Effect of Features Extraction Techniques on Cyberstalking Detection Using Machine Learning Framework

Arvind Kumar Gautam and Abhishek Bansal
Department of Computer Science, Indira Gandhi National Tribal University, Amarkantak, M.P., India
Email: analyst.igntu@gmail.com, abhishek.bansal@igntu.ac.in

***Abstract*—Various cybercriminals are active with predefined and preplanned agendas to carry out cybercrimes in the Internet world. Cyberstalking, cyberbullying, cyber terrorism, cyber hacking, data leakage, identity theft, phishing, and other types of cyber harassment continually occur in the virtual world. Cyberstalking and cyberbullying are near to close in content and intent, involving the same internet-based technology to harass, bully and undermine others online. This paper implemented a cyberstalking detection model and analyzed the effect of various feature extraction techniques on different machine learning classifiers for cyberstalking detection. For feature extraction, the proposed model applied Word2vec, BOW, TF-IDF, FastText, GloVe, ELMo, and BERT. Logistic Regression (LR), Support Vector Machine (SVM), K-Nearest Neighbor (KNN), Random Forest (RF), Naive Bayes (NB), and Decision Tree (DT) were used for classification. Effects of each feature extraction method to enhance the performance of the detection model were determined based on the performance results of applied classifiers with each feature extraction process. Experimental results show that BOW and TF-IDF outperformed advanced word embedding-based feature extraction methods. BOW (for LR) achieved the highest accuracy of 95.7%, highest precision of 97.9%, and highest F-Score of 97.3%. TF-IDF achieved the highest recall of 99.8% for NB. SVM classifier achieved the second-highest accuracy of 95.2% with TF-IDF. BERT model successfully obtained maximum accuracy of 90.9% and 90.7% for LR and SVM, respectively. ELMo model also performed well and produced maximum accuracy of 90.5% and 90.2% for LR and SVM, respectively. The SkipGram model of Word2Vec provided an accuracy of 85% for the LR classifier. GloVe provided 81.2% accuracy for the RF classifier. SkipGram and the CBOW model of FastText provided 85.7% and 82.2% accuracy, respectively, for the RF classifier.**

***Index Terms*—features extraction, word embedding, machine learning, cyberstalking detection, cyberbullying bag of words, TF-IDF, Word2Vec, GloVe, FastText, ELMo, BERT**

I. Introduction

In the web world, online social media applications and email technology are making habitually to the society. People often use social media platforms for legal and illegal activities, such as education, business, entertainment, fake news propaganda, publicity, and cybercrimes. Nowadays, cybercriminals are utilizing social media for various cyber fraud and cyberstalking activities. In the current time, many people or organizations are reporting cyberstalking cases in the cyber cell of the police station [1], [2]. Cyberstalking is a severe cyber attack [3] in which the attacker uses digital media to harass the victim or group through personal attacks and the disclosure of false or confidential information among other persons [4]. Cyberstalking and cyberbullying are two challenging issues of online abuse and are near to close in content and intent, which involve the same internet-based technology to harass, bully and undermine others in the online world. Cyberbullying mainly focuses on teenagers, while cyberstalking targets other users in the internet world for online harassment. Cyberstalking is systematic, repeated, and numerous cyber-attacks and does not occur on a single occurrence [5], [6]. Cyberstalking may be classified into Email stalking, Internet stalking, Computer stalking, Phone stalking, and Automated stalking [7]. In email stalking, stalkers use email technology to send hateful, offensive, threatening messages which contain spam and viruses. Email stalkers often use fake email IDs to fraud and harass the victim. In internet stalking, social media platforms are used by the stalker to bully, harass and troll the victim. In computer stalking, computers and personal accounts are hacked and controlled by the stalker to target the victim. In phone stalking, victims are harassed using repeated and unwanted phone calls, texts, and multimedia messages through mobile phones. Automated stalking is an advanced technology used by stalkers to target victims using mobile apps and automated computer programs controlled by suspicious servers. There are many examples of cyberstalking, like making and posting a real or fake sexual image of the victim to their loved ones, uploading personal information on public websites, and hacking social media and email accounts [8]. Trolling, flaming, excluding, masquerading, mobbing, denigrating, outing, harassing, and hacking are others kinds of cyberstalking [9]. Different types of cyberstalking are presented in Fig. 1.

Manuscript received December 16, 2021; revised March 6, 2022; accepted March 9, 2022.

doi: 10.12720/jait.13.5.486-502

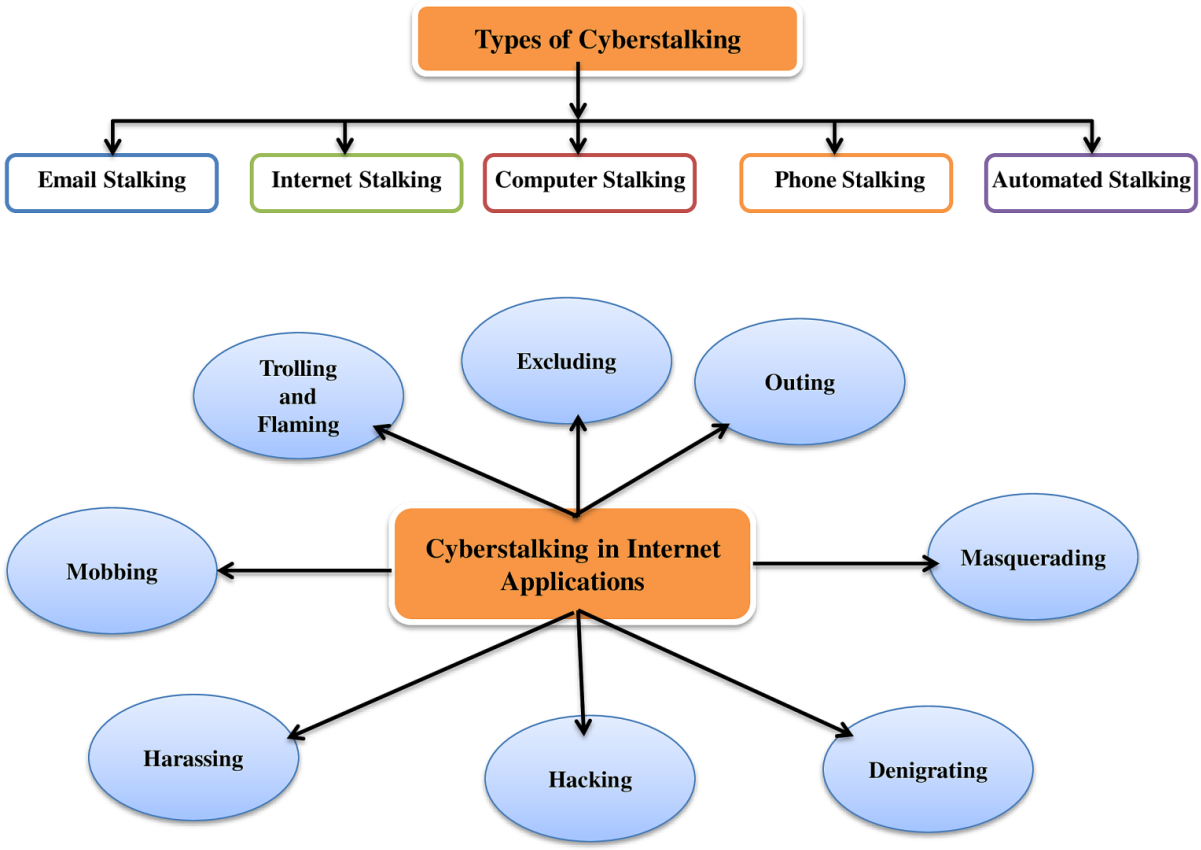

Figure 1. Types and examples of cyberstalking.

Cyberstalking should be observed in detection, prevention, and control to diminish its severe impact. In the literature, many cyberstalking detection techniques are inspired by various machine learning techniques [10], [11]. Machine Learning Algorithms (MLA) are the most well-known utilization of Artificial Intelligence (AI). Machine learning techniques have an automated learning capacity and produce accurate outcomes from learning experiences [12]. In machine learning, a classifier is an automated learning algorithm that is used by the machine to classify data. It is also known as the brain of any machine learning algorithm to filter good and bad data. The success of any machine learning-based model depends on the performance of classifiers in cyberstalking detection model. Many factors are involved in enhancing the performance of the classifiers. Some important factors are datasets circulation, pre-processing tasks, and proper feature extraction from datasets. However, the feature extraction techniques are most effective in improving the performance of the classifier in the cyberstalking detection model.

This paper analyzes the impact of various features extraction methods in machine learning classifiers to better performance for cyberstalking detection models. The significant contributions of this research work are as follows.

1) Some chosen quality papers were reviewed to explore the well-known feature extraction methods utilized by the researchers to improve the performance of the cyberstalking detection model.
2) Proposed a cyberstalking detection model utilizing machine learning with seven popular feature extraction methods: BOW (Bag of Word), TF-IDF (Term Frequency-Inverse Document Frequency), Word2Vec, FastText, GloVe (Global Vectors), ELMo, and BERT.
3) The proposed detection model was evaluated using six machine learning classifiers with all selected feature extraction methods on the same dataset to measure the effect of feature extraction methods in enhancing the performance of the detection model.
4) Finally, the study determined the effect of feature extraction methods and found that feature extraction is vital in improving the classification task.

The rest of the article is structured section-wise. Section II shows the survey of past work performed by the researchers for cyberstalking detection utilizing diverse feature extraction strategies with machine learning methods. Section III explains the materials and proposed system for cyberstalking detection using machine learning. The experimental outcomes from the proposed approach and discussions are explained in Section IV. The study is finalized with future directions in Section V of the paper.

II. Review of Literature

This section discusses the contributions of researchers that utilized the diverse feature extraction methods with MLA to detect cyberstalking and cyberbullying. In the literature, researchers have applied different feature extractions methods with machine learning approaches to enhance the performance of the detection model. In 2012, Vinita Nahar *et al.* [13] proposed a cyberbullying detection model using two feature selection methods. The BOW method was used for standard features extraction from offensive and non-offensive comments. The probabilistic idle semantic analysis technique performed sentiment features extraction from abusive messages. The authors achieved high accuracy using a support vector machine for text classification. In 2013, Maral Dadvar *et al.* [14] proposed a feature-based detection model and found that detection model performance could be improved using offensive-specific features such as timeline and chat history. The detection model utilized SVM for classifications and BOW for the feature extraction method on a YouTube dataset containing 4626 comments from 3858 distinct users. In 2014, another feature-extraction-based detection model for word similarity was implemented by Zhang *et al.* [15]. The authors applied the Word2Vec word embedding method on the WordNet synonym dictionary dataset and found better results. In 2015, Ghasem Z. *et al.* [16] suggested a framework for controlling and combating cyberbullying and cyberstalking using a machine learning-based approach. The authors used a hybrid machine learning approach with several feature selection methods to automatically detect and mitigate email-based cyberstalking. The proposed model applied SVM and neural network in a 5172 spam and genuine email dataset. The authors claimed that the detection model could collect necessary evidence for law enforcement. In 2016, Michele Di Capua [17] proposed a detection model using a machine learning approach on social media platforms. The authors experimented on several datasets from Twitter, YouTube, and Formspring using the Support Vector Machine algorithm for text classification while syntactic, semantic, sentiment and social features were used for features extraction. In 2017, S. Bhoir *et al.* [18] implemented a model to measure the performance of different word embedding techniques. The authors experimented using several word embedding-based feature extraction techniques, namely CBOW (Continuous Bag of Word), Skip-gram, GloVe (Global Vectors), and Principal Component Analysis (PCA). The authors found that GloVe performed better than other models with large and small datasets. The experimental results of the authors showed that CBOW and PCA models are required more data, while Skip-gram and GloVe are required fewer data to enhance the performance.

