International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 10 Issue: 03 | March - 2026 SJIF Rating: 8.659 ISSN: 2582-3930

Cyberbullying Detection: Identifying Hate Speech Using Machine Learning In Real Time Chat Application

ALLU ANEESHA[*1], KOTA HARSHITHA[2], CHITTIBOMMALA SURYA[3], GATTI HEMA SRIDEVI[4], PITTU MADHU BRAHMA REDDY[5]

[1]Assistant Professor, Department of CSE(CB), Bapatla Engineering College, Bapatla 522101, AP,India
[2]Student, Department of CSE(CB),Bapatla Engineering College, Bapatla 522101,AP, India
[3]Student, Department of CSE(CB), Bapatla Engineering College, Bapatla 522101, AP, India
[4]Student, Department of CSE(CB), Bapatla Engineering College, Bapatla 522101, AP, India
[5]Student, Department of CSE(CB), Bapatla Engineering College, Bapatla 522101, AP, India

***Abstract* —** Social media and online communications have become totally dominant in the digital age. However, with the growth of social media and the internet has come the emergence of both facets of bullying — cyberbullying and hate speech — online bullying that substantially negatively affects the psychological and social well-being of its victims, particularly among vulnerable populations. The goal of this paper is to provide a solution to automate the detection of cyberbullying and hate speech through machine learning techniques using natural language processing (NLP). The proposed system will utilize NLP techniques to identify relevant features of text data so that a number of classification algorithms, including logistic regression, support vector machines (SVMs), and several types of deep learning models (such as long short-term memory (LSTM) networks or BERT), can be trained and tested. A labeled dataset containing tweets and comments made on blogs will be used to create and evaluate the models, and the models will be evaluated based on accuracy, precision, recall, and F1 score. The results from this work will ultimately result in a comprehensive, scalable, and reliable means to identify abusive online content leading to a safer and more respectful online culture. Additionally, the result of this work could be incorporated into the social media sites to provide real-time monitoring and content moderation.

Key Words — Cyberbullying Detection, Hate Speech, Machine Learning, NLP, TF-IDF, SVM, LSTM, BERT, Text Classification, Content Moderation.

II. INTRODUCTION

There have been large increases in the use of social media like Twitter, Facebook, Instagram and other forms of online communication such as forums. These technologies gave us more ways than ever before to connect and share information with one another. However, they have also created new problems for people, including doing "harmful things" to others such as through cyberbullying and hate speech.

Cyberbullying includes a variety of sorts of abuse that are carried out using some digital channel of communication including harassing someone; defaming someone (making false statements about someone); impersonating someone; and using emotional manipulation.

There are serious and large-scale consequences to being the victim of cyberbullying. Many victims suffer from anxiety, depression, and social withdrawal (isolation) and in some cases commit suicide. The challenge is made even more difficult when considering that the daily amount of content generated by users is so large that manually moderating content is impractical and inefficient.

Keyword or rule-based filtering systems do not provide sufficient context to understand whether language is intended to be hateful or abusive. In addition, many new words are frequently created; new slang is frequently developed; and the meaning of words continually evolves; therefore, traditional filtering systems provide no way for an adequate understanding of current language or the many subtle ways that someone can express hatred or abusive behavior.

This paper proposes an automated intelligent system that uses machine learning and natural language processing techniques to detect both cyberbullying as well as hate speech in user-generated text. The proposed system will automatically classify text into three categories — cyberbullying, hate speech, and neither hateful nor abusive — with high levels of accuracy. The system has been designed and built to scale to support moderating user generated content as well as to improve the safety of online communities.

III. LITERATURE SURVEY

For over 20 years, research on detecting cyberbullying and hate speech with automation has changed from basic methods relying on rules to more complex approaches using deep learning. The following section discusses some of the most important contributions to this area of research.

3.1 Rule-Based and Keyword Filtering

The first automated systems for detecting offensive content used lexicons and keyword filtering as their basis for detection. These systems kept a "blacklist" of offensive words and flagged any content that included those words. While they were relatively easy to implement, the false alarm rate for the systems was very high because the methods did not account for context or tone, and therefore did not have any way to determine whether the original authors intended to use the flagged words in an appropriate manner. Many times, the

same words were also used positively or were written in an "ironic" manner. By the same token, misspelled words and coded language enabled serious offenses to evade detection completely. Yin et al. (2009) illustrated that using only a lexicon-based method failed to capture the complex semantic meanings of online abusive behavior.

