Detecting Cybercrime: An Evaluation of Machine Learning and Deep Learning Using Natural Language Processing Techniques on the Social Network

Abdullah Amer (Abdull-87@hotmail.com)
Aligarh Muslim University (Research Scholar
Tamanna Siddiqui
Aligarh Muslim University (Professor)
Belkacem Athamena
Professor College of Business Al Ain University

Research Article

Keywords: Natural Language Processing, Cybercrime, social networks, Machine Learning, Deep Learning.

Posted Date: October 25th, 2022

DOI: https://doi.org/10.21203/rs.3.rs-2184218/v1

Additional Declarations: No competing interests reported.

Detecting Cybercrime: An Evaluation of Machine Learning and Deep Learning Using Natural Language Processing Techniques on the Social Network

1st Abdullah Yahya Abdullah Amer

department of Computer Science

Aligarh Muslim University (Research Scholar)

Aligarh, India

ayaamer@myamu.ac.in or https://orcid.org/0000-0002-3556-7395

2nd Tamanna Siddiqui

department of Computer Science

Aligarh Muslim University (Professor)

Aligarh, India

ja_zu_siddiqui@hotmail.com ORCID https://orcid.org/0000-0002-0068-4241

3nd Prof. Belkacem Athamena

Professor College of Business Al Ain University, Al Ain, UAE

Abstract: The widespread use of online social networks has culminated in across-the-board social communication among users, resulting in a considerable amount of user-generated contact data. Cybercrime has become a significant issue in recent years with the rise of online communication and social network. Cybercrime has lately been identified as a severe national psychological concern among platform users, and building a reliable detection model is crucial. Cyberbullying is the phrase used to describe such online harassment, insults, and attacks. It has become challenging to identify such unauthorized content due to the massive number of user-generated content. Because deep neural networks have various advantages over conventional machine learning approaches, researchers are turning to them more frequently to identify cyberbullying. Deep learning and machine learning have several uses in text classification. This article suggested the novel neural network model through parameters of an algorithmic and optimization comparative analysis of nine category approaches, four neural networks, and five machine learning, in two scenarios with real-world datasets of cyberbullying.

Moreover, this work also analyzes the impact of word embedding and feature extraction techniques based on text mining and NLP on algorithms' performances. We performed extensive experiments on the two scenarios with a split dataset to demonstrate the merit of this research, comparing nine classification approaches through five feature extraction techniques. Our proposed cybercriminal detection model using neural networks, deep learning, and machine learning outperforms the existing state-of-the-art method of cybercriminal detection in terms of accuracy achieving higher performance.

Keywords: Natural Language Processing, Cybercrime, social networks, Machine Learning, Deep Learning.

1. Introduction

Information technology has revolutionized humanity, and artificial intelligence special is nowadays leading such a revolution, taking a central role able to influence the nearest future of society significantly. Consequently, researchers are interested in artificial intelligence. In recent years, social networking site services have exploded in popularity. Millions of people have utilized these platforms as new contact methods and dynamic information sources where they are able to create their profiles and interact with others irrespective of their geographic location or limitations in terms of their physical capabilities. As a result, these websites are becoming essential and omnipresent communication channels in this regard. Dataset collection of social networks can provide new insights into societies that were not previously difficult to

study on such a large size and scope. Furthermore, these digital technologies can be used to investigate human connections and behaviors outside of the real world.

Cybercriminals have turned to the social network as a new venue for perpetrating many sorts of cybercrime, including fraud [1], spamming, malware distribution [2], and cyberbullying. A growing trend in online communication through social media has led to a rise in cyberbullying [3]. Cyberbullying is the group and individual user using information and communication technologies for the purpose of harassing other users [4]. Cyberbullying is widely acknowledged as a serious issue, with victims displaying a dramatically increased risk of suicidal ideation [5].

There is a persistent form of cyberbullying with negative consequences for the victim that is more persistent than traditional bullying. A cyberbully has the ability to harass his or her victims in front of the entire internet group. Social networking sites, such as Facebook, Twitter, etc., become inextricably linked to the lives of their users. As a result, these platforms are the most common forms of cyberbullying victimization, and their admiration and widespread use have increased the number of cybercrime incidents [6]. This rise is generally attributed to traditional bullying being more challenging to perpetrate than cyberbullying, in that bullies bully their victims without being confronted directly through the associated author.

It is a well-known social networking site that enables users to send and receive messages of up to 140 characters. The Twitter network has over 500 million members, with 288 million actively communicating daily, generating a million tweets. About 80% of those activities, Twitter users use their smartphones to tweet. Twitter has become an essential tool for near-real-time communication [7], and the study found that it is becoming a "cyberbullying playground" [8].

In this study, we proposed Hybrid Models Based on Neural Networks and Machine Learning using text mining and NLP techniques. This study comprehensively studies different DL and ML classifiers. We experimented on the dataset and compared it to evaluate the performance of the nine classifiers model. We analyze the classification efficiency of NB, SVM, KNN, RF, and RL machine learning classification. We execute three kinds of features for these techniques: TF-IDF, count vectorization BoW, TF-IDF words unigram, and bigram. With deep learning, we proposed a novel model accommodation Long Short-Term Memory LSTM, Convolutional Neural Networks CNN, Bidirectional Long Short-Term Memory Bi-LSTM, and Gated Recurrent Unit GRU. Build a Shallow Neural Network from single-layer deep learning algorithms. It is to be stated that these shallow neural networks' SNN differs from "shallow learning," a subset of machine learning ML in which the demand for predetermined features is

made. As a result, feature extraction in the proposed SNN is an automated process using neural network neurons. Word2Vec and Fast Text are two embedding methods that have been tested for these SNNs. The effectiveness of models using feature extraction approaches is examined in this study. Four evaluation measures determine performance: recall, precision, F1-score, and accuracy [9][10]. Our proposed models utilized in this study outperform different existing study-related cybercriminal detections. The main contributions of this study are as follows:

1. We proposed a novel hybrid architecture for Cybercrime detection, which utilizes machine and deep learning by using many different feature extraction and word embedding for text representation.
2. We also built a Shallow Neural Network and confirmed cybercriminal classification detection's efficacy.
3. We experiment with our model in different scenarios and determine the best-suited approaches to text embedding and feature extraction for both neural networks and machine learning based on methods.

This paper rest structured as follows: existing work is reviewed and provided in section 2. The methodology is displayed in Section 3. Experimental analysis and performance metrics discusses in section 4. Results analysis discusses in section 5—finally, the conclusion.

2. Literature Survey

This section is mainly focused on a survey of Cybercrime CC classification and detection on social network data sets. Deep Learning and Machine learning-based approaches in various feature extraction techniques are extensively utilized in Cybercrime tweets classification. Balakrishnan et al. [9] introduce a model with different machine learning classifiers, such as J48, RF, and NB, which were applied to detect cyberbullying events in a dataset and classify tweets into cyberbullying classes routine aggressors, bullying, and spammer. According to findings, the emotional trait has no bearing on detecting the frequency of crime activity. Despite its qualifiers, the model is confined to tiny data set by reducing the number of labels. Nahar et al. [10] proposed the semi-learning strategy to identify CB within social networks using an augmented fuzzy SVM method and training dataset sample to detect CB. The augmented training method automatically unclassified streaming data is expanded and extracted as a training set. As an initial estimate, a small, constrained training set is used to perform the learning. The proposed solution solves stream data's dynamic and complex nature. Al Harbi B. et al. [11] Collaboratively leveraging social media data to investigate the novel topic of cyberbullying identification. This is a challenging task because of the intricacy of the numerous cross-modal linkages across various approaches, the structural correlations between

various social media sessions, and the substantial attribute data of various modalities. To address these issues, they offer XBully. This novel bullying detection system reformulates multimodal attempts to train node embedding representations on social media data as a heterogeneous network. In the last few decades, many studies on cyberbullying have concentrated on text analysis. Cyberbullying, on the other hand, is evolving into a multi-channel, multi-form, and multi-objectives phenomenon. Traditional text analytic techniques can't handle the diversity of bullying data on social media sites. Dadvar M. al.[12] Provide a strategy for detecting hate content across several social media platforms. The writers used 197,566 comments from Wikipedia, Reddit, YouTube, and Twitter. Of those, 80% were deemed to be non-hostile, while the remaining 20% were hateful. Experiments were carried out using various machine learning techniques to examine each feature individually and assess their accuracy depending on feature selection. In addition to classifiers for machine learning, Yuvaraj N.et al. [13]provided a better approach for integrating user-based, content-based, and roles typical of cyberbullying. With the combined use of all characteristics, the results indicated improved performance. Deep learning-based techniques for detecting cybercrime it has also been proposed in the literature to use tweets. Zhao et al. [14] Cyberbullying detection is suggested. Social media SDA generated representations that were both discriminative and resilient. After that, the learned numerical representation can be entered into SVM. Dewani A et al. [15] have expanded an SVM classifier, the insult to build cyberbullying characteristics based on word embedding, and the f-measure was 0.78%. A vocabulary of common phrases used by neurotics in social networks was also employed to help develop the unique features. The authors employed the text health dataset containing a semantic context using Word2Vec embedding model-based deep learning. Ward C. et al. [16] present a system for detecting cyberbullying based on various distinctive Twitter traits, such as tweet content, network, and user. They then created a machine learning model for supervised machine learning to identify cyberbullying. Their suggested framework performs pretty well for such a feature-driven model, with the F-measure of 0.936% and an average receiver-operating characteristic curve of 0.943% below the receiver-operating characteristic curve. Khairy M. et al. [17] developed a model for identifying cyberbullying based on representation learning. They have considered both BOW and semantic of latent features in addition to a predefined list of offensive terms. They used SVM as a text classifier and got an F1 score of 0.78. For the task, the authors also used data from Twitter. I wendi et al. [18]introduced a Bi-LSTM and RNN-based model to identify cyberbullying. This model demonstrated that while the RNN is capable of excellent performance, the Bi-LSTM still has very high efficiency. CNN also performs better in some

