PROTECTIVE USER FROM ONLINE HARASSMENT THROUGH AUTOMATED DELETION SYSTEM

A PROJECT REPORT

Submitted by

K.AKASH DURAI	**(623021104003)**
P.LALITH KUMAR	**(623021104025)**
P.SARAN	**(623021104048)**

in partial fulfilment for the award of the degree

of

BACHELOR OF ENGINEERING

IN

COMPUTER SCIENCE AND ENGINEERING

TAGORE INSTITUTE OF ENGINEERING AND TECHNOLOGY

ANNA UNIVERSITY: CHENNAI 600 025

MAY -2025

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **“PROTECTIVE USER FROM ONLINE HARASSMENT THROUGH AUTOMATED DELETION SYSTEM”** is the Bonafide work of **K.AKASH DURAI (623021104003), P.LALITH KUMAR (623021104025), and P.SARAN (623021104048),** who carried out the project work under my supervision.

SIGNATURE	**SIGNATURE**
Dr.M.BALAANAND, M.E,Ph.D.,	Ms.S.VISHNUPRIYA, M.E
HEAD OF THE DEPARTMENT	**SUPERVISOR**
ASSOCIATE PROFESSOR	**ASSOCIATE PROFESSOR**
Department of Computer	Department of Computer
Science and Engineering	Science and Engineering
Tagore Institute of Engineering	Tagore Institute of Engineering
and Technology,	and Technology,
Deviyakurichi- 636112.	Deviyakurichi- 636112.

Submitted to the viva-voce examination held on ……………………………….

INTERNAL EXAMINER	**EXTERNAL EXAMINER**

DECLARATION

We,

K.AKASH DURAI	**(623021104003)**
P.LALITH KUMAR	**(623021104025)**
P.SARAN	**(623021104048)**

Jointly declare that the project on "**PROTECTIVE USER FROM ONLINE HARASSMENT THROUGH AUTOMATED DELETION SYSTEM"** is the result of the original work done by us and to the best of our knowledge, a similar work has not been submitted earlier to the Anna University an any other institution for fulfilment of the requirement.

This project is submitted for partial fulfilment of the requirement for the Degree of Bachelor of Engineering (B.E) in Computer Science and Engineering for the Anna University.

NAME	**SIGNATURE**
K.AKASH DURAI	
P.LALITH KUMAR	
P.SARAN	

DATE :

PLACE : DEVIYAKURICHI

ACKNOWLEDGEMENT

At this pleasing moment of having successfully completed our project, we wish to convey our sincere thanks and gratitude to the management of our college and who provide all facilities to us.

We would like to express our sincere thanks to our honorable Principal **Dr.V.JAIGANESH M.E, Ph.D., MBA., MISTE., C,ENG,** for forwarding us to do our project and offering adequate duration in completing our project.

We are also grateful to the Head of the Department (Computer Science and Engineering) **Dr.M.BALAANAND.,M.E.,Ph..D.,** for his constructive suggestion and encouragement during our project.

With deep sense of gratitude, we extend our earnest and sincere thanks to our project supervisor **Ms.VISHNUPRIYA M.E.,** Assistant Professor, Department of Computer Science and Engineering for her kind guidance and encouragement during project.

We also express our indebted thanks to our project coordinator **Mrs.M.PRIYA, M.E.,** and all the staff members and lab technicians in Department of Computer Science and Engineering, Tagore Institute of Engineering and Technology, Deviyakurichi for their valuable suggestion and support during this project work.

Our boundless sense of gratitude to our parents, to all my friends, well-wishers and every person who directly or indirectly helped us wish hand and hand to completing this project.

Abstract

Cyberbullying, which occurs on digital platforms such as social media, messaging apps, and online gaming, involves the use of technology to harass, humiliate, or harm individuals. This form of bullying can result in lasting emotional distress as harmful or private information is shared publicly, creating a permanent record that can have long-term consequences. Despite efforts to detect and prevent cyberbullying, many existing approaches, particularly those based on Machine Learning (ML) and Natural Language Processing (NLP), often fail to capture the deeper semantic meaning of the text, limiting their effectiveness in accurately identifying bullying content. To address this challenge, the project proposes a novel model combining Long Short-Term Memory (LSTM) networks and Convolutional Neural Networks (CNN) for enhanced cyberbullying detection. The model leverages Word2Vec to generate custom word embedding's, which improves its understanding of the contextual relationships between words. LSTM networks are well-suited for handling sequential data, enabling the model to detect patterns in comments or tweets where important events may be separated by an unknown duration. Additionally, CNN layers are used to extract relevant information from larger chunks of text, providing a deeper understanding of the content. The proposed LSTM-CNN model aims to classify posts and comments as either bullying or non-bullying based on toxicity scores. Through experimental results, the model has demonstrated significant improvements in accuracy and efficiency in detecting cyberbullying attacks compared to existing methods. This advanced architecture offers a robust solution for analysing online interactions, ensuring better protection and detection of harmful content across digital platforms, and ultimately enhancing online safety.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVATION

S.NO	ABBREVIATION	EXPANSION
1.	ML	Machine Learning.
2.	NLP	Natural Language Processing.
3.	LSTM	Long Short-Team Memory.
4.	CNN	Convolutional Neural Network.
5.	TF-IDF	Term Frequency-Inverse Document Frequency.
6.	SVM	Support Vector Machine.
7.	RF	Random Forest.
8.	API	Application Programming Interface.
9.	SQL	Structured Query Language.
10.	DL	Deep Learning

CHAPTER 1

INTRODUCTION

1.1. OVERVIEW

Cyberbullying is bullying that takes place over digital devices like cell phones, computers, and tablets. Cyberbullying can occur through SMS, Text, and apps, or online in social media, forums, or gaming where people can view, participate in, or share content. Cyberbullying includes sending, posting, or sharing negative, harmful, false, or mean content about someone else. It can include sharing personal or private information about someone else causing embarrassment or humiliation. Some cyberbullying crosses the line into unlawful or criminal behaviour.

The most common places where cyberbullying occurs are:

- Social Media, such as Facebook, Instagram, Snapchat, and Tik Tok
- Text messaging and messaging apps on mobile or tablet devices
- Instant messaging, direct messaging, and online chatting over the internet
- Online forums, chat rooms, and message boards, such as Reddit
- Email
- Online gaming communities

1.1.1. Different Kinds of Cyberbullying

There are many ways that someone can fall victim to or experience cyberbullying when using technology and the internet. Some common methods of cyberbullying are:

Harassment – When someone is being harassed online, they are being subjected to a string of abusive messages or efforts to contact them by one person or a group of people. People can be harassed through social media as well as through their mobile phone (texting and calling) and email. Most of the contact the victim will receive will be of a malicious or threatening nature.

Doxing – Doxing is when an individual or group of people distribute another person's personal information such as their home address, cell phone number or place of work onto social media or public forums without that person's permission to do so. Doxing can cause the victim to feel extremely anxious and it can affect their mental health.

Cyberstalking – Similar to harassment, cyberstalking involves the perpetrator making persistent efforts to gain contact with the victim, however this differs from harassment – more commonly than not, people will cyberstalk another person due to deep feelings towards that person, whether they are positive or negative. Someone who is cyberstalking is more likely to escalate their stalking into the offline world.

Revenge porn – Revenge porn, is when sexually explicit or compromising images of a person have been distributed onto social media or shared on revenge porn specific websites without their permission to do so. Normally, images of this nature are posted by an ex-partner, who does it with the purpose of causing humiliation and damage to their reputation.

Swatting – Swatting is when someone calls emergency responders with claims of dangerous events taking place at an address. People swat others with the intention of causing panic and fear when armed response units arrive at their home or place of work. Swatting is more prevalent within the online gaming community.

Corporate attacks – In the corporate world, attacks can be used to send masses of information to a website in order to take the website down and make it non-functional. Corporate attacks can affect public confidence, damaging businesses reputations and in some instances, force them to collapse.

Account hacking – Cyberbullies can hack into a victim's social media accounts and post abusive or damaging messages. This can be particularly damaging for brands and public figures.

False profiles – Fake social media accounts can be setup with the intention of damaging a person or brand's reputation. This can easily be done by obtaining publicly available images of the victim and making the account appear as authentic as possible.

Slut shaming – Slut shaming is when someone is called out and labelled as a “slut” for something that they have done previously or even just how they dress. This kind of cyberbullying often occurs when someone has been sexting another person and their images or conversations become public. It is seen more commonly within young people and teenagers but anyone can fall victim to being slut shamed.

1.1.2. Effects of Cyberbullying

There are numerous effects that may be seen in those who are dealing with cyberbullying. It can be helpful to know what to expect to see in a victim, as this can be one way to identify when someone is being bullied online.

Some of these effects are even stronger than what is seen with traditional bullying, as the victim often cannot escape the abusive situation. They may include:

- Feelings of distress about the bullying
- Increased feelings of depression and mood swings
- Increased feelings of anxiety
- Problems falling asleep or staying asleep (e.g., insomnia)
- Suicidal ideation or suicide attempt
- Increased feelings of fearfulness
- Feelings of low self-esteem or self-worth
- Social isolation, withdrawing from friend groups, or spending a lot of time alone
- Avoiding doing things that they used to enjoy
- Poor academic performance
- Problems in relationships with family members and friends
- Symptoms of post-traumatic stress
- Self-harm (e.g., cutting, hitting yourself, head banging)
- Substance abuse
- Increased feelings of anger, irritability, or angry outburst