3.2 Old Methods of Artificial Intelligence (AI)

The restrictions of rule-based systems created a need for better solutions, so machine learning was conceived. Some of the best-known machine-learning standards used on text-classifying jobs were Naive Bayes classifiers (NB), Support Vector Machines (SVM), and Logistic Regression models using engineered features like n-grams, bag-of-words models, and TF-IDF vectors. Age-old Aspects Nobata et al. (2016) presented favourable results for abusive language detection by employing a model combining their linguistic, syntactic, and distributional features within a regression method; Davidson et al. (2017) shared an important dataset and a report that compares TF-IDF features with Logistic Regression to identify hate language from offensive language not perceivable as hate; however, they also elucidated the difficulty of determining the finer distinctions between these types of language.

These older types of methods providing better accuracies than their rule-based counterparts are also limited because they rely on human feature engineering for input data and linear models do not capture semantic relationships that are nonlinear and complex.

3.3 Deep Learning Approaches

Deep learning architectures revolutionised text classification and led to major advances in the accuracy of classifying text documents. Kim (2014) was among the first to explore using CNNs (Convolutional Neural Networks) for text data and achieved excellent performance on sentence classification tasks. RNNs (Recurrent Neural Networks) and LSTMs (Long Short Term Memory Networks) have been shown to capture sequential (temporal) dependencies in the text, which have proven to be very important for identifying patterns of hate speech (Hochreiter & Schmidhuber, 1997). Zhang et al. (2018) combined a CNN and a GRU (Gated Recurrent Unit) architecture with the goal of detecting Twitter hate speech; they achieved state-of-the-art performance at that time. Further improvements in performance gained through the implementation of attention mechanisms allowed models to focus on the most discriminating areas of input text when making predictions.

3.4 Transformer-Based Models

The introduction of Transformer Model with BERT (Bidirectional Encoder Representations from Transformers) by Devlin et al. (2019) gave rise to a complete change in how systems perform NLP (Natural Language Processing) tasks. When pre-trained on massive text corpora then fine-tuned for specific downstream tasks, BERT and its variants (RoBERTa, DistilBERT and ALBERT) have become the standard for achieving exceptional accuracy on virtually all major NLP benchmarks. It was shown that BERT fine-tuned for detecting hate speech (Mozafari et al., 2020) significantly outperformed all previous methods of detecting hate speech. At the same time, the coming computational power required for running BERT-related models will likely continue to inhibit their use in real-time on systems with limited computational resources.

Research gaps:

In terms of literature identify an imbalance between detection accuracy and operational efficiencies. Although BERT as a representation from a deep neural network shows outstanding performance, it also requires large amounts of computing power to operate. Traditional algorithms, while more efficient and able to be deployed in production at relatively low cost, tend to have lower detection accuracy for contextually complex specimens. Thus, there is a need for developing systems that strike a balance between excellent detection accuracy and acceptable operational costs. This research aims to provide an alternative method for achieving that balance.

IV. EXISTING SYSTEM

There are currently three primary types of methods for managing cyberbullying and hate speech on online platforms: manual moderation, rule-based automated filtering, and early machine learning.

4.1 Manual Content Moderation:

Flagged content (or randomly sampled content) are examined by human moderators to determine whether or not they violate platform policies. This is the predominant method across many of the largest social media platforms. Human moderators have the capacity to make nuanced moral judgments about whether or not an instance of potential hate speech violates platform policy; however, manual content moderation is unscalable due to the extreme volume of content generated across platforms like Twitter where there are in excess of 500 million tweets produced each and every day and therefore there is an insufficient number of human moderators to keep pace with (rate that they are processing) the volume of content produced. Additionally, humans will incur psychological trauma from the repeated exposure to abusive behavior, which has resulted in high staff turnover rate among moderators.

