del cyberbullying," *Cultura y Educación*, vol. 29, no. 4, pp. 703–727, 2017.

[3] S. Balakrishna, Y. Gopi, and V. K. Solanki, "Comparative analysis on deep neural network models for detection of cyberbullying on social media," *Ingeniería Solidaria*, vol. 18, no. 1, pp. 1–33, 2022.

[4] U. Bretschneider, T. Wöhner, and R. Peters, "Detecting online harassment in social networks," in *Proc. 35th Int. Conf. on Information Systems*, 2014, pp. 1–14.

[5] J. Bayzick, A. Kontostathis, and L. Edwards, "Detecting the presence of cyberbullying using computer software," 2011, pp. 1–2.

[6] K. Reynolds, A. Kontostathis, and L. Edwards, "Using machine learning to detect cyberbullying," in *Proc. 10th Int. Conf. on Machine Learning and Applications (ICMLA)*, vol. 2, Dec. 2011, pp. 241–244. doi: https://doi.org/10.1109/ICMLA.2011.152.

[7] J. Xu, K. Jun, X. Zhu, and A. Bellmore, "Learning from bullying traces in social media," in *Proc. Assoc. for Computational Linguistics: Human Language Technologies*, 2012, pp. 656–666.

[8] A. Kontostathis, L. Edwards, and A. Leatherman, "Text mining and cybercrime," in *Text Mining: Applications and Theory*. Hoboken, NJ: Wiley, 2010, pp. 1–14.

[9] J. Xu, K. Jun, X. Zhu, and A. Bellmore, "Learning from bullying traces in social media," in *Proc. Conf. of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies*, 2012, pp. 656–666.

[10] C. Van Hee, E. Lefever, B. Verhoeven, J. Mennes, B. Desmet, G. Pauw, and W. Daelemans, "Detection and fine-grained classification of cyber-bullying events," in *Proc. Int. Conf. Recent Advances in Natural Language Processing (RANLP)*, 2015, pp. 672–680.

[11] R. Salsabila, R. Sarno, I. Ghozali, and K. R. Sungkono, "Improving cyberbullying detection through multi-level machine learning," *Int. J. of Emerging Trends in Engineering Research*, vol. 14, no. 2, 2024.

[12] P. Yi and A. Zubiaga, "D-XCB: Data-independent debiasing for fair and accurate transformer-based cyberbullying detection," *arXiv preprint* arXiv:2402.16458, Feb. 2024.

[13] K. Maity, R. Jain, P. Jha, and S. Saha, "Explainable cyberbullying detection in Hinglish: A generative approach," *IEEE Trans. on Computational Social Systems*, 2023.

[14] A. Almomani, K. Nahar, M. Alauthman, M. A. Al-Betar, Q. Yaseen, and B. B. Gupta, "Image cyberbullying detection and recognition using transfer deep machine learning," *Int. J. of Cognitive Computing in Engineering*, vol. 5, pp. 14–26, 2023.

[15] H. Saini, H. Mehra, and R. Rani, "Enhancing cyberbullying detection: A comparative study of ensemble CNN–SVM and BERT models," *Social Network Analysis and Mining*, vol. 14, no. 1, 2023.

[16] R. Kumar and B. Bhat, "A study of machine learning-based models for detection, control, and mitigation of cyberbullying in online social media," *International Journal of Information Security*, vol. 21, pp. 1409–1431, 2022.

[17] M. Dadvar, D. Trieschnigg, R. Ordelman, and F. De Jong, "Improving cyberbullying detection with user context," 2015, pp. 2–5.

[18] H. Hosseinmardi, S. A. Mattson, R. I. Rafiq, R. Han, Q. Lv, and S. Mishra, "Detection of cyberbullying incidents on the Instagram social network," 2014.

[19] V. Nahar, S. Al-Maskari, X. Li, and C. Pang, "Semi-supervised learning for cyberbullying detection in social networks," in *Proc. Australasian Database Conf.*, 2014, pp. 160–171.

[20] J. M. Hilbe, *Logistic Regression Models*. Boca Raton, FL: CRC Press, 2009.

[21] Y. W. Chang and C. J. Lin, "Feature ranking using linear SVM," in *Causation and Prediction Challenge*, Dec. 2008, pp. 53–64.

[22] J. R. Quinlan, "Simplifying decision trees," *Int. J. of Man-Machine Studies*, vol. 27, no. 3, pp. 221–234, Sep. 1987.

[23] B. Xu, X. Guo, Y. Ye, and J. Cheng, "An improved random forest classifier for text categorization," *Journal of Computers (JCP)*, vol. 7, no. 12, pp. 2913–2920, 2012.

[24] S. Tan, "An effective refinement strategy for KNN text classifier," *Expert Systems with Applications*, vol. 30, no. 2, pp. 290–298, 2006.

[25] A. M. Kibriya, E. Frank, B. Pfahringer, and G. Holmes, "Multinomial naive Bayes for text categorization revisited," in *Proc. Australasian Joint Conf. on Artificial Intelligence*, Berlin, Germany: Springer, Dec. 2004, pp. 488–499.

[26] M. Zareapoor and P. Shamsolmoali, "Application of credit card fraud detection based on bagging ensemble classifier," *Procedia Computer Science*, vol. 48, pp. 679–685, 2015.

[27] M. A. Al-Ajlan and M. Ykhlef, "A deep learning algorithm for cyberbullying detection," *Int. J. of Advanced Computer Science and Applications*, no. 9, 2018.

[28] F. Chollet, "Keras," *GitHub*, 2015. [Online]. Available: https://github.com/fchollet/keras

[29] F. Pedregosa, G. Varoquaux, A. Gramfort, V. Michel, B. Thirion, O. Grisel, V. Blondel, P. Prettenhofer, R. Weiss, V. Dubourg, J. Vanderplas, et al., "Scikit-learn: Machine learning in Python," *J. Mach. Learn. Res.*, vol. 12, pp. 2825–2830, Nov. 2011.

[30] M. Abadi, P. Barham, J. Chen, Z. Chen, A. Davis, J. Dean, M. Devin, S. Ghemawat, G. Irving, M. Isard, M. Kudlur, et al., "TensorFlow: A system for large-scale machine learning," in *Proc. 12th USENIX Symp. on Operating Systems Design and Implementation (OSDI'16)*, 2016, pp. 265–283.

[31] Y. Bengio, R. Ducharme, P. Vincent, and C. Jauvin, "A neural probabilistic language model," *J. Mach. Learn. Res.*, vol. 3, no. Feb., pp. 1137–1155, 2003.

[32] C. Emmery, B. Verhoeven, G. De Pauw, G. Jacobs, C. Van Hee, E. Lefever, B. Desmet, V. Hoste, and W. Daelemans, "Current limitations in cyberbullying detection: On evaluation criteria, reproducibility, and data scarcity," *arXiv preprint* arXiv:1910.11922, 2019.