electronics

Article

Cyberbullying Detection: Hybrid Models Based on Machine Learning and Natural Language Processing Techniques

Chahat Raj [1], Ayush Agarwal [2], Gnana Bharathy [1], Bhuva Narayan [3] and Mukesh Prasad [1,*]

1 School of Computer Science, FEIT, University of Technology Sydney, Sydney, NSW 2007, Australia; chahatraj58@gmail.com (C.R.); gnana.bharathy@uts.edu.au (G.B.)
2 Department of Information Technology, Delhi Technological University, Delhi 110042, India; ayush286@gmail.com
3 School of Communication, FASS, University of Technology Sydney, Sydney, NSW 2007, Australia; bhuva.narayan@uts.edu.au
* Correspondence: mukesh.nctu@gmail.com

Abstract: The rise in web and social media interactions has resulted in the efortless proliferation of offensive language and hate speech. Such online harassment, insults, and attacks are commonly termed cyberbullying. The sheer volume of user-generated content has made it challenging to identify such illicit content. Machine learning has wide applications in text classification, and researchers are shifting towards using deep neural networks in detecting cyberbullying due to the several advantages they have over traditional machine learning algorithms. This paper proposes a novel neural network framework with parameter optimization and an algorithmic comparative study of eleven classification methods: four traditional machine learning and seven shallow neural networks on two real world cyberbullying datasets. In addition, this paper also examines the effect of feature extraction and word-embedding-techniques-based natural language processing on algorithmic performance. Key observations from this study show that bidirectional neural networks and attention models provide high classification results. Logistic Regression was observed to be the best among the traditional machine learning classifiers used. Term Frequency-Inverse Document Frequency (TF-IDF) demonstrates consistently high accuracies with traditional machine learning techniques. Global Vectors (GloVe) perform better with neural network models. Bi-GRU and Bi-LSTM worked best amongst the neural networks used. The extensive experiments performed on the two datasets establish the importance of this work by comparing eleven classification methods and seven feature extraction techniques. Our proposed shallow neural networks outperform existing state-of-the-art approaches for cyberbullying detection, with accuracy and F1-scores as high as ~95% and ~98%, respectively.

Keywords: cyberbullying; hate speech; offensive language; machine learning; neural networks; deep learning; natural language processing

Citation: Raj, C.; Agarwal, A.; Bharathy, G.; Narayan, B.; Prasad, M. Cyberbullying Detection: Hybrid Models Based on Machine Learning and Natural Language Processing Techniques. *Electronics* **2021**, *10*, 2810. https://doi.org/10.3390/electronics10222810

Academic Editors: Juan M. Corchado, Stefanos Kollias and Javid Taheri

Received: 15 October 2021
Accepted: 15 November 2021
Published: 16 November 2021

1. Introduction

Social media is an interactive tool that brings people together to share information. The primary function of Online Social Networks (OSNs) is to allow people to communicate virtually by using the internet. However, such technologies have also resulted in several additional social issues, one of them being 'cyberbullying.' Although bullying has existed in society before these technologies, the perceived protection of the online interfaces has resulted in increased cyberbullying. Cyberbullying is commonly defined as an intentionally violent or aggressive behaviour using electronic media carried out by an individual or a group targeting a victim online [1]. This action involves repeated online insulting, harassing, or attacking a target verbally [2]. Malicious social media users use sexist remarks, offensive language, hate speech, toxic comments, and abusive language to target victims. Such content torments social media users, adversely affecting their mental health, and demeaning the integrity of social networking platforms [3]. Manual flagging of such illicit content online is neither feasible nor fruitful [4]. The need for an automated detection mechanism for cyberbullying detection is evident.

Textual instances of cyberbullying, hate speech, and offensive language are primarily identified by employing traditional machine learning classifiers. In a supervised learning scenario, preprocessed text input is used for training algorithms on a subset of the dataset. This training set consists of data items and their labels (cyberbully or not) on which the algorithm learns in a supervised fashion. The performance of the algorithm is recognized by validation and testing of the algorithm on the remaining unseen data of the dataset. The goal is to develop a classifier that performs equally well on training and testing data without overfitting. Classification results evaluated by several metrics such as accuracy, precision, recall, and F1-score demonstrate the efficacy of the classifiers. Several traditional machine learning algorithms require explicit feature extraction from input data. Natural Language Processing (NLP) has wide applications in this domain, as researchers have employed several feature extraction techniques for textual content. Primary endeavors include supervised classification by using bag-of-words at character-level representation [5–7] by various traditional machine learning algorithms, such as Random Forest, Naïve Bayes, Linear Regression, Support Vector Machine (SVM), and XGBoost for cyberbullying detection [8]. The bag-of-words approach extracts features of the dataset by counting the occurrences of words within a document, disregarding the word order. Some authors have proposed novel directions of extracting features from the respective social media platforms pertaining to the data. Advancement in approaches have brought a user-level detection mechanism where author profiling is used to check if a user has been previously involved in any acts of cyberbullying [3,9]. Existing approaches have also focused on the background information and user characteristics, such as their demographic data for characterizing malicious users [10]. Deep learning techniques were employed to overcome the limitations of traditional machine learning, eliminating the manual feature extraction step and obtaining better results on large-scale datasets. Recurrent Neural Networks (RNNs) are among researchers' primary choices for text classification [4,11] due to their ability to mine the implicit semantic features in the text. It reduces the necessity of a feature extraction operation by automatically extracting and optimizing the features. With recent evolution, Text-based Convolutional Neural Networks (Text-CNNs) have come into focus [12]. Although they are primarily employed in visual tasks, CNNs have displayed tremendous performance in one-dimensional text classification [13,14]. The state-of-the-art techniques for cyberbullying detection largely rely on RNNs, CNNs, and transformersdue to their mproved accuracies than compared to traditional machine learning classifiers.

This paper provides a wide comparative study on various traditional machine learning and deep learning algorithms. We experiment on two real-world datasets and compare the performance of eleven classification algorithms. We examine the classification efficiencies of XGBoost, SVM, Naive Bayes, and Logistic Regression under the traditional machine learning category. We implement four types of feature extraction techniques for these methods: count vectorization, Term Frequency-Inverse Document Frequency (TF-IDF) word unigram, TF-IDF word bigram and trigram, and TF-IDF character bigram and trigram. In the deep learning category, we propose a novel framework accommodating CNN, Long Short-Term Memory (LSTM), Bidirectional Long Short-Term Memory (Bi-LSTM), Gated Recurrent Unit (GRU), Bidirectional Gated Recurrent Unit (Bi-GRU), CNN-BiLSTM, and Attention-BiLSTM. Since we propose single layer models for the above stated deep learning algorithms, we, hereafter, refer to them as 'shallow neural networks' in this paper. It is to be noted that these shallow neural networks are different from 'shallow learning,' which is a subset of machine learning wherein predefined features are required. However, in the proposed shallow neural networks, feature extraction is an automatic step carried out by the neurons of the neural networks. The embedding techniques experimented with these shallow neural networks include Global Vectors (GloVe), FastText, and Paragram. This comparative study examines the performance of algorithms and their feature extraction techniques. The results are compared by using four evaluation metrics, namely accuracy, precision, recall, and F1-score. The models employed in this study outperform several state-of-the-art cyberbullying detection mechanisms. Section 4.3 of this paper, Baseline Comparison, discusses the performance of models found in the literature along with the results derived from this study.

The contributions of this paper are as follows:

1. We propose a novel architecture for cyberbullying detection that employs a bidirectional GRU by using GloVe for text representation. The proposed mechanism outperforms the existing baselines that employed Logistic Regression, CNN, and Bidirectional Encoder Representations from Transformers (BERT).
2. We also propose a novel CNN-BiLSTM framework for the task which yields results comparable to the existing baselines.
3. We provide a comparative study on the classification performance of four traditional machine learning and seven neural-network-based algorithms.
4. We experiment with several feature extraction techniques and determine best-suited approaches for feature extraction and text embedding for both traditional machine learning and neural-network-based methods.
5. We establish the efficacy of shallow neural networks for cyberbullying classification, thus moderating the need of complexly structures deep neural networks.

The rest of the paper is organized as follows: Section 2 explains the existing literature and mechanisms developed for efficient cyberbullying detection. Section 3 describes the methodology of the proposed work, the mathematical background of algorithms used, and the novel framework to accommodate all neural network algorithms. Section 4 lists the experimental results obtained and compares their performance; finally, Section 5 concludes the contributions and future prospects.

2. Related Work

Cyberbullying is classified as generalized abuse, largely towards appearance, interests, intelligence, or previous posts of the recipients. Hate speech is differentiated from cyberbullying in being defined as abuse directed specifically towards a unique, non-controllable attribute of a group of people, such as race, sexuality, and gender identity. Davidson et al. [5] identified it as a derogatory language intended to humiliate or to insult the members of the targeted group. In certain instances, people use terms not belonging to hate speech but are offensive to specific groups. Warner and Hirschberg [15] suggest that some African Americans use the term 'nigga' in their day-to-day language, and terms such as 'hoe' and 'bitch' are frequently used on social media. Such terms do not fall within the boundaries of hate speech but are offensive to specific societal sections; hence, they are categorized as offensive speech.

There is a considerable availability of real-world datasets in the present scenario such as the Twitter dataset [10], the Chinese Sina Weibo dataset [13], and the Kaggle dataset [16] for cyberbullying, hate speech, and offensive language detection. These labeled datasets allow the use of machine learning and deep learning algorithms by aiding supervised and semi-supervised training. Ample availability of annotated training data aids in building efficient supervised frameworks. However, the task of online cyberbullying detection holds certain limitations that have to be addressed. The drawbacks lie in the low availability of positively labeled cyberbullying posts because the datasets available are highly imbalanced. Wulczyn et al. [17] crowdsourced and aggregated a vast corpus of annotated Wikipedia articles with over 100K items extracted from talk pages. Another annotated dataset was presented by Warner and Hirschberg [15], collecting commonly used hate-speech terms from Twitter. They identified that hate speech targets specific groups of a particular ethnicity, race, caste, or creed and found that a correlation exists between hate speech and stereotypical words.