In 2018, Waykole *et al.* [19] reviewed feature extraction methods and proposed an approach for text classification and measuring the effect of feature extraction techniques. The authors applied LR and RF classifiers with TF-IDF, BOW, and word2vec feature extraction methods and found that Word2Vec is the better feature extraction method with a random forest classifier for text classification. In 2019, John Hani *et al.* [20] implemented a model to detect cyberbullying on social media. Experimental works were performed based on sentiment analysis using neural networks and SVM classifier with TF-IDF feature extraction techniques on the Kaggle dataset. The authors achieved better accuracy (92.8% and 90.3% for neural network and SVM) than previous researchers on the same dataset. Ravinder Ahujaa *et al.* [21] implemented a sentiment-analysis-based detection model to determine the impact of features extraction methods. The authors applied TF-IDF and N-Grams for feature extraction on the Twitter dataset. The Authors used DT, SVM, KNN, RF, LR, and NB for classification and found that TF-IDF produced better results than n-gram. Al-Hashedi *et al.* [22] developed a deep learning-based model with word embeddings-based feature extraction to detect cyberbullying. The detection model applied word2vec, GloVe, and ELMo (Embeddings from Language Models) and found that BiLSTM (Bidirectional Long Short-Term Memory) with ELMo outperformed in the detection of cyberbullying texts. Modha et al. [23] implemented a model for textual aggression classification on multilingual social media data. The authors used and compared the various feature extraction methods in their experimental work using machine learning and deep learning methods. Authors observed that ML classifiers performed better with TF-IDF and BOW while word embedding-based methods (Word2Vec, GloVe, FastText) enhance the performance of deep learning models.

In 2020, Manowarul Islam *et al.* [24] proposed an approach to increase the performance of the model in cyberbullying and cyberstalking detection on online media networks. The authors assessed their proposed framework utilizing DT, RF, NB, and SVM classifiers with BOW and TF-IDF and accomplished better outcomes. Amgad Muneer *et al.* [25] offered a machine learning way to improve the detection accuracy on Twitter. The authors assessed their proposed framework utilizing DT, LR, RF, NB, SVM, LGBM, SGD, and AdaBoost classifiers. Word2Vec and TF-IDF strategies were used for feature extraction. Anant Khandelwal et al. [26] implemented a deep learning-based unified system for aggression identification on English-Hindi blended remarks utilizing Deep Pyramid CNN, Disconnected RNN, and Pooled BiLSTM. Authors applied emotion sensor features, parts-of-speech (pos), punctuation, sentiment analysis, topic signals, and TF-IDF on TRAC 2018 and Kaggle datasets. Hoyeon Park *et al.* [27] analyzed the impact of word embedding-based feature extraction methods. The model was experimented with using NB, SVM, RF, Gradient Boosting, and XGBoost classifiers with BoW, TF-IDF, and Word2Vec techniques. Authors found that applied ML classifiers achieved the highest accuracy of 84.27% with TF-IDF while classifiers achieved an accuracy of 79.8% with Word2Vec. Thavareesan *et al.* [28] implemented part of the speech tagging model using machine learning. The authors used

TF-IDF, Word2Vec, BOW, FastText, and GloVe for feature extraction and compared the performance of word embedding methods. As per experimental work, the authors found that the performance of TF-IDF and BOW were better than other applied word embedding models. Barrientos *et al.* [29] proposed a model for classifying textual erotic content using machine learning techniques. Authors used LR, SVM, RF, and KNN for classification, while TF-IDF, Word2Vec, and BOW were used for feature extraction. Experimental work of authors observed that TF-IDF performed better than BOW and Word2Vec.

In 2021, A. Asante *et al.* [30] proposed a substance-based technical solution for cyberstalking detection utilizing machine learning and data mining. The proposed system utilized modules for message recognition, separating, content detection, profiling offender, and substantiation modules. N. Dughyala *et al.* [31] suggested a model for automating the detection of cyberstalking utilizing MLA and NLP. The authors asserted that their proposed system would automatically detect cyberstalking and identify the stalker on the web. Marwa Tolba *et al.* [32] suggested hybrid group ways to deal with online harassment detection in profoundly imbalanced information. The authors applied word2vec, GloVe, and SSWE word-embedding with nine methods for balancing skewed class distribution. The authors observed that the performance of GloVe was superior to other strategies. H. S. Alatawi *et al.* [33] implemented a model for hate speech detection using BiLSTM, BERT (Bidirectional Encoder Representations from Transformers), and other word embeddings. The proposed approach experimented on a mixed dataset containing Twitter and Stormfront datasets. According to the result from the experiment, it was observed that BERT achieved a maximum f-score of 80% while BiLSTM accomplished a 75% f-score. Jain *et al.* [34] proposed a cyberbullying detection model using machine learning techniques. The authors used the BOW, TF-IDF, and Word2Vec for feature extraction and found that BOW and TF-IDF performed better than the Word2Vec model. Pericherla *et al.* [35] analyzed the performance of various feature extraction methods for cyberbullying detection. The authors utilized the LR and LightGBM machine learning algorithms to measure the performance of feature extraction methods. BOW, TF-IDF, Word2Vec, FastText, GloVe, ALBERT (A Lite version of BERT), ELECTRA (Efficiently Learning an Encoder that Classifies Token Replacements Accurately), XLNet, RoBERTa(Robustly Optimized BERT Pretraining Approach), and GPT-2(Generative Pre-trained Transformer) were applied in the experimental work. The authors claimed that GPT-2 and RoBERTa performed better than other features extraction methods based on the experiment. Raj *et al*. [36] proposed a hybrid model for cyberbullying detection using MLA and NLP. Authors applied machine learning (XG Boost, SVM NB, and LR) and deep learning algorithms (CNN, GRU, BiGRU, LSTM, BiLSTM, and CNN-BiLSTM) for classification while TF-IDF, BOW, GloVe, and FastText were used as feature extraction methods. Authors claimed that deep learning performed better than machine learning algorithms, although both techniques achieved almost similar results. The authors measured the effect of features extraction methods and observed that machine learning performed better with TF-IDF while deep learning algorithms performed better with GloVe.

Apart from these, several other researchers have also suggested different approaches to the detection model. Haoti Zhong *et al.* [37] proposed a detection model utilizing the Bag of Words and Word2Vec with several ML classifiers and deep learning techniques to detect cyberbullying on the Instagram social network. Chatzakou *et al.* [38] implemented a detection model utilizing the Bag of Words with machine learning classifiers to detect cyberbullying and cyber aggression in social media. The authors achieved more than 90% accuracy and AUC. Rui Zhao *et al.* [39] suggested a model for cyberbullying detection in social media. The authors implemented the model on Twitter and MySpace utilizing the Bag of Words with machine learning algorithms. Manuel *et al.* [40] proposed a model for early cyberbullying detection on social media. The authors used the Bag of Words with ML classifiers and found better results. Prabowo *et al.* [41] proposed sentiment analysis-based techniques for cyberbullying detection. The authors used the TF-IDF feature extraction with SVM and found promising results. Hugo Rosa *et al.* [42] suggested a cyberbullying detection model on social media. The authors implemented the model on Twitter Formspring and Google-News datasets by utilizing the Word2Vec with several ML classifiers and deep learning techniques as single and hybrid approaches. Bhagya *et al.* [43] proposed a model for cyberbullying detection on social media. The authors utilized the TF-IDF method with an SVM classifier and achieved 92% accuracy. Nijia *et al.* [44] proposed text-based cyberbullying detection on social media using the TF-IDF with character-level CNN model. The authors performed the experimental work on the Tweet dataset (English) and the Weibo dataset (Chinese). Safa Alsafari *et al.* [45] implemented a model to detect Hate speech and offensive speech on social media written in Arabic. Authors utilized Word2Vec, FastText, and other contextual feature extraction with deep learning and machine learning techniques. Sweta Agrawal *et al.* [46] suggested a model for cyberbullying detection across multiple social media networks. The authors implemented the model on Wikipedia, Twitter, and Formspring by utilizing the Word2Vec with deep learning techniques. Jianwei Zhang *et al.* [47] suggested a model for cyberbullying detection on Twitter (Japanese text) using multiple textual features. The authors utilized the Word2Vec with ML classifiers. Bandeh Ali *et al.* [48] develop a machine learning approach utilizing the Word2Vec method for cyberbullying detection and found better results. Benaissa *et al.* [49] suggested a system to detect cyberbullying in Arabic text using the FastText model with deep learning techniques. Devin Soni *et al.* [50] presented a machine learning approach for cyberbullying detection in

multimedia content using the GloVe model. Pinkesh Badjatiya *et al.* [51] also suggested a deep learning-based methodology for hate speech detection on Twitter using the GloVe model.

In the literature review, several related research papers between the years 2012 to 2021 were selected to find the popular feature extraction techniques, machine learning techniques, and contributions of previous work performed by researchers to detect cyberbullying, cyberstalking, and other cyber harassment. Machine learning, deep learning, fuzzy logic, and hybrid-based approaches are being broadly utilized in cyberstalking detection. Literature review showed that feature extraction methods are crucial reasons to enhance the performance of the cyberstalking detection model. BOW, TF-IDF, and Word2Vec are broadly used features extraction techniques. However, researchers are also utilizing advanced word embeddings-based and language model-based features extraction methods such as GloVe, FastText, InferSent, ELMo, BERT, GPT-2, ALBERT. Authors at [7], [13], [14], [19], [23], [24], [27]-[29], [34]-[40] utilized the BOW for feature extraction. Performance of the TF-IDF feature extraction method experimented by authors [7], [10], [19]-[22], [24]-[29], [34]-[36], [41], [43], [44]. Authors at [12], [15], [18], [19], [22], [23], [25], [27]-[29], [32], [34], [35], [42], [45]-[48] have experimented the Word2Vec model using machine learning and deep learning. Authors at [23], [28], [35], [36], [45], [49] suggested the FastText model for achieving better results from detection models. GloVe model tested and suggested by authors at [18], [22], [23], [28], [32], [35], [36], [50], [51]. Authors at [10], [22], [23], [28], [33], [35], [36] experimented the other word embedding and language model-based feature extraction methods. The majority of research suggested using the BOW and TF-IDF with machine learning for text classification with better results. Word2Vec with machine learning and deep learning is better for semantic relation. Advance word embedding and language models based feature extraction methods are better suited with large corpus and enhance the performance of detection tasks. Authors at [18], [23], [27], [28], [35] compared and analyzed the performance of feature extraction methods. However, there is still a lack of proper comparison among the feature extraction methods.

III. Material and Proposed Methodology

The proposed machine learning framework consists of four main phases for cyberstalking detection: pre-processing, features extraction, text classification, and cyberstalking detection. The basic layout of the proposed detection framework is explained in Fig. 2. The overall functioning of the proposed methodology used the following main steps.

Step 1: The messages, tweets, comments, and emails from different internet sources were collected for making the labeled dataset.

Step 2: Data from the dataset were cleaned through pre-processing tasks.

Step 3: Features vectors were calculated separately using different feature extraction techniques with word-level for syntactic features, semantic features, and sentiment features.

Step 4: Training and testing sets were prepared by splitting the cleaned dataset.

Step 5: From the training set, ML classifiers were trained.

Step 6: Data from datasets were classified as cyberstalking or non-cyberstalking text based on prediction probability scored provided by the trained classifiers.

Step 7: The performance of the detection model was measured using different parameters.

Step 8: Step 2 to step 7 were repeated for each feature extraction method: BOW, TF-IDF, Word2Vec, FastText, GloVe, ELMo, and BERT.

A. Dataset

We collected the datasets from Kaggle and other sources [52]-[57] and manually made a mixed dataset containing public and private Emails, Twitter tweets, and comments from Facebook and YouTube. Text of datasets was classified as non-cyberstalking text and cyberstalking text. The experimental dataset includes a total of 20375 unique records. Dataset was split by 15281 and 5094 for training and testing, respectively.

B. Pre-processing

The data collected from various sources contain raw text with unnecessary characters, blank spaces, blank lines, meaningless characters, and different symbols. Such types of text of the dataset can not be used directly for feature extraction and need to clean and arranged properly. Pre-processing tasks are required to properly prepare data before feature extraction and evaluating the ML algorithms. The different tasks of pre-processing must be performed carefully because it often affects the performance of the detection model. In this phase, the data of the datasets were filtered and normalized into a specific format. We performed several pre-processing tasks using Natural Language Processing (NLP).