instances. Agarwal et al. [19] used RNN based on Class Weighting and under samples. The RNN model outperformed the LSTM model thanks to these modifications. This suggests that adjusting the settings can improve the performance of RNNs.

3. Methodology

The all-component cybercriminal detection model is displayed in Finger. 1. The model contains the following phases: 3.1 data set, 3.2 data cleansing and pre-process, 3.3 features selection and extraction, and 3.4 ML and DL classification. Overall these components subsection showed the following.

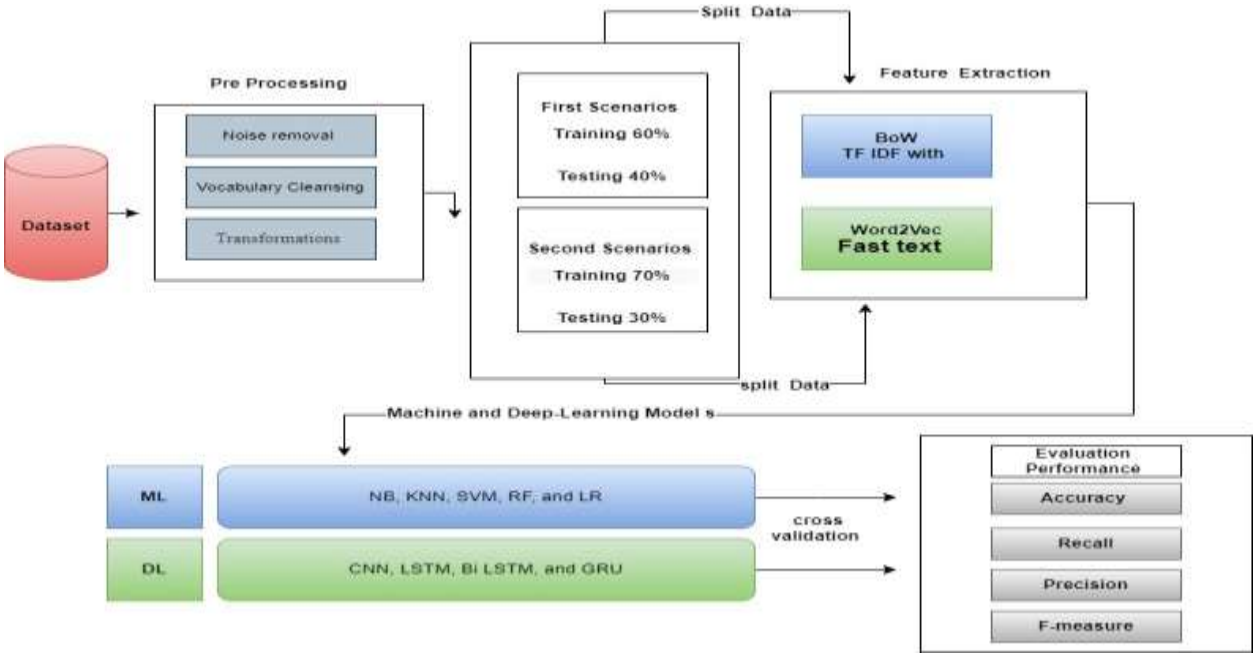

Figure 1. Methodology of the model.

3.1.Data set Collection

The frequency of life-threatening cybercrimes has been steadily increasing and has been recognized as a social threat. The suspicious tweet dataset contains 60001 tweets downloaded from Kaggle. The dataset continues with two columns, message and label. Suspicious it's a very general term the tagging of our dataset is based on three main types that are cyberbullying, terrorism, and threatening; therefore, (0) for suspicious and (1) for non-suspicious[1].

3.2. Data Preprocessing

[1] https://www.kaggle.com/datasets/syedabbasraza/suspicious-tweets

There are three sub-phases in the pre-process handles data phase. This procedure is used to the raw social Network dataset to produce the processed data specified in the previous dataset. Noises removals methods such as mentions/hashtag removal, URL removal, emoticon transformation, and symbol/ punctuation removal are conducted in the initial sub-phase [20]. Out of vocabulary, clean text, such as slang alteration, acronym expansion, spell checking, and elongation (duplicate Characters elimination), is undertaken in the second sub-phase. Data transformational such as tokenization, stop word filtering, stemming, and lower-case conversion is done in the final sub-phase. These sub-phases are used to improve the text data of the tweets and enhance our model results. Figure 2 illustrates the data cleansing and pre-processing procedures [21].

Stop words

This work used inconsequential words as languages capable of creating noise when performing text data classification as a useful attribute. These phrases are known as stop words. They can be seen being used in sentences that support the construction of the phrases and serve to connect our thoughts. For instance, articles, conjunctions, and prepositions pronouns are assumed to stop words. Our method extracted standard terms from the records, such as "a, like, are, for, an, by, at, where, how, are, how, from, how, in, this, is, in, on, or, the, these, too, was, when, where, how," etc. Following that, we prepare and save the data for the subsequent steps. [22].

Tokenization

In order to deal with a situation where a specified text will be divided into smaller pieces known as tokens, tokenization has been used in this procedure. They consist of letters, numbers, and punctuation. Furthermore, a sensitive dataset element by no purpose or value is used to replace another non-sensitive equivalent element. We made sure that the tokenization technique was safeguarded and examined in accordance with the highest industry standards for the security of sensitive data. Our method for using a tokenization framework provides authority and APIs for getting tokens for data processing applications. Whenever necessary and possible, confidential data can be detokenization [23]**.**

Lemmatization

Lemmatization's purpose is to minimize infectious form to a particular base form, just like stemming organizes. Instead of stemming, lemmatization may or may not break up infections. It only employs the lexical information's fundamentals to create the correct fundamental types of vocabulary [24].

3.3.Feature Selection and Extraction

The social network data set features are extracted by employing three natural language processing NLP, feature extraction techniques such as TF-IDF, BoW, fast text Word2Vec [25], adjectives, pronouns, and nouns, considered the major feature text content. In contrast, verbs also adverbs provide added info about content. Additionally, extracting function words, POS tags, then the content word feature could enhance a performance implementation. As commented in [26], many methods exist for selecting features. For determining cyberbullying circumstances, protruding features are specified using a process, and after that, feature subsets are fed the proposed classifier model.

Term frequency-inverse document frequency TF-IDF

Inverse Document Frequency (IDF) is a process of applying conjunction by term frequency to minimize the impact of essentially normal words on a dataset. IDF allows the high weight on items words huge or below frequency terms within a text dataset. Through this feature, extraction techniques are famously recognized as TF-IDF [27]. The mathematical symbol weight of the words within the text documents with TF-IDF as presented in Equation (1)

Then the end equation for

$$\text{IDF } W_D = \left(1 + \log(D/(|\text{d}|:\text{D}\ |\text{w} \in d\ |)\right) \quad (1)$$

Here D indicates a data set (d) shows the document's set of words w.

TF-IDF to a D dataset by d documents and w word

It is calculated as shown in Equation (2) [28]

$$\text{TF-IDF } W_{D,d} = \text{TF wd} * \text{IDF } W_D \quad (2)$$

Count Vectorization

This straightforward statistical technique produces embedded vectors of input text. To create an embedding vector for a term, we use the frequency with which it appears in a document. A matrix is formed for the complete Collection of documents, with each document's rows corresponding to its columns as words. The values of a term's frequency of occurrence in a document are contained in the cells. We train the conventional machine learning algorithms using this matrix as the feature representation [29].