Classification frameworks of cyberbullying posts primarily utilize Natural Language Processing (NLP) methods [18]. Term Frequency-Inverse Document Frequency (TF-IDF) is an established method for extracting textual features from the data [19]. It measures a word's importance in a given document by using its frequency and inverse frequency count. Several NLP techniques such as TF-IDF, Vector Space Model (VSM), Linear Discriminant Analysis (LDA), and Latent Semantic Analysis (LSA) have been designed for such feature extraction [20–22]. A Naïve Bayes approach was implemented by Kwok and Wang [23] on a Twitter dataset comprising racist and non-racist comments, which demonstrated average classification performance when using the unigram bag-of-words model for feature extraction. A combined approach utilizing linguistics, n-grams, syntactic, and distributed syntactic features was designed by Nobata et al. [7] for detecting online hate speech. Yin et al. [24] proposed a supervised approach with TF-IDF for cyberbullying detection that uses content, context, and sentiment as textual features. Warner and Hirschberg [15] performed hate speech classification by using Support Vector Machines (SVM) linked with word sense disambiguation and using a lexicon of stereotypical words as features. A systematic review of existing research in this domain is compiled by Tokunaga [25], discussing the cyberbullying typologies, detection frameworks, and potential directions.

With the advent of deep learning algorithms for cyberbullying detection, Recurrent Neural Network (RNN) and Convolutional Neural Network (CNN) approaches have primarily been employed. Themeli et al. [26] performed hate speech detection by employing traditional machine learning and deep neural network models. Their experimental results demonstrate higher performance of Bag of Words (BoW) over GloVe and N-gram graphs when combined with Logistic regression and a three-layered neural network. Bu and Cho proposed a novel ensemble framework that uses two deep learning models for knowledge transfer: a CNN for capturing character-level syntactic features of the text and a Long-term Recurrent Convolutional Network (LRCN) for extracting semantic features [2]. Agrawal and Awekar [27] experimented on deep learning models by domain transferring the knowledge by using CNN, LSTM, Bi-LSTM, and Bi-LSTM with attention using random, GloVe, and Sentiment-Specific Word Embedding (SSWE). Aroyehun and Gelbukh [28] established the efficacy of deep neural networks by using seven different combinations of CNN, LSTM, and Bi-LSTM models and compared the results with traditional machine learning algorithms for aggression detection, incorporating data augmentation, and pseudo-labeling for the same. Mishra et al. [3] proposed a novel method of Twitter user profiling for cyberbullying detection by using authors' community-based data in addition to textual information. Rawat et al. [9] also relied on user information for abusive content detection by employing web scraping and exploratory data analysis to analyze the characteristics of users involved in spreading hate speech by combining traditional machine learning algorithms, sentiment analysis, and topic modeling for malicious user detection. The offensive tweet detection model by Aglionby et al. [29] proposes a multi-layer RNN and Gradient Boost Decision Tree (GBDT) classifier framework with a self-attention mechanism that enhances text classification. Chen et al. [30] analyzed embedding methods for words and sequences experimenting with word-level and sentence-level embedding techniques. Chu et al. also explored deep learning models with word embeddings for abuse detection [31] by developing an RNN with LSTM and two CNNs with word and character-level embeddings. Anand and Eswari [32] developed an LSTM and a CNN network and analyzed their effect in the presence and absence of GloVe embeddings for cyberbullying detection. Badjatiya et al. [11] explored the performance of CNN and LSTM with various text embedding models observing GDBT combined with LSTM as their best performing model. Pavlopoulos et al. [33] established that their proposed RNN with attention mechanism outperforms Logistic Regression, a Multi-Layer Perceptron (MLP), and a vanilla-CNN model for cyberbullying detection. Banerjee et al. [34] proposed a simple convolutional network with GloVe embeddings performing better than RNN GloVe and several traditional machine learning techniques. A Bi-LSTM network with an attention mechanism was proposed by Agarwal et al. [35] in order to classify cyberbullying posts using under sampling and class weighting for avoiding class imbalance in the dataset.

3. Methodology

Online cyberbullying detection frameworks primarily rely on traditional machine learning algorithms. However, these traditional machine learning algorithms pose a disadvantage due to the inability to yield high accuracies on vast volumes of data for supervised classification. Existing studies are advancing towards utilizing neural networks that overcome this limitation and provide better results and robust mechanisms. This section covers various popular traditional machine learning and shallow neural network approaches. We discuss the proposed methodology of our classification frameworks and the architectures of all the proposed networks.

3.1. Preprocessing and Feature Extraction

Algorithms for text classification cannot process raw data due to their inability to understand high-level human language directly. The text undergoes conversion into vector notation in order to be processed by classification algorithms. Prior to this step, raw textual data undergoes several preprocessing steps, often referred to as data cleaning. Figure 1 illustrates the workflow of the proposed methodology. Input data are preprocessed by removing empty rows, punctuation, special characters, numerical values, stopwords, lowercasing text, tokenization, and stemming. In order to create vector notations of the input text, we experiment with several methods. For traditional machine learning models, we use four methods: Count Vectorization, TF-IDF word unigram, TF-IDF word bigram and trigram, and TF-IDF character bigram and trigram. For the proposed shallow neural networks, we use GloVe, FastText, and Paragram as the embedding representations. We employ a stratified 5-fold cross validation technique that splits the dataset into five sets of training and testing, preserving the class distribution of the original dataset in each split. This technique is employed to obtain statistically grounded results by averaging results from runs on individual splits of the original dataset.

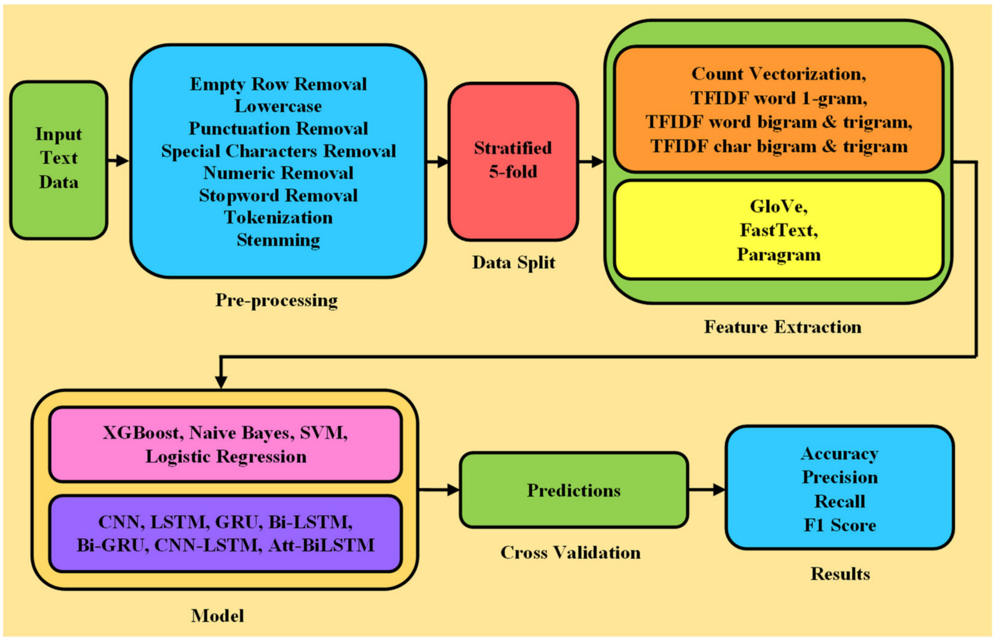
Input Text Data
Empty Row Removal
Lowercase
Punctuation Removal
Special Characters Removal
Numeric Removal
Stopword Removal
Tokenization
Stemming
Pre-processing
Stratified 5-fold
Data Split
Count Vectorization, TFIDF word 1-gram, TFIDF word bigram & trigram, TFIDF char bigram & trigram
GloVe, FastText, Paragram
Feature Extraction
XGBoost, Naive Bayes, SVM, Logistic Regression
CNN, LSTM, GRU, Bi-LSTM, Bi-GRU, CNN-LSTM, Att-BiLSTM
Model
Predictions
Cross Validation
Accuracy
Precision
Recall
F1 Score
Results

Figure 1. Workflow of the proposed framework.

Count Vectorization: This is a simple statistical method for generating embedded vectors of input text [36]. We use the frequency of the occurrence of a term in a document in order to generate its embedding vector. A matrix is created for the entire document set where rows contain each document and columns represent each word. The cells contain the values of occurrence frequency of a term in a document. We use this matrix as the feature representation for training the traditional machine learning algorithms.

TF-IDF: Term Frequency-Inverse Document Frequency (TF-IDF) [37] is a statistical approach that uses the occurrences of words as a measure for extracting textual features. A word's importance is directly proportional to its frequency in the document and inversely proportional to its frequency in the entire document set. For a term w_i in a document x_j, where its occurrence is $n_{i,j}$ in x_j, we calculate the term-frequency, $TF_{i,j}$, given by Equation (1).

$$TF_{i,j} = \frac{n_{i,j}}{\sum_k n_{k,j}} \tag{1}$$

Here, $\sum_k n_{k,j}$ denotes the sum of occurrences of a term w_i in the entire document set. Next, we compute the inverse document frequency (IDF) by taking the logarithm of the total number of documents divided by the number of documents with the term w_i as in Equation (2).

$$IDF_i = log\left(\frac{|D|}{\left|\{j : w_i \in x_j\}\right| + 1}\right) \tag{2}$$

Here, $|D|$ denotes the total number of documents, and $\left|\{j : w_i \in x_j\}\right|$ is the number of documents with the term w_i. Once we obtain the individual TF and IDF values, we compute the required TF-IDF of the term w_i, given by Equation (3).

$$(TF - IDF)_{w_i} = TF_{i,j} \times IDF_i \tag{3}$$

We use TF-IDF with word unigram, word bigram and trigram, and character bigram and trigram. Here, unigram means a single word, such as 'dog,' 'girl,' and 'book'. Bigram (for instance, 'grey house') and trigram (for instance, 'within walking distance') are the combinations of two and three words, respectively. The character bigram ('flavor': 'fl,' 'la,' 'av,' 'vo,' and 'or') and trigram ('flavor': 'fla,' 'lav,' 'avo,' and 'vor'), similarly, are the combinations of two and three characters, respectively.

GloVe: Global Vectors (GloVe) [38] for word representations is an unsupervised technique for deriving word embeddings from text input. We utilize an A $\times$ A term-based co-occurrence matrix to obtain representations. The cooccurrence matrix is used to examine the semantic relationship between terms. For instance, high cosine similarity is demonstrated between words such as 'queen' and 'king' or 'mother' and 'woman.' The technique learns from a large Wikipedia and Gigaword corpus in an unsupervised fashion. For a word i with its vector representation w_i, the objective function is denoted by Equation (4):

$$f\left(w_i - w_j, \widetilde{w}_k\right) = \frac{P_{ik}}{P_{jk}} \tag{4}$$

where i and k are words with a similar context, and P_{ik} is the probability of them occurring together. We use these co-occurrence probabilities as features by capturing both statistics and the context of the words.