4.2 Keyword-Based Filtering Rules

Automated keyword-based filtering flags content that contains predetermined lists of offensive language (e.g. swearing). Automated keyword based filtering systems are fast to set up and operational, but they have many important issues:

- They are complete disregard of context, resulting in flagging benign use of sensitive words whilst failing to catch abusive content that is disguised through intentional misspellings, l33tspeak, and obscure or coded language.
- They require ongoing manual intervention to keep up with changing language.
- They have high false positive rates (resulting in legitimate speech being blocked) and

high false negative rates (resulting in actual abuse being overlooked).

- They cannot identify indirect harassment, subtle discriminatory speech, or bullying behaviour in context.

This clearly indicates and underlines the need for an intelligent, context-sensitive, scalable system that is capable of accurately detecting cyberbullying and hate speech across the entire spectrum of online language.

V. PROPOSED SYSTEM

The suggested system overcomes the limitations of current methods by creating an end-to-end Machine Learning pipeline to automatically classify cyberbullying and hate speech. The pipeline accepts and ingests source material (i.e. raw text) through social media and other online communication, and processes this raw text through an NLP pipeline, to normalize the text, to extract features from the text, and then to classify the text to one of three classes; Cyberbullying, Hate Speech, or Non-Offensive.

5.1 Definitive Features of Suggested System

- Automated, NLP-based Pre-processing that processes informal writing (i.e. slang, abbreviations, noise, at social media).
- Tokenizer feature extraction will determine the importance of tokens to allow for more distinct class differentiation.
- Multiple classifiers evaluated in parallel (Logistic Regression, Naive Bayes, SVM, LSTM, and BERT) so that the best model can be selected based on the needs of deployment.
- Multi-classification that classifies the text into three distinct classes, rather than using a simple harmful/non-harmful label, or classifying the text as harmful/non-harmful.
- Scalable architecture for real-time integration with Social Media APIs.
- Evaluation of accuracy, precision, recall and F1-score, in accordance with industry-standard evaluations.

5.2 Advantages Over Existing Systems

Table 1: Comparison of Existing vs. Proposed System

Feature	Existing System	Proposed System
Language Understanding	Keyword matching only	Semantic NLP + ML classification
Context Awareness	None	TF-IDF, LSTM, BERT capture context
Scalability	Low (manual moderation)	High (automated pipeline)
Accuracy	Low to moderate	High (especially with BERT)
Adaptability	Manual rule updates required	Model retraining with new data
Classification Granularity	Binary (offensive / not)	Three-class: bully / hate / normal
Real-Time Detection	Limited	Fully supported

VI. SYSTEM ARCHITECTURE

The architecture for a cyberbullying detection system consists of multiple layers within a modular system of five major components including a data input layer, preprocessing, feature extraction, classification and output layers. Each component has a function assigned to it that allows it to communicate with its neighbouring module or component via clearly defined interfaces or protocols, allowing for easier maintenance and the ability to make changes later without impacting other areas of the system.

6.1 Architecture Overview

Fig 1: Proposed System Achitecture

6.2 Data Flow

Text data initially enters through the Data Input Layer (DIL) and can be ingested via multiple methods, including batch processing of labeled CSV datasets for training and real-time single input processing for application deployment. Once data is in the DIL, it passes sequentially through the Pre-processing Module where it is cleaned and normalised, then to the Feature Extraction Module where it generates Term Frequency-Inverse Document Frequency (TF-IDF) feature vectors. These feature vectors are provided as input to the trained Classification Module, which outputs class label and confidence score. The Output Layer formats the Classification Module's output and returns the result to either the calling application or moderation dashboard.

6.3 Module Interactions

- Processed text data will not affect other processed text data because the Preprocessing Module is stateless, and each piece of text is processed independently and thus can be processed in parallel.
- The TF-IDF Vectorizer is produced from the training corpus and serialised after the completion of training. During the inference period, the vectorizer will produce TF-IDF vector for each new input utilizing the vocabulary produced during the training of the vectorizer.
- The Classification Module permits interchangeable models to be plugged-and-played with the Classification Module. Additionally, any model used must not require modification to either the Preprocessing Module and/or Feature Extraction Module.
- The system generates a moderation log of all classifications, input text hashes, and confidence scores to facilitate the auditing and retraining of the system

VII. METHODOLOGY

Methodology follows a systematic pipeline, which is reproducible. The pipeline includes data collection, data preprocessing, feature engineering, model training and evaluating.