1) Removing stop words

In corpus, any articles, prepositions, and pronouns that do not sense sentiment are called stop words [58]. These stop words are often meaningless and make an unnecessary burden for determining the syntactic, semantic, and sentiment meaning in the classification task. All stop words from the text were removed.

2) Noise removal

Any dataset also contains various unnecessary and meaningless characters. In corpus, any digits, repeated words, symbols, blank space, special characters, and punctuation marks are called noise data [59]. All noise data were removed from the dataset using NLP.

3) Tokenization

Splitting the corpus sentence into individual words is called tokenization [59]. After removing the stop word and noise from the dataset, corpus sentences were divided into words using the tokenizer method. After that, all tokenized words were added to a list.

4) Normalization

Making the uniformity of corpus text is called normalization [59]. All tokenized words must be converted to either upper case or lower case letters as well as convert useful numbers to equivalent words for normalization. The normalization process is essential in pre-processing because tokenized words with different case letters refer to separate meanings. For example, the words' svm' and 'SVM' refer to non-identical terms in the vector. In the experiment in this paper, all words were converted into lower case letters.

5) Stemming

Converting the tokenized words into their original form is called steaming [60]. All unwanted computations of words were removed from the list using steaming. For example, trolls, trolling, and trolled were converted into 'troll'.

6) Lemmatization

Lemmatization is called merging the words into a single related word using synonyms [59]. In this step, all words with synonyms relation were merged into one word, and others were removed from the list. Steaming is a simple process, but Lemmatization is often more useful than steaming for morphological analysis of the words.

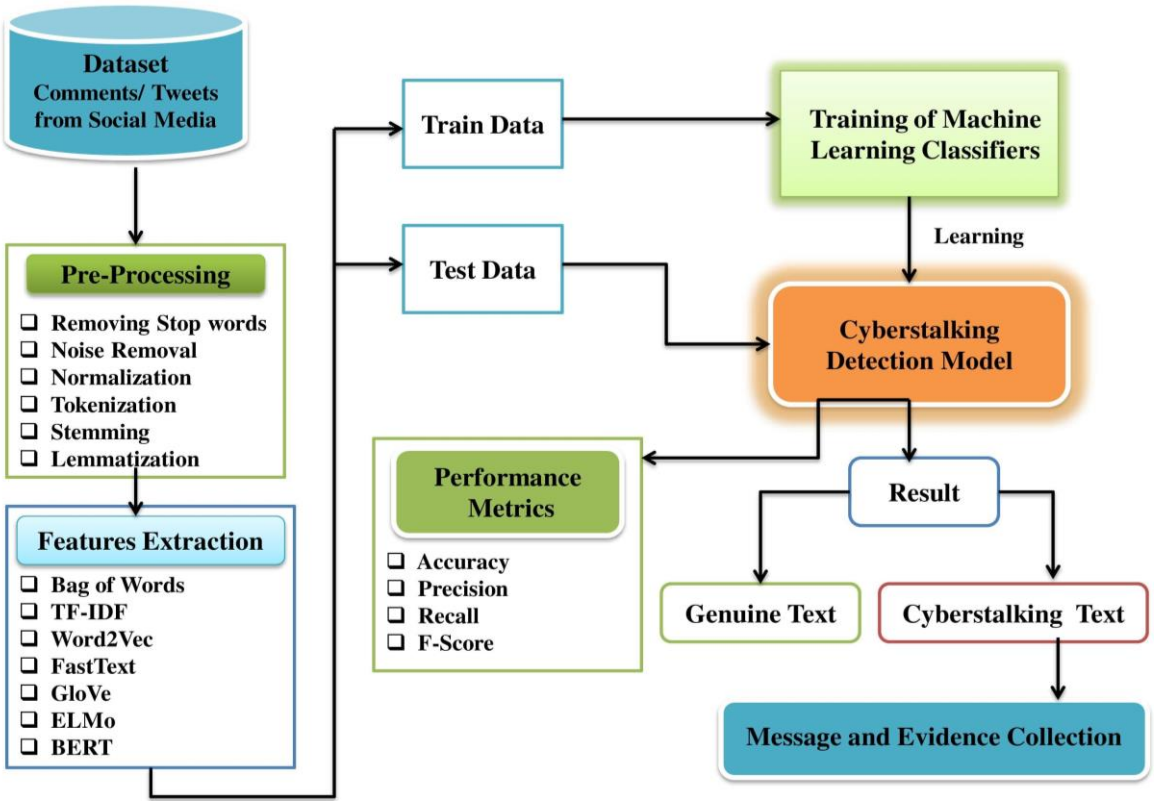

Figure 2. Proposed machine learning framework for cyberstalking detection.

C. Feature Extraction

In the wake of playing out the pre-processing tasks, the dataset was organized as required and prepared for features extraction strategies. Feature extraction techniques often influence the performance of the detection model. Features extraction is needed because the ML algorithms can not recognize information as text structure for classifications. In this stage, cleaned text data was changed into numbers, and the feature vectors and word references were arranged. Feature extraction techniques make a feature vector based on a gathering of predefined keywords utilizing registering the weights of the words. Various methods, namely filtration, fusion, mapping, and clustering are used for feature extraction. The filtration method is based on the word frequency, information gain, and mutual information, while fusion methods use the integration of specific classifiers for feature extraction. Mapping methods are widely used for features extraction, which uses the LSI (latent semantic index) and PCA (Principal Component Analysis) techniques. Clustering techniques consider the fundamental similarity of text to cluster text features. In the literature, many researchers have used several feature extraction techniques to enhance the performance of the detection model. BOW [61], TF-IDF [62], Word2Vec [63], GloVe [64], FastText [65], ELMo [66], BERT [67], ALBERT [68], ELECTRA [69], GPT-2 [70], XL-NET [71], RoBERTa [72], SBERT [73], Doc2VEC [74], InferSent [75], and Universal Sentence Encoder [76] are some popular examples of feature extraction methods. These methods are applied in the word-level, sentence-level, and n-gram levels for features extraction. In this paper, the proposed framework applied BOW, TF-IDF, Word2Vec, FastText, GloVe, ELMo, and BERT model separately with word-level for feature extractions to measure its effect in cyberstalking detection.

1) Bag of Word (BOW)

The BOW is a notable and least complex feature extraction process that portrays words inside a document and addresses the text into numbers. BOW counts the

frequencies of words as a feature in the text and places those words into a bag. The BOW does not have the positions, syntax, and design of words. Bag of Words utilizes a jargon of known words and measures available terms for features extraction. Each word includes as a feature in BOW and is given comparable importance [77]. Bag of words collects the data, designs a vocabulary, and finally scores the words in each document to create the document vectors. If the feature occurs in the input data, then feature frequency is represented by value 1; otherwise, 0 shows the feature frequency. Bag of Word may perform better than Word Embedding features extraction in the medium size dataset and domain-specific context. After completing the basic pre-processing tasks, this paper implemented Bag of Words in the word level.

2) TF-IDF

TF-IDF, known as Term Frequency-Inverse Document Frequency, is a well-recognized, perceived accurate computation that can determine the importance of any word of reports in an assortment of the corpus. In TF-IDF, the routinely occurring words are given more importance because frequently occurring words are more useful for the classification [78]. Term Frequency (TF) of any term is determined based on the quantity of incidence in the document to the complete words in that document. Inverse Document Frequency (IDF) is utilized to determine the significance of any term in the document. In this paper, we carried out the TF-IDF at the word level. The equation (1) is used to calculate the feature vector in the TF-IDF.

$$TF-IDF(T,D) = \frac{Count(T) \in D}{Number(W) \in D} \times Log\left(\frac{N}{(Occurance(T) \in N)+1}\right) \quad (1)$$

where:

$$\left.\begin{array}{l} \text{Count(T)} \in \text{D} = \text{Number of times word T} \\ \text{appears in a document "D"} \\ \text{Number(W)} \in \text{D} = \text{Total number of words in} \\ \text{the document "D"} \end{array}\right\} \rightarrow \textit{R}\text{epresents the Term Frequency}$$

Occurrence(T) ∈ N = {Total occurrence of Word "T" in total documents} → Represents the Document Frequency

N= Total Documents

3) Word2vec

Word2Vec is a predictive embedding model technique published in 2013 by Tomas Mikolov at Google. Word2vec is usually used to extract relatedness across words or items like semantic relatedness, equivalent identification, idea arrangement, selection inclinations, and similarity. Word2vec technique uses words as input data from a large corpus of text, and after learning the relation, it produces output as vector representation [79]. Word2Vec utilizes two unique strategies, Continuous Bag-of-Words (CBOW) and SkipGram, to work out the vector portrayals. In CBOW, the most probable word in the given context is predicted. CBOW model initially makes a vocabulary from the corpus and creates the context and target generator. After that, the model is created and trained to produce the vector representation of the word. The Skip-gram model predicts the situation utilizing a given word. SkipGram model uses the reverse approach of CBOW. Initially, it creates the vocabulary based on the given corpus and builds a SkipGram generator using target, context, and relevancy. Finally, SkipGram model architecture is designed and trained to obtain the vectors from words. In this paper, we carried out CBOW and SkipGram, the two models of Word2Vec, in word-level.

4) GloVe

The GloVe (Global Vectors) is an unsupervised embedding technique introduced by Stanford University in 2014. The GloVe is a count and co-occurrence-based model that finds the co-occur words in the corpus and creates a co-occurrence matrix to learn. GloVe executes the training on accumulated worldwide word-to-word co-event statistics [80]. The GloVe can perform parallel implementation and is easier to train on large data sizes. GloVe combines the benefits of the skip-gram model of word2vec, but like Word2Vec, GloVe cannot encode those unknown words that are not in the vocabulary. This paper used a pre-trained GloVe model in word-level for feature extraction. The mathematical equation (2) represents the GloVe vector model [80].

$$FV\_Glove = \sum_{i,j=1}^{VS} f(N_{ij})\ (W_i^t\ \widetilde{W}_j\ +\ B_i + \tilde{B}_j - Log(N_{ij}))^2 \quad (2)$$

where: VS is the vocabulary size, and $f(N_{ij})$ is the weight function. N_{ij} represents the frequency of word W_i appears with word W_j. B_i and B_j are biases of the word W_i and W_j.

5) FastText

FastText is an embedding technique developed by Facebook in 2015 for vector representation of words and sentences. FastText is an extension of Word2Vec and supports the method of skip-gram and CBOW (Continuous Word of Bag). FastText provide pre-trained models in 157 languages, and it also ropes on character level n-gram. FastText does not learn directly from words. First, it breaks the terms into several sub-words, represents them as n-gram of characters, and then feeds them into the neural network to obtain the vector representation [81]. FastText can obtain the vectors of those unknown words that are not in the vocabulary, which is the main advantage of this method. In this paper, we implemented CBOW and Skip-gram, both models of FastText in word-level.

6) ELMo

ELMo stands for "Embedding from Language Model," which is discovered by AllenNLP in the year 2018. ELMo is a prediction-based contextual embedding that utilizes a 2-layer bi-directional deep LSTM network to generate vector representation of words [82]. ELMo considers words inside which sets they have been used rather than reference words with their vector structure. ELMo calculates embeddings for a word using the entire sentence containing that word. ELMo embeddings can capture the context of the word and can produce different embeddings for a similar word utilized in an alternate context in various sentences. ELMo has character-based

embedding capacity, and it is also capable of creating the vector for out of vocabulary words. Using the bidirectional LSTM, ELMo can predict the next and previous probable word in the given sentence.

7) BERT

BERT is represented as "Bidirectional Encoder Representations from Transformer," an advanced prediction-based contextual embedding model developed by the Google team in 2018. BERT uses deep bidirectional layers of transformers encoders and generates two different word vectors for a different context. By utilizing transfer, BERT creates a contextual relation between words of any sentence to create the vector representation of the word. BERT uses 12, 24 transformers, 768, 1024 hidden layer size 12, 16 attention heads for the base and large model, respectively. BERT takes sentences as input and uses CLS, SEP, and MASK tokens for fine-tuning and specific tasks in training [83]. CLS is a starting token used as a classification token for conjunction with a softmax layer in classification tasks. SEP is a sequence delimiter used as a separation token for sequence-pair jobs at pre-training, while MASK token is used for a masked word in pre-training time. BERT creates the vector representation of a word that is dynamically informed by the words around them.