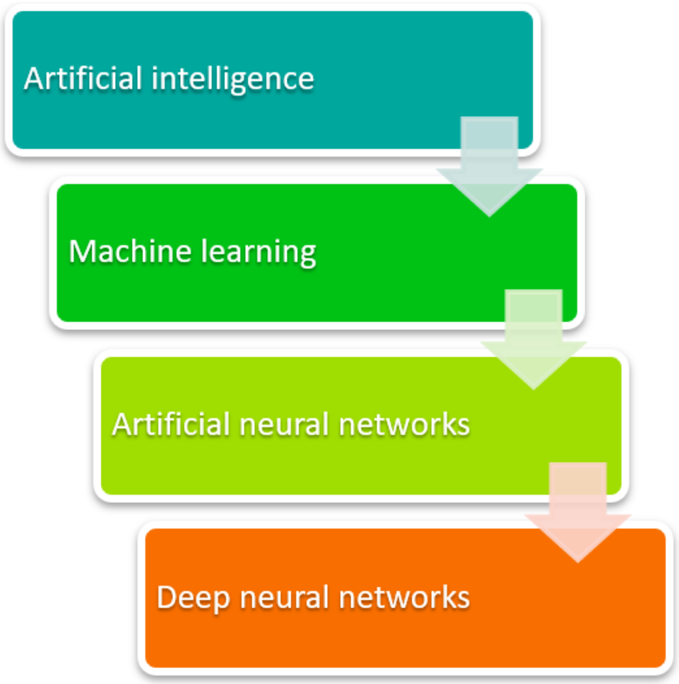
1.2. PROBLEM STATEMENT

Social media networks such as Facebook, Twitter, Flickr, and Instagram have become the preferred online platforms for interaction and socialization among people of all ages. While these platforms enable people to communicate and interact in previously unthinkable ways, they have also led to malevolent activities such as cyber-bullying. Cyberbullying is a type of psychological abuse with a significant impact on society. Cyber-bullying events have been increasing mostly among young people spending most of their time navigating between different social media platforms. Particularly, social media networks such as Twitter and Facebook are prone to CB because of their popularity and the anonymity that the Interne provides to abusers. In India, for example, 14 percent of all harassment occurs on Facebook and Twitter, with 37 percent of these incidents involving youngsters. Moreover, cyberbullying might lead to serious mental issues and adverse mental health effects. Most suicides are due to the anxiety, depression, stress, and social and emotional difficulties from cyber-bullying events. This motivates the need for an approach to identify cyberbullying in social media messages (e.g., posts, tweets, and comments). In this project, we mainly focus on the problem of cyberbullying detection on the Twitter platform. As cyberbullying is becoming a prevalent problem in Twitter, the detection of cyberbullying events from tweets and provisioning preventive measures are the primary tasks in battling cyberbullying threats. Therefore, there is a greater need to increase the research on social networks-based CB in order to get greater insights and aid in the development of effective tools and approaches to effectively combat cyberbullying problem. Manually monitoring and controlling cyberbullying on Twitter platform is virtually impossible. Furthermore, mining social media messages for cyberbullying detection is quite difficult. For example, Twitter messages are often brief, full of slang, and may include emojis, and gifs, which makes it impossible to deduce individuals' intentions and meanings purely from social media messages. Moreover, bullying can be difficult to detect if the bully uses strategies like sarcasm or passive-aggressiveness to conceal it. Despite the challenges that social media messages bring, cyberbullying detection on social media is an open and active research topic.

Cyberbullying detection within the Twitter platform has largely been pursued through tweet classification and to a certain extent with topic modelling approaches. Text classification based on supervised machine learning (ML) models are commonly used for classifying tweets into bullying and non-bullying tweets. Also, it may be suitable only for a pre-determined collection of events, but it cannot successfully handle tweets that change on the fly. Considering these limitations, an efficient tweet classification approach must be developed to bridge the gap between the classifier and the topic model so that the adaptability is significantly proficient. The deep learning model has been successfully used by current researchers to address various issues related to the text domain, including sentiment analysis, question answering, document classification, sentence classification, spam filtering and others. By following them, this project also uses a deep learning algorithm to address the cyber bullying detection issue. BiLSTM is capable of capturing the semantics of the sentence by performing the convolution operations over the tweets.

1.3. DEEP LEARNING

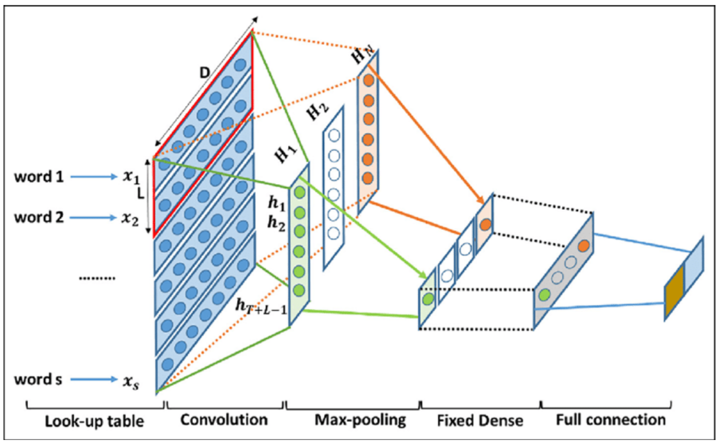
Deep Learning is a field which comes under Machine Learning and is related to the use of algorithms in artificial neural networks. It is majorly used to create a predictive model to solve the problems with just a few lines of coding. A Deep Learning system is an extensive neural network which is inspired by the function and structure of the brain. Deep learning models can be supervised, semi-supervised, or unsupervised (or a combination of any or all of the three). They're advanced machine learning algorithms used by tech giants, like Google, Microsoft, and Amazon to run entire systems and power things, like self-driving cars and smart assistants.



First, machine learning had to get developed. ML is a framework to automate (through algorithms) statistical models, like a linear regression model, to get better at making predictions. A model is a single model that makes predictions about something. Those predictions are made with some accuracy. A model that learns—machine learning—takes all its bad predictions and tweaks the weights inside the model to create a model that makes fewer mistakes. The learning portion of creating models spawned the development of artificial neural networks. ANNs utilize the hidden layer as a place to store and evaluate how significant one of the inputs is to the output. The hidden layer stores information regarding the input's importance, and it also makes associations between the importance of combinations of inputs

1.3.1. CONVOLUTIONAL NEURAL NETWORK

Convolution Neural Networks(CNNs) are multi-layered artificial neural networks with the ability to detect complex features in data, for instance, extracting features in image and text data. CNNs have majorly been used in computer vision tasks such as image classification, object detection, and image segmentation. However, recently CNNs have been applied to text problems. To classify text data using convolutions, use 1-D convolutional layers that convolve over the time dimension of the input. Trains a network with 1-D convolutional filters of varying widths. The width of each filter corresponds the number of words the filter can see (the n-gram length).



Define Network Architecture

Define the network architecture for the classification task. The following steps describe the network architecture.

- Specify an input size of 1, which corresponds to the channel dimension of the integer sequence input.
- Embed the input using a word embedding of dimension 100.
- For the n-gram lengths 2, 3, 4, and 5, create blocks of layers containing a convolutional layer, a batch normalization layer, a ReLU layer, a dropout layer, and a max pooling layer.
- For each block, specify 200 convolutional filters of size 1-by-*N* and a global max pooling layer.
- Connect the input layer to each block and concatenate the outputs of the blocks using a concatenation layer.
- To classify the outputs, include a fully connected layer with output size *K*, a softmax layer, and a classification layer, where *K* is the number of classes.

1.4. NATURAL LANGUAGE PROCESSING

Natural language processing (NLP) is a branch of artificial intelligence that helps computers understand, interpret and manipulate human language. NLP draws from many disciplines, including computer science and computational linguistics, in its pursuit to fill the gap between human communication and computer understanding.

NLP Techniques

Natural Language Processing (NLP) applies two techniques to help computers understand text: syntactic analysis and semantic analysis.

Syntactic Analysis

Syntactic analysis – or parsing – analyzes text using basic grammar rules to identify sentence structure, how words are organized, and how words relate to each other.

Some of its main sub-tasks include:

- **Tokenization** consists of breaking up a text into smaller parts called *tokens* (which can be sentences or words) to make text easier to handle.
- **Part of speech tagging (PoS tagging)** labels tokens as *verb, adverb, adjective, noun,* etc. This helps infer the meaning of a word (for example, the word "book" means different things if used as a verb or a noun).
- **Lemmatization & stemming** consist of reducing inflected words to their base form to make them easier to analyze.
- **Stop-word removal** removes frequently occuring words that don't add any semantic value, such as *I, they, have, like, yours,* etc.

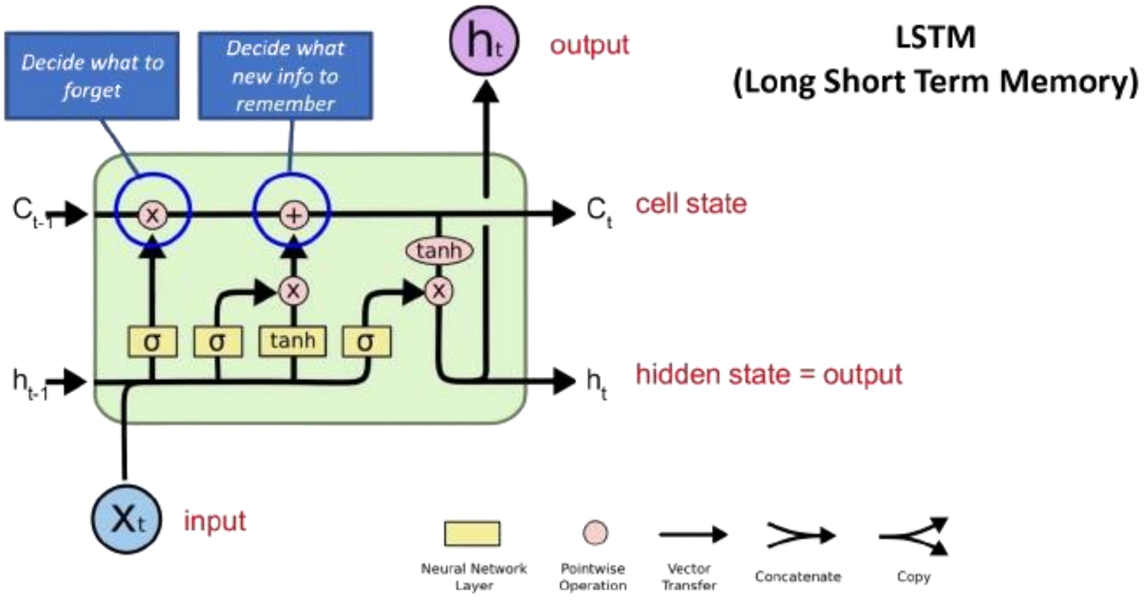
Semantic Analysis

Semantic analysis focuses on capturing the meaning of text. First, it studies the meaning of each individual word (lexical semantics). Then, it looks at the combination of words and what they mean in context. The main sub-tasks of semantic analysis are:

- **Word sense disambiguation** tries to identify in which sense a word is being used in a given context.
- **Relationship extraction** attempts to understand how entities (places, persons, organizations, etc) relate to each other in a text.

1.5. LSTM

LSTM stands for Long-Short Term Memory. LSTM is a type of recurrent neural network but is better than traditional recurrent neural networks in terms of memory. Having a good hold over memorizing certain patterns LSTMs perform fairly better. As with every other NN, LSTM can have multiple hidden layers and as it passes through every layer, the relevant information is kept and all the irrelevant information gets discarded in every single cell.



The **Embedding Layer** takes the input list of indexes generated by the vectorization function. We have initialized a layer with a number of embeddings equal to the length of vocabulary and embedding length to **50**. This initialization will create a weight tensor of shape **(vocab_len, embed_len)** which has an embedding vector of length **50** for each token of vocabulary. The layer is responsible for mapping the index of each token to a float vector of length **50** because we have set the embedding length to **50**. This layer takes tensor of shape **(batch_size, max_tokens)** and outputs tensor of shape **(batch_size, max_tokens, embed_len)**. Each token gets assigned its respective embedding vector based on the index value by this layer.