7.1 Datasets

The system will be built and evaluated from existing public labelled datasets of online text that have previously been used in research on hate speech and cyberbullying. Examples include the Twitter hate speech dataset by Davidson et al. (2017) and the Kaggle Cyberbullying Classification dataset. The combined dataset contains samples of text that have been labelled along three classes: Cyberbullying, Hate Speech and Non-Offensive. The class distribution of the combined dataset is similar to real-life situations. As such, this research will also apply stratified random sampling and class weights will be used where appropriate during model training.

7.2 Data Preprocessing

- All text samples go through the following series of steps of Natural Processing (NLP) in terms of pre-processing their data.
- Lowercase all text characters so they will match up as tokens no matter how many uppercase letters there are.
- Remove all hyperlinks and HTML markup from text using regular expressions.
- Remove punctuation marks (like periods, commas, semicolons, question marks, etc.), neutralize emoticons (smiley or sad faces), and remove any other characters in the text that have no meaning, including non-alphabetic and non-numeric.
- Split all forms of text into individual words using NLTK's word tokenizer.
- Remove high-frequency (low meaning) words as defined in the list of English stop words provided by NLTK.
- Convert all tokens in the form of words to their roots or base forms using Porters' Stemmer and WordNets' Lemmatizers. This will minimize the total number of different words in the overall vocabulary and help to make the document generated from this processing as general as possible.
- Normalise any remaining abbreviations, duplicated characters and informal 'slang' by using custom dictionary look-up.

7.3 Feature Extraction

The processed text, which is then converted into numerical feature representations suitable for use by machine learning algorithms. The main method used to accomplish this is TF-IDF (term frequency-inverse document frequency), which assigns a weight that is determined by how often the word occurs in relation to the total number of times that word appears throughout all documents within a corpus, thereby reflecting the usefulness of any given term as a discriminative feature.

When working with deep learning models (e.g., LSTM and BERT), instead of using TF-IDF or other traditional feature extraction methods, pre-trained word embeddings are utilized. Specifically, the LSTM model will use pre-trained GloVe (Global Vectors for Word Representation) embeddings of size 100, which encode information about the meaning of each word using dense vector representations (i.e., vectors with many elements).

BERT utilizes contextual sub-word embeddings that are created by the Transformer architecture. The context for each sub-word is bi-directional and does not require feature engineering to be explicitly defined.

7.4 Classification Models

The five classification model types are:

- Logistic Regression (LR): This is a linear classifier that calculates the log-odds of a class being present or absent using a linear function of the feature vector. It serves as the main baseline for comparison.
- Naive Bayes (NB): This is classified as a probabilistic classifier using Bayes' theorem based on the assumptions of feature independence (is a probabilistic classifier). Despite its simple assumptions regarding the independence of features, it usually performs well for text classification problems.
- Support Vector Machine (SVM): This type of model is characterized as a maximum-margin classifier based on support vectors and finding the hyperplane to best divide the two classes in the feature space. This classifier is especially well suited

for high-dimensional feature vectors created from TF-IDF. It uses a linear kernel.

- LSTM (Long Short-Term Memory): This type of model uses a recurrent neural network that can capture long-range sequential dependencies in text. It consists of an input embedding layer, two stacked LSTM layers, a dropout layer for regularization, and a softmax output layer.
- BERT (Bidirectional Encoder Representations From Transformers): This is a pre-trained language model called "bert-base-uncased," which will be fine-tuned to the cyberbullying dataset. BERT allows the entire input sequence to be processed bidirectionally, thereby recovering rich context between tokens in an input sequence.