D. Machine Learning Classifiers

In this phase, Machine learning algorithms were trained and tested using training and testing datasets. In this paper, Logistic Regression (LR), Support Vector Machines (SVM), Random Forest (RF), Decision Tree (DT), k-Nearest Neighbor (KNN), and Naive Bayes (NB), machine learning classifiers were applied with each feature extraction technique for classification and measuring the effect and performance. Logistic regression creates the separate hyper-plane between datasets and takes the features as input to provide the results as probability [84]. In K-Nearest Neighbor is lazy and instance learning which classified the new sample based on the distance from its neighbor [84]. Naïve Bayes is more efficient and straightforward, predicting the result using the probability of any object using the Bayes Theorem [85]. Decision Trees create nodes and leaves as tree structures to take and represent the decision [86]. Random Forest is an enhanced form of a DT classifier that uses multiple decision trees [87]. Decision Tree and random forest algorithms performed as expected, but both take more time to train and predict the results. SVM computes the distance between the line and support vector and differentiate the classes individually in n-dimensional to provide more accurate result [88].

E. Cyberstalking Detection

After training and testing the classifiers based on the dataset, classifiers were used for classification tasks in unlabeled data. In this phase, new unlabeled datasets containing posts/tweets from social media were classified into genuine posts or cyberstalking posts based on the prediction and prediction probability of trained machine learning classifiers. The detection model will also support cyberstalker identification, domain Identification, and header Identification. After classifying the unlabeled dataset or any text, all cyberstalking messages, facts, and stalker information were collected and saved to the database as evidence and documentation.

F. Performance Metrics

Performance metrics of classifiers with each feature extraction method were measured with the support of the confusion matrix. Metrics used to monitor and measure the performance of a model during training and testing time are called performance metrics. In this paper, several parameters of performance metrics such as accuracy, precision, f-score, and recall were used to measure the performance of cyberstalking detection model and the effect of feature extraction techniques. Accuracy shows the total no of correct predictions by the classifiers. Precision shows how many of the actual positive cases were able to predict correctly. Recall explain the sensitivity and calculate the proportion of true positive prediction to total Positive. F-Score represents the harmonic average between precision and recall. Accuracy, precision, recall, and f-score can be determined utilizing the following mathematical representations.

$$Accuracy = \frac{TP+TN}{TP+FP+FN+TN} \quad (3)$$

$$Precision = \frac{TP}{TP+FP} \quad (4)$$

$$Recall = \frac{TP}{TP+FN} \quad (5)$$

$$F-Score = \frac{2\times Precision\times Recall}{Precision+Recall} \quad (6)$$

where: TP=True Positive that shows the total number of correctly detected cyberstalking text. TN=True Negative, which explains the total number of correctly detected non-cyberstalking text. FP=False Positive, which represents the total number of incorrectly detected cyberstalking text. FN=False Negative refers to the total wrongly detected non-cyberstalking text.

IV. Experimental Results

This section discusses experimental work and results to measure the effect of feature extraction techniques on the proposed machine learning framework. The experiment used python language with Scikit Learn and other machine learning library packages to implement the proposed framework. In the first experiment, the BOW method was applied for feature extraction, and the model was tested using the different machine learning classifiers. In Table I, the performance of algorithms with Bag of Words (BOW) is shown. As per the experimental result shown in Table I and Fig. 3, it was found that logistic regression produced better accuracy of 95.7%, a better f-score of 97.3%, and better precision of 97.9%. Naïve Bayes provided better recall of 98.3% than other machine learning classifiers. The DT and SVM classifiers obtained 95.5% and 95.3% accuracy, respectively. The performance of LR, DT, and SVM classifiers was almost similar in accuracy, precision, and f-score, while all classifiers obtained almost similar and better recall values. The KNN classifier obtained the lowest accuracy of

87.5%, precision of 88.1%, f-score of 92.5%, and recall of 97.4%.

TABLE I. PERFORMANCE METRICS OF MACHINE LEARNING ALGORITHMS WITH BAG OF WORDS

ML Algorithm	Accuracy	Precision	Recall	F-Score
LR	**0.9568**	**0.9785**	0.9670	**0.9727**
Decision Tree	0.9548	0.9766	0.9665	0.9715
SVM	0.9529	0.9723	0.9684	0.9703
RF	0.9334	0.9443	0.9739	0.9588
Naive Bayes	0.9056	0.9063	**0.9830**	0.9431
KNN	0.8751	0.8812	0.9746	0.9255

In the second experiment, the same machine learning algorithms were applied on the same dataset with TF-IDF methods for features extraction. Table II and Fig. 4 show the performance of algorithms with TF-IDF. The experimental results found that the support vector machine produced better accuracy of 95.2%, a better f-score of 97%, and better precision of 97.2%. Naïve Bayes provided better recall (99.8%) than other machine learning classifiers. Performance of SVM, DT, LR, and RF classifiers was almost similar and better in terms of accuracy, precision, and f-score, while all classifiers provided better and almost identical recall. The KNN and NB classifiers obtained the lowest accuracy of 85.3% and 82.5%, respectively. The KNN and NB classifiers also gave the most inadequate precision and f-score. Both feature extraction methods, BOW and TF-IDF, achieved better and satisfactory results. Based on the experimental results, LR, SVM, and DT classifiers outperformed for both method BOW and TF-IDF.

TABLE II. PERFORMANCE METRICS OF MACHINE LEARNING ALGORITHMS WITH TF-IDF

ML Algorithm	Accuracy	Precision	Recall	F-Score
SVM	**0.9521**	**0.9723**	0.9684	**0.9703**
Decision Tree	0.9470	0.9676	0.9657	0.9667
LR	0.9380	0.9465	0.9773	0.9617
RF	0.9356	0.9451	0.9758	0.9602
KNN	0.8533	0.8582	0.9773	0.9139
Naive Bayes	0.8247	0.8206	**0.9980**	0.9006

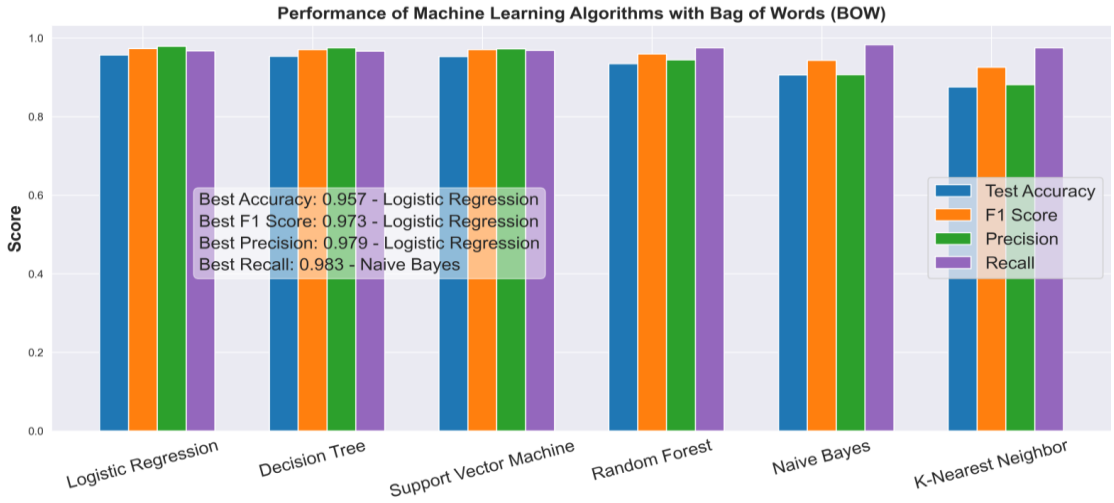

Figure 3. Classification summary of algorithms with Bag of Words (BOW).

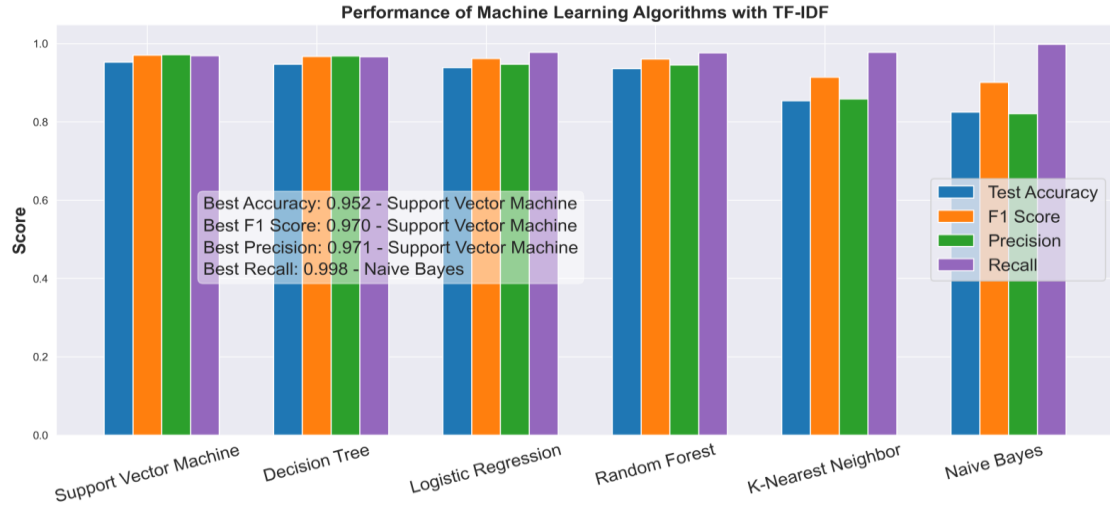

Figure 4. Classification summary of algorithms with TF-IDF.

The third experiment used the Word2Vec word embedding method with CBOW and SkipGram models for features extraction on the same dataset. Table III, Fig. 5 and Fig. 6 show the performance of algorithms with the

CBOW and SkipGram models of Word2vec. As per the experimental result of the CBOW model of Word2Vec, it was found that random forest achieved the highest accuracy of 82.6%, highest f-score of 90%, and highest recall of 97.7%. Naïve Bayes achieved the highest precision of 88% than other machine learning classifiers. The LR and SVM classifiers obtained 82.2% accuracy, and performance was very near to the random forest classifier. KNN, NB, and DT classifiers provided the lowest accuracy of 80.8%, 75.9%, and 75%, respectively. In the case of the SkipGram model of the Word2Vec method, logistic regression achieved the highest accuracy of 85% and the highest f-score of 91%. The random forest classifier achieved the highest recall of 97%. Naïve Bayes again provided better precision of 88.1% than other classifiers. SVM and RF both classifiers obtained an accuracy of 84.8% and 84.3%, respectively, which was near the LR classifier's accuracy. NB and DT classifiers again failed to provide better accuracy and obtained the lowest accuracy of 79% and 76.7, respectively. As per the results, CBOW and Skip-Gram, models of Word2Vec, gave almost similar results, although the Skip-Gram model of Word2Vec performed better and enhanced the performance of machine learning classifiers. Overall performance of RF, LR, and SVM classifiers was better with both models of Word2Vec. The overall performance of the Word2Vec feature extraction method was not good as expected in comparison to BOW and TF-IDF. However, the recall value provided by Word2Vec was satisfactory and almost near to BOW and TF-IDF methods.