Word embedding

Word embedding is a technique for representing any word as a vector. Expressed in a vector space, this can be regarded as words with the same meaning or synonyms being close to each other. The basic goal is to understand some form of relationship between words, whether it's in terms of importance, morphology, milieu, or context. Thus, by expressing words in space, the textual data's dimension can be decreased, which could otherwise be quite vast because each word i Another reason to employ word embedding is to encode words into numerical form, as computers do not grasp words as well as numerical data by default. The importance of word embedding cannot be overstated in approaches based on deep learning architecture because data must be in numerical form. Many word embeddings have been proposed to date, and we evaluated the frameworks proposed using the two most prevalent word embeddings regarded as separate dimensions [30].

Word2Vec

Word embedding architecture has been improved, and word-to-vector representation is used. The Word2Vec method employs constructing a vector using continuous bag-of-words (CBOW), the Skip-gram model, and two hidden layers of shallow neural networks with high dimensions for each word; take the next step in our research. Using Skip-grams [30] delves into the corpus of word w and their context. The purpose is to increase the likelihood as much as possible:

$$\textbf{Arg max} = \prod_{c \in T} \boldsymbol{\theta} \; [\prod_{c \in T} \boldsymbol{p}(\boldsymbol{c}|\boldsymbol{w}\,\boldsymbol{\theta})] \qquad \textbf{(3)}$$

Where T denotes Text data, and θ is a parameter of p (c | w q).

Figure .2 displays the CBOW simple model that, using the words from earlier, seeks to find the words, whereas Skip-gram looks for words that might appear near each word. V ×N [31] is represented as a matrix of w by the weights between the input and output layers:

$$\mathbf{h}=\boldsymbol{W}_{\boldsymbol{I}}^{\boldsymbol{T}}\boldsymbol{c} = \boldsymbol{W}_{\boldsymbol{k}}^{\boldsymbol{T}} \coloneqq \boldsymbol{v}_{\boldsymbol{wi}}^{\boldsymbol{T}} \qquad (4)$$

This technique is an exploration tool that is very powerful correlations in data corpora similarity between words is also important. In this case, in the vector space, several assignments are assigned to words like "large" and "bigger," embedding would regard the two words "big" and "bigger" to be close together.

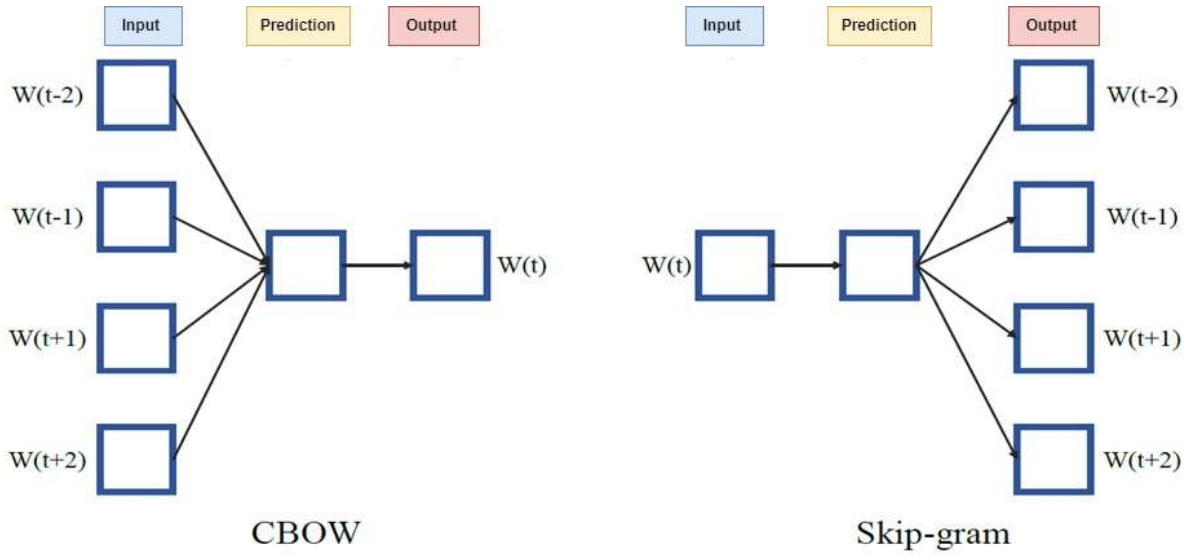

Figure 2. According to Skip-gram architecture, surrounding words are predicted using previous words, while continuous bag-of-words (CBOW) predict words using their context.

Fast Text

Various alternative word embedding forms disregard the morphology of words by giving each word its own vector. [32]. Through developing the new word embedding method dubbed Fast Text, fake news detection introduced a technique to overcome this issue. An n-gram of characters represents each letter in the word w. As known the word "introduction" and n = 3, FastText will generate the ensuing character tri-gram representation**:**

< rod , in, uce , int, tro, duc, ntr, ce, odu>

Inform that the trigram and the sequence matching the word there are different < Int> found in the phrase introduces. Assume the dictionary of n-grams of size G and are given a word w that is associated with each n-gram g as a vector representationZ_g. In this situation, the derived scoring function [33] is

$$S(W, C) = \sum_{g \in g_w} Z_g^T v_c \quad (5)$$

Where $g_w \in$ **{1, 2, 3, 4. . . G}.**

3.4. CLASSIFIER MODEL

3.4.1. MACHINE LEARNING CLASSIFICATION MODELS

We applied five popular machine learning classifiers for cybercriminal detection: Naïve Bayes, K-nearest Neighbor, Support Vector Machine, and Logistic Regression. Steps are coming after

the cleaning data & pre-processing, and feature extraction stages and Vectorizer text was fed into these classifiers to see how well they performed.

a. Naïve Bayes (NB)

For classifying separable features as text categorization or word processing documents, Naive Bayes (NB) is useful. The Bayes theorem variables V1, V2, and the multinomial order are developed via naive Bayes. C is independent in class. Equations (6) and (7) deal with NB classification in the data:

$$P(C|V) = \frac{p(V|C)}{p(V)} \quad (6)$$

$$P(C|(v1, v2... vn) = \frac{p(V1|C)P(V2|C).....P(Vn|C)P(C)}{p(V1)P(V2).....P(Vn)} \quad (7)$$

Hence, C stands for class, and a vector's features are described by V = (v1, v2,..., vn). Consequently, discover that certain qualities remain conditionally independent. Analyze the residue in accordance with every input in Equations (8).

$$NC = GMA\ C\ p(C) \prod_{i=1}^{n} p(vi|C) \quad (8)$$

Equation (10) is connected to picking the output with the highest probability after evaluating the likelihood of a dispersed collection of inputs to each likely value of class C. According to the text data, La Place semi-thing applied and was likely of a class that fit.[33].

b. K-nearest Neighbor Classifier (KNN)

For each feature type, KNN provides an approach to classification that is straightforward to implement. The model here handles binary class and multi-class situations far too easily. Due to data set restrictions, KNN has severe search challenges while locating the nearest neighbor. Additionally, this approach is a powerful data-dependent algorithm because KNN's execution performance depends on obtaining a substantial space function in Equations (9) (10).

$$F(x) = \arg\max S(x, Cj) \quad (9)$$

$$\sum_{di \in kNN} sim(x, di)y(di, Cj) \quad (10)$$

Where S points to score values by admiration on S(x, Cj), those score values of contender i for the label of j, and output of f (x) is a class for datasets test set [34].

K-NN Classification of Adjusted Weight

K-nearest neighbor classification of weight-adjusted (KNNWA) is the version of KNN that try to learn a weight vector to classifier [36]. The weighted cosine model is estimated as below:

$$\text{Cos(x, y, w)}\frac{\sum_{t\in T}(x1\times wt)\times(yt\times wt)}{\sqrt{\sum_{t\in T}(x1\times wt)^2}\times\sqrt{\sum_{t\in T}(y1\times wt)^2}} \qquad (11)$$

Where T indicates a set of an item and xt and yt are TF, as presented. To a training our model (d 2 D), made Nd = fn1, n2, . . . , nkg do a set of k-NN of d. as Nd, the association sum of d neighbors which relate to label c is determined as below (12):

$$S_c \sum_{\text{n}\in\text{N,C(n)=c}} \cos(\text{d, ni, w}) \qquad (12)$$

The whole similarity is estimated as below:

$$\text{T=}\sum_{\text{c}\in\text{C}} S_c \qquad (13)$$

The contribution of d is determined into a word of Sc of label c also T in Equations (14):

$$\text{Cont (d) =}\begin{cases}1\\0\end{cases} if\ \forall c\ \in C\ \neq label(d) \qquad (14)$$

Where cont (d) stands for contributions (d) [35]

c. Support Vector Machine (SVM)

SVM was first created for binary classification tasks—problems utilizing that dominant technique [36]. The classifier linear is also non-linear and applied to the dataset's dimension. A non-linear mapping is used by the support vector machine (SVM) algorithm to transform the real training data into a higher dimension. It was looked for during the linear optimal division hyperplane in the new size. When a sufficient non-linear mapping separates tuples from various

classes into a large enough high dimension data set, the result is a "decision limit" known as a hyperplane. [37].