FastText: Introduced by Facebook's AI Research Lab (FAIR), FastText [39,40] is a skip-gram-based model [41] for enhanced word representations. The effectiveness of this technique lies in its consideration of the morphology of words in a language. Other embedding techniques denote each word as a distinct vector. In contrast, FastText is specially designed to handle words of the same root using character n-grams. Naturally, it contains the sub word information for every word by dividing each word into a bag of its n-gram combinations. For example, for a word 'language' with n as 3, the bag of character n-grams will contain 'la,' 'lan,' 'ang,' 'ngu,' 'gua,' 'uag,' 'age,' and 'ge'. FastText enables understanding the context of unknown words by breaking them into smaller forms and matching the similarities with those within its training corpus.

Paragram: Paragram [42] is another word representation technique designed to capture better contextual similarities. It uses the ParaPhrase DataBase (PPDB) and performs fine-tuning [43], counter-fitting [44], and attract-repel [45] in order to inject synonym and antonym features as a vectorization constraint. The technique is comparatively robust due to better contextual understanding.

3.2. Traditional Machine Learning Approaches

We employed four popular machine learning approaches for cyberbullying detection: XGBoost, Naïve Bayes, SVM, and Logistic Regression. After the preprocessing and feature extraction phases, the vectorized text was input to these classifiers in order to evaluate their performance.

XGBoost: Extreme Gradient Boosting (XGBoost) [46] is a Gradient Boosting Decision Tree (GBDT) enhancement. The algorithm employs multiple decision trees with low accuracy (weak learners) and combines them to provide higher accuracy. The trees are built in stages, and the residuals or errors of prior models from previous stages inform the next stage of trees by using gradient descent. That is standard gradient boosting. Building on top of gradient boosting principle, XGBoost incorporates several other optimizations such as parallelized implementation, tree pruning, hardware optimization, regularization, sparsity awareness, cross validation, and weighted quantile sketch to render its performance more efficient and effective.

The algorithm advances in the direction of the tree, which minimizes the objective function. For a document $D = \{(x_i, y_i)||D| = n, x_i \in R^m, y_i \in R\}$ with n samples and m eigenvalues, where x_i denotes a sample and y_i denotes its category, the predictions are calculated by using Equation (5):

$$\hat{y}_i = \theta(x_i) = \sum_{n=1}^{N} f_n(x_i) \tag{5}$$

where $f_n(x_i)$ is the error value between true and predicted classes. The solution of the above objective function is calculated by using maximum likelihood estimation as discussed by Chen and Guestrin [47].

Naïve Bayes: Naïve Bayes (NB) algorithm [48] is used for probabilistic classification. It is widely used for various practical applications due to its efficiency in reducing computational costs. It is a scalable algorithm applicable to large-sized datasets, also resulting in high classification accuracies. Its principally assumes that a feature in a category is independent of its presence in another category. The probability of a document D pertaining to a class C is given by Equation (6).

$$P(C|D) = \frac{P(D_j|C)P(C)}{P(D_j)} = \frac{P(D_j|C)P(C)}{P(D_j|C)P(C) + P(D_j|C)P(\overline{C})} \tag{6}$$

In order to predict that a data point x' with features $(a'_i \ldots a'_d)$ belongs to a particular category, the prediction $\theta(x')$ is given by Equation (7).

$$\theta(x') = argmax_{y_i \in C} P(y_i) \prod_{j=1}^{d} P(a'_j|y_i) \tag{7}$$

SVM: Support Vector Machine (SVM) [49] is a supervised algorithm that uses the separation margin between data points of classes as a classification criterion. The original m-dimensional feature space is reduced to a user-defined dimensional space. Support vectors are then determined to optimize the margin distance among data points of different categories. The algorithm automatically determines these support vectors found nearest to the separating margins (hyperplanes). Equation (8) defines a linear SVM optimization equation:

$$\alpha^* = maximize_\alpha \sum_{i=1}^{l} \alpha_i - \frac{1}{2} \sum_{i=1}^{l} \sum_{j=1}^{l} y_i y_j \alpha_i \alpha_j (x_i x_j) \tag{8}$$

$$\begin{aligned} &\textit{subject to the constraints } \sum_{i=1}^{l} y_i \alpha_i = 0 \\ &0 \le \alpha_i \le C,\ i = 1 \ldots l \end{aligned} \tag{9}$$

where α_i denotes the term weight, and C represents the model's error and relative importance. In order to predict that a data point x' belongs to a particular category, the prediction $\theta(x')$ is given by Equation (10):

$$\theta(x') = sign\left(\sum_{i=1}^{l} \alpha_i y_i (x_i, x') + b\right) = sign((w^*, x') + b) \tag{10}$$

where, $w^* = \sum_{i=1}^{l} \alpha_i y_i x_i$.

Logistic Regression: Logistic Regression (LR) [50] is another statistical algorithm that works on predicting probabilities rather than classes. The logistic function is used to form a hyperplane in order to classify data points in the given classes. Textual features are input to the algorithm employed to generate forecasts about a data point belonging to a particular class. The function is given by Equation (11) where the positive class is determined by $h_\theta(x) \ge 0.5, y = 1$, and the negative class is determined by $h_\theta(x) \le 0.5, y = 0$.

$$h_\theta(x) = \frac{1}{1 + e^{-x_i \theta}} \tag{11}$$

Here, θ is the parameter (weight) on an input variable x parameterizing the space of linear functions mapping from X to Y.

3.3. Neural Network Approaches

The elimination of manual feature extraction has made neural networks extremely popular in the research community. Neurons within the network are responsible for automatically extracting essential features that help to differentiate content belonging to different classes. The need of neural networks arises due to large dataset sizes that most of the traditional machine learning algorithms fail to accommodate. Additionally, neural networks offer robustness and higher classification results. We compare the following architectures of popular neural networks for cyberbullying classification: CNN, LSTM, Bi-LSTM, GRU, Bi-GRU, CNN-BiLSTM, and Attention-BiLSTM. To execute our approach, we design a novel framework accommodating each of these models, depicted by Figures 2–5. The methodology and architectures of these models are discussed below.

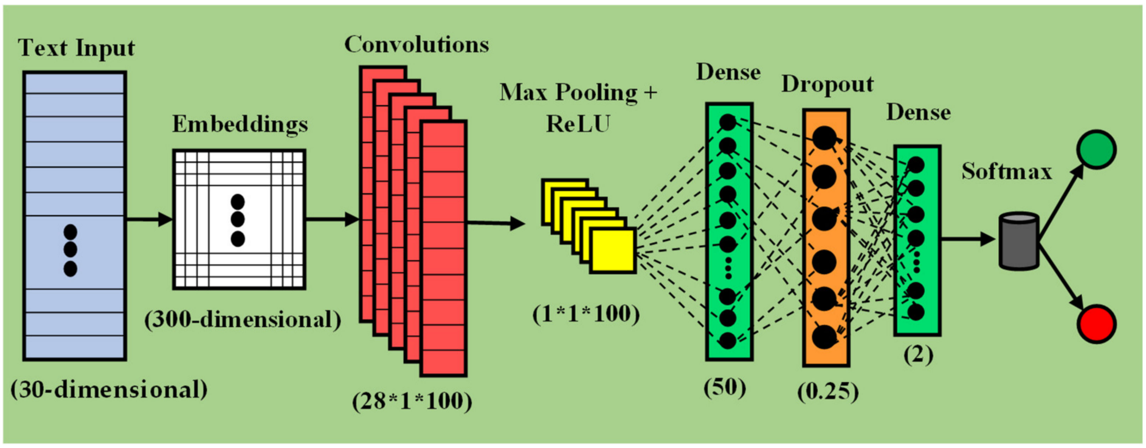

Figure 2. Architectural representation of proposed CNN framework.

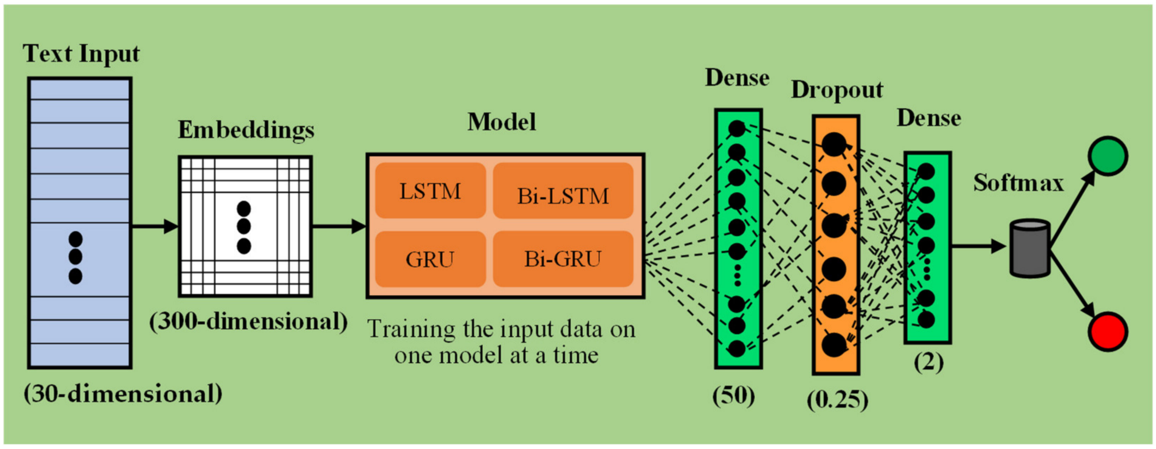

Figure 3. Architectural representation of framework for using LSTM, Bi-LSTM, GRU, and Bi-GRU.

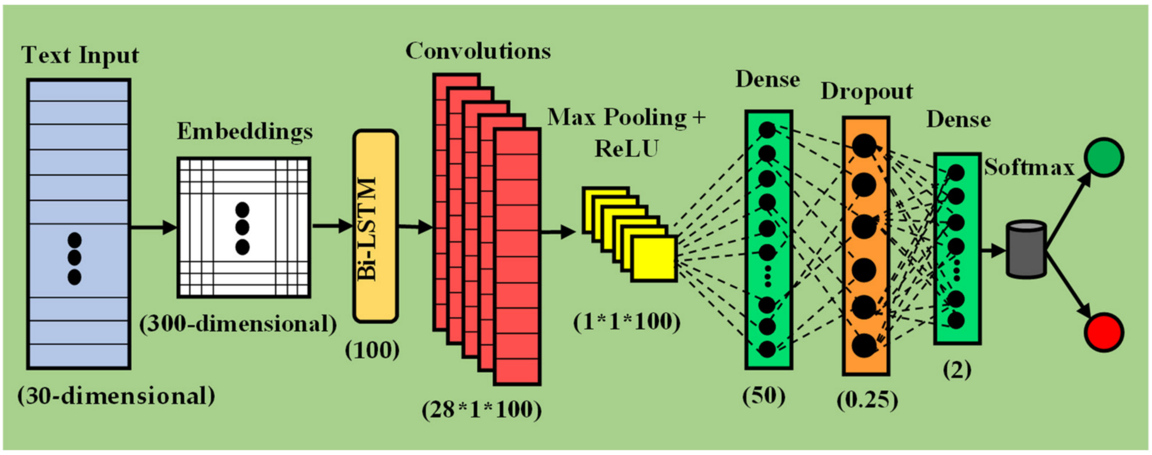

Figure 4. Architectural representation of CNN-BiLSTM framework.