1.6. AIM AND OBJECTIVE

Aim

The aim of the project is to design and implement a comprehensive framework for detecting, blocking, and preventing cyberbullying in social networking sites. The focus is on leveraging advanced deep learning techniques, specifically LSTM (Long Short-Term Memory) and CNN (Convolutional Neural Network) algorithms, to analyze and classify user-generated content, identifying instances of cyberbullying. The overarching goal is to create a safer online environment by proactively addressing and mitigating instances of harmful behaviour on social networking platforms.

Objectives

- To develop a robust LSTM-CNN model for cyberbullying detection.
- To implement user authentication and management features.
- To apply pre-processing techniques for effective text analysis.
- To enable real-time cyberbullying detection capabilities.
- To design a user-friendly interface for seamless interaction.
- To implement dynamic actions for handling cyberbullying incidents.
- To explore integration with major social networking platforms.
- To incorporate features for user education and awareness.
- To ensure scalability and optimal system performance.
- To implement robust data privacy and security measures.
- To conduct thorough testing, including unit and user acceptance tests.
- To provide comprehensive documentation for system understanding.

1.7. SCOPE OF THE PROJECT

The project's scope is defined by a comprehensive set of objectives aimed at creating an effective framework for cyberbullying prevention within the realm of social networking sites. The primary focus is on developing an advanced LSTM-CNN model capable of accurately detecting instances of cyberbullying in real-time. This involves the implementation of robust preprocessing techniques, including tokenization and stemming, to enhance the accuracy of text analysis. The project also includes the establishment of a secure user authentication and management system, ensuring authorized access, accountability, and safeguarding user privacy. The user interface is designed to be intuitive and user-friendly, fostering seamless interaction and engagement with the platform. Dynamic action handling mechanisms, such as user blocking and content moderation, are implemented to effectively address and mitigate cyberbullying incidents. Exploring integration possibilities with major social networking platforms extends the project's impact, broadening its reach. Additionally, features promoting user education and awareness are incorporated, aiming to cultivate responsible online behavior. Scalability and optimal performance are emphasized to accommodate increasing volumes of user-generated content. Ensuring data privacy and security is paramount, with robust measures in place to adhere to best practices and legal standards. Thorough testing, including unit and user acceptance tests, is conducted to validate the functionality and performance of the system. The project concludes with the development of comprehensive documentation, covering system architecture, implementation details, and guidelines for system understanding and maintenance.

CHAPTER 2
SYSTEM SPECIFICATION

2.1. HARDWARE REQUIREMENTS

• **Processors**	: Intel® Core™ i5 processor, 2.60 GHz,8 GB of DRAM
• **Disk space**	: 320 GB
• **Operating systems**	: Windows® 10

2.2. SOFTWARE REQUIREMENTS

• **Programming**	: Python 3.7.4(64-bit) or (32-bit)
• **Framework**	: Flask 1.1.
• **Database**	: MySQL 5.
• **Web Server**	: Wampserver 2i
• **Packages**	: Pandas, Sikit Learn, Numpy, matplotlib, seaborn
• **Libraries**	: NLTK and SpaCy for text preprocessing

CHAPTER 3
SYSTEM ANALYSIS

3.1. EXISTING SYSTEM

In this chapter existing machine learning classifiers utilized for tweet classification will be discussed. This chapter analysed five supervised machine learning algorithms: Support Vector Machines (SVM), Naive Bayes (NB), Random Forest (RF), Decision Tree (DT), Gradient Boosting model (GBM), Logistic Regression (LR) and Voting Classifier (Logistic Regression C Stochastic Gradient Descent classifier).

- **Random Forest**

RF is a tree based classifier in which input vector generated trees randomly. RF uses random features, to create multiple decision trees, to make a forest. Then class labels of test data are predicted by aggregating voting of all trees. Higher weights are assigned to the decision trees with low value error. Overall prediction accuracy is improved by considering trees with low error rate.

- **Support Vector Machine**

The Support vector machine (SVM) is understood that executes properly as sentiment analysis. SVM typifies preference, confines and makes usage of the mechanisms for the assessment and examines records, which are attained within the index area. Arrangements of vectors for every magnitude embody crucial details. Information (shown in form of vector) has been arranged in type to achieve this target. Next, the border is categorized in two training sets by stratagem. This is a long way from any area in the training samples. Support-vector machines in machine learning includes focused learning models connected to learning evaluations which inspect material that is exploited to categorize, also revert inspection.

- **Naive Bayes**

Ordering approach, Naive Bayes(NB), with sturdy (naive) independent assumptions among stabilities, depends on Bayes' Theorem. NB classifier anticipates that the proximity of a specific element of class that is confined to the closeness of a couple of different variables. For instance, a natural organic product is presumably viewed as an apple, if its shading is dark red, if type of it is round and it is roughly 3 creeps in expansiveness. In machine learning, Naive Bayes classifiers are a gathering of essential "probabilistic classifiers" considering applying Bayes' speculation with gullible opportunity assumptions between the features. They are considered as the minimum problematic Bayesian network models.

- **Gradient Boosting Machine**

GBM is a ML based boosting model and is widely being used for regression and classification tasks, which works by a model formed by ensemble of weak prediction models, commonly decision trees. In boosting, weak learners are converted to strong learners. Every new generated tree is a modified form of previous one and use gradient as loss function. Loss calculate the efficiency of model coefficients fitting over underlying data. Logically loss function is used for model optimization.

- **Logistic Regression**

In LR class probabilities are estimated on the basis of output such as they predict if the input is from class X with probability x and from class Y with probability y. If x is greater than y, then predicted output class is X, otherwise Y. Insight, a logistic approach used for demonstrating the probability of a precise group or else, occurrence is obtainable, e.g., top/bottom, white/black, up/down, positive/negative or happy/unhappy. This is able to stretch out and to show a small number of classes about events, for example, to make a decision if an image includes a snake, hound, deer, etc., every article being famous in the image would be appointed a probability wherever in the series of 0 and 1 with whole addition to one.

- **Stochastic Gradient Descent**

Gradient Descent's types include Stochastic Gradient Descent (SGD). SDGD is an iterative strategy for advancing a target work with appropriate perfection properties (for example differentiable or sub differentiable). Degree of advancement is calculated by it in light of development of alternative variables. It is very well, may be viewed as a stochastic guess of inclination plummet advancement, since it replaces the genuine angle (determined from the whole informational index) by a gauge thereof (determined from an arbitrarily chosen subset of the information).

- **Voting Classifier**

Voting Classifier(VC) is a cooperative learning which engages multiple individual classifiers and combines their predictions, which could attain better performance than a single classifier. It has been exhibited that the mixture of multiple classifiers could be more operative compared to any distinct ones. The VC is a meta classifier for joining tantamount or hypothetically exceptional ML classifiers for order through greater part throwing a voting form. It executes "hard" and "soft" casting a ballot. Hard voting gives the researcher the chance to foresee the class name in place of the last class mark that has been anticipated often through models of characterization.

3.1.1. Disadvantages

- Process of reporting such cases is long, tedious job.
- Difficult to track.
- Most of the cyberbullying cases go unreported.
- Low accuracy.
- Time consuming process.
- Problem is not automatically detected and not promptly report bullying message.
- Response time is slow.
- Basic features and common classifier accuracy is low.
- Data are manually labelled using online services or custom applications.
- Usually data limited only to a small percentage

3.2. PROPOSED SYSTEM

The proposed system aims to provide a robust solution for tackling cyberbullying on social networking sites. It utilizes a combination of deep learning techniques, specifically LSTM (Long Short-Term Memory) and CNN (Convolutional Neural Network), to analyze and classify text data. The framework involves several modules to detect, block, and prevent cyberbullying instances.

Data Collection

The system will collect data from social networking sites through APIs or web scraping. This data includes tweets, comments, and messages that will be used for training and testing the models.

Pre-processing Module

This module will handle the pre-processing of textual data:

- **Tokenization:** Breaking down text into individual words or tokens.
- **Stemming:** Reducing words to their root form.
- **Stop Word Removal:** Eliminating common words that don't carry significant meaning.
- **Emoji Removal:** Stripping out emojis to focus on text content.
- **URL Removal:** Eliminating URLs for cleaner text.
- **Normalization:** Standardizing the text to a common format.

Feature Extraction

TF-IDF (Term Frequency-Inverse Document Frequency) will be employed to extract features from the pre-processed text.

LSTM-CNN Classification

The combined LSTM-CNN model will be trained on the labeled dataset to classify text as cyberbullying (1) or non-cyberbullying (0). This hybrid architecture leverages the temporal understanding of LSTM and spatial understanding of CNN for improved accuracy.

Bullying Prediction and Actions

Users can post messages on their profiles or others' profiles. The system will analyze these messages to identify potential instances of cyberbullying. The system will predict if a posted message contains cyberbullying. If identified, the system will:

- Send a warning email to the user who posted the message.
- Block the user if the offense is severe or repeated.
- Remove the cyberbullying message to prevent its spread.

3.2.2. Advantages

- It successfully classifies the tweets in various classes.
- Auto report generator generates a simple report for probable accusers.
- Several analytics and report can be sent to the crime department.
- Accuracy is high.
- Foul language on any given page, removes it, and can highlight words as well,
- This method detects the offensive post or messages it block that user id.
- The “filtered content” is displayed at back to the page, in such a way preventing the display of explicit content.
- An automatically generate a report for each incident is also provided.

3.3. SYSTEM STUDY

System study, also known as systems analysis, is a process of gathering and interpreting facts, understanding the complexities of systems, identifying problem areas within those systems, and proposing feasible solutions. It is a crucial phase in the system development life cycle (SDLC) that precedes system design and implementation. The primary goal of a system study is to provide a comprehensive understanding of the existing system, its strengths, weaknesses, opportunities, and threats, leading to informed decision-making for system improvement or development.

2.3.1. Feasibility Study

A **feasibility study** serves as the cornerstone in the assessment of a proposed project's viability, offering stakeholders a comprehensive understanding to facilitate well-informed decisions. This study encompasses a multifaceted analysis, encompassing technical, operational, economic, legal, and scheduling aspects.

- **Technical Feasibility Analysis**

Technical feasibility evaluates the project's alignment with existing technology infrastructure, considering factors such as available technology, required expertise, and compatibility with current systems. This phase ensures that the proposed solution can be seamlessly integrated from a technological perspective.

- **Operational Feasibility Analysis**

In the operational feasibility analysis, the focus shifts to how well the proposed solution harmonizes with day-to-day operations. This involves assessing the impact on personnel, identifying training requirements, and gauging the overall synergy with existing workflows. A project can only be successful if it aligns with the operational realities of the organization.

- **Economic Feasibility Analysis**

Economic feasibility involves a meticulous cost-benefit analysis, scrutinizing development costs, ongoing operational expenses, and the anticipated benefits. This financial examination provides stakeholders with a clear picture of the project's economic viability and potential return on investment.

