IJCRT.ORG ISSN : 2320-2882

INTERNATIONAL JOURNAL OF CREATIVE RESEARCH THOUGHTS (IJCRT)

An International Open Access, Peer-reviewed, Refereed Journal

Cyberbullying Detection On Social Media Using Machine Learning And NLP

Neha Uttam Tak

Dept. Data Science

(Zeal college of Engineering And Research)

Pune, India

Abstract: Cyberbullying has become a growing concern with the rise of social media platforms like Instagram, Twitter, Facebook, etc, where individuals can face harassment, threats, and abusive language. Early detection of such behaviour is crucial to ensure safe and healthy online environments. This study focuses on the development of a cyberbullying detection system using Machine Learning (ML) and Natural Language Processing (NLP) techniques. The proposed model processes user-generated content from social media platforms, extracting linguistic features such as sentiment, profanity, and semantic context. Pre-processing steps include text normalization, tokenization, and stop-word removal. Various supervised ML algorithms such as Logistic Regression, Support Vector Machines (SVM), Random Forest, and deep learning models like LSTM are trained on annotated datasets containing labelled examples of bullying and non-bullying content. Performance metrics such as accuracy, precision, recall, and F1-score are used to evaluate the models. The results demonstrate that combining NLP with ML can effectively identify cyberbullying, enabling timely intervention and contributing to safer digital spaces.

Keywords- Cyberbullying, Machine Learning, Natural Language Processing (NLP), Instagram, Twitter, Facebook.

Introduction: The widespread use of social media platforms has transformed the way people communicate, share information, and connect globally. However, this digital revolution has also given rise to negative behaviors, one of the most prominent being cyberbullying the act of using electronic communication to bully, harass, or threaten individuals, often repeatedly and anonymously. Victims of cyberbullying may suffer from emotional distress, mental health issues, and in severe cases, it can lead to tragic consequences such as self-harm or suicide.

Traditional methods of detecting and addressing cyberbullying, such as manual reporting and moderation, are often ineffective due to the high volume of content generated every second across platforms like Twitter, Instagram, and Facebook. As a result, there is a growing need for automated systems that can **detect** harmful or abusive language in real time, providing faster and more consistent responses to cyberbullying incidents.

In recent years, Machine Learning (ML) and Natural Language Processing (NLP) have

emerged as powerful tools for analyzing and understanding human language. By leveraging these technologies, it is possible to build models capable of identifying patterns in textual data that are indicative of cyberbullying. NLP techniques help preprocess and extract meaningful features from text, while ML algorithms use these features to classify content as bullying or non-bullying.

This study aims to explore the application of ML and NLP techniques in building an effective cyberbullying detection system. The goal is to create a model that can automatically identify and flag abusive content on social media, thereby contributing to safer and more respectful online communities.

II. Related works:

Recently several approaches have been introduced for detecting social media bullying. In this segment, the closely related research work that has been previously done detecting cyberbullying on social media sites is briefly described. Natural language processing (NLP) and machine learning (ML) have significantly contributed to the development of automated cyberbullying detection systems on social media platforms. Early research primarily relied on rule-based approaches and keyword filtering, which proved to be inadequate due to the complexity and contextual nature of online abuse.

Dinakar et al. [1] developed a classifier for cyberbullying detection in YouTube comments using machine learning algorithms, specifically focusing on sensitive topics like racism and sexuality. Their work illustrated the importance of topic-specific models in improving detection accuracy. Xu et al. [2] introduced a multi-modal framework that incorporated both textual and social features, such as user interaction patterns, which demonstrated improved performance compared to using text features alone.

Chavan and Shylaja [3] proposed a system using Support Vector Machines (SVM) to detect aggressive content on Twitter. Their model emphasized the need for linguistic feature extraction, including part-of-speech tagging and sentiment analysis, to accurately capture the emotional tone of abusive posts. Similarly, Nahar et al. [4] utilized a hybrid model combining text mining with graph-based techniques, which helped identify hidden relationships between users and potentially abusive content.

In recent years, deep learning methods have gained popularity in the field. Badjatiya et al. [5] employed word embeddings with LSTM (Long Short-Term Memory) networks and gradient-boosted decision trees to achieve better performance than traditional machine learning classifiers. This study highlighted the ability of deep neural networks to learn complex patterns in unstructured text data.

Mozafari et al. [6] explored the use of transformer-based architectures, particularly BERT (Bidirectional Encoder Representations from Transformers), to detect offensive language. Their fine-tuned BERT model outperformed previous approaches by effectively capturing semantic context and handling linguistic nuances such as sarcasm and idioms. Furthermore, studies such as those by Samghabadi et al. [7] demonstrated the benefits of multi-task learning, enabling models to simultaneously learn related tasks like hate speech and cyberbullying detection.