TABLE III. PERFORMANCE OF MACHINE LEARNING ALGORITHMS WITH WORD2VEC

Model: Word2Vec With CBOW Model Vector Size 100				
ML Algorithm	Accuracy	Precision	Recall	F-Score
RF	**0.8255**	0.8323	**0.9775**	**0.8991**
LR	0.8218	0.8429	0.9537	0.8949
SVM	0.8216	0.8377	0.9620	0.8956
KNN	0.8078	0.8591	0.9071	0.8824
Naive Bayes	0.7588	**0.8798**	0.8068	0.8417
Decision Tree	0.7502	0.8480	0.8355	0.8417
Model: Word2Vec With Skip-Gram Model Vector Size 100				
ML Algorithm	Accuracy	Precision	Recall	F-Score
LR	**0.8503**	0.8704	0.9537	**0.9102**
SVM	0.8474	0.8644	0.9583	0.9089
RF	0.8427	0.8523	**0.9704**	0.9075
KNN	0.8289	0.8774	0.9123	0.8945
Naive Bayes	0.7902	**0.8813**	0.8506	0.8657
Decision Tree	0.7671	0.8543	0.8525	0.8534

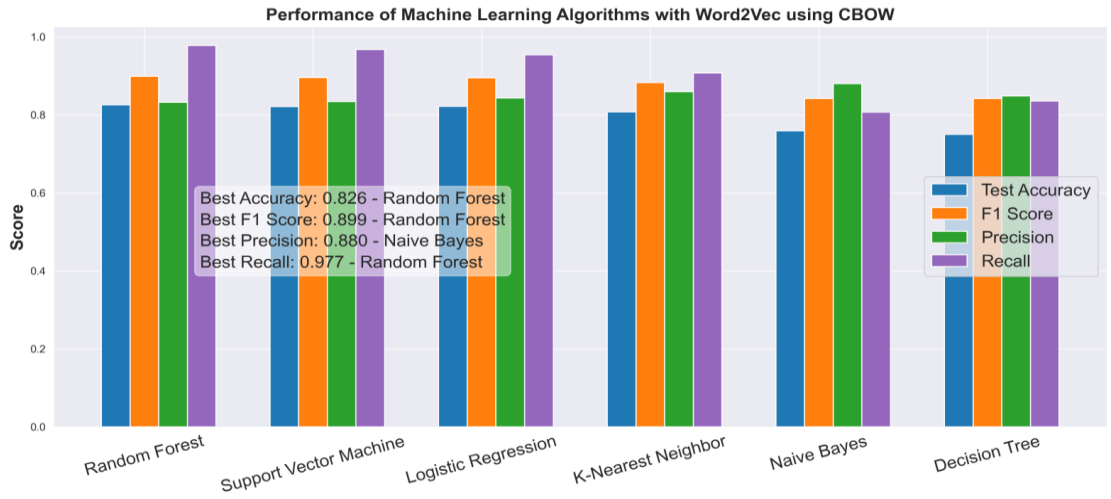

Figure 5. Classification summary of algorithms with Word2Vec using CBOW model.

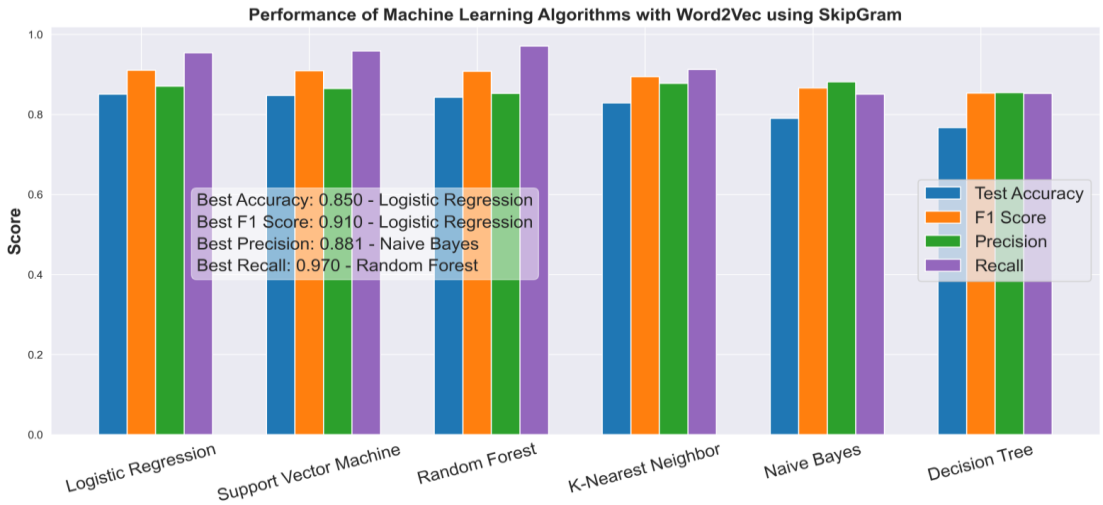

Figure 6. Classification summary of algorithms with Word2Vec using skip-gram model.

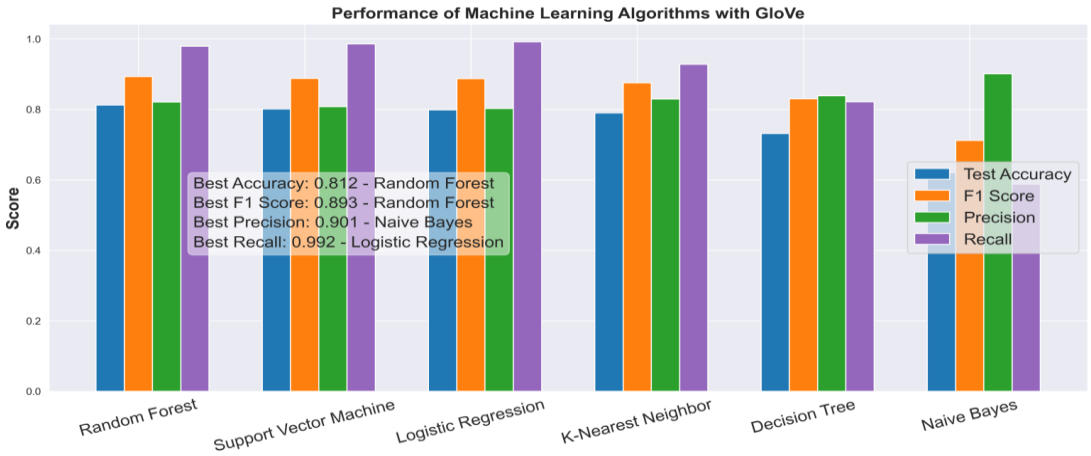

Figure 7. Classification summary of algorithms with GloVe.

The fourth experiment utilized the GloVe word embedding method with a vector size of 300 for features extraction. Again, the same dataset and classifiers were used to evaluate the model. Table IV and Fig. 7 show the performance of algorithms with GloVe. As per the experimental result, it was found that random forest achieved the highest accuracy of 81.2% and the highest f-score of 89.2%. Naïve Bayes provided better precision of 90.1%, while logistic regression obtained the highest recall of 99.2%, compared to other machine learning classifiers. Support vector machine provided an accuracy of 80.2% near the accuracy of random forest. The NB and DT both classifiers obtained the lowest accuracy, recall, and f-score while RF, SVM, and LR were again best performer classifiers. Overall performance of the GloVe model was not as per expectation compared to BOW and TF-IDF. However, the GloVe model provided outstanding recall values like BOW and TF-IDF methods.

TABLE IV. PERFORMANCE MACHINE LEARNING ALGORITHMS WITH GLOVE

Model: pre-trained GloVe model (GloVe.840B.300d.txt) with vector size=300				
ML Algorithm	Accuracy	Precision	Recall	F-Score
RF	**0.8123**	0.8204	0.9789	**0.8927**
SVM	0.8016	0.8076	0.9860	0.8879
LR	0.7986	0.8023	**0.9918**	0.8870
KNN	0.7897	0.8291	0.9274	0.8755
Decision Tree	0.7311	0.8382	0.8213	0.8296
Naive Bayes	0.6199	**0.9010**	0.5878	0.7114

In the fifth experiment, the FastText word embedding method with CBOW and SkipGram models were utilized for features extraction on the same dataset. Table V, Fig. 8, and Fig. 9 show the performance of algorithms with FastText. As per the experimental results of the CBOW model of FastText, it was found that the random forest classifier achieved the maximum accuracy of 82.2%, the maximum f-score of 90%, and the maximum recall of 98%. K-Nearest Neighbor provided the highest precision of 85% than other machine learning classifiers. The LR and SVM classifiers obtained 81% and 80.7% accuracy, respectively, while the DT and NB classifiers provided the lowest accuracy of 74.4% and 74.5%, respectively. In the case of the SkipGram model of FastText, Random Forest provided the highest accuracy of 85.7%, highest f-score of 91.5%, and highest recall of 96.7%. Naïve Bayes again provided better precision of 91.1% than other classifiers. The LR and SVM classifiers achieved an accuracy of 85.5% and 85.4%, respectively, which was very near to the accuracy of the random forest classifier. Again, the NB and DT classifiers failed to provide better accuracy and obtained the lowest accuracy of 78.7% and 79%, respectively. As per the results, CBOW and Skip-Gram, both FastText gave almost similar results. However, the Skip-Gram model of FastText performed better and enhanced the performance of machine learning classifiers. The RF, SVM, and LR were again best performer classifiers. Overall performance of FastText model was better than GloVe and near to word2vec but not enough compared to BOW and TF-IDF model. However, the recall value provided by the FastText model was as per expectation like Word2Vec and GloVe model and near to BOW and TF-IDF model.

TABLE V. PERFORMANCE MACHINE LEARNING ALGORITHMS WITH FASTTEXT

Model: FastText with CBOW Model Vector Size 100				
ML Algorithm	Accuracy	Precision	Recall	F-Score
Random Forest	**0.8221**	0.8281	**0.9796**	**0.8975**
LR	0.8096	0.8277	0.9605	0.8891
DVM	0.8069	0.8168	0.9759	0.8893
KNN	0.7929	**0.8513**	0.8960	0.8731
Decision Tree	0.7438	0.8475	0.8265	0.8369
Naive Bayes	0.74479	0.8470	0.8287	0.8377
Model: FastText with Skip-Gram Model Vector Size 100				
ML Algorithm	Accuracy	Precision	Recall	F-Score
LR	0.8554	0.8716	0.9596	0.9135
SVM	0.8542	0.8671	0.9645	0.9132
Random Forest	**0.8567**	0.8681	**0.9667**	**0.9147**
KNN	0.8218	0.8807	0.8975	0.8890
Decision Tree	0.7892	0.8731	0.8599	0.8664
Naive Bayes	0.7867	**0.9115**	0.8105	0.8580

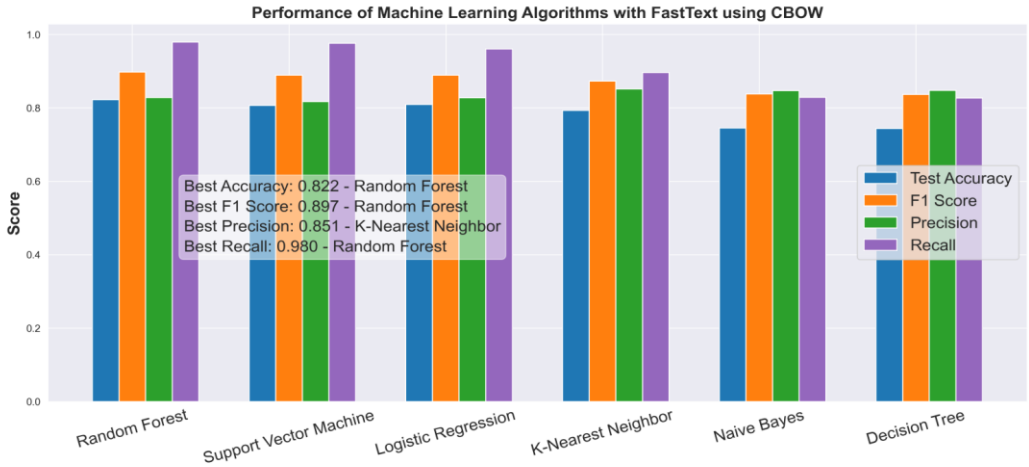

Figure 8. Classification summary of algorithms with FastText using CBOW model.