CHAPTER 4
SODTWARE DESCRIPTION

4.1. PYTHON 3.7.4

Python is a general-purpose interpreted, interactive, object-oriented, and high-level programming language. It was created by Guido van Rossum during 1985- 1990. Like Perl, Python source code is also available under the GNU General Public License (GPL). This tutorial gives enough understanding on Python programming language.

Python is a high-level, interpreted, interactive and object-oriented scripting language. Python is designed to be highly readable. It uses English keywords frequently where as other languages use punctuation, and it has fewer syntactical constructions than other languages. Python is a MUST for students and working professionals to become a great Software Engineer specially when they are working in Web Development Domain.

Python is currently the most widely used multi-purpose, high-level programming language. Python allows programming in Object-Oriented and Procedural paradigms. Python programs generally are smaller than other programming languages like Java. Programmers have to type relatively less and indentation requirement of the language, makes them readable all the time. Python language is being used by almost all tech-giant companies like – Google, Amazon, Facebook, Instagram, Dropbox, Uber… etc. The biggest strength of Python is huge collection of standard library which can be used for the following:

- Machine Learning
- GUI Applications (like Kivy, Tkinter, PyQt etc.)
- Web frameworks like Django (used by YouTube, Instagram, Dropbox)
- Image processing (like OpenCV, Pillow)
- Web scraping (like Scrapy, BeautifulSoup, Selenium)
- Test frameworks
- Multimedia
- Scientific computing
- Text processing and many more.

Tensor Flow

TensorFlow is an end-to-end open-source platform for machine learning. It has a comprehensive, flexible ecosystem of tools, libraries, and community resources that lets researchers push the state-of-the-art in ML, and gives developers the ability to easily build and deploy ML-powered applications.

TensorFlow

TensorFlow provides a collection of workflows with intuitive, high-level APIs for both beginners and experts to create machine learning models in numerous languages. Developers have the option to deploy models on a number of platforms such as on servers, in the cloud, on mobile and edge devices, in browsers, and on many other JavaScript platforms. This enables developers to go from model building and training to deployment much more easily.

Keras

Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation.

K Keras

Simple. Flexible. Powerful.

- Allows the same code to run on CPU or on GPU, seamlessly.
- User-friendly API which makes it easy to quickly prototype deep learning models.
- Built-in support for convolutional networks (for computer vision), recurrent networks (for sequence processing), and any combination of both.
- Supports arbitrary network architectures: multi-input or multi-output models, layer sharing, model sharing, etc. This means that Keras is appropriate for building essentially any deep learning model, from a memory network to a neural Turing machine.

Pandas

pandas is a fast, powerful, flexible and easy to use open source data analysis and manipulation tool, built on top of the Python programming language. pandas is a Python package that provides fast, flexible, and expressive data structures designed to make working with "relational" or "labeled" data both easy and intuitive. It aims to be the fundamental high-level building block for doing practical, real world data analysis in Python.

Pandas is mainly used for data analysis and associated manipulation of tabular data in Data frames. Pandas allows importing data from various file formats such as comma-separated values, JSON, Parquet, SQL database tables or queries, and Microsoft Excel. Pandas allows various data manipulation operations such as merging, reshaping, selecting, as well as data cleaning, and data wrangling features. The development of pandas introduced into Python many comparable features of working with Data frames that were established in the R programming language. The panda's library is built upon another library NumPy, which is oriented to efficiently working with arrays instead of the features of working on Data frames.

NumPy

NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays. Using NumPy, mathematical and logical operations on arrays can be performed.

NumPy is a general-purpose array-processing package. It provides a high-performance multidimensional array object, and tools for working with these arrays.

Matplotlib

Matplotlib is a comprehensive library for creating static, animated, and interactive visualizations in Python. Matplotlib makes easy things easy and hard things possible.

Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK.

Seaborn

Seaborn is a library for making statistical graphics in Python. It builds on top of matplotlib and integrates closely with pandas data structures. Visualization is the central part of Seaborn which helps in exploration and understanding of data.

seaborn

Seaborn offers the following functionalities:

- Dataset oriented API to determine the relationship between variables.
- Automatic estimation and plotting of linear regression plots.
- It supports high-level abstractions for multi-plot grids.
- Visualizing univariate and bivariate distribution.

Scikit Learn

scikit-learn is a Python module for machine learning built on top of SciPy and is distributed under the 3-Clause BSD license.

scikits learn machine learning in Python

Scikit-learn (formerly scikits. learn and also known as sklearn) is a free software machine learning library for the Python programming language. It features various classification, regression and clustering algorithms including support-vector machines, random forests, gradient boosting, k-means and DBSCAN, and is designed to interoperate with the Python numerical and scientific libraries NumPy and SciPy.

NLTK

NLTK is a leading platform for building Python programs to work with human language data. It provides easy-to-use interfaces to over 50 corpora and lexical resources such as WordNet, along with a suite of text processing libraries for classification, tokenization, stemming, tagging, parsing, and semantic reasoning, wrappers for industrial-strength NLP libraries, and an active discussion forum.

NLTK (Natural Language Toolkit) Library is a suite that contains libraries and programs for statistical language processing. It is one of the most powerful NLP libraries, which contains packages to make machines understand human language and reply to it with an appropriate response.

WordCloud

A word cloud (also called tag cloud or weighted list) is a visual representation of text data. Words are usually single words, and the importance of each is shown with font size or color. Python fortunately has a wordcloud library allowing to build them.

The wordcloud library is here to help you build a wordcloud in minutes. A word cloud is a data visualization technique that shows the most used words in large font and the least used words in small font. It helps to get an idea about your text data, especially when working on problems based on natural language processing.

4.2. MYSQL

MySQL is a relational database management system based on the Structured Query Language, which is the popular language for accessing and managing the records in the database. MySQL is open-source and free software under the GNU license. It is supported by Oracle Company. MySQL database that provides for how to manage database and to manipulate data with the help of various SQL queries. These queries are: insert records, update records, delete records, select records, create tables, drop tables, etc. There are also given MySQL interview questions to help you better understand the MySQL database.

MySQL is currently the most popular database management system software used for managing the relational database. It is open-source database software, which is supported by Oracle Company. It is fast, scalable, and easy to use database management system in comparison with Microsoft SQL Server and Oracle Database. It is commonly used in conjunction with PHP scripts for creating powerful and dynamic server-side or web-based enterprise applications. It is developed, marketed, and supported by MySQL AB, a Swedish company, and written in C programming language and C++ programming language. The official pronunciation of MySQL is not the My Sequel; it is My Ess Que Ell. However, you can pronounce it in your way. Many small and big companies use MySQL. MySQL supports many Operating Systems like Windows, Linux, MacOS, etc. with C, C++, and Java languages.

4.3. WAMPSERVER

WampServer is a Windows web development environment. It allows you to create web applications with Apache2, PHP and a MySQL database. Alongside, PhpMyAdmin allows you to manage easily your database.

WAMPServer is a reliable web development software program that lets you create web apps with MYSQL database and PHP Apache2. With an intuitive interface, the application features numerous functionalities and makes it the preferred choice of developers from around the world. The software is free to use and doesn't require a payment or subscription.

4.4. BOOTSTRAP 4

Bootstrap is a free and open-source tool collection for creating responsive websites and web applications. It is the most popular HTML, CSS, and JavaScript framework for developing responsive, mobile-first websites.

It solves many problems which we had once, one of which is the cross-browser compatibility issue. Nowadays, the websites are perfect for all the browsers (IE, Firefox, and Chrome) and for all sizes of screens (Desktop, Tablets, Phablets, and Phones). All thanks to Bootstrap developers -Mark Otto and Jacob Thornton of Twitter, though it was later declared to be an open-source project.

Easy to use: Anybody with just basic knowledge of HTML and CSS can start using Bootstrap

Responsive features: Bootstrap's responsive CSS adjusts to phones, tablets, and desktops

Mobile-first approach: In Bootstrap, mobile-first styles are part of the core framework

Browser compatibility: Bootstrap 4 is compatible with all modern browsers (Chrome, Firefox, Internet Explorer 10+, Edge, Safari, and Opera)

4.5. FLASK

Flask is a web framework. This means flask provides you with tools, libraries and technologies that allow you to build a web application. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.

Flask
web development,
one drop at a time

Flask is often referred to as a micro framework. It aims to keep the core of an application simple yet extensible. Flask does not have built-in abstraction layer for database handling, nor does it have formed a validation support. Instead, Flask supports the extensions to add such functionality to the application. Although Flask is rather young compared to most Python frameworks, it holds a great promise and has already gained popularity among Python web developers. Let's take a closer look into Flask, so-called "micro" framework for Python. Flask is part of the categories of the micro-framework. Micro-framework are normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins.

CHAPTER 5
PROJECT DETAILS

5.1. PROJECT DESCRIPTION

In the proposed framework shown in Figure 5.1., the process of detecting cyberbully activities begins with input dataset from social network. Input is text conversation collected from twitter. Input is given to data pre-processing which is applied to improve the quality of the project data and subsequent analytical steps, this includes removing stop words, extra characters and hyperlinks. After performing pre-processing on the input data, it is given to Feature Extraction. Feature Extraction is done to obtain features like Noun, Adjective and Pronoun from the text and statistics on occurrence of word (frequency) in the text. The cyberbullying words are given as training dataset. With the training dataset the preprocessed online social network conversation is tested for bullying word presence. Feature Vector distance algorithm detects the cyberbully words present in the conversation and displays it. For cyberbully Classification, BiLSTM is used.