Despite these advancements, challenges persist. Many datasets used in these studies suffer from class imbalance, making it difficult to train models that generalize well. Additionally, the dynamic nature of language on social media—characterized by code-switching, evolving slang, and multilingual content—poses further complications. Researchers have begun exploring data augmentation, transfer learning, and ensemble methods to address these limitations [8].

For the training and testing of social platform bullying items, two classifier models were used, namely, Support Vector Machine (SVM) [9] and Naive Bayes [10]. Both the classifiers were able to identify the true positive scenarios, maintaining 71.25% and 52.70% accuracy individually. But SVM surpassed Naive Bayes on the same dataset for equivalent work.

III. Problem Statements And Objective

The rapid growth of social media platforms has revolutionized online communication, but it has also led to the emergence of negative

behaviours such as cyberbullying. Cyberbullying involves the use of digital platforms to harass, threaten, or humiliate individuals, often resulting in severe emotional and psychological consequences. It is a major issue via web- based networking media sites like Facebook and Twitter. Numerous people, particularly teenagers, endure antagonistic impacts, for example, discouragement, restlessness, brought down confidence and even absence of inspiration to live when being focused by menaces via web- based networking media.

Manual monitoring and reporting mechanisms are insufficient due to the massive volume and dynamic nature of online content. Additionally, the use of informal language, slang, sarcasm, and evolving vocabulary makes it challenging to detect abusive content using traditional rule-based systems.

Cyberbullying then again can be hard to distinguish and stop because of it happening on the web, regularly avoided the eyes of guardians and educators. The issue is to thought of an innovative methodology that can help in programmed discovery of tormenting via web-based networking media. The methodology explored is a framework able to do consequently distinguishing and revealing occasions of harassing via web-based networking media stages.

The objective of this undertaking is to produce information on how a programmed framework for distinguishing harassing via web-based networking media can be built. To accomplish the objective inquiring about cyberbullying is begin. Characterizing the idea and to what degree it happens via webbased networking media.

The primary objective of cyberbullying detection on social media using machine learning and NLP is to automatically identify and classify posts or comments that are abusive, hateful, or harmful, thus enabling platforms to take appropriate action and mitigate the harm caused by cyberbullying. This involves leveraging NLP techniques to understand the meaning and context of text, and machine learning models to learn patterns and predict whether a given piece of text is likely to be bullying-related.

IV. Proposed Methodology

In this work, we propose an end-to-end methodology for detecting cyberbullying content on social media using Natural Language Processing (NLP) techniques and Machine Learning (ML) models. The methodology consists of several key stages, including data collection, preprocessing, feature extraction, model training, and evaluation. The overall workflow of the proposed approach is illustrated in **Fig. 1**, which depicts the interaction between each stage.

A. Data Collection

We have used Dataturks' Tweet Dataset for Cybertroll Detection obtained from Kaggle [1] for reaching the final results. Because of the seriousness of the issue we aim to resolve, it was crucial to choose a dataset that was complete, reliable, relevant, and to the point. While we considered many other datasets as well, many of them either had missing attributes, were too low in quality, or were found to have irrelevant data after manual inspection. Thus, after having tried out of many other open sourced datasets, we came down to [1] as it seemed in line with all the parameters required. Here is the Detailed Description of the dataset: 1) It is a partially manually labelled dataset. 2) Total Instances: 20001 The dataset has 2 attributes- tweet and label [0 corresponds to No while 1 corresponds to Yes] B. Data Cleaning The dataset used was set in a json format. Since the fields of the dataset were relatively simple to interpret, the original set of fields in the annotation attribute was removed, and filled with the label values to simplify the next step.

B. Data Cleaning

The dataset used was set in a json format. Since the fields of the dataset were relatively simple to interpret, the original set of fields in the annotation attribute was removed, and filled with the label values to simplify the next step.

C. Data Preprocessing

The pre-processing steps were done as follows using the nltk library along with regex:

1) Word Tokenization: A Token is a single entity that is building blocks for sentence or paragraph. Word Tokenization converts our text to separate words in a list.

2) Stop words filtering is done using nltk.corpus.stopwords.words('english') to fetch a list of stopwords in the English dictionary, after which they are removed. Stop words are words such as "the", "a", "an", "in", which are not significant and do not affect the meaning of the data to be interpreted.

3) To remove punctuation, we save only the characters that are not punctuation, which can be checked by using string.punctuation .