In binary label SVM by the state of text data set classification, see x1, x2, x3, ..., xl make trained samples relating for one label X,

Where X is the compressed sub-set of RN [21]. Next, we could express the binary classifier s as shown in Equations (15):

$$\text{Min}\,\frac{1}{2}\parallel\ \omega\ \parallel\ +\frac{1}{2}\sum_{i=1}^{l}\delta - p \qquad (15)$$

Subject to:

$$(\omega, \emptyset(x_i)) \geq p\ \delta\ i = 1,2,3, \ldots \ldots l\ \delta \geq 0 \qquad (16)$$

If w and p resolve that difficulty, then a decision function is provided with:

$$\text{F(x) = sign } ((\omega, \emptyset(x)) - \text{p}) \qquad (17)$$

d. Random Forest classifier (RF)

The group learning method for text classification uses random forests, often known as random choice forests. T. Kam developed this method, which used t tree-like parallel [38], in 1995. This led to the development of a technique in 1999 by L. Bremen, who achieved convergence for RF as border measurements (mg(X, Y)) as in Equations (4.15):

$$\text{Mg(X, Y) =avk I (hk (X) = Y) – Avk I (hk(X) = j)} \qquad (18)$$

Where I (.) points to indicator function.

Next, after the whole train, trees as forests and prediction are allocated based on voting [41] as in Equations (19):

$$\delta_v \arg\max \sum_{\boldsymbol{i}\text{j:j6=j}}^{\boldsymbol{i}} \boldsymbol{i}_{\{\text{ij>ji}\}} \qquad (19)$$

Such that

$$r_{ij} + r_{ij} = 1$$

The Random Forest (RF) technique uses a collection of autonomous trees (decision trees). A "Gini index" is used for each branch's acquisition decision. Therefore is the estimated index in Equations (20):

$$T = 1\text{-}\sum_{i=1}^{c}(pi) \quad (20)$$

Around this point, pi showed the likelihood of class I, and c displayed the total number of classes. Then, we used 100 trains in the forest whenever an internal node split is denoted with a 'T.' RF is an ensemble of the decision tree classified as two kinds of nodes: internal and external. Internal nodes feature fundamental to building classification, while external nodes describe the decision class. Total of features analyzed through the split to determine the most suitable split into each node. In this Pseudocode of the RF, the algorithm is illustrated during algorithm (2).

e. **Logistic Regression (LR)**

Logistic Regression is appropriate to the binary classification text issue by this algorithm. Equations (21)–(23) determine the logistic function, which defines the Logistic Regression output:

$$h_0(x) = \frac{1}{1+ep(-0x)} \quad (21)$$

$$C(0) = \frac{1}{m}\sum_{i-1}^{m} c(h_0(x,y) \quad [32] \quad (22)$$

$$(h_0(x,y) = \begin{cases} \log(1-h(x))ify = 0 \\ \log(h(x))ify = 1 \end{cases} \quad (23)$$

Here, m symbolizes the whole of the train, h (xi) performs a hypothesis function from the ith train, and then yi is the input label ith train. So we applied the model to punish the classifier, including 'lbfgs' optimizer[39].

3.4.2. DEEP LEARNING MODELS DESCRIPTION

Deep learning models have produced cutting-edge results in various fields, including a wide range of NLP applications. Deep learning to categorize text data uses three different deep learning architectures in tandem. Below is a detailed description of each model [40].

Neural networks have become increasingly popular in the research community due to the elimination of manual feature extraction. The network's neurons are in charge of automatically extracting crucial properties that assist in distinguishing information from distinct classes. Large dataset sizes, which most machine learning algorithms fail to handle, need the use of neural networks. Neural networks are also more reliable and produce better categorization results. To implement our technique, we compared the models of standard neural networks for cybercrimes classification with Bi-LSTM, CNN, GRU, and LSTM.

A. Convolutional Neural Networks CNN

The convolutional neural network (CNN) is a deep learning architecture usually utilized for classifying hierarchical data documents [40]. Text categorization has been effectively accomplished using CNNs despite their image processing roots. In a straightforward CNN for image analysis, an image sensor is convolved with a collection of kernels of size $d \times d$. Convolution layers called "feature maps" can be stacked to provide a variety of input processors. From one layer to the next, CNN employs pooling to reduce output size, reducing computational complexity. Various pooling strategies are applied to minimize outputs while keeping critical properties [41].

Max pooling, which chooses the maximum element in the window, is the most used pooling technique. The maps are flattened into a single column to transfer the pooled output from stacked featured maps to the following layer. The final layers are often fully connected in CNNs. The weights and the feature detector filters are adjusted in a CNN's backpropagation step. When using CNN for text classification, the number of "channels," S, could be a problem. Applications for image categorization frequently have a few channels. E.g., only three RGB channels), and S for text classification applications can be quite extensive (e.g., 50 K) [42], resulting in extremely high dimensionality.

B. Long Short-Term Memory LSTM

LSTM is a sort of recurrent neural network (RNNs) that have the edge over RNNs in terms of information retention. Traditional RNNs have the problem of vanishing gradient descent, which LSTMs solve [43]. Because of their large memory capacity, LSTMs are ideal for text classification and predictive modeling applications. This type of network determines whether

data must be sent to additional neurons and that dataset could be forgotten or discarded. Backpropagation and a gated mechanism are used in these networks.

A fundamental LSTM network consists of an input, output, and the forget gate, represented by Equations (24)–(26).

$$\text{Input}_t = \sigma(w_i[h_t - 1, x_t] + b_i) \quad (24)$$

$$\text{output}_t = \sigma(w_o[h_t - 1, x_t] + b_o) \quad (25)$$

$$\text{forget gate}_{\ t} = \sigma(w_f[h_t - 1, x_t] + b_f) \quad (26)$$

Here, x_t symbolizes an input text, and h_t Denotes the input's state, with ht indicating the current state and ht1 representing the initial state. The weights and bias for each gate are W and b, respectively.

C. Bidirectional LSTM (Bi-LSTM)

In LSTM networks, it is a reliable approach for improving backpropagation. While data in an LSTM flows in only one direction, data in a Bi-LSTM can travel in both directions [44]. A Bi-LSTM may reverse and serially process inputs. It's just two LSTMs combined in opposite directions in terms of architecture. Using the forward LSTM layer, the network can remember information from the past to the future, while the backward LSTM layer can remember information from the future to the previous. Equation (27) illustrates the Bi-LSTM network's output:

$$Y_t = Y_t - 1, Y_t, \ldots, Y_t + \text{n} \quad (27)$$

Where $Y_t = \sigma$(h h), and σ is a concatenation operation

Two hundred neurons make up the BLSTM mode, while 400 neurons make up the second layer. There are three dense layers; the first dense layer has 128 neurons, the second has 64 neurons, and the third has 32 neurons. To prevent over-fitting, we also employed three dropout layers.

D. Gated Recurrent Unit GRU

Gated Recurrent Units GRUs is the mechanism of gating to RNN by dealing with the exploding and vanishing gradient formulated by [45]. The LSTM design is simplified in the GRU architecture. However, a GRU differs from an LSTM because it only has two gates and no internal memory.

In addition, a second non-linearity is not used. The following is a process explanation of a GRU cell: To input vector x_t At the same time, t by vectors, parameters, and matrices such as

W, U, and b, respectively, the updated gate and reset entrance are provided by Equations (28) and (30):

$$Z_t = \sigma(w_z x_t + U_z h_t - 1 + b_t) \quad (28)$$

$$r_t = \sigma(w_r x_t + U_r h_t - 1 + b_r) \quad (29)$$

Where h_t is determined in Equation (30)

$$h_t = (1 + z_r)\theta h_t - 1 + z_t h_t) \quad (30)$$

Implementation

4. EXPERIMENTAL ANALYSIS

In this section, we implement all experiments carried out using Python libraries. Our model is evaluated the data set from Twitter using the following metrics: accuracy, precision, recall, and F-measure. The collection dataset is illustrated in section 3.1.

We employed Regex Tokenizer for words tokenization, Porter Stemmer through stemming, and WordNet Lemmatize for a pre-processing task. As conferred above, the embedding dimension is set for 300. The hyperparameters and meta-parameters in the external neural network, such as the input dimension, length of recurrent dense layers, activation and dropout, and function, are determined upon different experiments to achieve the better possible outcomes. Part one used five machine learning models in subsection Two dealing with cybercrime: RF, DT, and LR. In the second part, we used four models of deep learning. Namely, CNN, Bi-LSTM, LSTM, and GRU is applied to evaluation with our proposed Cybercrime model. These classifiers have been designated for state-of-the-art cybercrime detection in date social networks. The same setup parameter configurations of the supposed baseline models in the original articles are employed.