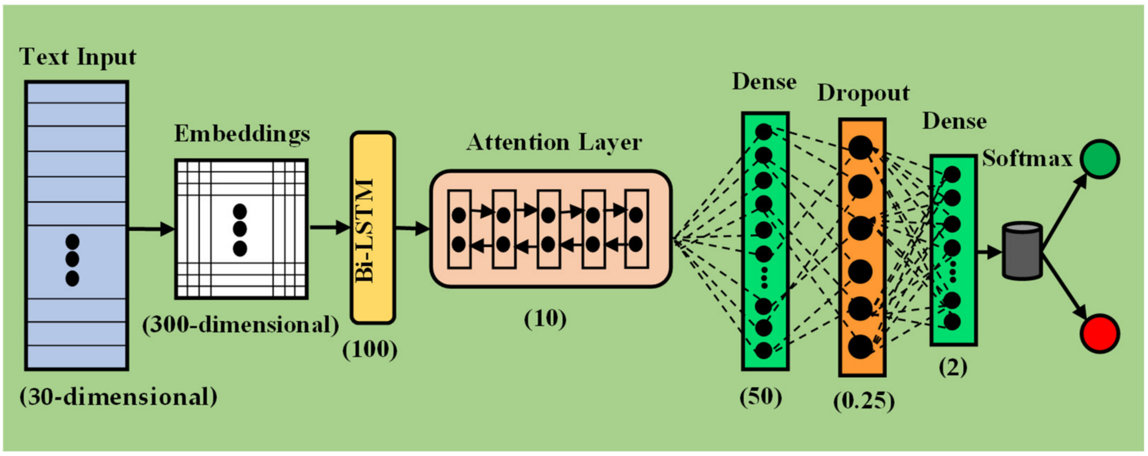

Figure 5. Architectural representation of Attention-BiLSTM framework.

CNN: After demonstrating extreme efficiency in image classification tasks, convolutional neural networks [51] are widely adopted for text classification. We develop Text-CNN with a single hidden layer and employ it for cyberbullying detection. A convolution operation $*$ on functions f and g is performed by reversing and shifting one of these functions as described by Equation (12).

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t-\tau)d\tau \tag{12}$$

A representation of the proposed architecture incorporation of a CNN is shown in Figure 2. The model takes in preprocessed 30-dimensional text input and performs the respective embedding by using GloVe, FastText, and Paragram with 300-dimensional word vectors for each of them. The embedded text passes through a one-dimensional convolutional layer with a kernel that moves over the convolutions with a filter size of three and stride of one. The information is then passed through a max-pooling layer that outputs the max value from each resulting data matrix. The ReLU activation function is used after, which the input is fed to a series of fully connected layers. A dense layer of dimension 50 sends the information through a dropout layer of probability 0.25 to a final dense layer. The dimension of the recurrent dense layer is set to two, which is equal to the number of classes. Predictions are generated by using a softmax function that outputs the labels for each item in the dataset.

LSTM: Long Short-Term Memory networks (LSTMs) [52] are a special type of Recurrent Neural Networks (RNNs) that are more advantageous compared to RNNs in terms of information retention. LSTMs overcome the problem of vanishing gradient descent encountered in traditional RNNs [53]. LSTMs are highly preferred for tasks such as text classification and predictive modeling due to their extensive memory capacity. Such a network selectively decides what information is necessary to be transferred to further neurons and which data can be forgotten or omitted. These networks employ backpropagation and a gated mechanism. A basic LSTM network consists of an input (i_t), output (o_t), and a forget gate (f_t), represented by Equations (13)–(15).

$$i_t = \sigma(W_i \cdot [h_{t-1}, x_t] + b_i) \tag{13}$$

$$f_t = \sigma\left(W_f \cdot [h_{t-1}, x_t] + b_f\right) \tag{14}$$

$$o_t = \sigma(W_o \cdot [h_{t-1}, x_t] + b_o) \tag{15}$$

Here, x_t denotes an input text, h is used to represent the state of the input where h_t is called current state, and h_{t-1} denotes the previous state. W and b are the weights and bias for each gate, respectively. Here, σ denotes the activation function used, which is ReLU in the case of the proposed model.

Bi-LSTM: Bidirectional LSTM (Bi-LSTM) [54] is a robust mechanism used to enhance backpropagation in LSTM networks. While the information in an LSTM travels unidirectionally, Bi-LSTM allows data to move in both forward and backward directions. A Bi-LSTM processes inputs both reverse and serially. Architecturally, it is simply combining two LSTMs but in opposite directions. This allows the network to remember information from past to future by using the forward layer and future to past layer by using the backward LSTM layer. For a given sequence of inputs $x_{t-1}, x_t, x_{t+1}, \ldots, x_n$, the output from the forward layer $\overrightarrow{h}$ is calculated, whereas for a reverse sequence, $x_n, x_{n-1}, x_{n-2}, \ldots, x_{t-1}$, the output $\overleftarrow{h}$ is calculated through the backward layer where $h = o_t * tanh(C_t)$ and where C_t is a vector produced by the activation function. The output of the Bi-LSTM network is denoted by Equation (16):

$$Y_T = y_{t-1}, y_t, \ldots, y_{t+n} \tag{16}$$

where $y_t = \sigma\,(\overrightarrow{h}, \overleftarrow{h})$, and σ is a concatenation operation.

GRU: Gated Recurrent Units (GRUs) [55] are also a type of RNN with a gated mechanism designed to deal with the vanishing and exploding gradient problem. These provide more testing accuracies than traditional RNNs because of the ability to remember long-term dependencies. GRUs are a more straightforward and dynamic version of LSTM networks specifically designed for updating or resetting information in their memory cells. The network constitutes an update gate that combines input and a forget gate present in LSTMs. Additionally, there is a reset gate for refreshing the memory contents. These are lightweight and have fewer parameters than LSTMs. For an input vector x_t at time t with vector, parameter, and matrices such as b, W and U, respectively, the update gate and reset gate are given by Equations (17) and (18):

$$z_t = \sigma_g(W_z x_t + U_z h_{t-1} + b_z) \tag{17}$$

$$r_t = \sigma_g(W_r x_t + U_r h_{t-1} + b_r) \tag{18}$$

where h_t is defined by Equation (19) with $\odot$ as the Hadamard product.

$$h_t = (1 - z_t) \odot h_{t-1} + z_t \odot \hat{h}_t \tag{19}$$

$$\hat{h}_t = \varnothing_h(W_h x_t + U_h(r_t \odot hh_{t-1}) + b_h) \tag{20}$$

σ_g and $\varnothing_h$ are the activation function and hyperbolic tangent, respectively.

Bi-GRU: A bidirectional GRU [54] is a dual-layered structure similar to a Bi-LSTM with forward and backward neural networks. The idea of this structure is to transfer entire contextual information from the input to the output layer. Similarly to a bidirectional LSTM, in a Bi-GRU, the input information travels through a neural network in the forward direction and a neural network in the backward direction. The outputs from both these forward and backward layers are fused to provide the final output. An architectural representation for classification using LSTM, Bi-LSTM, GRU, and Bi-GRU is displayed in Figure 3. In order to use a specific model at a time, the outputs from the embedding matrix are fed to the supposed neural network model and then sent to the series of fully connected layers. The parameters of fully connected layers stay the same as in the CNN for all models proposed in this work.

CNN-BiLSTM: We propose a combination network constituting a convolutional and a BiLSTM layer, illustrated in Figure 4. The input text, after undergoing embedding, is initially fed to a BiLSTM layer of size 100. Features from this layer further undergo convolution operation by using a one-dimensional convolutional layer with a ReLU activation. Outputs are processed under max-pooling operation and passed on to the series of fully connected layers. This combination allows us to use the retentive power of LSTMs and feature extraction capability of CNNs, thus forming a more robust classifier.

Attention-BiLSTM: Another model that we propose combines a Bi-LSTM network with a hierarchical attention model. As suggested by the name, the attention model [56] pays special attention to words possessing higher importance in the document. In the proposed architecture represented in Figure 5, information processed through the Bi-LSTM network is passed through an attention layer with multiple neurons and then to the fully connected layers. The mechanism encodes only selective valuable information by understanding the context and enhancing the final output. This allows the model to run successfully on sufficiently large input texts. We adopt the self-attention mechanism proposed by Vaswani et al. [56]. The model assigns non-zero weights to all input items. We employ scaled dot product as the similarity function. The attention value for a given query Q is calculated by using key-value pairs as source by obtaining the similarities between each key K and the query. This is mathematically represented by Equation (21). A softmax function is used to normalize the weights in order to calculate the final attention provided by Equation (22), where d_k is the key dimension.

$$Attention(Query,\ Source) = \sum_{i=1}^{t_x} Similarity(Query,\ Key_i) \times Value_i \tag{21}$$

$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right)V \tag{22}$$

3.4. Implementation Details

The implementation for all experiments is carried out using Python 3 on Google Colab with a RAM of 13.53 GB. For the preprocessing tasks, we employed RegexTokenizer for word tokenization, Porter Stemmer for stemming, and WordNet Lemmatizer. As discussed above, the embedding dimension is set to 300. The meta-parameters and hyperparameters in the shallow neural networks such as the input dimension, size of recurrent dense layers, dropout, and activation function are chosen upon several experimentation to yield the best possible results. A dense layer of 50 units provided the lowest loss value and highest accuracies in all experiments. The consequent dense layer is of size two depending on the number of classes in the datasets. A dropout value of 0.25 reduced overfitting considerably. The meta-parameters specific to each neural network such as the kernel size, stride in the CNN network, and the sizes of the LSTM, Bi-LSTM, GRU, and Bi-GRU networks are also decided upon trial-and-error experimentation with several values among their individual ranges. We employed the Adam optimizer and binary cross entropy for the models. Training is executed on a batch size of 64 instances for five epochs each.