4) Stemming: Stemming is a process of linguistic normalization, which reduces words to their word root word. We stem the tokens using nltk.stem.porter.PorterStemmer to get the stemmed tokens. For example, connection, connected, connecting word reduce to a common word "connect".

5) Digit removal: We also filtered out any numeric content as it doesn't contribute to cyberbullying.

6) Now the next step was to extract features so that it can be used with ML algorithms, for which we used TF-IDF Transform using Python's sklearn libary. TF-IDF is a statistical measure to evaluate the relevance of a word, which is basically calculated by multiplying the number of times that words appeared in the document by the inverse document frequency of the word. TF-IDF uses the method diminishing the weight (importance) of words appeared in many documents in common, considered them incapable of discerning the documents, rather than simply counting the frequency of words as CountVectorizer does. The outcome matrix consists of each document (row) and each word (column) and the importance (weight) computed by tf * idf (values of the matrix). If a word has high tf-idf in a document, it has most of the times occurred in given documents and must be absent in the other documents. So the words must be a signature word.

Attribute evaluation is done manually as can be seen where we have printed the top 25 words according to the calculated tf-idf score. Some Top ranked words for the dataset were: [hate, fuck, damn, suck, ass, that, lol, im, like, you, it, get, what, no, would, bitch].

D. Data Resampling

As the data was skewed, Resampling had to be performed on the training data, Firstly the data was split into Training and Test in 80:20 ratio and resampling was performed on the training data.

- As we had ample data to work with, we used oversampling of the minority class. This means that if the majority class had 1,000 examples and the minority class had 100, this strategy would oversampling the minority class so that it has 1,000 examples.
- For Oversampling, RandomOverSample function is used from imblearn package for all the "not majority" classes which in our case, was only the 1 minority class.

After resampling, the training data had 9750 CB & NON-CB instances.

V. DESCRIPTION OF METHODS

A. Gaussian Naive Bayes

Naive Bayes classifiers are a collection of classification algorithms based on Bayes' Theorem of mathematics. In simple words, the Bayes' theorem describes the probability of an event, based on prior knowledge of conditions that might be related to the event. It is not a single algorithm but a family of algorithms where all of them share a common principle, i.e. every pair of features being classified is independent of each other. Naive Bayes is a classification algorithm for binary (two-class) and multi-class classification problems. The technique is easiest to understand when described using binary or

categorical input values. It is called naive Bayes because the calculation of the probabilities for each hypothesis are simplified to make their calculation tractable. Naive Bayes can be extended to real-valued attributes, most commonly by assuming a Gaussian distribution. This extension of Naive Bayes is called Gaussian Naive Bayes. Beside the Gaussian Naive Bayes there are also existing the Multinomial naive Bayes and the Bernoulli naive Bayes. We picked the Gaussian Naive Bayes because it is the most popular one and one of the simplest to implement because we only need to estimate the mean and the standard deviation from the training data. The classifier was implemented using sklearn.naive bayes package.

B. Logistic Regression

Regression analysis is a predictive modelling technique that analyzes the relation between the target or dependent variable and independent variable in a dataset. Regression analysis techniques get used when the target and independent variables show a linear or non-linear relationship between each other, and the target variable contains continuous values. Regression analysis involves determining the best fit line, which is a line that passes through all the data points in such a way that distance of the line from each data point is minimized. Logistic regression is one of the types of regression analysis technique, which gets used when the dependent variable is discrete. Example: 0 or 1, true or false, etc. This means the target variable can have only two values, and a sigmoid curve denotes the relation between the target variable and the independent variable, by mapping any real value to a value between 0 and 1. We chose Logistic Regression as the size of our data set was large, and it had almost equal occurrence of values to come in target variables. Moreover, there was no correlation between independent variables in the dataset. The classifier was implemented using sklearn.linear model package.

C. Decision Tree Classifier

A Decision Tree is constructed by asking a series of questions with respect to the dataset. Each time an answer is received, a follow-up question is asked until a conclusion about the class label of the record. The series of questions and their possible answers can be organised in the form of a decision tree, which is a hierarchical structure consisting of nodes and directed edges. It has 3 types of nodes: Root, Internal, and Leaf nodes. In a decision tree, each leaf node is assigned a class label. The non-terminal nodes, which include the root and other internal nodes, contain attribute test conditions to separate records that have different characteristics. Using the decision algorithm, we start at the tree root and split the data on the feature that results in the largest information gain (IG) (reduction in uncertainty towards the final decision). In an iterative process, we can then repeat this splitting procedure at each child node until the leaves are pure. This means that the samples at each leaf node all belong to the same class. The classifier was implemented using sklearn.tree package.