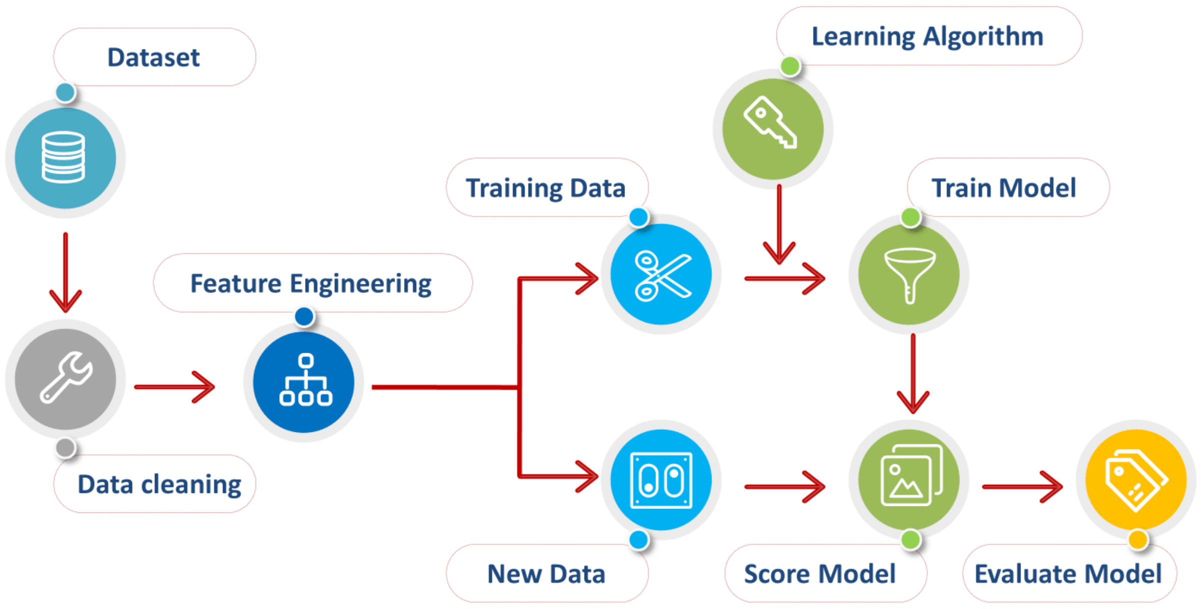


Figure 5.1. Cyberbullying Detection Proposed Model

5.2. MODULE DESCRIPTION

5.2.1. Social Networking Web App

Build a social networking service is an online platform which people use to build social networks or social relationships with other people who share similar personal or career interests, activities, backgrounds or real-life connections. Social networking services vary in format and the number of features. The classification model has been exposed as a REST API which was consumed by a Web application built using Python's Flask framework. The main features include an Admin dashboard for visualization of cyberbullying activities, an option to search tweets, and automatic generation and emailing of reports of cyberbullying activity.

Aadhar User Account Management

- **New User**

Create user account with aadhar number.

- **Existing User**

The existing users of Facebook will also have to upload a scanned copy of their Aadhar Card. If they fail to do so, their profile will be suspended within the next 15 days.

Cyberbullying Analysis API

In this module we developed the API for cyberbullying analytics on chat or post user data. It focuses on keywords and analyzes chat or post according to a two-pole scale (positive and negative).

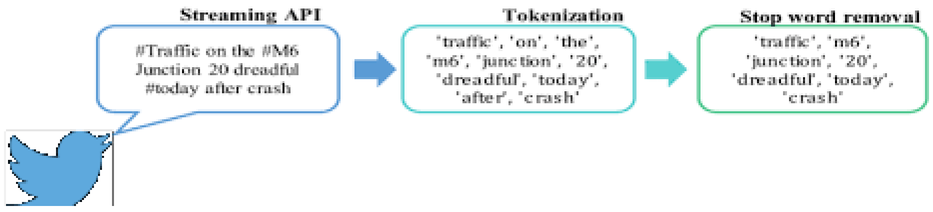
5.2.2. Training Phase: Cyberbullying Tweet Classification

- **Cyberbullying Data Set Annotation**

We used cyberbullying data from Kaggle. The dataset in consisted of two labels, positive and negative, while was composed of three labels of positive, neutral, and negative. Furthermore, the dataset in was composed of five labels of positive, somewhat positive, neutral, somewhat negative, and negative.

- **Pre-processing**

Datasets contain unnecessary data in raw form that can be unstructured or semi-structured. Such unnecessary data increases training time of the model and might degrades its performance. Pre-processing plays a vital role in improving the efficiency of DL models and saving computational resources. Text pre-processing boosts the prediction accuracy of the model. The preprocessing step is essential in cyberbullying detection. It consists of both cleaning of texts (e.g., removal of stop words and punctuation marks), as well as spam content removal. In the proposed model, it has been applied to remove and clean unwanted noise in text detection. For example, stop words, special characters, and repeated words were removed. Then, the stemming for the remaining words to their original roots has been applied as a result of this preprocessing, and the dataset containing clean tweets is produced for the proposed model to be run and predicted. In this project, we executed various data preprocessing steps such as tokenization, spelling correction, stop words removing, punctuation removing, digit removing, removing a non-Bangla character, removing Emoticons, word normalization, and lemmatization and data splitting.


Streaming API
#Traffic on the #M6 Junction 20 dreadful #today after crash
Tokenization
'traffic', 'on', 'the', 'm6', 'junction', '20', 'dreadful', 'today', 'after', 'crash'
Stop word removal
'traffic', 'm6', 'junction', '20', 'dreadful', 'today', 'crash'

Tokenization: Tokenization refers to splitting up a sentence, phrase, or word into numerous smaller linguistic units named "Tokens". These tokens help to comprehend the NLP model and decipher the significance of the content by investigating the arrangement of the words. In natural language processing, the language needs to be analyzed and scrutinized under certain constraints and conditions. Tokenization breaks up a text to facilitate the whole process of analyzing a language in detail.

Spelling correction: The collected text data was full of spelling blunders, and hence the raw data had to undergo a decent scrutinization. In this dataset, most of the errors were type-error. Consequently, we made corrections to these words. This process was primarily done by a deep learning-based Bengali spell checker module which used the "Bangla Academy" word database as the training system. We pass every article and get a text without spelling errors.

Stop words removal: Stop words are the words being used in a language without coordinating meaningful information. In English, there are a handful of stop words that do not denote any particular meaning rather than helping another phrase or words to make sense of. In order to train a model, it's very important to remove the stop words since stop words exist in a high quantity without providing any meaningful and unique sense.

Punctuation removal: Punctuation can play an important role when it comes to creating an emotional vibe to the expression. Apart from that, punctuation has no role when data has to be converted to a word embedding method. In our project, the word embedding method was incorporated, and for that reason, punctuations were to be removed beforehand.

Digit removing: The data used in this investigation was mostly based on news coverage or bulletins in written form. There existed plenty of numerical characters inscribed in many forms, such as the date and time of an event, address, phone number, and game score, and so on. Generally, these numerical values do not provide any contextual information that can be used to classify the contents. Therefore, all sorts of numerical characters were discarded before training by the models.

Non-English characters removal: The core purpose of our research was grounded on using an English corpus. Nevertheless, there existed a lot of Non-English Characters. They were insignificant to the objective of our project and negligible for the scheme we have propounded. Therefore, if there were any Non-English Characters, we handled them by either translating them into relevant Bengali characters or by discarding them depending on the context and intendment of those characters.

Emoticon removal: The Internet has opened the horizon of globalization through social media. Emoticons play a vital role in expressing situational emotion while communicating with another person on social media. Nonetheless, the emoticons do not convey a message and do not contain any meaning themselves. Thereupon, Emoticons were removed from the corpus for further investigation.

Word normalization: Many words were not used in their standard form in our corpus. Some of the spellings were distorted, and some of them were in an informal format. For example, Bengali currency Taka is sometimes written in "tk". We converted those words into their original spelling automatically by python programming. This process is named word normalization, and it is a prerequisite for a well-developed NLP model.

Lemmatization: Lemmatization is the method of changing a word to its base structure. The contrast between stemming and lemmatization will be, lemmatization considers the unique circumstance and converts the word to its significant base structure, while stemming simply eliminates the last couple of characters, frequently prompting off base implications and spelling blunders. Lemmatization in phonetics is the way toward gathering the versatile types
of a word so they can be broken down as a solitary thing, recognized by the word's lemma or word reference structure.

Data splitting: Training, validation, and testing are supposed to be the most important phase in the realm of machine learning. In order to train the models, the corpus was methodized in a way, so it becomes compatible with the computation process of the algorithms. Training allows the models to learn from their respective trials and error. Our dataset was split into three segments such as 70% and 10%, and 20%. 70% of our dataset was used for training the models, and the remaining 10% was kept for validation purposes, and the rest 20% is for the testing model. Validation helps the model to evaluate itself and grease the wheels for training repeatedly. After the models were trained, the testing data was used as the testing set for testing the model's performance.

- **Feature Extraction**

After the data pre-processing step, the next essential step is the choice of features on a refined dataset. Supervised deep learning classifiers require textual data in vector form to get trained on it. The textual features are converted into vector form using TF and TF-IDF techniques in this project. Features extraction techniques not only convert textual features into vector form but also helps to find significant features necessary to make predictions. For the most part all features do no contribute to the prediction of the target class. That is the reason feature extraction is the important part in the recognition of happy and unhappy related tweets. What actually Term Frequency(TF) means that, according to what often the term arises within the document? It's measured by TF. This will be achievable with the intention of a term would seem a lot further in lengthy documents than short documents because every document is variant in extent. Like the mode about standardization:

$$\text{TF(t)} = \frac{\text{No. of times term t shows in a document}}{\text{Total no. of terms inside document}}$$

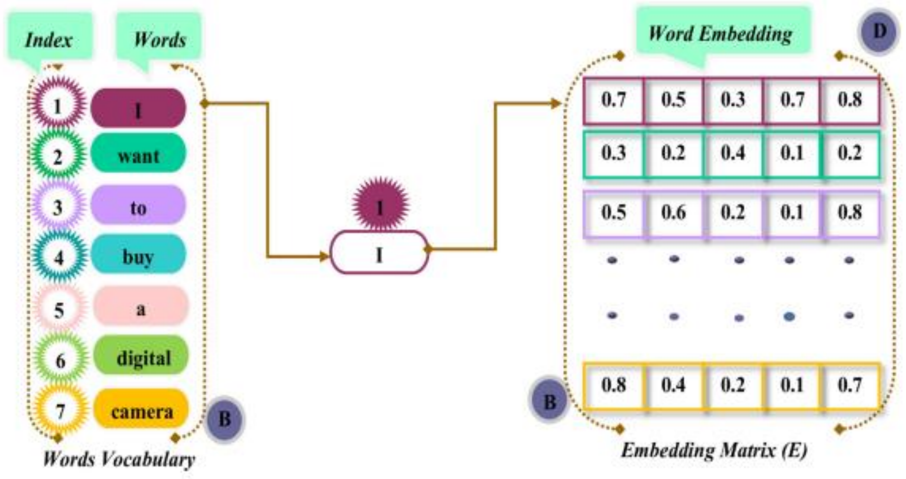
The term frequency be frequently divided with the document length (the total number of terms in the document). IDF: Inverse documents frequency proceeds to find how much a term is significant within the text. Every term is measured equally when TF is computed. Nevertheless, it is recognized that convinced terms, like "is", "of", and "that", can show much more times except contain small prominence. Therefore, frequent terms are needed to be weighed down as level up exceptional ones, through calculating following

$$\text{IDF(t)} = \text{D log(e)} \frac{\text{Total No. of documents}}{\text{No. of documents through term t in it}}$$

Term frequency (TF) is utilized regarding data recovery and shows how regularly an articulation (term, word) happens in a tweets.

Word Embedding's

The semantic meanings of words are provided by word embedding in this project, which is first used in the semantic word cloud generation to the best of our knowledge. We prepare the related text corpus and then train our word embedding by using the continuous bag-of-words (CBOW) model, which is implemented in the open-source toolkit word2vec. CBOW is orders of magnitude faster than the others for training datasets and yields significant gains for dependency parsing. After training, we extract the semantic meanings of all important key words from the word embedding. By using the pre-trained word embedding, each word corresponds a vector in the low dimensional space, typically 50-500.