7.5 Evaluation Metric

Each of these models is evaluated based on their performance on a test set that was not used for training and makes up 20% of the total data set. Evaluation metrics include the following:

Table 2: Evaluation Metrics

Metric	Formula	Interpretation
Accuracy	TP + TN / (TP + TN + FP + FN)	Overall proportion of correct predictions
Precision	TP / (TP + FP)	Fraction of positive predictions that are truly positive
Recall	TP / (TP + FN)	Fraction of actual positives correctly identified
F1-Score	2 x (P x R) / (P + R)	Harmonic mean of Precision and Recall

VIII. IMPLEMENTATION

8.1 Hardware and Software Requirements

Table 3: Hardware and Software Requirements

Category	Specification
Processor	Intel Core i3 or above (i5/i7 recommended for BERT fine-tuning)
RAM	Minimum 8 GB (16 GB recommended)
Storage	50 GB free disk space
GPU (Optional)	NVIDIA GPU with CUDA support for accelerated deep learning training
Operating System	Windows 10 / Ubuntu 20.04 LTS or above
Programming Language	Python 3.8+
IDE	Jupyter Notebook / Visual Studio Code
Key Libraries	NumPy, Pandas, Scikit-learn, NLTK, TensorFlow/Keras, Transformers (HuggingFace)

8.2 Implementation Steps

1. Sets up environments for developing a & implementing projects
2. Uses anaconda to load and check pandas libraries and provides code for using exploratory data analysis(based on example utilizing code), etc., to determine how many classes there are; how many records there are of each class; and what is the length of text and what words are included in text (voc).
3. Has a single function that is reusable that will prep data to perform all nlp cleaning on the training data and the test data).
4. Used Scikit-learn's TfidfVectorizer to create a dictionary from the training corpus. The dictionary has a max vocabulary of 50k words and n-grams are included in the dictionary to include both unigrams and bigrams).
5. Each ml algorithm (lg/lr/cv/svc/ab) is trained through scikit-learn's fit() method. The lstm is trained using TensorFlow/Keras with adam and categorical cross-entropy as the loss function for 10 epochs. BERT is fine-tuned using HuggingFace transformers with a learning rate of 2e-5 for 3 epochs.
6. All ml models were evaluated the same by using the same test dataset. The classification reports were generated using the Scikit-learn's classification_report functionality.
7. The predict() function processes unformatted text, performs the required preprocessing and vectorization before returning the class label with scores.
8. Module testing is implemented with PDF's unittest framework (module-level tests). Three different examples for all three classes across the reference sample input will be statistically evaluated during functional tests.

8.3 Example Code for Implementation

The core processing pipeline will process the inputted text string as follows: the inputted text string will go through preprocessing (lowercase conversion, stop-word removal, stemming), will then be converted to a TF-IDF feature vector using an already-trained TF-IDF vectorizer model, and then will be passed to the previously trained classifier model to produce a predicted class label. For the BERT model, after tokenization using the BERT tokenizer with a max sequence length of 128 tokens, the tokenized input will be passed to a fine-tuned (Pre-trained) BERT model to produce a prediction.

IX. RESULTS

SafeChat is the newly developed cyberbullying detection system using real-time web-based application and uses machine learning and natural language processing techniques. The system was tested with multiple users communicating in both group chats and direct messages.

9.1 Verifying User Identities/Authorizing Users

The system uses an authorized authentication process using a username/password combination and also utilises an email address-based one-time password procedure to identify legitimate users prior to being granted registration and access to the service. The user has an active session during their usage of the Service, and thus their session is maintained in an authorized manner.