D. Adaboost Classifier

AdaBoost is an iterative ensemble method. The general idea behind boosting methods is to train predictors sequentially, each trying to correct its predecessor. AdaBoost classifier builds a strong classifier by combining multiple poorly performing classifiers so that you will get high accuracy strong classifier. The basic concept behind Adaboost is to set the weights of classifiers and training the data sample in each iteration such that it ensures the accurate predictions of unusual observations. Any machine learning algorithm can be used as base classifier if it accepts weights on the training set. At a high level, AdaBoost is similar to Random Forest as they both tally up the predictions made by each decision trees within the forest to decide on the final classification. There however, lie some subtle differences. In AdaBoost, the decision trees have a depth of 1 (i.e. 2 leaves). In addition, the predictions made by each decision tree have varying impact on the final prediction made by the model. Rather than taking the average of the predictions made by each decision tree in the forest (or majority in the case of classification), in the AdaBoost algorithm, every decision tree contributes a varying amount to the final prediction. The classifier was

implemented using sklearn.ensemble package.

E. Random Forest Classifier

As its name implies, Random Forest Classifier consists of a large number of individual decision trees that operate as an ensemble. Each individual tree in the random forest spits out a class prediction and the class with the most votes becomes our model's prediction. The low correlation between models is the key as they can produce ensemble predictions that are more accurate than any of the individual predictions, as the trees protect each other from their individual errors. The process of Bagging is used to diversify models as each individual tree is allowed to randomly sample from the dataset with replacement. The classifier was implemented using sklearn.ensemble package.

VI. EXPERIMENT AND RESULTS

For our supervised learning technique analysis, we've used Naive Bayes(Gaussian), Logistic regression, and J48 Decision Tree as the standard methods. As Ensemble methods, we have used AdaBoost and RandomForest Classifiers. In our research, we found that the Gaussian Naive Bayes classifier performed the poorest, whereas the Random Forest Classifier gave the best result in terms of every metric.[fig1 & fig2]. It wasn't surprising to see the Random Forest classifier performing the best. The Decision Tree classifier performed better than Naive Bayes classifier and Logistic Regression. The Random Forest Classifier came out on top in all the performance metrics, which was expected as it is an extension of the Decision Tree classifier, averaging out results of multiple recursions of the same. The Metrics used for determining the performance of models are as follows:

- $$F - Measure = \frac{(2 \times Precison \times Recall)}{Precision + Recall}$$
- $$Accuracy = \frac{TP + TN}{TP + TN + FP + FN}$$
- $$Precision = \frac{TP}{TP + FP}$$
- $$Recall = \frac{TP}{TP + FN}$$
- ROCArea, which denotes the area under the curve formed by plotting TP rate.

where,
TP = No. of True Positives
TN = No. of True Negatives
FP = No. of False Positives
FN= No. of False Negatives

Traditional Supervised Learning used: NaiveBayes, Logistic Regression and J48 Decision Trees classifier The Ensemble Learning Methods used: AdaBoost and Random Forest classifier

Figure 1 and Figure 2 shows a graphical comparison between the aforementioned algorithms.

Note: Table IV represents the weighted average using both the classes(hate speech and non hate speech) for Precision, Recall, and F1 score.

First column and row of the confusion matrices represents Cyberbullying class whereas the second row and column represents Non-cyberbullying class.

TABLE IV
SUPERVISED TRADITIONAL METHODS

	NaiveBayes	Regression	DecisionTree
Accuracy	0.62	0.80	0.85
Precision	0.79	0.81	0.88
Recall	0.62	0.80	0.85
F1-Score	0.59	0.81	0.85
ROCArea	0.68	0.81	0.87
Confusion Matrix	925 1504 31 1541	1920 509 274 1298	1896 533 67 1505

TABLE V
SUPERVISED ENSEMBLE METHODS

	AdaBoost	Random Forest
Accuracy	0.71	0.92
Precision	0.74	0.92
Recall	0.71	0.92
F1-Score	0.72	0.92
ROCArea	0.73	0.92
Confusion Matrix	1616 813 332 1240	2175 254 73 1499