Embedding Layer

Word Similarity Graph Construction

Important key words are represented as points in a high dimensional space which is hard to understand and difficult to visualize. The goal is to analyse and visualize the relations of them and gain insight of clusters. Dimensionality reduction techniques can be directly applied to project points to a lower dimensional subspace, such as MDS. However, they tend to create a sparse layout. In order to create a sematic word layout with a pleasing layout, we construct a word similarity graph, and then use the graph related algorithms to generate a semantic-preserving and aesthetic word layout. The word similarity graph contains a node set and an edge set with from and to vertex tuples. The node set is the collection of the selected important key words. An edge is constructed between two words if the semantic similarity between two words is above the user specified threshold. The similarity between two vectors can be measured by many metrics, such as the cosine and the jacard similarity. Since the semantic meaning of a word is described by the direction of the vector in word embedding, the cosine similarity is widely used for the NLP task. Thus, we use the cosine similarity to measure the similarity between two words represented in word embedding. The higher the value of the cosine similarity, the more similar the two corresponding words in semantics. The cosine similarity value is also used as the weight of the edge. We then conduct a clustering algorithm based Newman and Girvan’s modularity on the graph, which uses modularity to evaluate the strength of division of a graph into clusters. These clusters can better present the themes in the text.

BULLYING
SCARED
SOCIAL
VIOLENCE
BAD
STRESS
EDUCATION
POWER
BEHAVIOR
HATE
HOME
PAINFUL
CRYING
MOBBING
STOP
RUMOR
THREAT
JOKE
PROBLEM
CONFLICT
PRESSURE
SCHOOL
ABUSE
VICTIM
ANGRY
CYBER
UPSET
WORRY
DESPAIR
ORDER
PREVENTING
SAYING
FEAR
LAUGHING
EDUCATION
RELATIONAL
HOPELESS
HARASSMENT
GOSSIP
ATTITUDE
ALONE
NEGATIVE
EMOTIONAL
FORCE

Word Cloud Layout

After constructing the word similarity graph, we need to project the graph to a 2D space. We first apply MDS to initialize the nodes of the graph in 2D and then apply a force-directed graph layout with the energy model to further optimize the positions of words to generate an aesthetically pleasing semantic word cloud. MDS approximately preserves the distance between nodes, i.e., the **semantic** similarity between words, which can accelerate the convergence of the force-directed graph drawing algorithm by 4% - 7%. The energy model is based on the constructed similarity graph to perform the force-directed graph layout. It assigns the energy among the set of nodes and the set of edges with three kinds of energy, including the attraction energy, the repulsion energy and the gravitation energy. The algorithm iteratively searches for all possible solutions and the minimal energy value corresponds a good layout.

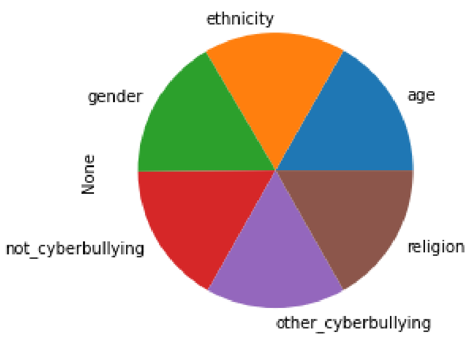
Word Cloud Visualization

With the initial positions of words, we can visualize words without overlapping. The words are positioned by the cluster. In the same cluster, words are rendered in the order of their importance. If the overlap or collision occurs, the word follows the Archimedean spiral to find a new position to render. Clusters are encoded by the colours to help users understand different themes in the texts, which also indicate that the force-directed algorithm satisfies layout requirements. The font size is proportional to the importance value of the word. As a word cloud should guide users fast read and understand the text, the framework supports the exploration of a specific word. When we click an interested word in the word cloud, the texts are sorted by the frequency of the word in the text, and the word is also highlighted in the texts. When we hover the mouse over an interested word, the contexts of the word in the document collection, which are defined as the words appearing in the same sentence with it, are highlighted and other words are faded, with coloured by the number of co-occurrence from cool to warm.

- **CNN – LSTM Classification**

The combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks is a potent architecture for sequence classification, particularly in the context of cyberbullying detection within textual content. The workflow involves meticulous data preprocessing, utilizing techniques such as tokenization, padding, and embedding to convert words into vectors using pre-trained word embeddings or embedding layers. Subsequently, the architecture incorporates one-dimensional convolutional layers (CNN) to capture local patterns in the text. Multiple convolutional filters with varying kernel sizes are employed to detect features at different scales. Max pooling is then applied to extract the most salient features. Following this, the output of the CNN is passed through an LSTM layer to capture long-range dependencies and sequential patterns. The LSTM layer is configured to return only the output at the last timestep, representing the entire sequence.

A classification layer is introduced, comprising a dense layer with a softmax activation function suitable for binary or multi-class classification. The model is trained on labeled data using an optimizer, such as Adam, and monitoring relevant metrics like accuracy, precision, recall, and loss. During the training process, hyperparameter tuning is essential. Experimentation with parameters like learning rate, dropout rates, and the number of filters and LSTM units aids in optimizing model performance. Evaluation on a separate validation set ensures generalization to unseen data, with metrics guiding model assessment. Post-evaluation, fine-tuning or architectural iterations may be performed to enhance performance further. Once satisfied, the model is deployed in a production environment for real-time cyberbullying detection. Ongoing monitoring and maintenance mechanisms are crucial, ensuring the model adapts to evolving language patterns through regular updates with new data. This comprehensive approach leverages the strengths of both CNNs and LSTMs, making it well-suited for nuanced tasks such as cyberbullying detection.



5.2.3. Testing Phase: Cyberbullying Predictor

In this module, Finally, we used matrix factorization to predict cyberbullying words or text and block the post or the user. In this module, user login and give comments about the post and also chat with the other social networking users.

The Cyberbullying Predictor is a user-centric system designed to ensure a safer online experience. It offers a range of features catering to Social Networking (SN) users, including seamless login, input of posts or tweets, cyberbullying detection, warning mechanisms, and user blocking and removal functionalities.

SN User

- **Login**

Provides a secure and user-friendly login interface for authenticating users.

- **Input Post Messages or Tweets**

Enables users to input their messages or tweets in the platform, facilitating normal interaction.

- **Bullying Detection**

Implements advanced machine learning algorithms, including a combination of Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks, to detect instances of cyberbullying within the user's content.

- **Warning Mechanisms**

Issues warnings to users when potentially harmful or offensive content is detected, promoting awareness about responsible online behavior.

- **User Blocking**

Empowers users to block individuals engaging in cyberbullying, limiting further interaction with the offending party.

- **User Removal**

Provides a mechanism to report and remove users engaging in severe or repeated cyberbullying incidents, ensuring a more secure and positive online community.

SN Admin dashboard

The admin dashboard features a graphical visualization of the results generated by the model. The graphs displayed are, namely, monthly and weekly distribution of cyberbullying activities, sentiment analysis distribution of bullying and non-bullying reports, website-wise distribution of cyberbullying instances, list of recent instances of cyberbullying, and worldwide distribution of cyberbullying instances.

- **Report generation:**

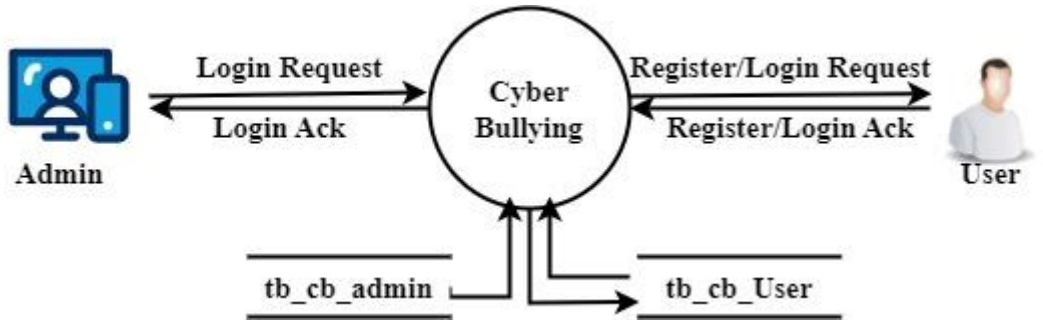
A report is generated to document the incident and can be sent to the authorities for further actions. It is auto-generated using Python docxtpl in .docx format by the application and consists of the following details:

- Name, role, and signature of the admin
- Date and time of the incident
- Description of the incident, containing the URL to the tweet
- Name of the people involved Follow up action

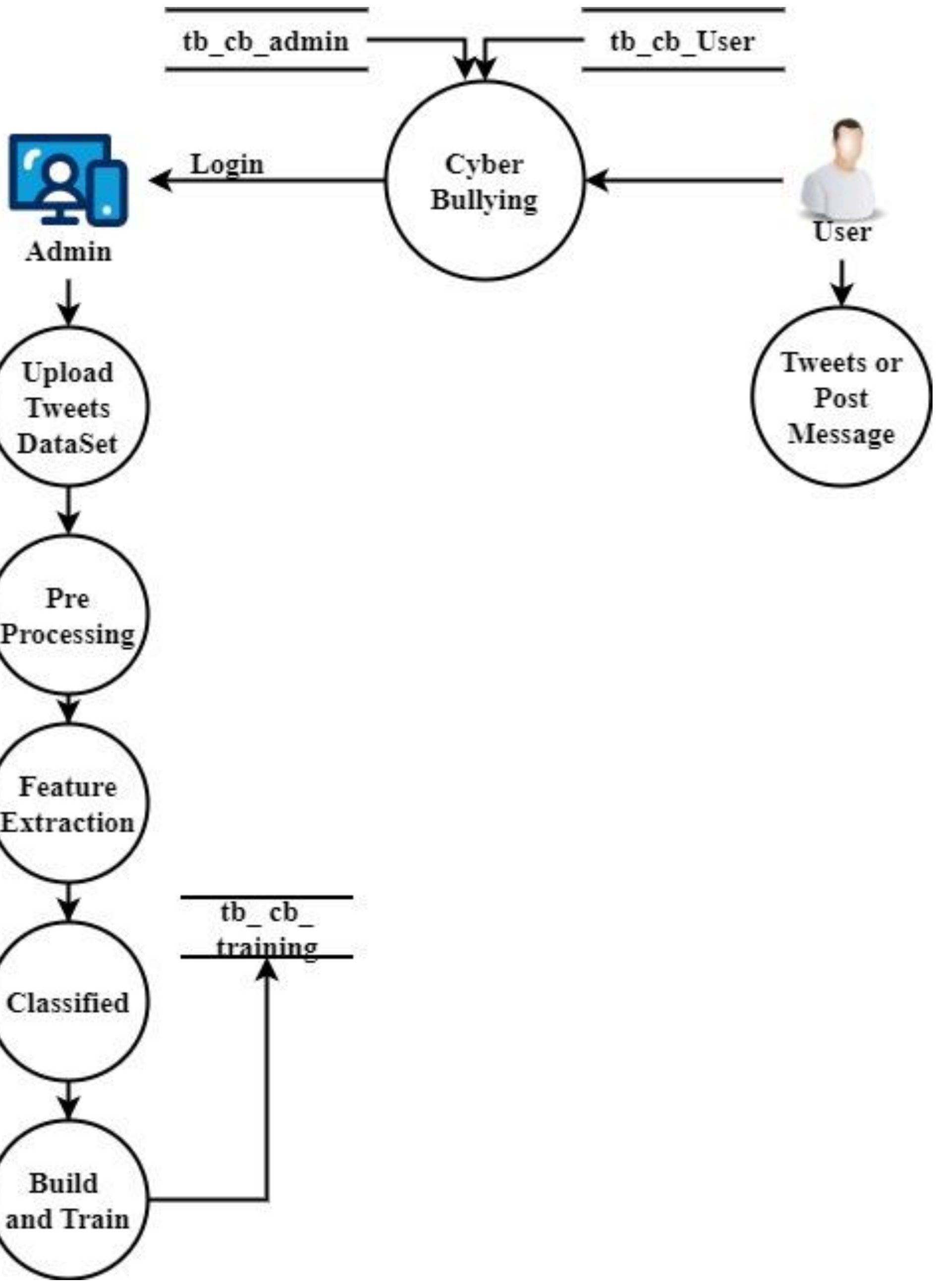
Upon generation, the report can be auto mailed to the pertinent authorities. These details can also be edited by the admin before mailing.