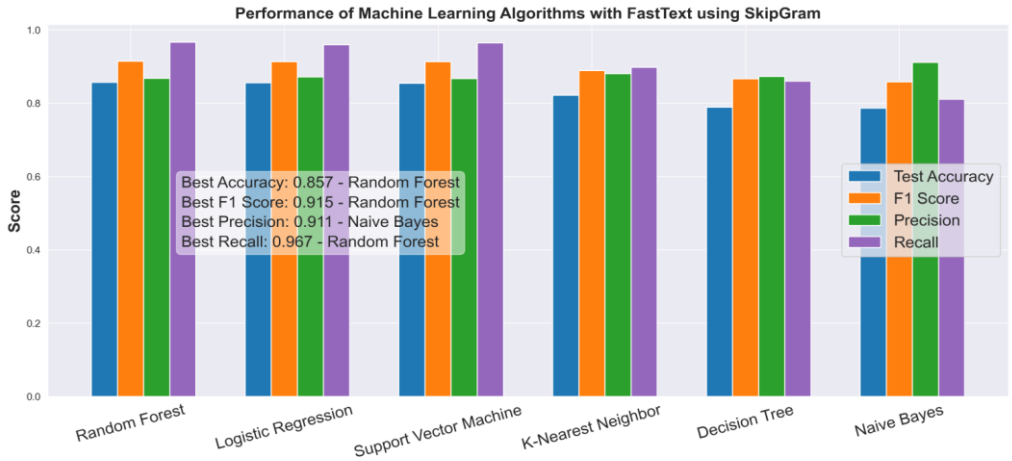

Figure 9. Classification summary of algorithms with FastText using Skip-Gram model.

TABLE VI. PERFORMANCE MACHINE LEARNING ALGORITHMS WITH ELMO

pre-trained ELMo model (https://tfhub.dev/google/elmo/2				
ML Algorithm	Accuracy	Precision	Recall	F-Score
LR	**0.9051**	0.9279	0.9551	**0.9413**
SVM	0.9024	**0.9318**	0.9467	0.9392
Random Forest	0.8492	0.8532	0.9790	0.9118
KNN	0.8333	0.8345	**0.9862**	0.9040
Decision Tree	0.7807	0.8661	0.8570	0.8616
Naive Bayes	0.7116	0.8566	0.7660	0.8088

TABLE VII. PERFORMANCE MACHINE LEARNING ALGORITHMS WITH BERT

Model: pre-trained BERT base model (https://bert-as-service.readthedocs.io/en/latest/section/get-start.html#download-a-pre-trained-bert-model)				
ML Algorithm	Accuracy	Precision	Recall	F-Score
LR	**0.9093**	0.9370	0.9499	**0.9434**
SVM	0.9071	**0.9379**	0.9460	0.9419
Random Forest	0.8687	0.8757	**0.9731**	0.9219
KNN	0.8624	0.8709	0.9711	0.9183
Decision Tree	0.7988	0.8775	0.8686	0.8730
Naive Bayes	0.7931	0.9363	0.7941	0.8594

In the sixth experiment, the pre-trained ELMo model (https://tfhub.dev/google/elmo/2) was utilized for feature extraction. Again, the same dataset and classifiers were utilized to evaluate the model. In Table VI and Fig. 10, the performances of algorithms with ELMo are described. As per the experimental result, it was found that logistic regression produced the highest accuracy of 90.5% and the highest f-score of 94.1%. SVM provided better precision of 93.2%, while KNN obtained the highest recall of 98.6% compared to other machine learning classifiers. SVM provided an accuracy of 90.2% near the accuracy of logistic regression. The NB and DT both classifiers obtained the lowest accuracy, recall, and f-score while SVM and LR were again best performer classifiers. Overall performance of the ELMo model was higher than Word2Vec, GloVe, and FastText but less than BOW and TF-IDF.

In the seventh and last experiment, the pre-trained BERT base model was utilized for feature extraction in the same dataset and classifiers to evaluate the detection model. In Table VII and Fig. 11, the performances of algorithms with BERT are presented. As per the experimental result, it was found that logistic regression produced the highest accuracy of 90.9% and the highest f-score of 94.3%. SVM provided better precision of 93.8%, while random forest obtained the highest recall of 97.3%, compared to other machine learning classifiers. SVM provided an accuracy of 90.7% near the accuracy of logistic regression. The NB and DT both classifiers obtained the lowest accuracy, recall, and f-score while SVM and LR were again best performer classifiers.

Overall performance of the BERT model was very near but a little higher than ELMo, much better than Word2Vec, GloVe, and FastText, but less than BOW and TF-IDF.

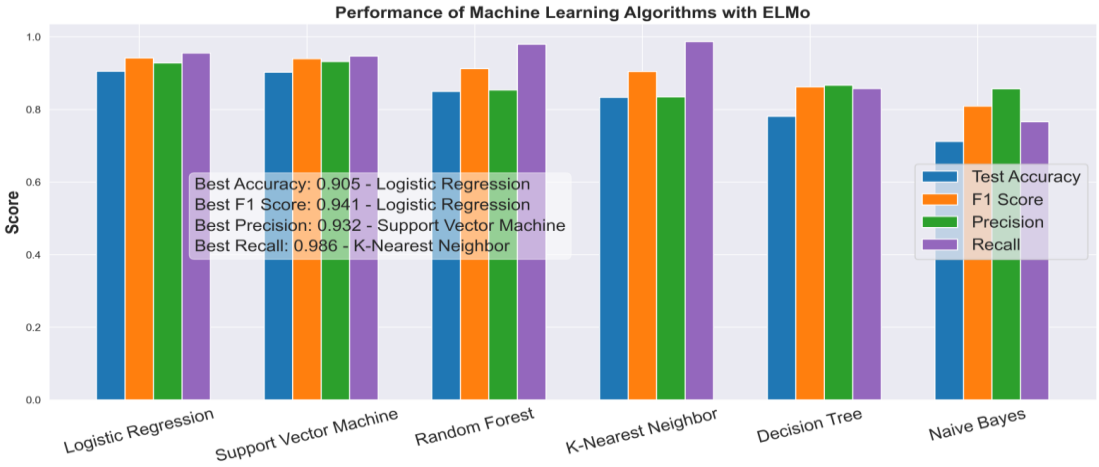

Figure 10. Classification summary of algorithms with ELMo model.

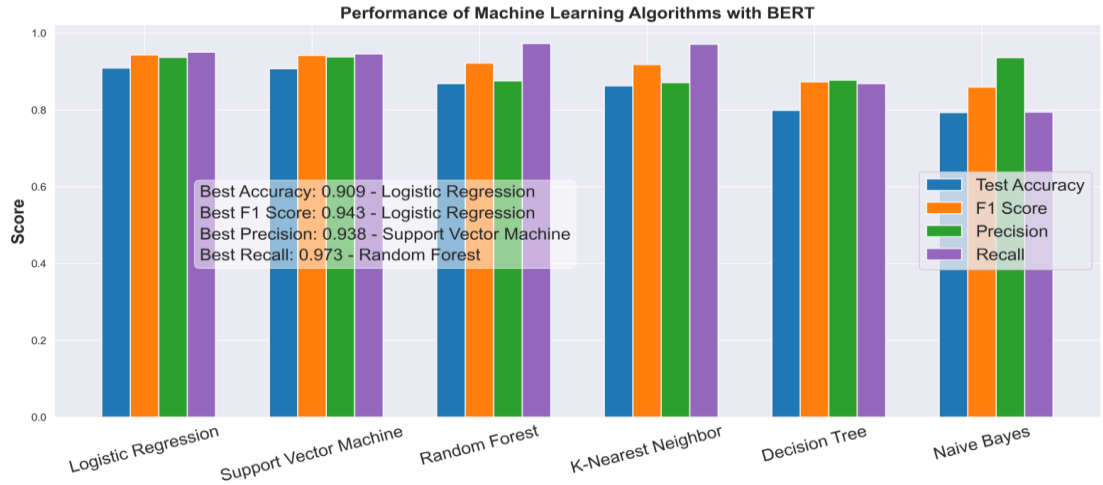

Figure 11. Classification summary of algorithms with BERT model.

TABLE VIII. PERFORMANCE OF FEATURE EXTRACTION METHODS

Maximum Score obtained by ML Classifiers with Feature Extraction Methods				
FE Methods	Accuracy	Precision	Recall	F-Score
BOW	**0.9568**	**0.9785**	0.9830	**0.9727**
TF-IDF	0.9521	0.9723	**0.9980**	0.9703
Word2Vec-CBOW	0.8255	0.8798	0.9775	0.8991
Word2Vec-SkipGram	0.8503	0.8813	0.9703	0.9102
GloVe	0.8123	0.9010	0.9918	0.8927
FastText-CBOW	0.8221	0.8513	0.9796	0.8975
FastText-SkipGram	0.8567	0.9115	0.9667	0.9147
ELMO	0.9052	0.9318	0.9862	0.9413
BERT	0.9093	0.9379	0.9731	0.9434

The comparative effect of various feature extraction methods is explained in Table VIII, Fig. 12, and Fig. 13. The performance (in terms of accuracy) of machine learning classifiers with each feature extraction method is represented in Fig. 12. The highest accuracy, precision, recall, and f-score achieved by each feature extraction method is explained in Fig. 13. Experimental results found that BOW and TF-IDF outperformed other word embedding and enhanced the performance of machine learning classifiers for cyberstalking detection in the case of small and medium-size datasets. BERT and ELMo performed well and better than Word2Vec, GloVe, and FastText. Although Word2Vec, GloVe, and FastText provided almost similar results and were not performed as expected in the case of small and medium datasets. ML classifiers achieved the highest accuracy of 95.7% (LR) and 95.2% (SVM) with BOW and TF-IDF, respectively. The highest precision of 97.9% and the highest f-score of 97.3% was achieved by BOW, while TF-IDF achieved an outstanding recall of 99.8%. Overall experimental results show that BOW and TF-IDF are better than other word embedding-based and contextual-based methods (Word2Vec, FastText, GloVe, ELMo, and BERT) in accuracy, precision, and f-score.

However, all the applied feature extraction models achieved better recall. Word2Vec, FastText, GloVe, ELMo, and BERT embedding methods may be better for large corpus and valuable for other applications. The performance of these word embedding methods is also dependent on vector size and other parameters to enhance the performance of classifiers. The experiment in this paper used a small vector size (100-300) to train and test the word embedding models.

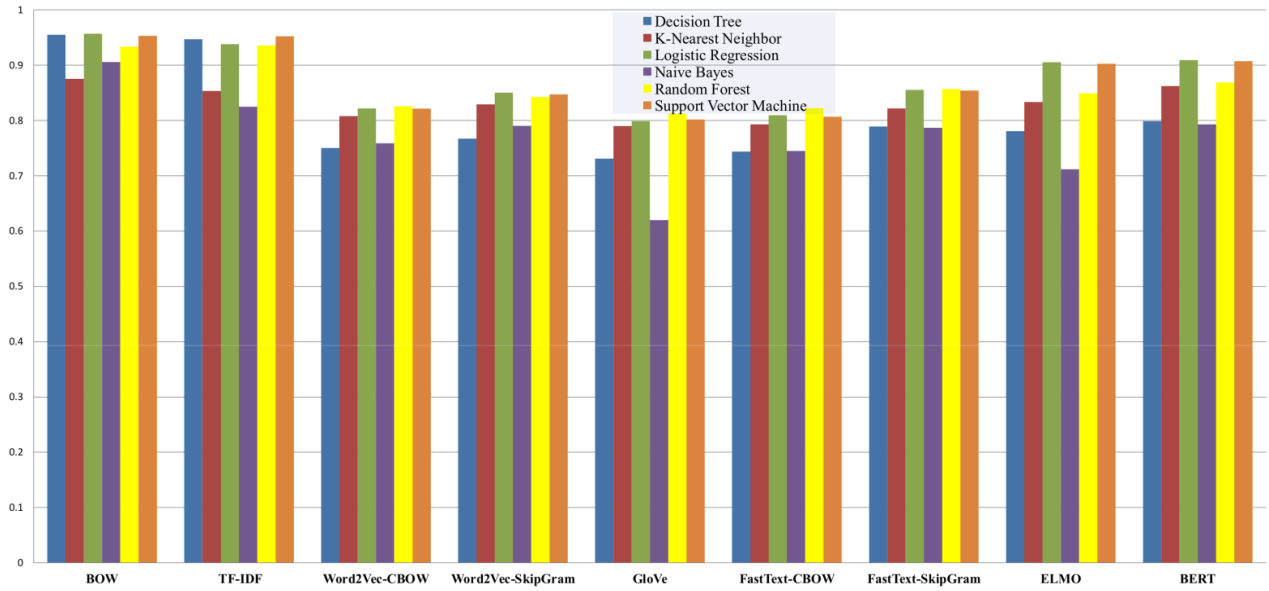

Figure 12. Accuracy of algorithms with different feature extraction methods.