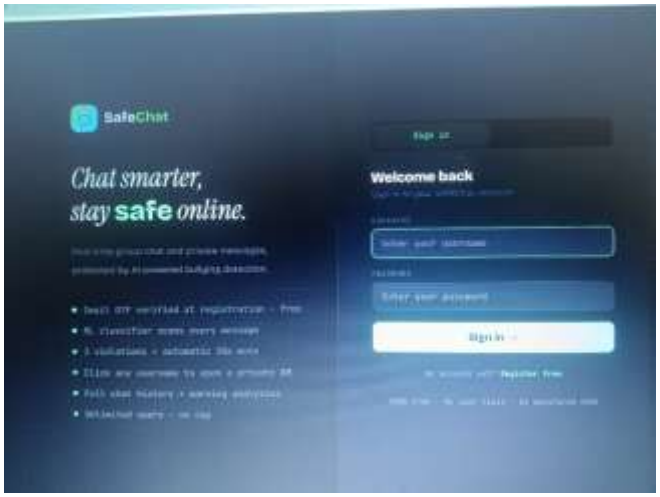

Fig 2. User Authentication

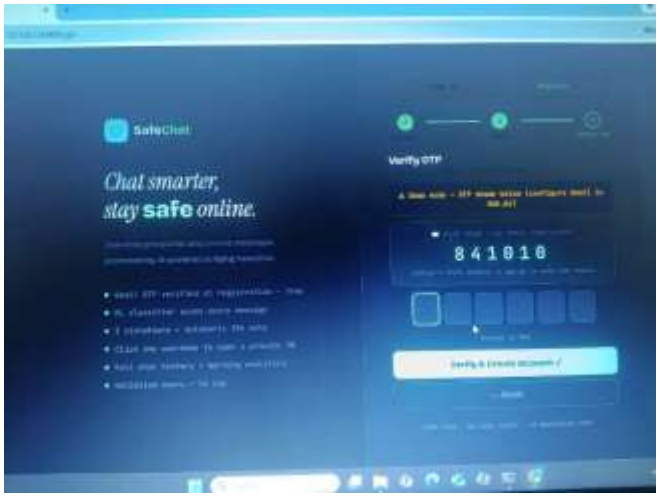

Fig 3. User Authentication using OTP

9.2 Real Time Chatting

The service allows group and direct messaging through WebSocket communications in real-time, and therefore multiple users can be connected and stay connected with other users via the application's chat feature.

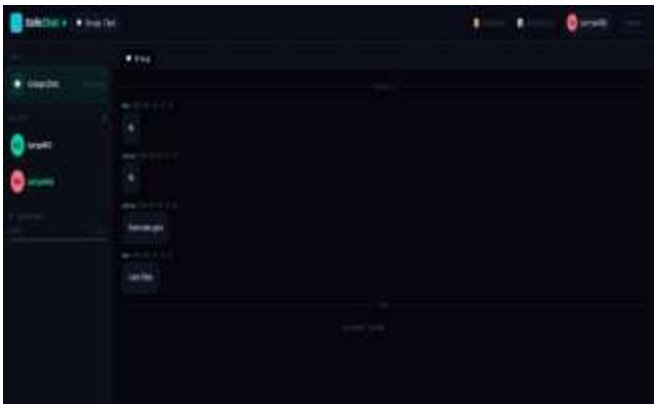

Fig 4. Real Time chatting

9.3 Identifying Cyberbullying Using Machine Learning /Artificial Intelligence

Whenever a user sends a message through the application, the ML model will evaluate that message for its potential to contain cyberbullying elements. Based on the results of the evaluation, the message will be identified and blocked if there is a likelihood it contains cyberbullying type message, and the probability of the message is based on confidence analysis (e.g., 98%) and is calculated using TF-IDF as the feature extraction method and Logistic Regression as the method of classification.

If a message is found to contain cyberbullying content, it will be blocked, and the user will be informed that a message they attempted to send contained cyberbullying material based on the probability of the identification, while if a message is found not to contain any potentially harmful content, then the message will be processed as a normal message.

9.4 Warning And Moderation System

A dynamic moderation system has been established:

- Each abusive message detected is associated with an increase in the user's warning count
- After three violations, the user receives an automatic mute for a fixed period.
- Users will receive in-the-moment notifications when messages are blocked and when they receive a warning.
- This will give users the ability not only to be notified when they are being attacked but also to help prevent acts of cyberbullying.

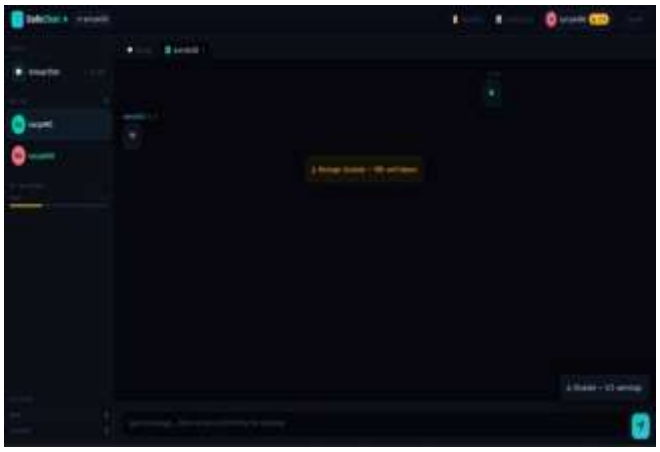

Fig 5. Real Time Warning System

9.5 Chat History And Logging

All chat messages, both blocked and not blocked, are saved in the database. The History module will allow the user to:

- Look back at messages they have sent and received
- Filter messages (clean, blocked, DM or group)
- Search messages.