4. Experimental Result Analysis

In this section, we describe the datasets used and report the experiment results. The results are evaluated using four metrics: accuracy, precision, recall, and F1-scores. We graphically compare the performance of all algorithms used on two real-world datasets. The baseline comparison with existing literature is provided in Section 4.3.

4.1. Datasets

Wikipedia Attack Dataset: This corpus was crowdsourced by Wulczyn et al. [17] in 2017 using Wikipedia articles. The dataset consists of discussion comments in the English language extracted from the 'talk page' of Wikipedia websites. The comments are extracted by accessing the revision history of Wikipedia pages to obtain all interactions, including removed comments. The collected corpus is cleaned by removing HTML content and keeping plain text only. The dataset is strapped out of all bot messages, and only human-made comments were retained. The dataset version that we use consists of 115,864 user comments with 13,590 cyberbullying text and 102,274 non-cyber bullying comments. Figure 6 illustrates the word clouds for attack and non-attack classes for this dataset.

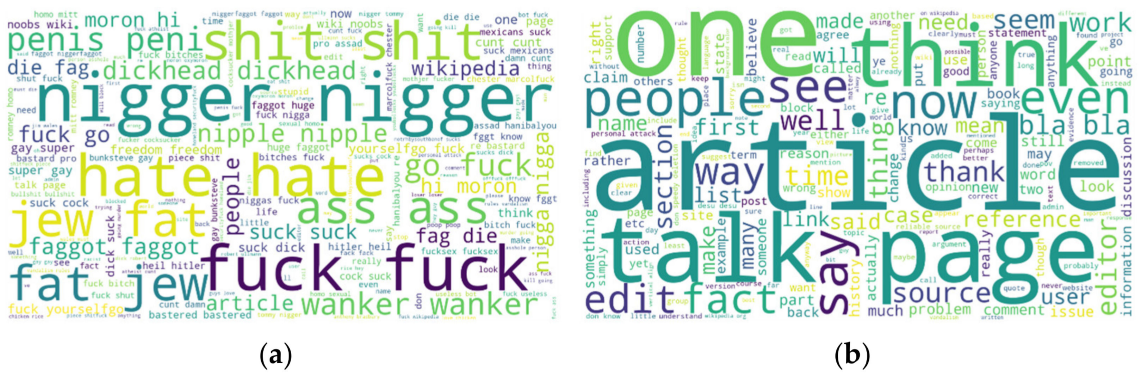

Figure 6. Word clouds of Wikipedia Attack dataset: (**a**) Attack and (**b**) Non-Attack.

Wikipedia Web Toxicity Dataset: Another corpus of comments is proposed by Wulczyn et al. [17] from Wikipedia collected from 'article talk namespace.' It is a binary labeled corpus with 159,689 comments containing 15,365 toxic and 144,324 non-toxic comments. The scraping procedure of the dataset is the same as the Wikipedia Attack dataset using the revision history of article pages. The discussion comments are collected, and administrative and bot comments are removed to constitute the final corpus. Figure 7 illustrates the word clouds for toxic and non-toxic classes for this dataset.

(**a**) (**b**)

Figure 7. Word clouds of Wikipedia Web Toxicity dataset: (**a**) Toxic and (**b**) Non-Toxic.

4.2. Result Analysis

We evaluate and compare the results of all classifiers used on two datasets. Table 1 illustrates the experimental results of traditional machine learning algorithms by using four feature extraction approaches. The results on XGBoost, Naïve Bayes, SVM, and Logistic Regression are graphically represented in Figures 8 and 9. Table 2 illustrates the results of the proposed shallow neural networks and their performance comparison is shown in Figures 10 and 11. While accuracy is the simplest and most intuitive metric of model performance, it is not suitable for unbalanced datasets. The precision and recall as well as F1-score (which is the harmonic mean of precision and recall) have also been reported, and their performances are also over 90% for the majority of cases.

Table 1. Results of four traditional machine learning models with four feature extraction techniques on two datasets.

Feature Extraction	ML Algorithms	Wikipedia Attack Dataset				Wikipedia Web Toxicity			
		A	P	R	F1	A	P	R	F1
Count Vectorization	XG Boost	94.43	99.55	95.36	97.14	94.19	99.14	94.35	97.21
	Naïve Bayes	95.44	99.73	96.27	96.92	95.62	98.48	96.9	98.14
	SVM	93.8	96.78	96.6	96.24	95.55	98.65	97.82	97.88
	Logistic Regression	94.55	98.34	96.11	96.79	96.49	99.39	97.55	97.73
TFIDF Word unigram	XG Boost	95.23	99.51	95.6	97.86	94.33	99.95	94.46	96.82
	Naïve Bayes	91.34	93.12	90.67	95.42	92.57	97.56	92.89	96.36
	SVM	95.02	98.94	96.41	98.12	96.29	99.93	97.59	98.77
	Logistic Regression	95.11	99.09	95.27	97.18	95.91	99.33	96.43	98.55
TFIDF Word bigram and trigram	XG Boost	93.26	98.71	92.93	96.79	92.38	96.46	91.74	96.12
	Naïve Bayes	89.11	92.12	89.25	94.77	91.81	98.12	90.91	95.54
	SVM	93.58	98.52	94.51	96.31	95.32	100	95.64	97.97
	Logistic Regression	91.6	94.55	91.76	96.13	93.24	97.17	92.93	96.82
TFIDF Char bigram and trigram	XG Boost	93.87	99.97	94.22	96.86	95	99.74	94.62	97.8
	Naïve Bayes	91.24	99.9	90.91	96.05	92.07	99.8	92.91	96.64
	SVM	81.01	98.92	69.05	81.24	96.21	99.35	97.34	98.32
	Logistic Regression	95.16	99.52	95.29	97.13	96.13	99.46	96.45	98.15

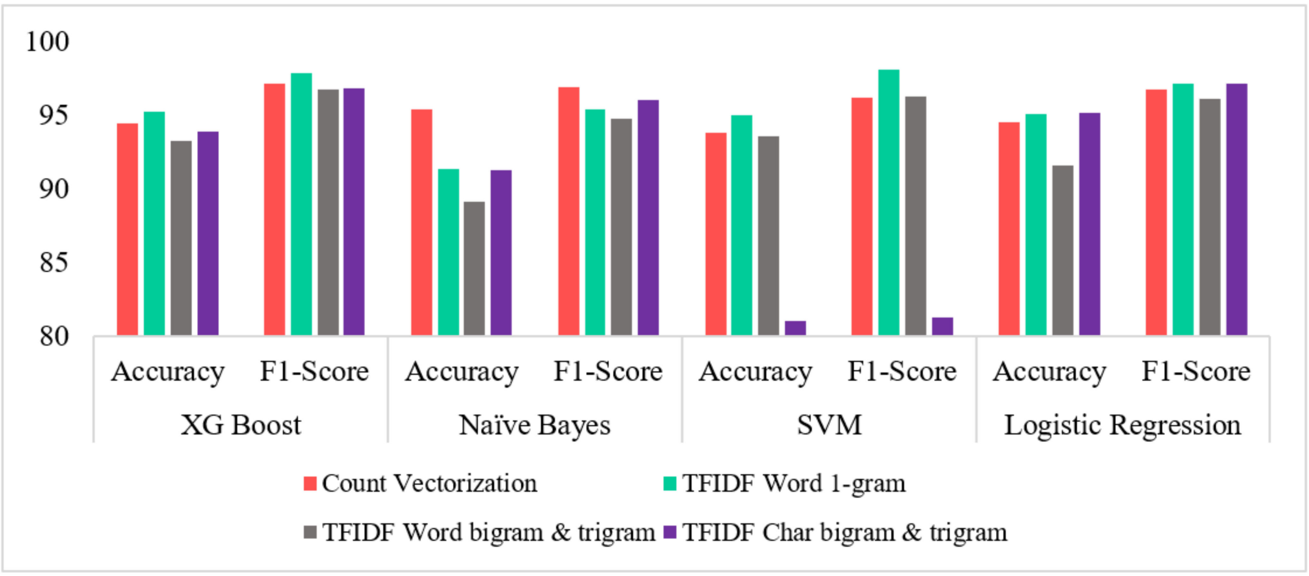

Figure 8. Performance comparison of traditional machine learning techniques on Wikipedia Attack dataset.

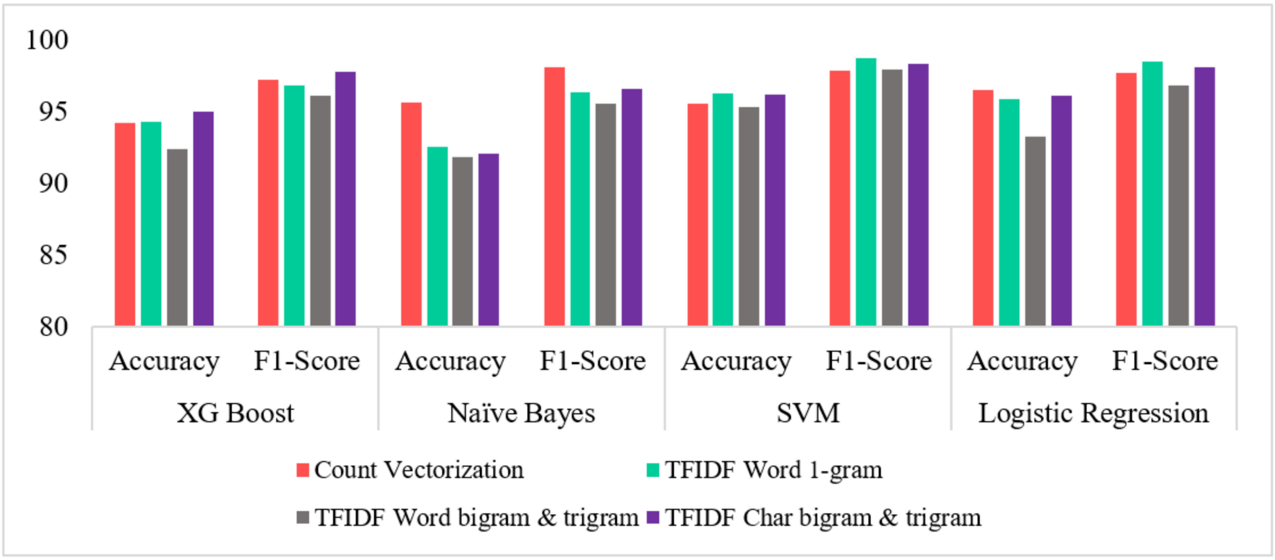

Figure 9. Performance comparison of traditional machine learning on Wikipedia Web Toxicity dataset.