5.3. DATA FLOW DIAGRAM

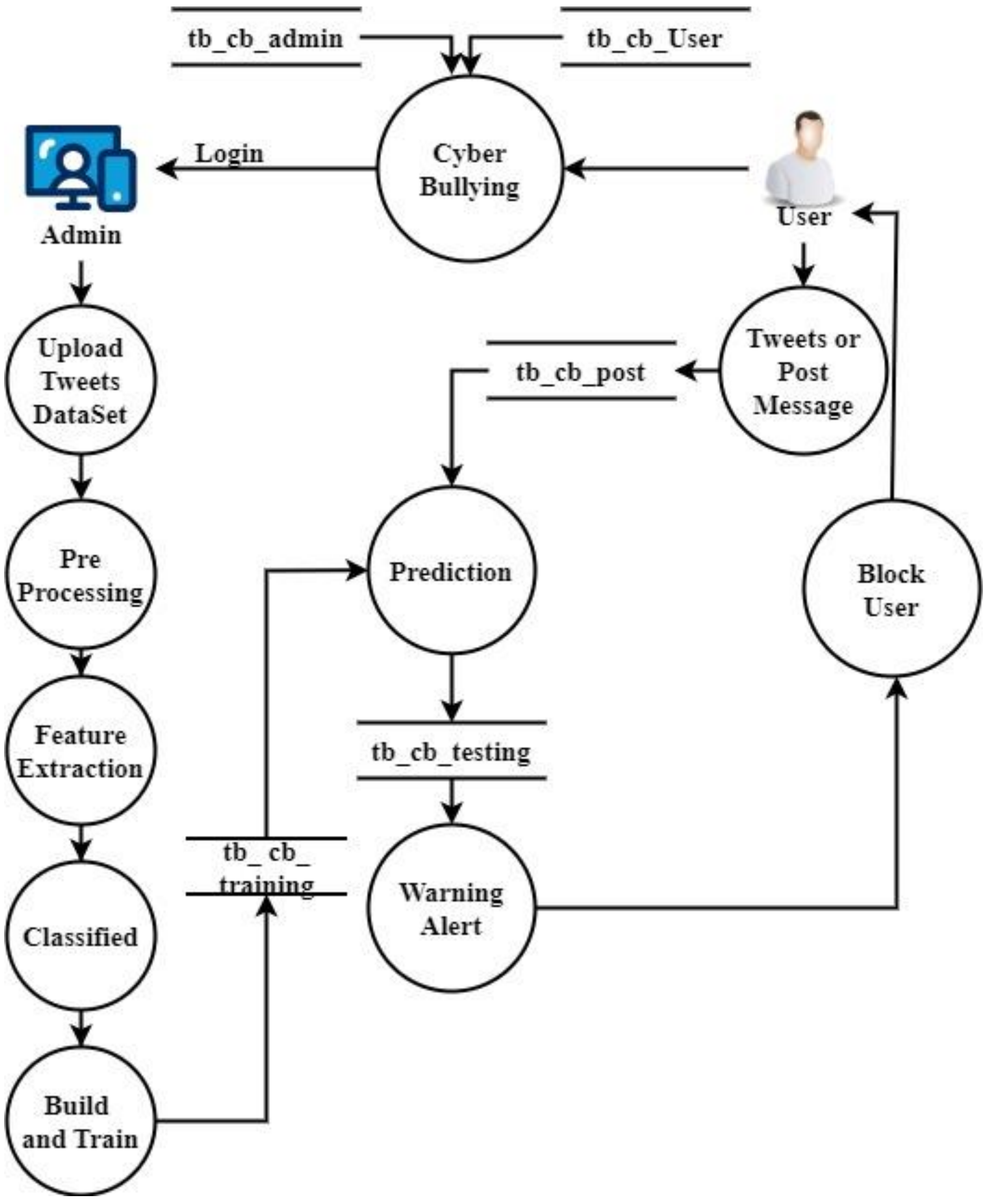
5.3.1. LEVEL 0



5.3.2. LEVEL 1


tb_cb_admin
tb_cb_User
Login
Cyber Bullying
Admin
User
Upload Tweets DataSet
Tweets or Post Message
Pre Processing
Feature Extraction
tb_ cb_ training
Classified
Build and Train

5.3.3. LEVEL 2


tb_cb_admin
tb_cb_User
Cyber Bullying
Login
Admin
User
Upload Tweets DataSet
Tweets or Post Message
tb_cb_post
Pre Processing
Prediction
Block User
Feature Extraction
tb_cb_testing
tb_ cb_ training
Warning Alert
Classified
Build and Train

5.4. DATABASE DESIGN

Table Name: Admin					
S.no	**Field**	**Data Type**	**Field Size**	**Constraint**	**Description**
1	User name	Varchar	20	Null	Admin name
2	Password	Varchar	20	Null	Admin Password

Table Name: BullyNetModel: Build and Train					
S.no	**Field**	**Data Type**	**Field Size**	**Constraint**	**Description**
1	Id	Varchar	20	Null	Id
2	Tweets Dataset	Text	20	Null	Tweets Dataset
3	Bully Net Trained Model	Varchar	20	Null	Bully Net Trained Model
4	Severity Index	Int	11	Null	Severity Index
5	Date time	Timestamp	Timestamp	Null	Date time

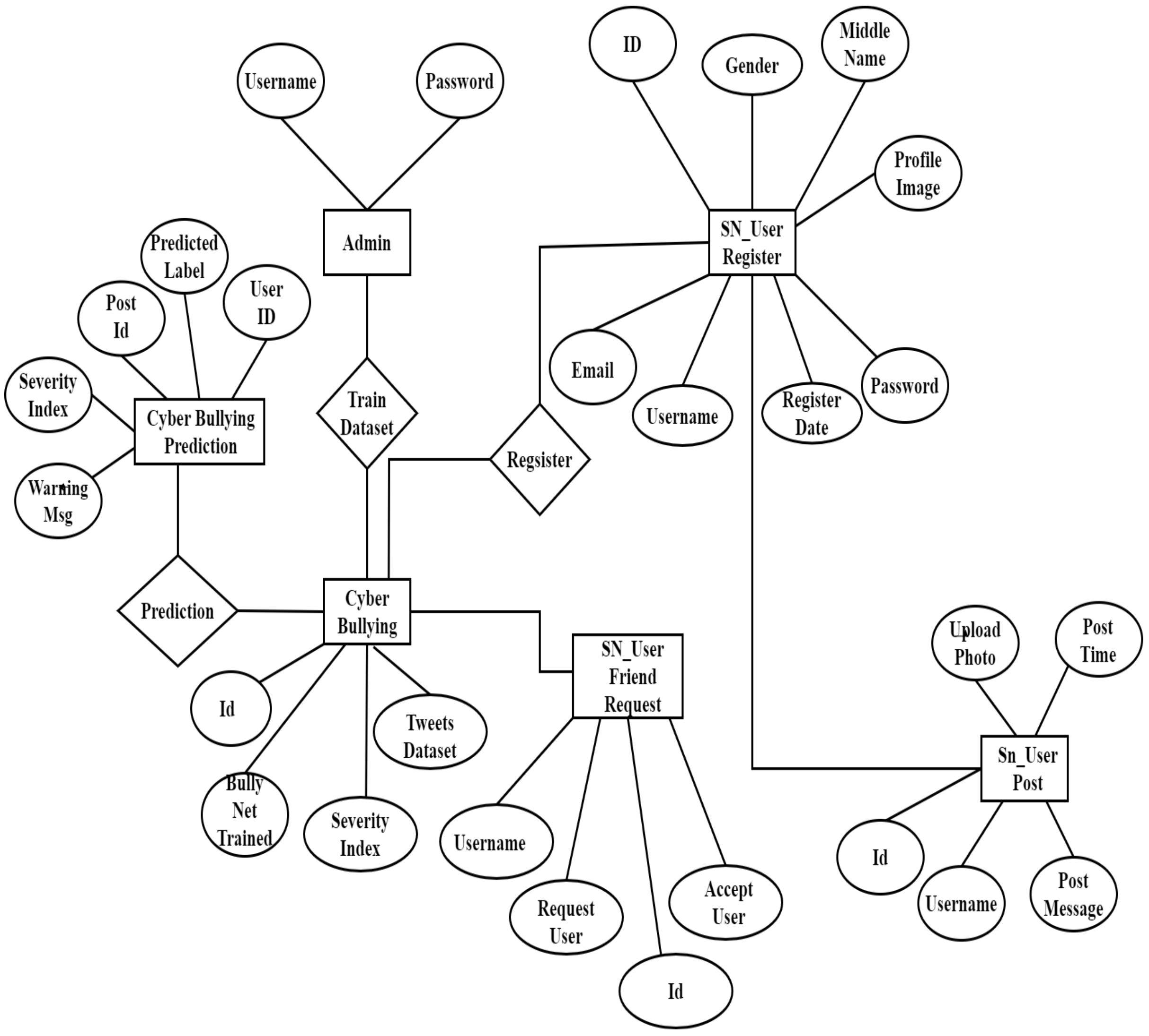
Table Name: Register					
S.no	**Field**	**Data Type**	**Field Size**	**Constraint**	**Description**
1	Id	Int	11	Null	Id
2	Name	Varchar	20	Null	Name
3	Address	Varchar	50	Null	Address
4	Gender	Varchar	20	Null	Gender
5	Email	Varchar	50	Null	Email
6	User Name	Varchar	20	Null	User Name
7	Profile Image	Varchar	30	Null	Profile Image
8	Password	Varchar	20	Null	Password
9	Cover image	Varchar	20	Null	Cover image
10	Create date	Timestamp	Timestamp	Null	Create date