Moreover, Python various 3.7.4 were used in our experiments. Some required libraries were used in the implementation and experiment configurations, such as Scikitlearn, NLTK, Keras, NumPy, TensorFlow, etc. The experimental evaluation is approved. Windows 10, Intel Core-i5 CPU, and 16 GB RAM hardware are used on a personal system through configuration. Hence, we are dealing with the Collection dataset, and pre-processing steps are performed as proposed in [58] applied to the NLTK package of Python. The input data set used for scenarios is divided into training and testing sets. To evaluate better, we also classified our model by using two scenarios, and the first one split 60% for training and 40% for testing; this is the scenario called (Scenario 1) and the second one divided 60% for training and 40% for testing; this is a scenario called (Scenario 2). The evaluation metrics are determined to show the better

performance of the dataset classification of each scenario. In each executed scenario, 5-fold cross-validation is adopted to achieve each evaluation metric's average value. Then we utilized a binary cross-entropy in the model.

Figure 3. ML and DL libraries

A. EVALUATION METRICS

The assessment measures used in this study to assess the effectiveness of the cybercriminal detection model are briefly highlighted in this subsection. The evaluation method is based on the following metrics: computing training time, precision, f score, recall, and accuracy. To acquire findings for each evaluation metric, however, each approach is carried out independently for each experiment [46][47].

Accuracy

Accuracy is the ratio of accurately expected observations to total observations, which is the most intuitive performance metric [48]. When there is highly dependable data, accuracy may be assumed, and the model is best. The following is the Equation (31):

$$\text{Accuracy} = \frac{\text{TP+TN}}{\text{TP+TN+FP+FN}} \tag{31}$$

Precision

Precision is the ratio of accurately anticipated positive observations to overall predicted positive statements, known as precision. This statistic answers the question of how much misclassification occurred in each class. Precision is linked to a low false-positive rate. As shown below is the Equation:

$$\text{Precision} = \frac{\text{TP}}{\text{TP+FP}} \tag{32}$$

Recall

The recall is the proportion of successfully predicted optimistic results to overall actual class results. The recall function asks, "How many of the rest of the data have been labeled?" A score of more than 0.5 is considered exceptional. As shown in the Equation be:

$$\text{Recall} = \frac{\text{TP}}{\text{TP+FN}} \quad (33)$$

A. F-Measure

F-measure uses the recall and precision results to evaluate total performance. The formulation presented in the Equation could be used to calculate the values of the f-measure.

$$\text{F} - \text{Measure} = \frac{2\times\text{pre}\times\text{Rcall}}{\text{pre+Rcall}} \quad (34)$$

5. EXPERIMENTAL RESULTS

We evaluate and compare the results of all classifiers used on three main types: cyberbullying terrorism and threatening datasets. Table 1 presents the experimental results of machine learning algorithms by employing five algorithms, NB, SVM, KNN, RF, and LR, and four feature extraction techniques such as Count Vectorization BoW, TFIDF, and TFIDF Word Unigram and bigram, with two Scenarios. First Scenarios for Training 60% and Testing 40% and Second Scenarios for Training 70% and Testing 30%. The first scenario's results are graphically illustrated in Figure 8 and Table 2. The first scenario's results are graphically illustrated in Figures 8 and 9. Table 2 results of the proposed cc model neural networks and their performance comparison are shown in Figures 10 and 2.

While neural network deep learning applied four algorithms CNN, LSTM, Bi-LSTM, and GRU. And two-word embedding techniques Word2Vec and fast text with two Scenarios. First Scenarios for Training 60% and Testing 40% and Second Scenarios for Training 70% and Testing 30%.

Although accuracy is the most straightforward and obvious criterion of model performance, it is not appropriate for datasets that are not evenly distributed. The F1-score (the harmonic mean of accuracy and recall), recall, and precision has been described. Their performance is likewise around 90% in most instances.

5.1. Evaluating cybercriminals detection for Machine Learning Models with two Scenarios

Table 1. Evaluating cybercriminals detection for Machine Learning Models with two Scenarios

Split data	Feature extraction	ML Algorithms	Accuracy	Recall	Precision	F-measure
First Scenarios For Training 60% and Testing 40%	**Count Vectorization BoW**	**NB**	**86.70**	**90.34**	**88.20**	**88.71**
		KNN	**73.01**	**77.68**	**76.46**	**74.42**
		SVM	**82.11**	**81.30**	**80.00**	**80.24**
		RF	**90.40**	**90.71**	**90.11**	**91.00**
		RL	**87.64**	**87.32**	**88.27**	**87.56**
	TFIDF	**NB**	**89.71**	**90.32**	**89.21**	**89.46**
		KNN	**74.00**	**77.62**	**77.41**	**74.00**
		SVM	**82.37**	**81.33**	**81.64**	**82.27**
		RF	**91.63**	**90.73**	**90.79**	**91.65**
		RL	**89.72**	**90.35**	**89.21**	**89.61**
	TFIDF Word Unigram and bigram	**NB**	**89.72**	**90.23**	**89.18**	**89.71**
		KNN	**91.61**	**80.66**	**79.46**	**79.00**
		SVM	**83.10**	**81.94**	**82.91**	**82.38**
		RF	**91.85**	**90.87**	**90.95**	**91.65**
		RL	**90.12**	**90.00**	**89.82**	**89.92**
Second Scenarios For Training 70% and Testing 30%	**Count Vectorization BoW**	**NB**	**86.71**	**90.34**	**88.22**	**88.72**
		KNN	**73.11**	**77.66**	**76.46**	**74.45**
		SVM	**82.12**	**81.37**	**80.87**	**80.31**
		RF	**90.43**	**90.77**	**90.17**	**91.14**
		RL	**87.66**	**87.35**	**88.20**	**87.79**
	TFIDF	**NB**	**89.72**	**90.34**	**89.22**	**89.72**
		KNN	**74.01**	**77.66**	**77.46**	**74.00**
		SVM	**82.38**	**81.37**	**81.91**	**82.38**

		RF	**91.66**	**90.77**	**90.88**	**91.66**
		RL	**89.72**	**90.35**	**89.22**	**89.72**
	TFIDF Word Unigram and bigram	**NB**	**89.72**	**90.34**	**89.22**	**89.72**
		RF	**76.01**	**80.66**	**79.46**	**79.00**
		SVM	**83.11**	**82.33**	**82.91**	**82.38**
		KNN	**91.98**	**90.98**	**90.93**	**91.79**
		RL	**90.12**	**90.00**	**89.97**	**89.98**

For the First scenario, the performance evaluating machine learning model observed that with KNN classifier achieves the highest accuracy of 91.61% through TF-IDF word unigram and bigram feature extraction, followed by another classifier named RF providing 91.85%, respectively. Hence TF-IDF words Unigram and bigram are shown as the best-achieved results than others feature as Logistic Regression achieving 91.12% F1 scores with a Count Vectorization and TF-IDF respectively. RF, LR, and KNN appear to perform better than NB and SVM in the first scenario, notwithstanding Naïve Bayes providing the better accuracy of 95.44% when using Count Vectorization. SVM demonstrates a poor performance compared to machine learning classifiers on this dataset.

In the second scenario with Machine Learning classifiers, the performance evaluation observed KNN achieves the best accuracy of 91.98% using the same feature extraction in the first scenario TF-IDF with word unigram and bigram. Here we explain that we used TF-IDF with the word unigram and bigram. Results improve much better than used TF-IDF only, as we observed in both scenarios. LR achieved 91.12% accuracy in different classifiers, and KNN provided 91.79% F1 scores with TF-IDF with word unigram and bigram. We observed a Count Vectorization in both scenarios achieved poor results compared to other feature extraction techniques of machine learning.