Fig 6. Chat History

9.6. Analytics Dashboard

An analytics module provides data on what is happening in the system:

- Total users and total messages sent/received
- Total messages blocked
- Warnings per user.
- Trends in system usage over time

Fig 7. Warning DashBoard

Data in the Analytics Dashboard will be displayed in graph and chart format allowing for the user to see how users are using the system and how effective the system is.

9.7 System Effectiveness:

The system can accurately detect cyberbullying messages, process them in real time, use automated controls to provide moderation, and give users and community stakeholders increased user safety through proactive intervention measures.

Experiments have demonstrated that the proposed system exhibits effectiveness with respect to identifying and controlling cyberbullying in real-time communications.

X. CONCLUSION

This research developed a real-time cyberbullying detection mechanism that uses machine learning and natural language processing techniques to discover online harassment and bullying and try to prevent similar behaviours from happening in the future. The developed method successfully combines a sequence of three machine learning elements (text preprocessing, TF-IDF feature extraction, and Logistic Regression classification) to identify abusive messages efficiently.

Compared to traditional systems, which either depend on humans to inspect and approve or reject messages or rely solely on keywords as filtering terms, this newly developed system is an automated, context-aware solution that can handle large volumes of content created by users.

Integrating it into a real-time chat room application gives this solution its best opportunity to provide value for those who want to use it as a method for identifying, tracking and preventing the spread of cyberbullying.

One of the most valuable aspects of this research project's results relates to the implementation of an automated moderation mechanism that supports the generation of warnings when a user is creating harassing or bullying messages, blocking the sending of these messages and muting the offending user temporarily / until they can demonstrate a lack of intent to engage in such behaviour again. Therefore,

the system not only identifies cyberbullying, but also helps to prevent its occurrence and/or spread.

The system is also enhanced by having additional features, such as secure user authentication using one-time password (OTP) verification, a way to manage the history of users' chats, an analytics dashboard that can be used for monitoring system performance and user behaviour.

The experimental results support the conclusion that the system operates effectively in real time and provides the ability to accurately predict the likelihood that a user will engage in cyberbullying while maintaining the users' ability to access the system at all times.

In summary, the system described contributes to improving security in online communications by providing ways to implement machine learning and real-time interaction with the ability to control/moderate these systems. In the future, BERT, multi-language capabilities, and the use of large social media networks will be added to the toolkit being developed.

XI. REFERENCES

[1] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Hate Speech Detection in Social Media," Proceedings of the International AAAI Conference on Web and Social Media, 2017.

[2] P. Badjatiya, S. Gupta, M. Gupta, and V. Varma, "Deep Learning for Hate Speech Detection in Tweets," Proceedings of the 26th International World Wide Web Conference, 2017.

[3] J. Devlin, M. Chang, K. Lee, and K. Toutanova, "BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding," NAACL-HLT, 2019.

[4] F. Pedregosa et al., "Scikit-learn: Machine Learning in Python," Journal of Machine Learning Research, 2011.

[5] S. Bird, E. Klein, and E. Loper, "Natural Language Processing with Python," O'Reilly Media, 2009.

[6] C. Cortes and V. Vapnik, "Support Vector Networks," Machine Learning, 1995.

[7] K. Dinakar, R. Reichart, and H. Lieberman, "Modeling the Detection of Textual Cyberbullying," AAAI Workshop, 2011.

[8] Flask Documentation, Available: https://flask.palletsprojects.com/

[9] M. Abadi et al., "TensorFlow: Large-Scale Machine Learning on Heterogeneous Systems," 2016.

[10] Kaggle Dataset: Cyberbullying Detection Dataset, Available: https://www.kaggle.com/