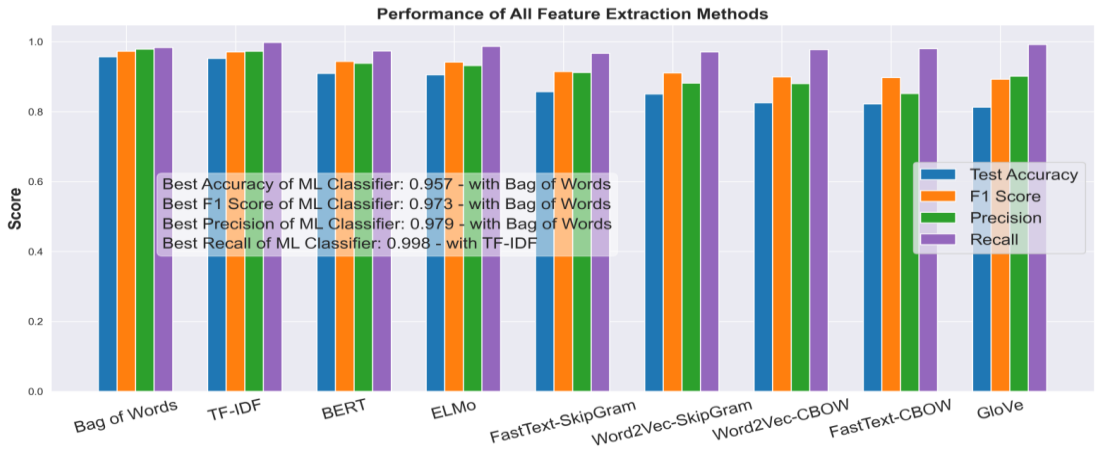

Figure 13. Performance of all feature extraction methods.

V. Conclusion and Future Work

Trendy uses of internet applications are making habitual and criminal society in the virtual world. Cyberstalkers and other cybercriminals are creating the negative and fear face of social media and other internet applications. This paper proposed a machine learning framework for cyberstalking detection by implementing various popular features extraction methods with six machine learning classifiers. Experimental works were conducted on medium-size mixed datasets to measure the effect of feature extraction techniques for enhancing the performance of classifiers. The proposed cyberstalking detection model provided the highest accuracy of 95.7% for logistic regression with BOW and 95.2% for SVM with TF-IDF. CBOW model of Word2Vec obtained maximum accuracy of 82.6% for the random forest, while the SkipGram model of Word2Vec obtained an accuracy of 85% for logistic regression. GloVe model provided maximum accuracy of 81.2% for the random forest. FastText method with CBOW model received maximum accuracy of 82.2% for random forest while FastText with SkipGram model achieved maximum accuracy of 85.7% for the random forest. ELMo model provided maximum accuracy of 90.5% for the logistic regression. BERT model provided maximum accuracy of 90.9% for the logistic regression. Experimental results show that in terms of accuracy, precision, and f-score, the performance of the basic feature extraction method (BOW, TF-IDF) was better than advanced feature extraction methods (Word2Vec, GloVe, FastText, ELMo, and BERT). Advanced word embedding-based feature extraction methods (Word2Vec, FastText, and GloVe) did not produce better outcomes as per expectation. BERT and ELMo performed very well compared to word2vec, GloVe, and FastText. However, all implemented feature extraction methods obtained almost similar and better results in terms of recall. SkipGram model performed better than the CBOW model with Word2Vec and FastText feature extraction methods. The Word2Vec, GloVe, FastText, ELMO, and BERT models can achieve more accurate and better results in the case of a large dataset. SVM, RF, and LR classifiers outperformed with all implemented feature extraction

methods, and the results of these classifiers are almost similar. Experimental results show that the feature extraction methods affected the performance of the detection model. Its performance is also dependent on the size and distribution of the dataset, pre-processing tasks, and classifiers. However, there are no best feature extraction methods for all cases, and the selection of feature extraction methods should be according to the problem and datasets. The performance of basic word embedding, advanced word embedding, and language model-based feature extraction methods should be experimented with and compared on different aspects. Other comparison backgrounds are dataset size, text language, time complexity, and application context. Future work will include analyzing these basic and advanced feature extraction methods with machine learning and deep learning algorithms on small, medium, and large datasets for proper comparison. Additionally, future work will also be focused on utilizing the different basic and advanced feature extraction methods to develop an automatic cyberstalking detection system to detect cyberstalking in real-time. In addition, it also includes analyzing the performance of these feature extraction methods for different aspects of datasets with hybrid methods using machine learning, deep learning, and fuzzy logic algorithms.

CONFLICT OF INTEREST

The authors declare that there is no conflict of interest.

AUTHOR CONTRIBUTIONS

Arvind Kumar Gautam conducted the research work, collected data, and wrote the paper. Dr. Abhishek Bansal supervised the work and approved the final version.

REFERENCES

[1] (2020). IndiaSpend Website. [Online]. Available: https://www.indiaspend.com/1-in-10-indian-adolescents-faces-cyberbullying-half-dont-report-study/

[2] (2022). The Indian Express Website. [Online]. Available: https://indianexpress.com/article/cities/ahmedabad/5000-people-mainly-women-bullied-stalked-and-harassed-on-social-media-in-pandemic-year-ahmedabad-police-7430922/

[3] P. E. Mullen, M. Pathé, and R. Purcell, "Stalking: New constructions of human behavior," *Australian and New Zealand Journal of Psychiatry*, vol. 35, pp. 9-16, 2001.

[4] M. Baer, "Cyberstalking and the internet landscape we have constructed," *Virginia Journal of Law & Technology*, vol. 154, no. 15, pp. 153-227, 2020.

[5] J. L. Truman, "Examining intimate partner stalking and use of technology in stalking victimization," Ph.D. thesis, University of Central Florida Orlando, Florida, 2010.

[6] S. B. WinkelmAn, J. Oomen-Early, A. D. Walker, L. Chu, and A. Yick-Flanagan, "Exploring cyber harassment among women who use social media," *Universal Journal of Public Health*, vol. 3, no. 5, p. 194, 2015.

[7] A. K. Gautam and A. Bansal, "Performance analysis of supervised machine learning techniques for cyberstalking detection in social media," *Journal of Theoretical and Applied Information Technology*, vol. 100, no. 2, 2022.

[8] N. M. Zainudin, K. H. Zainal, N. A. Hasbullah, N. A. Wahab, and S. Ramli, "A review on cyberbullying in Malaysia from digital forensic perspective," in *Proc. International Conference on Information and Communication Technology*, 2016, pp. 246-250.

[9] E. Kritzinger, M. Bada, and J. R. C. Nurse, "A study into the cybersecurity awareness initiatives for school learners in South Africa and the UK," in *Proc. IFIP World Conference on Information Security Education*, 2017, pp. 110-120.

[10] A. K. Gautam and A. Bansal, "A review on cyberstalking detection using machine learning techniques: Current trends and future direction," *International Journal of Engineering Trends and Technology*, vol. 70, no. 3, pp. 95-107, 2022.

[11] A. Muhammad, "A systematic review of machine learning algorithms in cyberbullying detection: Future directions and challenges," *Journal of Information Security and Cybercrimes Research*, vol. 4, no. 1, pp. 1-26, 2021.

[12] A. Muneer and S. M. Fati, "A comparative analysis of machine learning techniques for cyberbullying detection on twitter," *Future Internet*, vol. 187, no. 12, 2020.

[13] V. Nahar, S. Unankard, X. Li, and C. Pang, "Sentiment analysis for effective detection of cyber bullying," in *Proc. 14th Asia-Pacific International Conference on Web Technologies and Applications*, April 2012.

[14] M. Dadvar, D. Trieschnigg, R. Ordelman, and F. D. Jong, "Improving cyberbullying detection with user context," in *Proc. European Conference on Information Retrieval*, 2013, pp. 693-696.

[15] W. Zhang, W. Xu, G. Chen, and J. Guo, "A feature extraction method based on word embedding for word similarity computing," in *Natural Language Processing and Chinese Computing, NLPCC, Communications in Computer and Information Science*, C. Zong, J. Y. Nie, D. Zhao, and Y. Feng, Eds., Springer, Berlin, Heidelberg, 2014, vol. 496.

[16] Z. Ghasem, I. Frommholz, and C. Maple, "Machine learning solutions for controlling cyberbullying and cyberstalking," *International Journal of Information Security*, vol. 6, no. 2, pp. 55-64, 2015.

[17] M. D. Capua, E. D. Nardo, and A. Petrosino, "Unsupervised cyber bullying detection in social networks," in *Proc. 23rd International Conference on Pattern Recognition*, Cancún Center, México, December 2016.

[18] S. Bhoir, T. Ghorpade, and V. Mane, "Comparative analysis of different word embedding models," in *Proc. International Conference on Advances in Computing, Communication and Control*, 2017, pp. 1-4.

[19] R. N. Waykole and A. D. Thakare, "A review of feature extraction methods for text classification," *International Journal of Advance Engineering and Research Development*, vol. 5, no. 4, 2018.

[20] J. H. Mounir, M. Nashaat, M. Ahmed, and E. A. Amer, "Social media cyberbullying detection using machine learning," *International Journal of Advanced Computer Science and Applications*, vol. 10, no. 5, 2019.

[21] R. Ahujaa, A. Chuga, S. Kohlia, S. Gupta, and P. Ahuja, "The impact of features extraction on the sentiment analysis," *Procedia Computer Science*, vol. 152, pp. 341-348, 2019.

[22] M. Al-Hashedi, L. K. Soon, and H. N. Goh, "Cyberbullying detection using deep learning and word embeddings: An empirical study," in *Proc. 2nd International Conference on Computational Intelligence and Intelligent Systems*, 2019, pp. 17-21

[23] S. Modha and P. Majumder, "An empirical evaluation of text representation schemes on multilingual social web to filter the textual aggression," arXiv preprint arXiv:1904.08770, 2019.

[24] M. Islam and S. Sharmin, "Cyberbullying detection on social networks using machine learning approaches," in *Proc. IEEE Asia-Pacific Conference on Computer Science and Data Engineering*, 2020.

[25] A. Muneer and S. M. Fati, "A comparative analysis of machine learning techniques for cyberbullying detection on twitter," *Future Internet*, vol. 187, no. 12, 2020.

[26] A. Khandelwal and N. Kumar, "A unified system for aggression identification in english code-mixed and uni-lingual texts," in *Proc. 7th ACM IKDD CoDS and 25th COMAD*, 2020, pp. 55-64.

[27] H. Park and K. Kyoung-jae, "Impact of word embedding methods on performance of sentiment analysis with machine learning techniques," *Journal of the Korea Society of Computer and Information*, vol. 25, no. 8, pp. 181-188, 2020.

[28] S. Thavareesan and S. Mahesan, "Word embedding-based part of speech tagging in Tamil texts," in *Proc. IEEE 15th International Conference on Industrial and Information Systems*, 2020.