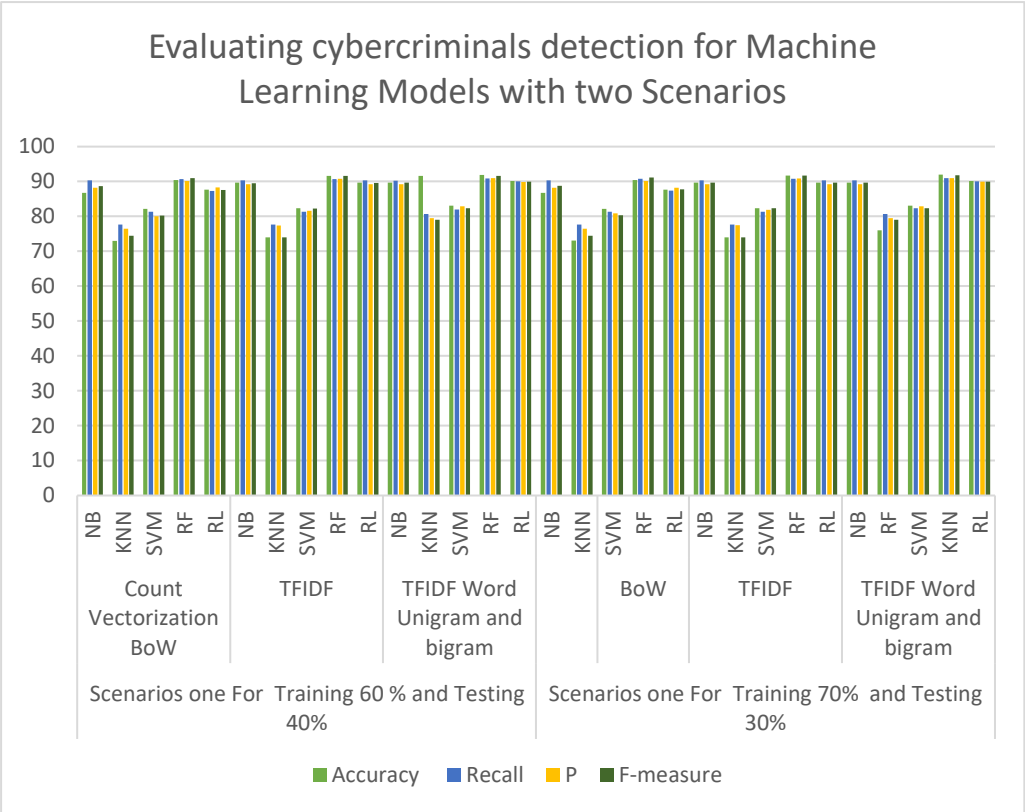

Figure 4. Evaluating cybercriminals detection for Machine Learning Models with two Scenarios

5.2. Evaluating cybercriminals detection for Deep Learning Models with two Scenarios

Table 2. Evaluating cybercriminals detection for Deep Learning Models with two Scenarios

Split data Training and Testing	**Feature extraction**	**Deep Learning Algorithms**	**Accuracy**	**Recall**	**P**	**F score**
Scenarios one For Training 60%	Word2Vec	CNN	**94.16**	**93.12**	**92.65**	**92.15**
		LSTM	**96.00**	**95.01**	**94.54**	**95.00**
		Bi-LSTM	**97.01**	**94.10**	**94.18**	**94.00**
		GRU	**93.90**	**93.65**	**92.61**	**92.99**

and Testing 40%	Fast text	CNN	**95.00**	**95.00**	**94.44**	**95.09**
		LSTM	**96.14**	**94.25**	**94.11**	**94.01**
		Bi-LSTM	**97.20**	**93.13**	**92.61**	**93.00**
		GRU	**95.19**	**95.09**	**95.00**	**95.10**
Scenarios one For Training 70% and Testing 30%	Word2Vec	CNN	**95.00**	**94.81**	**95.00**	**95.02**
		LSTM	**96.10**	**92.67**	**92.65**	**93.29**
		Bi-LSTM	**97.76**	**95.62**	**95.00**	**95.50**
		GRU	**95.07**	**94.93**	**95.00**	**95.13**
	Fast text	CNN	**94.16**	**93.12**	**92.65**	**92.15**
		LSTM	**96.00**	**95.01**	**94.54**	**95.00**
		Bi-LSTM	**97.18**	**94.10**	**94.18**	**94.00**
		GRU	**93.90**	**93.65**	**92.61**	**92.99**

Table 1. Evaluating cybercriminals detection for Deep Learning Models with two Scenarios

For the First scenario, we observed that Bi-LSTM achieves the highest accuracy of 97.01% using Word2Vec and 97.20 % using Fast Text, followed by LSTM providing 96.14% using Fast Text and CNN providing 94.16% accuracy with Word2Vec, respectively. Bi-LSTM, LSTM, and CNN appear to perform better than GRN in the first scenario.

For the second scenario, we observed that Bi-LSTM achieves the highest accuracy of 97.76% using Word2Vec and 97.18 % using Fast Text, followed by LSTM providing 96.00% using Fast Text and CNN providing 94.16% accuracy with Word2Vec, respectively. Bi-LSTM, LSTM, and CNN appear to perform better than GRN in the first scenario.

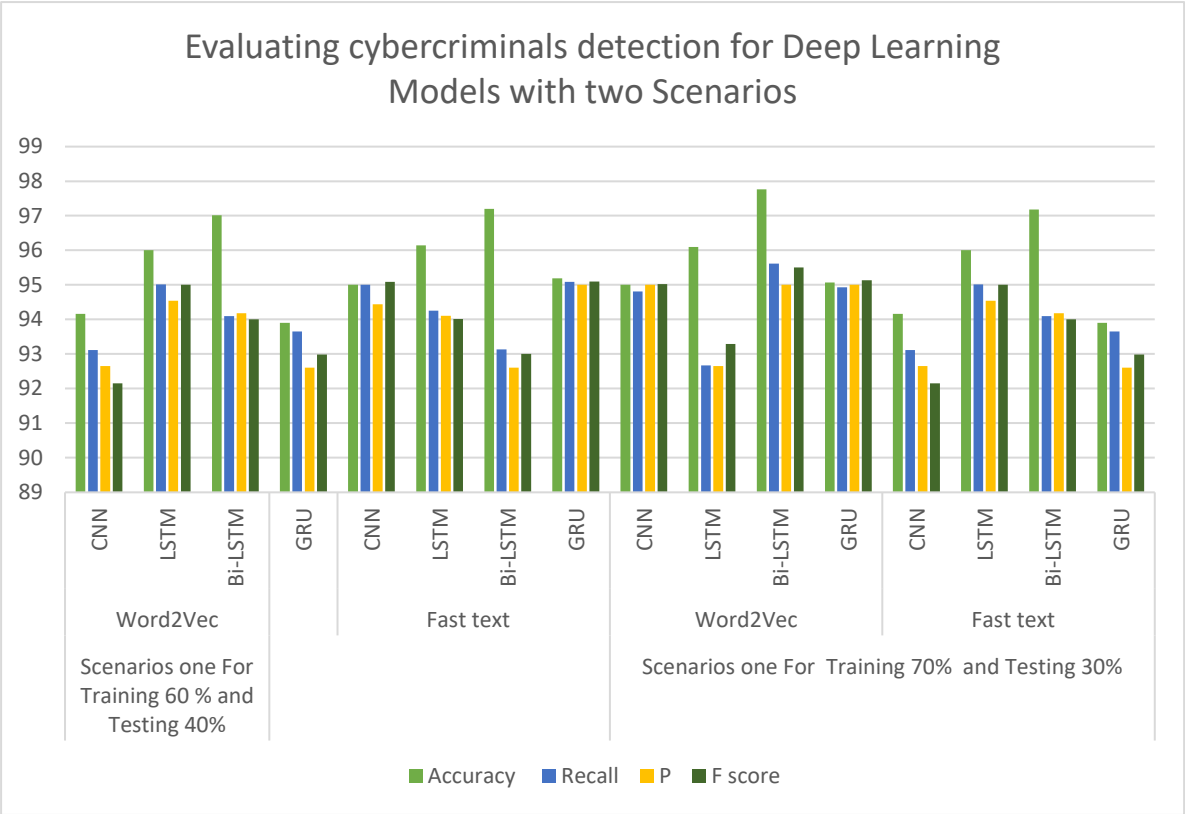

Figure 5. Evaluating cybercriminals detection for Deep Learning Models with two Scenarios

5.3.DISCUSSION RESULT

Performance Progress Rate displays the enhancements of the proposed cybercriminals detection model in term metrics: particularity, accuracy, precision, recall, and f-measure. The overall result is determined by comparing the execution of the proposed model by the different existing methods, five machine learning and four deep learning method evaluated to an evaluation method.

In machine learning, in the first scenario, the KNN classifier achieves the highest accuracy of 91.61% through TF-IDF word Unigram and bigram feature extraction, followed by another classifier named RF providing 91.85%, respectively. In the second scenario, KNN achieved the best accuracy of 91.98%, LR achieved 91.12% accuracy, and KNN provided 91.79% F1 scores with TF-IDF with word unigram and bigram. We observed using TF-IDF with words unigram and bigram, and results improve much better than using TF-IDF only or other feature extraction, as shown in both scenarios. In deep neural networks, in the first scenario, Bi-LSTM achieved the highest accuracy of 97.01% using Word2Vec and 97.20 % using Fast Text. In the

second scenario, Bi-LSTM achieved the highest accuracy of 97.76% using Word2Vec and 97.18 % using Fast Text, followed by LSTM providing 96.00% using Fast Text and CNN providing 94.16% accuracy with Word2Vec, respectively. We monitor that deep neural networks' highest-rated outperform machine learning classifiers. The Bi-LSTM mechanism shown performs very well. We show that in all situations, bidirectional neural networks perform better. Additionally, it is seen that the Bi-LSTM mechanism works remarkably well. The results of this study pave the path for creating more effective defenses against these online diseases.