For the Wikipedia Attack Dataset, we observed that SVM achieves the highest F1-score of 98.12% using TF-IDF word unigram, followed by XG Boost providing 97.14% and Logistic Regression providing 97.13% F1-scores with Count Vectorization and TF-IDF character bigram and trigram, respectively. XG Boost and Logistic Regression appear to perform better than Naïve Bayes on this dataset despite Naïve Bayes achieving the highest accuracy of 95.44% when using Count Vectorization. SVM demonstrates a lower performance on this dataset, especially with TFIDF character bigram and trigram.

For the Wikipedia Web Toxicity dataset, the highest F1-score of 98.77% is displayed by SVM with TFIDF word unigram embeddings. SVM with 98.32% F1-score follows it using TFIDF character bigram and trigram. Naïve Bayes using Count Vectorization follows it with 98.14% score. Overall, Logistic Regression demonstrates high results with all types of feature extraction techniques. SVM follows it, which is followed by XG Boost and then Naïve Bayes. However, when compared against the proposed shallow neural networks, traditional machine learning is observed to have lower scores and demonstrated inconsistency.

Table 2. Results of seven shallow neural networks with three embedding techniques on two datasets.

Feature Extraction	DL Algorithms	Wikipedia Attack Dataset				Wikipedia Web Toxicity			
		A	P	R	F1	A	P	R	F1
GloVe	CNN	95.6	98.43	96.69	97.16	96.53	99.33	97.56	98.69
	LSTM	95.08	98.53	96.85	97.61	96.44	99.14	97.24	98.34
	GRU	95.46	99.19	96.17	97.16	96.24	98.99	97.28	98.42
	Bi-LSTM	96.88	98.29	97.01	98.1	95.99	98.94	97.48	98.05
	Bi-GRU	96.98	99.22	96.74	98.56	96.01	99.45	96.8	98.63
	CNN-BiLSTM	95.36	98.88	96.84	98.03	96.74	99.63	96.78	98.07
	Att-BiLSTM	95.4	98.45	96.91	97.88	96.84	98.5	97.84	98.06
FastText	CNN	94.94	98.21	96.54	97.6	96.36	99.19	97.87	97.88
	LSTM	95.34	99.22	96.82	97.25	96.57	99.64	96.72	98.37
	GRU	95.8	98.72	96.72	98.23	95.93	98.42	97.17	98.55
	Bi-LSTM	95.93	98.81	96.58	98.15	96.51	99.18	97.34	98.07
	Bi-GRU	95.5	99.37	96.06	97.68	96.45	98.12	96.61	98.58
	CNN-BiLSTM	95.1	98.72	96.06	97.05	96.1	99.71	97.23	98.2
	Att-BiLSTM	95.01	99.12	96.51	97.81	96.6	99.18	97.06	98.65
Paragram	CNN	95.25	98.2	97.36	97.36	96.31	99.41	97.06	98.61
	LSTM	95.32	98.28	96.51	98.01	96.18	99.09	96.82	97.81
	GRU	95.78	98.03	97.39	97.48	96.58	98.92	96.83	98.22
	Bi-LSTM	95.12	99.17	96.74	97.95	95.87	99.41	96.63	98.46
	Bi-GRU	95.88	98.66	97.6	97.82	96.65	99.92	97.27	98.43
	CNN-BiLSTM	94.94	97.76	96.63	97.64	96.53	99.35	97.31	98.34
	Att-BiLSTM	95.12	98.21	96.73	98.03	96.77	98.91	97.72	98.49

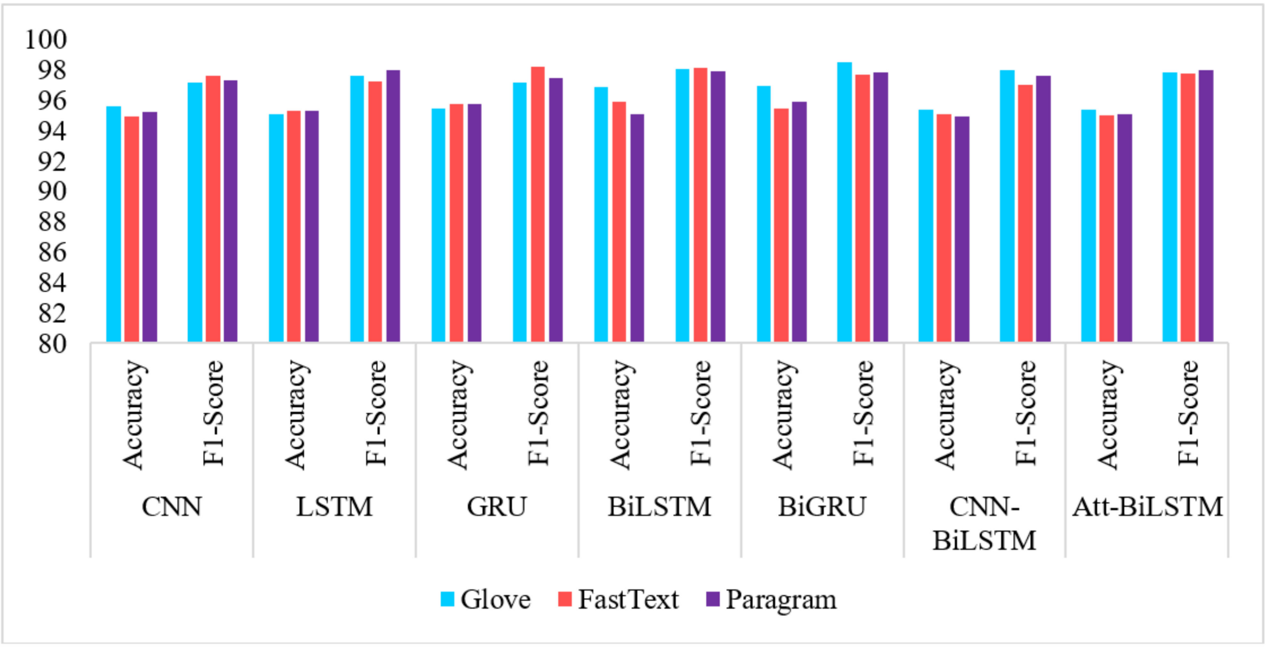

Figure 10. Performance comparison of seven shallow neural networks on Wikipedia Attack dataset.

By observing the performance of shallow neural networks, the majority of results are over 95% considering all evaluation measures. Moreover, these figures are higher than those reported for traditional machine learning models (see Section 4.3, Baseline Comparison). For the Wikipedia Attack dataset, Bi-GRU with GloVe embeddings is the best performing model with 98.56% F1-score and 96.98% accuracy. We observed that GloVe embeddings offered a higher and more consistent rate in better classification than Paragram and FastText. Although the other two embedding methods have also performed well, GloVe is simply the winner. Amongst the proposed shallow neural networks, Bi-LSTM and Bi-GRU models performed better than the rest. For the Wikipedia Web Toxicity Dataset, Bi-LSTM models have demonstrated great performance. The highest goes to 98.69% F1-score with GloVe embeddings followed by 98.65 with Attention-BiLSTM using FastText and then 98.61% with CNN using Paragram. On this dataset, the best performing models are CNN-BiLSTM, Attention-BiLSTM, and BiGRU. The results over 95% are quite similar; thus, we assume that the proposed framework allows all the above models to perform classification with high accuracies.

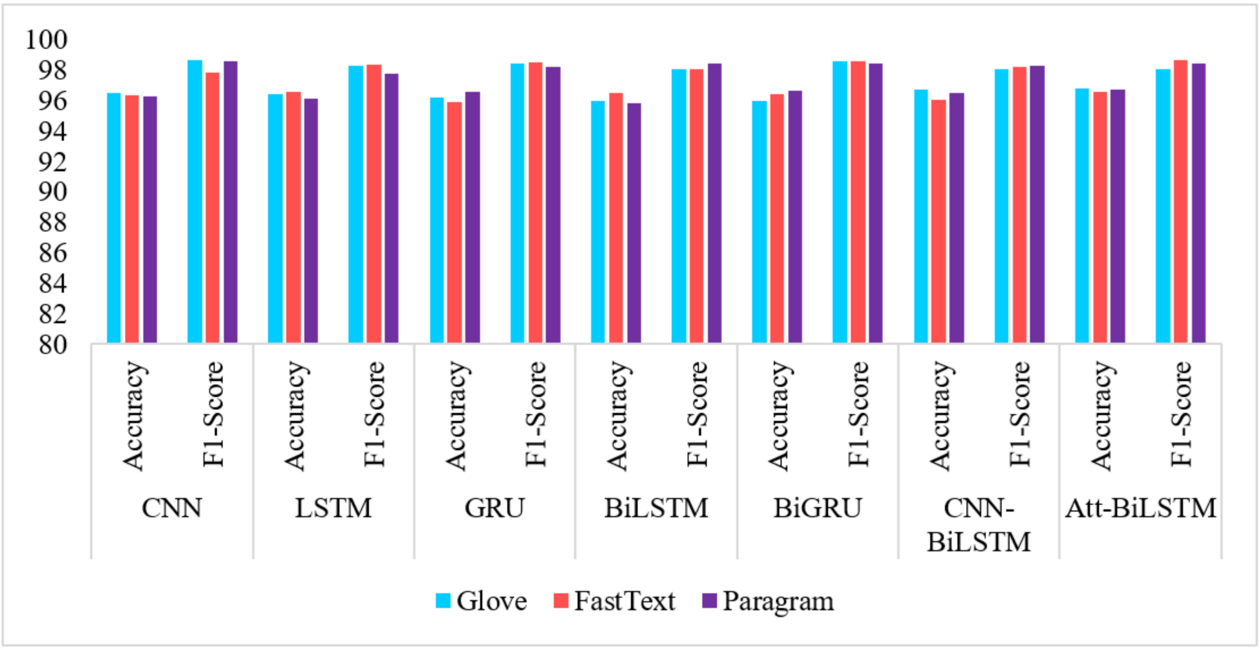

Figure 11. Performance comparison of seven shallow neural networks on Wikipedia Web Toxicity dataset.