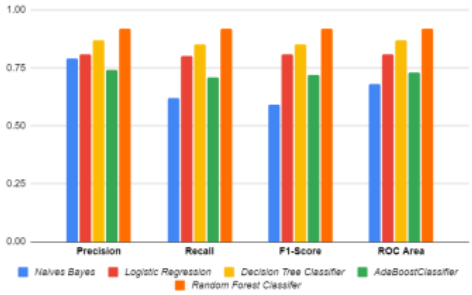

Fig. 1. Precision, Recall, F1-Score, ROCArea

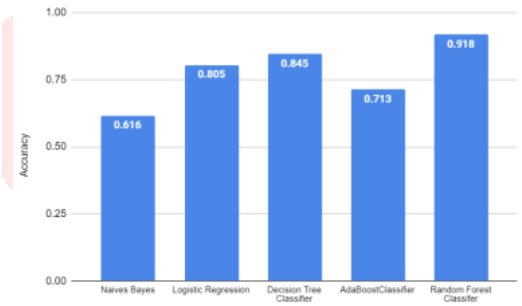

Fig. 2. Accuracy

VII. CONCLUSION AND FUTURE SCOPE

In this paper, We did a comparative study between various Supervised algorithms, additionally also comparing various Supervised Ensemble methods as well. The overall best performance was shown by Random Forest classifier, giving an accuracy of about 92%. The Ensemble methods performed equal to, or better than the Supervised methods but still, We observed a high True positive rate for the cyberbullying class in all the ensemble methods, which is much more desirable. Naive Bayes performed the worst, giving just 61% accuracy. Through this paper, we evaluated our approach and compared it with other papers in the section "Related Work". We also observed that none of the studied past researches used any semi-supervised methods, probably because they are not that popular or effective an didn't give any commendable result in comparison to the supervised methods. A very notable fact to be addressed is also the lack of labelled datasets and non-holistic consideration of cyberbullying by researchers when developing detection systems. These are two key challenges facing cyberbullying detection research. Another challenge faced was the lack of resources, due to which we were not able to analyze the performance of SVM(Support Vector Machine) or MultiLayer Perceptron(Neural Networks) classifiers. They have however been mentioned in our study for reference. Future work on cyberbullying can also benefit by using Dimensionality Reduction as the number of features in this case can be quite high as seen in our example. PCA(Principal Component Analysis) and LDA(Linear Discriminant Analysis) are few common techniques used for this purpose which have the ability to play a really important role in machine learning, especially when working with thousands of features. Principal Components Analysis are one of the top dimensionality reduction algorithms, and in addition to making the work of feature manipulation easier, it can also help to improve the results of the classifier. The idea is to explore advantages and disadvantages of each one and check its results individually and combined as well.

REFERENCES

[1] DataTurks. (2018, July 12). Tweets Dataset for Detection of Cyber-Trolls. Retrieved November 07, 2020, from https://www.kaggle.com/dataturks/dataset-for-detection-ofcybertrolls?select=Dataset+for+Detection+of+Cyber-Trolls.json

[2] Samghabadi, Niloofar Safi, et al. "Detecting nastiness in social media." Proceedings of the First Workshop on Abusive Language Online. 2017.

[3] Yao, Mengfan, Charalampos Chelmis, and Daphney? Stavroula Zois. "Cyberbullying ends here: Towards robust detection of cyberbullying in social

media." The World Wide Web Conference. 2019.

[4] Huang, Qianjia, Vivek Kumar Singh, and Pradeep Kumar Atrey. "Cyberbullying detection using social and textual analysis." Proceedings of the 3rd International Workshop on Socially-Aware Multimedia. 2014.

[5] T. Bin Abdur Rakib, L. K. Soon, in Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics) (Springer Verlag, 2018), vol. 10751 LNAI, pp. 180–189.

[6] Y. N. Silva, D. L. Hall, C. Rich, BullyBlocker: toward an interdisciplinary approach to identify cyberbullying. Social Network Analysis and Mining. 8 (2018), doi:10.1007/s13278-018-0496-z.

[7] E. Raisi, B. Huang, Weakly supervised cyberbullying detection with participant-vocabulary consistency. Social Network Analysis and Mining. 8 (2018), doi:10.1007/s13278-018-0517-y.

[8] Homa Hosseinmardi, Sabrina Arredondo Mattson, Rahat Ibn Rafiq, Richard Han, Qin Lv, Shivakant Mishra. (2015). Detection of Cyberbullying Incidents on the Instagram Social Network. "

[9] Dadvar, Maral Eckert, Kai. (2018). Cyberbullying Detection in Social Networks Using Deep Learning Based Models; A Reproducibility Study. 10.13140/RG.2.2.16187.87846.

[10] Nandhini, B. Sri, and J. I. Sheeba. "Cyberbullying detection and classification using information retrieval algorithm." Proceedings of the 2015 International Conference on Advanced Research in Computer Science Engineering Technology (ICARCSET 2015). 2015. APPENDIX A FURTHER STUDIES ON CYBERBULLYING DETECTIO