When all assessment metrics are taken into account, the majority of findings obtained from studying the performance of external neural networks are over 95%. These numbers are also greater than those for conventional machine learning models that have been published. The Bi-LSTM with quick text embedding is the model that performs most accurately in the second case. Compared to Word2Vec and Fast Text, we found that fast text embedding delivered a higher and more reliable rate in better classification. Word2Vec is the winner, even if the other two embedding techniques have also done well. Compared to the other suggested shallow neural networks, the Bi-LSTM and CNN models fared better. The performance of Bi-LSTM models has been excellent for the second case. Fast text embedding had the best accuracy, which was followed by Bi-LSTM using Fast Text (98.65%) and CNN using Word2Vec (98.61%). The Bi-LSTM, LSTM, and CNN models perform the best in this case. We assume that the suggested framework enables all of the above models to conduct classification with high accuracy because the results over 95% are relatively similar.

In summarizing the findings, all relevant measures show strong performance. The accuracy of the proposed external neural networks is considerably over 90%, while the accuracy of all conventional machine learning models across all datasets is just over 80%. The neural network approaches outperform conventional machine learning algorithms in terms of performance. With just one hidden layer, each neural network architecture offers exceptional performance. Comparable to other suggested shallow networks in terms of capability is the CNN Bi LSTM combo framework.

Furthermore, the network's parameters, including its neuron number, the depth of its fully connected dense layers, and the dropout probabilities that were chosen through experimentation, produce the best outcomes. With all of the neural networks used, the suggested architecture performs well. We list the main findings in the following manner:

1. Neural networks displayed the highest implementation than state-of-the-art machine learning classifiers due to their capability and robustness for managing big data sets.
2. Count Vectorization, although an old feature extraction technique for statistical techniques, unfailingly provides satisfactory outcomes compared to other methods.
3. Across overall pre-processing phases, KNN and RF demonstrated a higher average performance among overall machine learning classifiers, followed by NB, SVM, and LR in the said order.
4. Fast Text embedding resulted in a maximum number of higher outcomes than Word2Vec, although similar results were achieved by the other two processes similarly.
5. Accuracy conveys the highest-rated performance by overall neural network models. Regarding the F1 measures, we conclude that RNN networks such as Bi-LSTM and GRU presented increased performance.

6. Conclusion

As the online world has grown, cybercrime has become a serious issue with far-reaching effects on individuals and society. Our research analyzes and investigates different dimensions of cybercrime as bullying detection. We explored nine classification approaches in our proposed cybercrime detection model, including deep learning, shallow neural networks, and traditional machine learning. Moreover, we have even employed seven kinds of embedding and feature extraction techniques. The findings are confirmed by experiments using a dataset and two scenarios. We developed optimal network parameters, dense and dropout layer sizes, and a novel neural network framework. The framework accommodates several classifiers and generally provides the highest outcomes, outperforming different baselines. We achieved a relative analysis discussion performance of overall approaches used—the benefit of this analysis fibs in determining critical mechanisms for cyberbullying detection text from social media. Further, the suggestion of shallow neural networks moderates the demand for complicated deep learning, hence scrimping resources.

In machine learning, in the first scenario, the KNN classifier achieves the highest accuracy of 91.61% through TF-IDF word Unigram and bigram feature extraction, followed by another classifier named RF providing 91.85%, respectively. In the second scenario, KNN achieved the best accuracy of 91.98%, LR achieved 91.12% accuracy, and KNN provided 91.79% F1 scores with TF-IDF with word unigram and bigram. We observed using TF-IDF with words unigram and bigram; results improve much better than using TF-IDF only or other feature extraction, as shown in both scenarios. In deep neural networks, in the first scenario, Bi-LSTM achieved the highest accuracy of 97.01% using Word2Vec and 97.20 % using Fast Text. In the

second scenario, Bi-LSTM achieved the highest accuracy of 97.76% using Word2Vec and 97.18 % using Fast Text, followed by LSTM providing 96.00% using Fast Text and CNN providing 94.16% accuracy with Word2Vec, respectively. We monitor that deep neural networks' highest-rated outperform ML classifiers. We prove that bidirectional neural networks are more effective in every situation. The Bi-LSTM mechanism observed in performs very well. We show that in all situations, bidirectional neural networks perform better. Additionally, it is seen that the Bi-LSTM mechanism works remarkably well. The results of this study pave the path for creating more effective defenses against these online diseases.

Conflicts of Interest

The authors declare that they have no conflicts of interest to report regarding the present study.

References

[1] M. Dadvar and K. Eckert, "Cyberbullying detection in social networks using deep learning-based models," *Lect. Notes Comput. Sci. (including Subser. Lect. Notes Artif. Intell. Lect. Notes Bioinformatics)*, vol. 12393 LNCS, no. June, pp. 245–255, 2020, doi: 10.1007/978-3-030-59065-9_20.

[2] A. K. Gautam and A. Bansal, "Performance Analysis of Supervised Machine Learning Techniques for Cyberstalking Detection in Social Media," *J. Theor. Appl. Inf. Technol.*, vol. 100, no. 2, pp. 449–461, 2022.

[3] M. A. Al-Ajlan and M. Ykhlef, "Deep learning algorithm for cyberbullying detection," *Int. J. Adv. Comput. Sci. Appl.*, vol. 9, no. 9, pp. 199–205, 2018, doi: 10.14569/ijacsa.2018.090927.

[4] F. Elsafoury, S. Katsigiannis, Z. Pervez, and N. Ramzan, "When the Timeline Meets the Pipeline: A Survey on Automated Cyberbullying Detection," *IEEE Access*, vol. 9, no. July, pp. 103541–103563, 2021, doi: 10.1109/ACCESS.2021.3098979.

[5] J. O. Atoum, "Cyberbullying Detection through Sentiment Analysis," *Proc. - 2020 Int. Conf. Comput. Sci. Comput. Intell. CSCI 2020*, pp. 292–297, 2020, doi: 10.1109/CSCI51800.2020.00056.

[6] M. M. Islam, M. A. Uddin, L. Islam, A. Akter, S. Sharmin, and U. K. Acharjee, "Cyberbullying Detection on Social Networks Using Machine Learning Approaches," *2020 IEEE Asia-Pacific Conf. Comput. Sci. Data Eng. CSDE 2020*, no. April, 2020, doi: 10.1109/CSDE50874.2020.9411601.

[7] A. Muneer and S. M. Fati, "A comparative analysis of machine learning techniques for

cyberbullying detection on twitter," *Futur. Internet*, vol. 12, no. 11, pp. 1–21, 2020, doi: 10.3390/fi12110187.

[8] X. Zhang *et al.*, "Cyberbullying Detection with a Pronunciation Based Convolutional Neural Network," *2016 15th IEEE Int. Conf. Mach. Learn. Appl.*, pp. 740–745, 2017, doi: 10.1109/icmla.2016.0132.

[9] V. Balakrishnan, S. Khan, and H. R. Arabnia, "Improving cyberbullying detection using Twitter users' psychological features and machine learning," *Comput. Secur.*, vol. 90, p. 101710, Mar. 2020, doi: 10.1016/J.COSE.2019.101710.

[10] V. Nahar, S. Al-maskari, X. Li, and C. Pang, "Semi-supervised Learning for Cyberbullying," *Databases Theory Appl.*, pp. 160–171, 2014.

[11] B. Y. AlHarbi, M. S. AlHarbi, N. J. AlZahrani, M. M. Alsheail, J. F. Alshobaili, and D. M. Ibrahim, "Automatic cyber bullying detection in Arabic social media," *Int. J. Eng. Res. Technol.*, vol. 12, no. 12, pp. 2330–2335, 2019.

[12] M. Dadvar, R. Ordelman, F. De Jong, and D. Trieschnigg, "Improved cyberbullying detection using gender information," *Dutch-Belgian Inf. Retr. Work. DIR 2012*, pp. 23–26, 2012.

[13] N. Yuvaraj *et al.*, "Automatic detection of cyberbullying using multi-feature based artificial intelligence with deep decision tree classification," *Comput. Electr. Eng.*, vol. 92, pp. 1–24, 2021, doi: 10.1016/j.compeleceng.2021.107186.

[14] R. Zhao and K. Mao, "Cyberbullying Detection Based on Semantic-Enhanced Marginalized Denoising Auto-Encoder," *IEEE Trans. Affect. Comput.*, vol. 8, no. 3, pp. 328–339, 2017, doi: 10.1109/TAFFC.2016.2531682.

[15] A. Dewani, M. A. Memon, and S. Bhatti, "Cyberbullying detection: advanced pre-processing techniques & deep learning architecture for Roman Urdu data," *J. Big Data*, vol. 8, no. 1, 2021, doi: 10.1186/s40537-021-00550-7.

[16] C. Graney-ward, B. Issac, L. Ketsbaia, and S. M. Jacob, "Detection of Cyberbullying Through BERT and Weighted Detection of Cyberbullying Through BERT and Weighted Ensemble of Classifiers Ensemble of Classifiers," pp. 0–12, 2022, doi: 10.36227/techrxiv.17705009.v1.