Summarizing the results, all common metrics indicate good performance. Accuracy is well over 90% for the proposed shallow neural networks and just over 80% for all traditional machine learning models across all datasets. The neural network approaches demonstrate better performance than traditional machine learning algorithms. With only a single hidden layer, each neural network architecture provides high performance. The CNN-BiLSTM combination framework is equally capable as other proposed shallow networks. Additionally, the parametric settings of the network, neuron count, size of fully connected dense layers, and dropout probabilities, which have been decided upon experimentation, yield optimum results. The proposed architecture is observed to work well with all the neural networks utilized. We summarize the key observations as follows:

1. Neural networks demonstrated higher performance than state-of-the-art traditional machine learning algorithms due to their robustness and capability to handle large datasets.
2. Count Vectorization, although being an old statistical technique, manages to consistently provide good results.
3. Across all preprocessing steps, Logistic Regression displayed the highest average performance amongst all machine learning techniques used, followed by SVM, XG Boost, and Naïve Bayes in the said order.
4. GloVe embeddings resulted in a maximum number of high outputs than FastText and Paragram, although similar results were achieved by the other two methods in a similar fashion.
5. F1-measures convey high performance through all neural network models. By observing the accuracy scores, we conclude that RNN networks such as GRU, Bi-GRU, and

Bi-LSTM offered highest performance. Attention mechanism is also close to achieving results similar to these.

4.3. Baseline Comparison

In order to validate the efficacy of our work, we performed a baseline comparison with recent state-of-the-art techniques for cyberbully detection. The comparison of results on two datasets is provided in Table 3. The techniques used for comparison have been re-implemented in accordance with the environment settings mentioned in the existing studies. Bourgonje et al. [57] performed attack detection by using Bayes, Bayes Expectation Maximization, C4.5 Decision Trees, Multivariate Logistic Regression, Maximum Entropy, and Winnow2 on the Wikipedia dataset [17]. Logistic Regression and Bayes Expectation Maximum obtained 80.90% and 82.70% accuracies on the Wikipedia Attack dataset. Both these methods obtained 80.42% and 82.10% accuracies on the Wikipedia Web Toxicity dataset. The highest performance was obtained by their Bayes implementation on the Wikipedia Attack dataset with 83.11% accuracy and on Wikipedia Web Toxicity dataset with 82.19% accuracy. Agrawal and Awekar [27] used CNN, LSTM, Bi-LSTM, and Att-BiLSTM on the Wikipedia dataset, achieving highest accuracies of 92.91% and 93.52% with their CNN model. Bodapati et al. [58] achieved accuracy scores of 95.34% and 95.69% on Wikipedia Attack and Wikipedia Web Toxicity datasets, respectively, when using Bidirectional Encoder Representations from Transformers (BERT).

Table 3. Baseline comparison on Wikipedia Attack and Wikipedia Web Toxicity datasets.

Dataset	Method	A	P	R	F1
Wikipedia Attack Dataset	Bourgonje et al. [57] (Logistic Regression)	80.90	79.36	80.97	79.74
	Bourgonje et al. [57] (Bayes Exp. Max)	82.70	81.33	82.83	81.36
	Bourgonje et al. [57] (Bayes)	83.11	81.78	83.14	81.58
	Agrawal and Awekar [27] (CNN)	92.91	92.09	83.78	88.63
	Bodapati et al. [58] (BERT)	95.34	92.61	93.57	95.70
	Our Approach (Bi-GRU with GloVe)	**96.98**	**99.22**	**96.74**	**98.56**
Wikipedia Web Toxicity Dataset	Bourgonje et al. [57] (Logistic Regression)	80.42	78.91	80.46	79.23
	Bourgonje et al. [57] (Bayes Exp. Max)	82.10	80.60	81.87	80.57
	Bourgonje et al. [57] (Bayes)	82.19	80.68	82.01	80.60
	Agrawal and Awekar [27] (CNN)	93.52	92.79	88.67	91.56
	Bodapati et al. [58] (BERT)	95.69	92.71	95.11	96.82
	Our Approach (Bi-GRU with GloVe)	**96.01**	**99.45**	**96.8**	**98.63**

As observed by Table 3, the results obtained by our proposed methods have outperformed the existing approaches on these datasets. On the Wikipedia Attack dataset, our proposed model, Bi-GRU with GloVe embedding technique, achieves 96.98% accuracy and 98.56% F1-score, which is higher than the existing methods. On the Wikipedia Web Toxicity Dataset, the results achieved by Bi-GRU with GloVe embeddings outperformed the existing baselines with 96.01% accuracy and 98.63% F1-score. The achieved results are ~2–3% higher than the state-of-the-art methods in terms of F1 measure. This indicates that the proposed framework is also capable of handling class imbalances in the datasets. The evaluation metrics detailed in Table 3 validate the efficiency of our proposed methods. In addition, most of our proposed models outperformed the existing state of the-art, as observable in Table 2. It is notable that the proposed single layer neural network displays higher classification efficiency than the existing CNN and BERT deep models. For the Wikipedia Attack Dataset, the precision, recall and F1 measures achieved from all our shallow neural networks are higher than the existing methods [27,35,57,58]. For the Wikipedia Web Toxicity dataset, all the results achieved using neural network methods are exceptionally higher than the existing ones.

5. Conclusions and Future Prospects

With the expansion in the online space, cyberbullying has emerged as a ubiquitous problem having dire consequences on people and society. This research focuses on investigating several dimensions of cyberbullying detection. We explored eleven classification techniques, including traditional machine learning and shallow neural networks. In addition, we have also used seven types of feature extraction and embedding techniques. The results are established by performing experiments on two real-world datasets. We propose a novel neural network framework, establishing optimum network settings, dense, and dropout layer sizes. The framework accommodates various classifiers and achieves high results overall, outperforming several baselines. We provided a comparative study discussing the performance of all the methods utilized. The results are compared on a scale of four evaluation metrics in order to establish the concreteness of this study. The usefulness of this study lies in identifying robust mechanisms for online cyberbullying detection. Additionally, the proposal of shallow neural networks moderates the need of complex deep neural networks, thus economizing resources. We observe that neural networks highly outperform traditional machine learning algorithms. We establish that bidirectional neural networks perform better in all scenarios. The attention mechanism is also observed to perform exceptionally well. We observe that traditional machine learning algorithms such as SVM, Naïve Bayes, XGBoost, and Logistic Regression provide lower results compared to the shallow neural networks. Overall, we suggest using bidirectional RNNs and attention-based models for further advances in cyberbullying detection. This study paves a way towards developing better mechanisms to fight this online ailment.

Author Contributions: Conceptualization, G.B., B.N. and M.P.; methodology, A.A., C.R. and M.P.; software, A.A. and C.R.; validation, C.R., G.B. and M.P.; formal analysis, G.B. and M.P.; investigation, C.R. and B.N.; resources, C.R. and A.A.; data curation, C.R. and A.A.; writing—original draft preparation, C.R.; writing—review and editing, C.R., A.A., B.N. and M.P.; visualization, C.R.; supervision, M.P.; project administration, G.B. and B.N.; funding acquisition, M.P. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: Not applicable.

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

References

1. Moreno, M.A. Cyberbullying. *JAMA Pediatrics* **2014**, *168*, 500. [CrossRef] [PubMed]
2. Bu, S.J.; Cho, S.B. A hybrid deep learning system of CNN and LRCN to detect cyberbullying from SNS comments. In Proceedings of the International Conference on Hybrid Artificial Intelligence Systems, Oviedo, Spain, 20–22 June 2018; Springer: Cham, Switzerland, 2018; pp. 561–572. [CrossRef]
3. Mishra, P.; del Tredici, M.; Yannakoudakis, H.; Shutova, E. Author Profiling for Abuse Detection. In Proceedings of the 27th International Conference on Computational Linguistics, Santa Fe, NM, USA, 20–26 August 2018; pp. 1088–1098.
4. Pavlopoulos, J.; Malakasiotis, P.; Bakagianni, J.; Androutsopoulos, I. Improved Abusive Comment Moderation with User Embeddings. In Proceedings of the 2017 EMNLP Workshop: Natural Language Processing meets Journalism, Copenhagen, Denmark, 2 May 2017. [CrossRef]
5. Davidson, T.; Warmsley, D.; Macy, M.; Weber, I. Automated hate speech detection and the problem of offensive language. In Proceedings of the International AAAI Conference on Web and Social Media, Montreal, QC, Canada, 15–18 May 2017.
6. Djuric, N.; Zhou, J.; Morris, R.; Grbovic, M.; Radosavljevic, V.; Bhamidipati, N. Hate Speech Detection with Comment Embeddings. In Proceedings of the WWW 15 Companion: Proceedings of the 24th International Conference on World Wide Web, Florence, Italy, 18–22 May 2015. [CrossRef]
7. Nobata, C.; Tetreault, J.; Thomas, A.; Mehdad, Y.; Chang, Y. Abusive Language Detection in Online User Content. In Proceedings of the 25th International Conference on World Wide Web, Montreal, QC, Canada, 11–15 April 2016. [CrossRef]
8. Muneer, A.; Fati, S.M. A Comparative Analysis of Machine Learning Techniques for Cyberbullying Detection on Twitter. *Futur. Internet* **2020**, *12*, 187. [CrossRef]