[29] G. M. Barrientos, *et al*., "Machine learning techniques for the detection of inappropriate erotic content in text," *International*

Journal of Computational Intelligence Systems, vol. 13, no. 1, pp. 591-603, 2020.
[30] A. Asante and X. Feng, "Content-Based technical solution for cyberstalking detection," in *Proc. 3rd International Conference on Computer Communication and the Internet*, 2021, pp. 89-95.
[31] N. Dughyala, S. Potluri, S. Kj, and V. Pavithran, "Automating the detection of cyberstalking," in *Proc. Second International Conference on Electronics and Sustainable Communication Systems*, 2021, pp. 887-892.
[32] M. Tolba, S. Ouadfel, and S. Meshoul, "Hybrid ensemble approaches to online harassment detection in highly imbalanced data," *Expert Systems with Applications*, p. 175, 2021.
[33] H. S. Alatawi, A. M. Alhothali, and K. M. Moria, "Detecting white supremacist hate speech using domain specific word embedding with deep learning and BERT," *IEEE Access*, vol. 9, p. 10636, 2021.
[34] V. Jain, V. Kumar, V. Pal, and D. K. Vishwakarma, "Detection of cyberbullying on social media using machine learning," in *Proc. 5th International Conference on Computing Methodologies and Communication*, 2021, pp. 1091-1096.
[35] S. Pericherla and E. Ilavarasan, "Performance analysis of word embeddings for cyberbullying detection," in *Proc. IOP Conference Series: Materials Science and Engineering*, 2021.
[36] C. Raj, *et al*., "Cyberbullying detection: Hybrid models based on machine learning and natural language processing techniques," *Electronics*, vol. 22, no. 10, p. 2810, 2021.
[37] H. Zhong, *et al*., "Content-Driven detection of cyberbullying on the instagram social network," in *Proc. 25th Int. Jt. Conf. Artif. Intell*., July 2016, pp. 3952-3958.
[38] D. Chatzakou, *et al.*, "Detecting cyberbullying and cyberaggression in social media," *ACM Trans. Web (TWEB)*, vol. 13, no. 3, pp. 1-51, 2019.
[39] R. Zhao and K. Mao, "Cyberbullying detection based on semantic-enhanced marginalized denoising autoencoder," *IEEE Trans. Affect. Comput*., vol. 8, no. 3, pp. 328-339, 2017.
[40] M. F. López-Vizcaíno, F. J. Nóvoa, V. Carneiro, and F. Cacheda, "Early detection of cyberbullying on social media networks," *Future Gener. Comput. Syst.*, vol. 118, pp. 219-229, 2021.
[41] W. A. Prabowo and F. Azizah, "Sentiment analysis for detecting cyberbullying using TF-IDF and SVM," *Jurnal RESTI (Rekayasa Sistem Dan Teknologi Informasi)*, vol. 4, no. 6, pp. 1142-1148, 2020.
[42] H. Rosa, D. Matos, R. Ribeiro, L. Coheur, and J. P. Carvalho, "A deeper look at detecting cyberbullying in social networks," in *Proc. International Joint Conference on Neural Networks*, 2018, pp. 1-8.
[43] J. Bhagya and P. S. Deepthi, "Cyberbullying detection on social media using SVM," in *Inventive Systems and Control*, Singapore: Springer, 2021, pp. 17-27.
[44] N. Lu, G. Wu, Z. Zhang, Y. Zheng, Y. Ren, and K. K. R. Choo, "Cyberbullying detection in social media text based on character-level convolutional neural network with shortcuts," *Concurrency and Computation: Practice and Experience*, vol. 32, no. 23, 2020.
[45] S. Alsafari, S. Sadaoui, and M. Mouhoub, "Hate and offensive speech detection on arabic social media," *Online Soc. Netw. Media*, p. 100096, 2020.
[46] S. Agrawal and A. Awekar, "Deep learning for detecting cyberbullying across multiple social media platforms," in *Proc. Eur. Conf. Inf. Retr. ECIR 2018, in Advances in Information Retrieval, in Lecture Notes in Computer Science*, 2018, pp. 141-153.
[47] J. Zhang, T. Otomo, L. Li, and S. Nakajima, "Cyberbullying detection on twitter using multiple textual features," in *Proc. IEEE 10th Int. Conf. Aware.Sci. Technol*., Japan, 2019, pp. 1-6.
[48] B. A. Talpur and D. O'Sullivan, "Cyberbullying severity detection: A machine learning approach," *PLOS One*, vol. 15, no. 10, p. e0240924, 2020.
[49] B. A. Rachid, H. Azza, and H. H. B. Ghezala, "Classification of cyberbullying text in arabic," in *Proc. Int. Jt. Conf. Neural Netw*., UK, 2020, pp. 1-7.
[50] D. Soni and V. K. Singh, "See no evil, hear no evil: Audiovisual-textual cyberbullying detection," *Proc. ACM Hum.-Comput. Interact*, vol. 2, pp. 1-26, 2018.
[51] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep learning for hate speech detection in tweets," in *Proc. 26th Int. Conf. World Wide Web Companion*, 2017, pp. 759-760.
[52] (2020). Mendeley Cyberbullying datasets. [Online]. Available:https://data.mendeley.com/datasets/jf4pzyvnpj/1
[53] (2020). The Kaggle website-dataset. [Online]. Available: https://www.kaggle.com/mrmorj/hate-speech-and-offensive-language-dataset
[54] (2022). The Kaggle website-dataset. [Online]. Available: https://www.kaggle.com/andrewmvd/cyberbullying-classification
[55] (2021). The Kaggle website-dataset. [Online]. Available: https://www.kaggle.com/sanamps/toxiccommentclassification
[56] (2014). The Kaggle website-dataset. [Online]. Available: https://www.kaggle.com/c/detecting-insults-in-social-commentary/data
[57] (2019). The Kaggle website-dataset. [Online]. Available: https://www.kaggle.com/venky73/spam-mails-dataset
[58] S. Vijayarani, M. J. Ilamathi, and M. Nithya, "Pre-processing techniques for text mining-an overview," *International Journal of Computer Science & Communication Networks*, vol. 5, no. 1, pp. 7-16, 2015.
[59] (2019). Towards Data Science Website. All you need to know about text pre-processing for NLP and Machine Learning. [Online]. Available: https://towardsdatascience.com/all-you-need-to-know-about-text-preprocessing-for-nlp-and-machine-learning-bc1c5765ff67
[60] A. I. Kadhim, "An evaluation of pre-processing techniques for text classification," *International Journal of Computer Science and Information Security*, vol. 16, no. 6, pp. 22-32, 2018.
[61] Z. L. Chia, M. Ptaszynski, F. Masui, G. Leliwa, and M. Wroczynski, "Machine learning and feature engineering-based study into sarcasm and irony classification with application to cyberbullying detection," *Information Processing & Management*, vol. 58, no. 4, 2021.
[62] N. M. G. D. Purnamasari, M. A. Fauzi, and L. S. D. Indriati, "Cyberbullying identification in twitter using support vector machine and information gain based feature selection," *Indonesian Journal of Electrical Engineering and Computer Science*, vol. 18, no. 3, pp. 1494-1500, 2020.
[63] N. Yuvaraj, *et al*., "Nature-inspired-based approach for automated cyberbullying classification on multimedia social networking," *Mathematical Problems in Engineering*, 2021.
[64] M. Anand and R. Eswari. "Classification of abusive comments in social media using deep learning," in *Proc. 3rd International Conference on Computing Methodologies and Communication*, 2019.
[65] K. Wang, Y. Cui, J. Hu, Y. Zhang, W. Zhao, and L. Feng, "Cyberbullying detection, based on the fasttext and word similarity schemes," *ACM Transactions on Asian and Low-Resource Language Information Processing*, vol. 20, no. 1, pp. 1-15, 2020.
[66] S. D. Swamy, A. Jamatia, and B. Gambäck, "Studying generalisability across abusive language detection datasets," in *Proc. the 23rd Conference on Computational Natural Language Learning*, 2019.
[67] L. H. Tran, T. Tran, and A. Mai, "Text classification problems via BERT embedding method and graph convolutional neural network," arXiv e-prints, arXiv-2111, 2021.
[68] Z. Lan, M. Chen, S. Goodman, K. Gimpel, P. Sharma, and R. Soricut, "Albert: A lite bert for self-supervised learning of language representations," arXiv preprint arXiv:1909.11942, 2019.
[69] K. Clark, M. T. Luong, Q. V. Le, and C. D. Manning, "Electra: Pre-training text encoders as discriminators rather than generators," arXiv preprint arXiv:2003.10555, 2020.
[70] K. Ethayarajh, "How contextual are contextualized word representations? Comparing the geometry of BERT, ELMo, and GPT-2 embeddings," arXiv preprint arXiv:1909.00512, 2019.
[71] Z. Yang, Z. Dai, Y. Yang, J. Carbonell, R. R. Salakhutdinov, and Q. V. Le, "Xlnet: Generalized autoregressive pre-training for language understanding," *Advances in Neural Information Processing Systems*, vol. 32, 2019.
[72] Y. Liu, *et al*., "Roberta: A robustly optimized bert pre-training approach," arXiv preprint arXiv:1907.11692, 2019.
[73] N. Reimers and I. Gurevych, "Sentence-bert: Sentence embeddings using siamese bert-networks," arXiv preprint arXiv:1908.10084, 2019.
[74] S. Tomkins, L. Getoor, Y. Chen, and Y. Zhang, "A socio-linguistic model for cyberbullying detection," in *Proc. IEEE/ACM*

International Conference on Advances in Social Networks Analysis and Mining, 2018, pp. 53-60.
[75] S. Modha, P. Majumder, and T. Mandl, "An empirical evaluation of text representation schemes to filter the social media stream," *Journal of Experimental & Theoretical Artificial Intelligence*, pp. 1-27, 2021.
[76] S. Paul, S. Saha, and M. Hasanuzzaman, "Identification of cyberbullying: A deep learning based multimodal approach," *Multimedia Tools and Applications*, pp. 1-20, 2020.
[77] W. Rui, K. Xing, and Y. Jia, "BOWL: Bag of word clusters text representation using word embeddings," in *Proc. International Conference on Knowledge Science, Engineering and Management*, 2016.
[78] B. Das and S. Chakraborty, "An improved text sentiment classification model using TF-IDF and next word negation," arXiv preprint arXiv:1806.06407, 2018.
[79] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," arXiv preprint arXiv:1301.3781, 2013.
[80] D. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proc. Conference on Empirical Methods in Natural Language Processing*, 2014.
[81] A. Joulin, E. Grave, P. Bojanowski, and T. Mikolov, "Bag of tricks for efficient text classification," arXiv preprint arXiv:1607.01759, 2016.
[82] M. E. Peters, M. Neumann, M. Iyyer, M. Gardner, C. Clark, K. Lee, and L. Zettlemoyer, "Deep contextualized word representations," arXiv:1802.05365v2, 2018.
[83] J. Devlin, M. W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," arXiv preprint arXiv:1810.04805, 2018.
[84] F. Y. Osisanwo, J. E. Akinsola, O. Awodele, J. O. Hinmikaiye, O. Olakanmi, and J. Akinjobi, "Supervised machine learning algorithms: Classification and comparison," *International Journal of Computer Trends and Technology*, vol. 48, no. 3, pp. 128-138, 2017.
[85] I. Rish, "An empirical study of the naive bayes classifier," *IJCAI 2001 Workshop on Empirical Methods in Artificial Intelligence*, vol. 22, no. 3, pp. 41-46, 2001.
[86] B. Mahesh, "Machine learning algorithms - A review," *International Journal of Science and Research*, vol. 9, pp. 381-386, 2020.
[87] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5-32, 2001.
[88] S. Ray, "A quick review of machine learning algorithms," in *Proc. International Conference on Machine Learning, Big Data, Cloud and Parallel Computing*, 2019, pp. 35-39.

Arvind Kumar Gautam was born in Rewa, Madhya Pradesh, India. He received his Master of Philosophy degree in Computer Science in 2009 from APS University, Rewa, Madhya Pradesh, India. He is a Ph.D. research scholar in the Department of Computer Science, Indira Gandhi National Tribal University, Amarkantak, Madhya Pradesh, India. He is also working as a System Analyst for 9 years at Indira Gandhi National Tribal University, Amarkantak, Madhya Pradesh. He has more than 10 years of working experience in server administration, networking, cyber security, web programming, and teaching. He has published several research papers in international journals and conferences. His academic research interests mainly include Cyber Security, Machine Learning, and Web Engineering.

Abhishek Bansal received the MCA degree from Dr. B. R. Ambedkar University, Agra, Uttar Pradesh, India, in 2004, and the Ph.D. degree from Delhi University, Delhi, India. He is currently working as a Senior Assistant Professor with the Department of Computer Science, Indira Gandhi National Tribal University, Amarkantak, Madhya Pradesh, India. He has more than 12 years of teaching and research experience and supervised several Ph.D., research scholars. He has also published several papers in reputed journals and conferences.