[17] M. Khairy, T. M. Mahmoud, and T. Abd-El-Hafeez, "Automatic Detection of Cyberbullying and Abusive Language in Arabic Content on Social Networks: A Survey," *Procedia CIRP*, vol. 189, pp. 156–166, 2021, doi: 10.1016/j.procs.2021.05.080.

[18] C. Iwendi, G. Srivastava, S. Khan, and P. K. R. Maddikunta, "Cyberbullying detection

solutions based on deep learning architectures," *Multimed. Syst.*, 2020, doi: 10.1007/s00530-020-00701-5.

[19] A. Agarwal, A. S. Chivukula, M. H. Bhuyan, T. Jan, B. Narayan, and M. Prasad, "Identification and Classification of Cyberbullying Posts: A Recurrent Neural Network Approach Using Under-Sampling and Class Weighting," *Commun. Comput. Inf. Sci.*, vol. 1333, pp. 113–120, 2020, doi: 10.1007/978-3-030-63823-8_14.

[20] S. B. Kotsiantis and D. Kanellopoulos, "Data pre-processing for supervised leaning," *Int. J. ...*, vol. 1, no. 2, pp. 1–7, 2006, doi: 10.1080/02331931003692557.

[21] T. Ahmad and M. N. Aziz, "Data pre-processing and feature selection for machine learning intrusion detection systems," *ICIC Express Lett.*, vol. 13, no. 2, pp. 93–101, 2019, doi: 10.24507/icicel.13.02.93.

[22] S. Sarica and J. Luo, "Stopwords in technical language processing," *PLoS One*, vol. 16, no. 8 August, 2021, doi: 10.1371/journal.pone.0254937.

[23] V. S and J. R, "Text Mining: open Source Tokenization Tools – An Analysis," *Adv. Comput. Intell. An Int. J.*, vol. 3, no. 1, pp. 37–47, 2016, doi: 10.5121/acii.2016.3104.

[24] A. W. Pradana and M. Hayaty, "The Effect of Stemming and Removal of Stopwords on the Accuracy of Sentiment Analysis on Indonesian-language Texts," *Kinet. Game Technol. Inf. Syst. Comput. Network, Comput. Electron. Control*, vol. 4, no. 3, pp. 375–380, 2019, doi: 10.22219/kinetik.v4i4.912.

[25] S. Khalid, T. Khalil, and S. Nasreen, "A survey of feature selection and feature extraction techniques in machine learning," *Proc. 2014 Sci. Inf. Conf. SAI 2014*, no. August 2014, pp. 372–378, 2014, doi: 10.1109/SAI.2014.6918213.

[26] V. Lampos, B. Zou, and I. J. Cox, "Enhancing feature selection using word embeddings: The case of flu surveillance," *26th Int. World Wide Web Conf. WWW 2017*, no. January 2018, pp. 695–704, 2017, doi: 10.1145/3038912.3052622.

[27] R. Corizzo, E. Zdravevski, M. Russell, A. Vagliano, and N. Japkowicz, "Feature extraction based on word embedding models for intrusion detection in network traffic," *J. Surveillance, Secur. Saf.*, pp. 140–150, 2020, doi: 10.20517/jsss.2020.15.

[28] T. Dodiya, "Using Term Frequency - Inverse Document Frequency to find the Relevance of Words in Gujarati Language," *Int. J. Res. Appl. Sci. Eng. Technol.*, vol. 9, no. 4, pp. 378–381, 2021, doi: 10.22214/ijraset.2021.33625.

[29] T. H. E. Effect *et al.*, "Impact Factor : International Scientific Journal Theoretical & Applied Science THE EFFECT OF DIVERSITY OF THE NATIONALITY , BOARD OF DIRECTOR , INVESTMENT DECISION , FINANCING DECISION , AND

Impact Factor :," no. June, 2000, doi: 10.15863/TAS.

[30] M. Kowsher *et al.*, "An Enhanced Neural Word Embedding Model for Transfer Learning," *Appl. Sci.*, vol. 12, no. 6, 2022, doi: 10.3390/app12062848.

[31] E. L. Goodman, C. Zimmerman, and C. Hudson, "Packet2Vec: Utilizing Word2Vec for Feature Extraction in Packet Data," no. April, 2020.

[32] G. Forman and E. Kirshenbaum, "Extremely fast text feature extraction for classification and indexing," *Int. Conf. Inf. Knowl. Manag. Proc.*, pp. 1221–1230, 2008, doi: 10.1145/1458082.1458243.

[33] C. Elkan, "Naive Bayesian Learning," no. December 1998, pp. 1–4, 2007.

[34] X. Y. Wang and Z. O. Wang, "Improved K-nearest neighbor algorithm," *Dianzi Yu Xinxi Xuebao/Journal Electron. Inf. Technol.*, vol. 27, no. 3, pp. 487–491, 2005.

[35] A. K. Nikhath, K. Subrahmanyam, and R. Vasavi, "Building a K-Nearest Neighbor Classifier for Text Categorization," *Int. J. Comput. Sci. Inf. Technol.*, vol. 7, no. 1, pp. 254–256, 2016.

[36] F. R. Lumbanraja, E. Fitri, Ardiansyah, A. Junaidi, and R. Prabowo, "Abstract Classification Using Support Vector Machine Algorithm (Case Study: Abstract in a Computer Science Journal)," *J. Phys. Conf. Ser.*, vol. 1751, no. 1, 2021, doi: 10.1088/1742-6596/1751/1/012042.

[37] L. Wei, B. Wei, and B. Wang, "Text Classification Using Support Vector Machine with Mixture of Kernel," *J. Softw. Eng. Appl.*, vol. 05, no. 12, pp. 55–58, 2012, doi: 10.4236/jsea.2012.512b012.

[38] M. D. M. Manessa, K. T. Setiawan, M. Haidar, S. Supriatna, A. Pataropura, and A. H. Supardjo, "Optimization of the random forest algorithm for multispectral derived bathymetry," *Int. J. Geoinformatics*, vol. 16, no. 3, pp. 1–6, 2020, doi: 10.1007/978-981-15-0978-0.

[39] B. Fatemi, S. M. Kazemi, and D. Poole, "A Learning Algorithm for Relational Logistic Regression: Preliminary Results," no. 2004, 2016.

[40] M. P. Véstias, R. P. Duarte, J. T. de Sousa, and H. C. Neto, "Moving deep learning to the edge," *Algorithms*, vol. 13, no. 5, pp. 1–33, 2020, doi: 10.3390/A13050125.

[41] K. Sekaran, P. Chandana, N. M. Krishna, and S. Kadry, "Deep learning convolutional neural network (CNN) With Gaussian mixture model for predicting pancreatic cancer," *Multimed. Tools Appl.*, vol. 79, no. 15–16, pp. 10233–10247, 2020, doi: 10.1007/s11042-019-7419-5.

[42] S. Indolia, A. K. Goswami, S. P. Mishra, and P. Asopa, "Conceptual Understanding of

Convolutional Neural Network- A Deep Learning Approach," *Procedia Comput. Sci.*, vol. 132, pp. 679–688, 2018, doi: 10.1016/j.procs.2018.05.069.

[43] Y. Hua, Z. Zhao, R. Li, X. Chen, Z. Liu, and H. Zhang, "Deep Learning with Long Short-Term Memory for Time Series Prediction," *IEEE Commun. Mag.*, vol. 57, no. 6, pp. 114–119, 2019, doi: 10.1109/MCOM.2019.1800155.

[44] J. L. Wu, Y. He, L. C. Yu, and K. Robert Lai, "Identifying Emotion Labels from Psychiatric Social Texts Using a Bi-Directional LSTM-CNN Model," *IEEE Access*, vol. 8, pp. 66638–66646, 2020, doi: 10.1109/ACCESS.2020.2985228.

[45] M. Mohd, F. Qamar, I. Al-Sheikh, and R. Salah, "Quranic optical text recognition using deep learning models," *IEEE Access*, vol. 9, pp. 38318–38330, 2021, doi: 10.1109/ACCESS.2021.3064019.

[46] A. Yahya, A. Amer, and T. Siddiqui, "Detection of Covid-19 Fake News text data using Random Forest and Decision tree Classifiers," *Int. J. Comput. Sci.*, vol. 18, no. 12, pp. 88–100, 2020, doi: 10.5281/zenodo.4427205.

[47] T. Siddiqui, A. Y. A. Amer, and N. A. Khan, "Criminal Activity Detection in Social Network by Text Mining: Comprehensive Analysis," *2019 4th Int. Conf. Inf. Syst. Comput. Networks, ISCON 2019*, pp. 224–229, 2019, doi: 10.1109/ISCON47742.2019.9036157.

[48] K. Kowsari, K. J. Meimandi, M. Heidarysafa, S. Mendu, L. Barnes, and D. Brown, "Text classification algorithms: A survey," *Inf.*, vol. 10, no. 4, pp. 1–68, 2019, doi: 10.3390/info10040150.