9. Rawat, C.; Sarkar, A.; Singh, S.; Alvarado, R.; Rasberry, L. Automatic Detection of Online Abuse and Analysis of Problematic Users in Wikipedia. In Proceedings of the 2019 Systems and Information Engineering Design Symposium (SIEDS), Charlottesville, VA, USA, 26 April 2019. [CrossRef]
10. Waseem, Z.; Hovy, D. Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter. In Proceedings of the NAACL Student Research Workshop, San Diego, CA, USA, 13–15 June 2016. [CrossRef]
11. Badjatiya, P.; Gupta, S.; Gupta, M.; Varma, V. Deep Learning for Hate Speech Detection in Tweets. In Proceedings of the 26th International Conference on World Wide Web Companion—WWW '17 Companion, Perth, Australia, 3–7 April 2017. [CrossRef]
12. Kim, Y. Convolutional neural networks for sentence classification. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing, Doha, Qatar, 25–29 October 2014. [CrossRef]
13. Lu, N.; Wu, G.; Zhang, Z.; Zheng, Y.; Ren, Y.; Choo, K.R. Cyberbullying detection in social media text based on character-level convolutional neural network with shortcuts. *Concurr. Comput. Pr. Exp.* **2020**, *32*, e5627. [CrossRef]
14. Zhang, X.; Tong, J.; Vishwamitra, N.; Whittaker, E.; Mazer, J.P.; Kowalski, R.; Hu, H.; Luo, F.; Macbeth, J.; Dillon, E. Cyberbullying Detection with a Pronunciation Based Convolutional Neural Network. In Proceedings of the 2016 15th IEEE International Conference on Machine Learning and Applications (ICMLA), Anaheim, CA, USA, 18–20 December 2016. [CrossRef]
15. Warner, W.; Hirschberg, J. Detecting hate speech on the world wide web. In Proceedings of the LSM'12 Proceedings of the Second Workshop on Language in Social Media, Montreal, QC, Canada, 7 June 2012.
16. Reynolds, K.; Kontostathis, A.; Edwards, L. Using Machine Learning to Detect Cyberbullying. In Proceedings of the 2011 10th International Conference on Machine Learning and Applications and Workshops, Honolulu, HI, USA, 18–21 December 2011; Volume 2. [CrossRef]
17. Wulczyn, E.; Thain, N.; Dixon, L. Ex Machina. In Proceedings of the 26th International Conference on World Wide Web, Perth, Australia, 3–7 April 2017. [CrossRef]
18. Schmidt, A.; Wiegand, M. A Survey on Hate Speech Detection using Natural Language Processing. In Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media, Valencia, Spain, 3 April 2017. [CrossRef]
19. Qaiser, S.; Ali, R. Text Mining: Use of TF-IDF to Examine the Relevance of Words to Documents. *Int. J. Comput. Appl.* **2018**, *181*, 25–29. [CrossRef]
20. Allahyari, M.; Pouriyeh, S.; Assefi, M.; Safaei, S.; Trippe, E.D.; Gutierrez, J.B.; Kochut, K. A brief survey of text mining: Classification, clustering and extraction techniques. *arXiv* **2017**, arXiv:1707.02919.
21. Shah, F.P.; Patel, V. A review on feature selection and feature extraction for text classification. In Proceedings of the 2016 International Conference on Wireless Communications, Signal Processing and Networking (WiSPNET), Chennai, India, 23–25 March 2016. [CrossRef]
22. Dzisevic, R.; Sesok, D. Text Classification using Different Feature Extraction Approaches. In Proceedings of the 2019 Open Conference of Electrical, Electronic and Information Sciences (eStream), Vilnius, Lithuania, 25 April 2019. [CrossRef]
23. Kwok, I.; Wang, Y. Locate the hate: Detecting tweets against blacks. In Proceedings of the Twenty-Seventh AAAI Conference on Artificial Intelligence, Bellevue, WA, USA, 14 July 2013.
24. Yin, D.; Xue, Z.; Hong, L.; Davison, B.D.; Kontostathis, A.; Edwards, L. Detection of Harassment on Web 2.0. In Proceedings of the Content Analysis in the WEB, Madrid, Spain, 21 April 2009; Volume 2.
25. Tokunaga, R.S. Following you home from school: A critical review and synthesis of research on cyberbullying victimization. *Comput. Hum. Behav.* **2010**, *26*, 277–287. [CrossRef]
26. Themeli, C.; Giannakopoulos, G.; Pittaras, N. A study of text representations in Hate Speech Detection. *arXiv* **2021**, arXiv:2102.04521.
27. Agrawal, S.; Awekar, A. Deep Learning for Detecting Cyberbullying Across Multiple Social Media Platforms. In *Advances in Information Retrieval*; Springer: Cham, Switzerland, 2018; Volume 10772. [CrossRef]
28. Aroyehun, S.T.; Gelbukh, A. Aggression Detection in Social Media: Using Deep Neural Networks, Data Augmentation, and Pseudo Labeling. In Proceedings of the First Workshop on Trolling, Aggression and Cyberbullying, Santa Fe, NM, USA, 25 August 2018.
29. Aglionby, G.; Davis, C.; Mishra, P.; Caines, A.; Yannakoudakis, H.; Rei, M.; Shutova, E.; Buttery, P. CAMsterdam at SemEval-2019 Task 6: Neural and graph-based feature extraction for the identification of offensive tweets. In Proceedings of the 13th International Workshop on Semantic Evaluation, Minneapolis, MN, USA, 6–7 June 2019. [CrossRef]
30. Chen, H.; McKeever, S.; Delany, S.J. The use of deep learning distributed representations in the identification of abusive text. In Proceedings of the International AAAI Conference on Web and Social Media, Münich, Germany, 11–14 June 2019.
31. Chu, T.; Jue, K.; Wang, M. Comment Abuse Classification with Deep Learning. *Glob. J. Comput. Sci. Technol.* **2012**, *12*.
32. Anand, M.; Eswari, R. Classification of Abusive Comments in Social Media using Deep Learning. In Proceedings of the 2019 3rd International Conference on Computing Methodologies and Communication (ICCMC), Erode, India, 27–29 March 2019. [CrossRef]
33. Pavlopoulos, J.; Malakasiotis, P.; Androutsopoulos, I. Deep Learning for User Comment Moderation. In Proceedings of the First Workshop on Abusive Language Online, Vancouver, BC, Canada, 4 August 2017. [CrossRef]
34. Banerjee, V.; Telavane, J.; Gaikwad, P.; Vartak, P. Detection of Cyberbullying Using Deep Neural Network. In Proceedings of the 2019 5th International Conference on Advanced Computing & Communication Systems (ICACCS), Coimbatore, India, 15–16 March 2019. [CrossRef]

35. Agarwal, A.; Chivukula, A.S.; Bhuyan, M.H.; Jan, T.; Narayan, B.; Prasad, M. Identification and Classification of Cyberbullying Posts: A Recurrent Neural Network Approach Using Under-Sampling and Class Weighting. In *Information Processing and Management of Uncertainty in Knowledge-Based Systems*; Springer: New York, NY, USA, 2020; Volume 1333.
36. Salton, G.; Wong, A.; Yang, C.S. A vector space model for automatic indexing. *Commun. ACM* **1975**, *18*, 613–620. [CrossRef]
37. Shi, C.Y.; Xu, C.J.; Yang, X.J. Study of TFIDF algorithm. *J. Comput. Appl.* **2009**, *29*, 167–170.
38. Pennington, J.; Socher, R.; Manning, C.D. GloVe: Global Vectors for Word Representation. In Proceedings of the 2014 Conference on Empirical Methods in Natural Language Processing (EMNLP), Doha, Qatar, 25–29 October 2014. [CrossRef]
39. Bojanowski, P.; Grave, E.; Joulin, A.; Mikolov, T. Enriching Word Vectors with Subword Information. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 135–146. [CrossRef]
40. Joulin, A.; Grave, E.; Bojanowski, P.; Mikolov, T. Bag of tricks for efficient text classification. In Proceedings of the 15th Con-ference of the European Chapter of the Association for Computational Linguistics, Valencia, Spain, 3–7 April 2017; Volume 2. [CrossRef]
41. Mikolov, T.; Chen, K.; Corrado, G.; Dean, J. Efficient estimation of word representations in vector space. *arXiv* **2013**, arXiv:1301.3781v3.
42. Wieting, J.; Bansal, M.; Gimpel, K.; Livescu, K. From Paraphrase Database to Compositional Paraphrase Model and Back. *Trans. Assoc. Comput. Linguist.* **2015**, *3*, 345–358. [CrossRef]
43. Vulić, I.; Mrkšić, N.; Reichart, R.; Séaghdha, D.Ó.; Young, S.; Korhonen, A.; Barzilay, R.; Kan, M.-Y. Morph-fitting: Fine-Tuning Word Vector Spaces with Simple Language-Specific Rules. In Proceedings of the 55th Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Vancouver, BC, Canada, 30 July–4 August 2017; Volume 1. [CrossRef]
44. Mrkšić, N.; Séaghdha, D.Ó.; Thomson, B.; Gašić, M.; Rojas-Barahona, L.M.; Su, P.-H.; VanDyke, D.; Wen, T.-H.; Young, S. Counter-fitting Word Vectors to Linguistic Constraints. In Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, San Diego, CA, USA, 12–17 June 2016. [CrossRef]
45. Mrkšić, N.; Vulić, I.; Séaghdha, D.Ó.; Leviant, I.; Reichart, R.; Gašić, M.; Korhonen, A.; Young, S. Semantic Specialization of Distributional Word Vector Spaces using Monolingual and Cross-Lingual Constraints. *Trans. Assoc. Comput. Linguist.* **2017**, *5*, 309–324. [CrossRef]
46. Chen, T.; He, T.; Benesty, M. XGBoost: eXtreme Gradient Boosting. R package version 0.71-2. 1 August 2015.
47. Chen, T.; Guestrin, C. XGBoost: A Scalable Tree Boosting System. In Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, 13–17 August 2016. [CrossRef]
48. Sulzmann, J.-N.; Fürnkranz, J.; Hüllermeier, E. On Pairwise Naive Bayes Classifiers. *Lect. Notes Comput. Sci.* **2007**, *4701*, 371–381. [CrossRef]
49. Sarkar, A.; Chatterjee, S.; Das, W.; Datta, D. Text Classification using Support Vector Machine Anurag. *Int. J. Eng. Sci. Invent.* **2015**, *8*, 33–37.
50. Wright, R.E. Logistic Regression. In *Reading and Understanding Multivariate Statistics*; Grimm, L.G., Yarnold, P.R., Eds.; American Psychological Association: Washington, DC, USA, 1995; pp. 217–244.
51. Kalchbrenner, N.; Grefenstette, E.; Blunsom, P. A Convolutional Neural Network for Modelling Sentences. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers), Baltimore, MD, USA, 23–25 June 2014; Volume 1. [CrossRef]
52. Hochreiter, S.; Schmidhuber, J. Long short-term memory. *Neural Comput.* **1997**, *9*, 1735–1780. [CrossRef] [PubMed]
53. Hochreiter, S. The Vanishing Gradient Problem During Learning Recurrent Neural Nets and Problem Solutions. *Int. J. Uncertain. Fuzziness Knowl.-Based Syst.* **1998**, *6*, 107–116. [CrossRef]
54. Schuster, M.; Paliwal, K. Bidirectional recurrent neural networks. *IEEE Trans. Signal Process.* **1997**, *45*, 2673–2681. [CrossRef]
55. Cho, K.; Van Merriënboer, B.; Gulcehre, C.; Bahdanau, D.; Bougares, F.; Schwenk, H.; Bengio, Y. Learning phrase representations using RNN encoder-decoder for statistical machine translation. *arXiv* **2014**, arXiv:1406.1078. [CrossRef]
56. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, Ł.; Polosukhin, I. Attention is all you need. In Proceedings of the 31st Conference on Neural Information Processing Systems (NIPS 2017), Long Beach, CA, USA, 4–9 December 2017.
57. Bourgonje, P.; Moreno-Schneider, J.; Srivastava, A.; Rehm, G. Automatic Classification of Abusive Language and Personal Attacks in Various Forms of Online Communication. In *Transactions on Computational Science XI*; Springer Science and Business Media LLC: Cham, Switzerland, 2018; Volume 10713. [CrossRef]
58. Bodapati, S.; Gella, S.; Bhattacharjee, K.; Al-Onaizan, Y. Neural Word Decomposition Models for Abusive Language Detection. In Proceedings of the Third Workshop on Abusive Language Online, Florence, Italy, 1 August 2019. [CrossRef]