Table Name: Friend Request					
S.no	**Field**	**Data Type**	**Field Size**	**Constraint**	**Description**
1	Id	Int	11	Null	Id
2	User name	Varchar	20	Foreign key	User name
3	Request user	Varchar	20	Null	Request user
4	Accept status	Int	11	Null	Accept status

Table Name: User Post					
S.no	**Field**	**Data Type**	**Field Size**	**Constraint**	**Description**
1	Id	Int	11	Null	Id
2	User Name	Varchar	20	Null	User Name
3	Post Image	Text	200	Null	Post Image
4	Upload photo	Varchar	100	Null	Upload photo
5	Post time	Timestamp	Timestamp	Null	Post time

Table Name: Bully Profile Detection					
S.no	**Field**	**Data Type**	**Field Size**	**Constraint**	**Description**
1	User Id	Int	11	Null	User Id
2	Post id	Int	11	Null	Post id
3	Predicted Label	Varchar	20	Null	Predicted Label
4	Severity Index	Int	11	Null	Severity Index
5	Warning meg	Text	200	Null	Warning meg
6	Blocked User	Varchar	20	Null	Blocked User
7	Date time	Timestamp	Timestamp	Null	Date time

5.5. ER DIAGRAM



CHAPTER 6
SAMPLE CODING

```
##Load data and Pre-processing
df =  pd.read_csv('dataset/cyberbullying_tweets.csv')
stemmer = SnowballStemmer("english")
lematizer=WordNetLemmatizer()
from wordcloud import STOPWORDS
STOPWORDS.update(['rt', 'mkr', 'didn', 'bc', 'n', 'm',
'im', 'll', 'y', 've', 'u', 'ur', 'don',
'p', 't', 's', 'aren', 'kp', 'o', 'kat',
'de', 're', 'amp', 'will'])
def lower(text):
return text.lower()
def remove_hashtag(text):
return re.sub("#[A-Za-z0-9_]+", ' ', text)
def remove_twitter(text):
return re.sub('@\S+|https?:\S+|http?:\S|[^A-Za-z0-9]+', ' ', text)
def remove_stopwords(text):
return " ".join([word for word in
str(text).split() if word not in STOPWORDS]
def stemming(text):
return " ".join([stemmer.stem(word) for word in text.split()])
def lemmatizer_words(text):
return " ".join([lematizer.lemmatize(word) for word in text.split()])
def cleanTxt(text):
text = lower(text)
text = remove_hashtag(text)
text = remove_twitter(text)
text = remove_stopwords(text)
text = stemming(text)
text = lemmatizer_words(text)
return text
```

```
#cleaning the text
df['tweet_clean'] = df['tweet_text'].apply(cleanTxt)
#show the clean text
dat=df.head()
#data visulization
df['tweet_list'] = df['tweet_clean'].apply(lambda x:str(x).split())
top = Counter([item for sublist in df['tweet_list'] for item in sublist])
tweet_list1 = pd.DataFrame(top.most_common(20))
tweet_list1.columns = ['Words','Count']
tweet_list1.style.background_gradient(cmap='Greens')
##TF-IDF Feature Extraction
not_cyberbullying_type = df[df['cyberbullying_type']=='not_cyberbullying']
gender_type = df[df['cyberbullying_type']=='gender']
religion_type = df[df['cyberbullying_type']=='religion']
other_cyberbullying_type = df[df['cyberbullying_type']=='other_cyberbullying']
age_type = df[df['cyberbullying_type']=='age']
ethnicity_type = df[df['cyberbullying_type']=='ethnicity']
# Words in not cyberbullying Tweet
top = Counter([item for sublist in not_cyberbullying_type['tweet_list'] for item in sublist])
type_nc = pd.DataFrame(top.most_common(20))
type_nc.columns = ['Top of Words','Count']
type_nc.style.background_gradient(cmap='Greens')
data2=[]
for ss2 in type_nc.values:
data2.append(ss2)
#graph2
nc_fig = px.bar(type_nc, x="Count", y="Top of Words", title='Top 20 Words in not
cyberbullying Tweet', orientation='h',
width=700, height=700,color='Top of Words')
#nc_fig.show()
# Words in Gender cyberbullying Tweet
top20_gender = Counter([item for sublist in gender_type['tweet_list'] for item in sublist])
type_g = pd.DataFrame(top20_gender.most_common(20))
type_g.columns = ['Top of Words','Count']
type_g.style.background_gradient(cmap='Greens')
data3=[]
for ss3 in type_g.values:
data3.append(ss3)
#g3
g_fig = px.bar(type_g, x="Count", y="Top of Words", title='Top 20 Words in Gender
Cyberbullying Tweet', orientation='h',
width=700, height=700,color='Top of Words')
#g_fig.show()
# Words in Age Cyberbullying Tweet
top20_a = Counter([item for sublist in age_type['tweet_list'] for item in sublist])
type_a = pd.DataFrame(top20_a.most_common(20))
type_a.columns = ['Top of Words','Count']
type_a.style.background_gradient(cmap='Greens')
data6=[]
for ss6 in type_a.values:
data6.append(ss6)
#g6
a_fig = px.bar(type_a, x="Count", y="Top of Words", title='Top 20 Words in Age
cyberbullying Tweet', orientation='h',
width=700, height=700,color='Top of Words')
#a_fig.show()
#Top Words in Ethnicity Cyberbullying Tweet
top20_e = Counter([item for sublist in ethnicity_type['tweet_list'] for item in sublist])
type_e = pd.DataFrame(top20_e.most_common(20))
type_e.columns = ['Top of Words','Count']
type_e.style.background_gradient(cmap='Greens')
data7=[]
for ss7 in type_e.values:
data7.append(ss7)
##g7
e_fig = px.bar(type_e, x="Count", y="Top of Words", title='Top 20 Words in Ethnicity
Cyberbullying Tweet', orientation='h',
width=700, height=700,color='Top of Words')
```

```
#e_fig.show()
#spliting the data
labels = df['cyberbullying_type'].tolist()
df.cyberbullying_type.unique()
ClassIDMap = {'not_cyberbullying': 1, 'gender':2,
'religion':3, 'other_cyberbullying': 4,
'age': 5, 'ethnicity': 6 }
ClassIDMap
corpus, target_labels, target_names = (df['tweet_clean'],
[ClassIDMap[label] for
label in df['cyberbullying_type']],
df['cyberbullying_type'])
df = pd.DataFrame({'tweet text': corpus, 'cyberbullying Label':
target_labels, 'cyberbulying Name': target_names})
#CNN-LSTM Training Loop
def train_data(train_loader):
total_step = len(train_loader)
total_step_val = len(valid_loader)
early_stopping_patience = 4
early_stopping_counter = 0
valid_acc_max = 0 # Initialize best accuracy top 0
for e in range(EPOCHS):
#lists to host the train and validation losses of every batch for each epoch
train_loss, valid_loss  = [], []
#lists to host the train and validation accuracy of every batch for each epoch
train_acc, valid_acc  = [], []
#lists to host the train and validation predictions of every batch for each epoch
y_train_list, y_val_list = [], [
#initalize number of total and correctly classified texts during training and validation
correct, correct_val = 0, 0
total, total_val = 0, 0
running_loss, running_loss_val = 0, 0
####TRAINING LOOP####
model.train()
```

```
for inputs, labels in train_loader:
inputs, labels = inputs.to(DEVICE), labels.to(DEVICE) #load features and targets in device
h = model.init_hidden(labels.size(0))
model.zero_grad() #reset gradients
output, h = model(inputs,h) #get output and hidden states from CNN-LSTM network
loss = criterion(output, labels)
loss.backward()
running_loss += loss.item()
optimizer.step()
y_pred_train = torch.argmax(output, dim=1) #get tensor of predicted values on the training set
y_train_list.extend(y_pred_train.squeeze().tolist()) #transform tensor to list and the values to the list
correct += torch.sum(y_pred_train==labels).item() #count correctly classified texts per batch
total += labels.size(0) #count total texts per batch
train_loss.append(running_loss / total_step)
train_acc.append(100 * correct / total)
####VALIDATION LOOP####
with torch.no_grad():
model.eval()
for inputs, labels in valid_loader:
inputs, labels = inputs.to(DEVICE), labels.to(DEVICE)
val_h = model.init_hidden(labels.size(0))
output, val_h = model(inputs, val_h)
val_loss = criterion(output, labels)
running_loss_val += val_loss.item()
y_pred_val = torch.argmax(output, dim=1)
y_val_list.extend(y_pred_val.squeeze().tolist())
correct_val += torch.sum(y_pred_val==labels).item()
total_val += labels.size(0)
valid_loss.append(running_loss_val / total_step_val)
valid_acc.append(100 * correct_val / total_val)
#Save model if validation accuracy increases
if np.mean(valid_acc) >= valid_acc_max:
```

```
torch.save(model.state_dict(), './state_dict.pt')
print(f'Epoch {e+1}:Validation accuracy increased ({valid_acc_max:.6f} -->
{np.mean(valid_acc):.6f}).  Saving model ...')
valid_acc_max = np.mean(valid_acc)
early_stopping_counter=0 #reset counter if validation accuracy increases
else:
print(f'Epoch {e+1}:Validation accuracy did not increase')
early_stopping_counter+=1 #increase counter if validation accuracy does not increase
if early_stopping_counter > early_stopping_patience:
print('Early stopped at epoch :', e+1)
break
print(f'\tTrain_loss : {np.mean(train_loss):.4f} Val_loss : {np.mean(valid_loss):.4f}')
print(f'\tTrain_acc : {np.mean(train_acc):.3f}% Val_acc : {np.mean(valid_acc):.3f}%')
data2=[v1,v2]
data1=['Cyberbullying','Not Cyberbullying']
doc = dd1 #list(data.keys())
values = dd2 #list(data.values())
# creating the bar plot
plt.bar(doc, values, color ='blue',
width = 0.2)
plt.ylim((1,tot))
plt.xlabel("Class")
plt.ylabel("Count")
plt.title("")
fn="classify.png"
plt.xticks(rotation=0)
plt.savefig('static/graph/'+fn)
```

User Twits

```
if request.method == 'POST':
post= request.form['message']
if 'file' not in request.files:
flash('No file Part')
return redirect(request.url)
file= request.files['file']
```

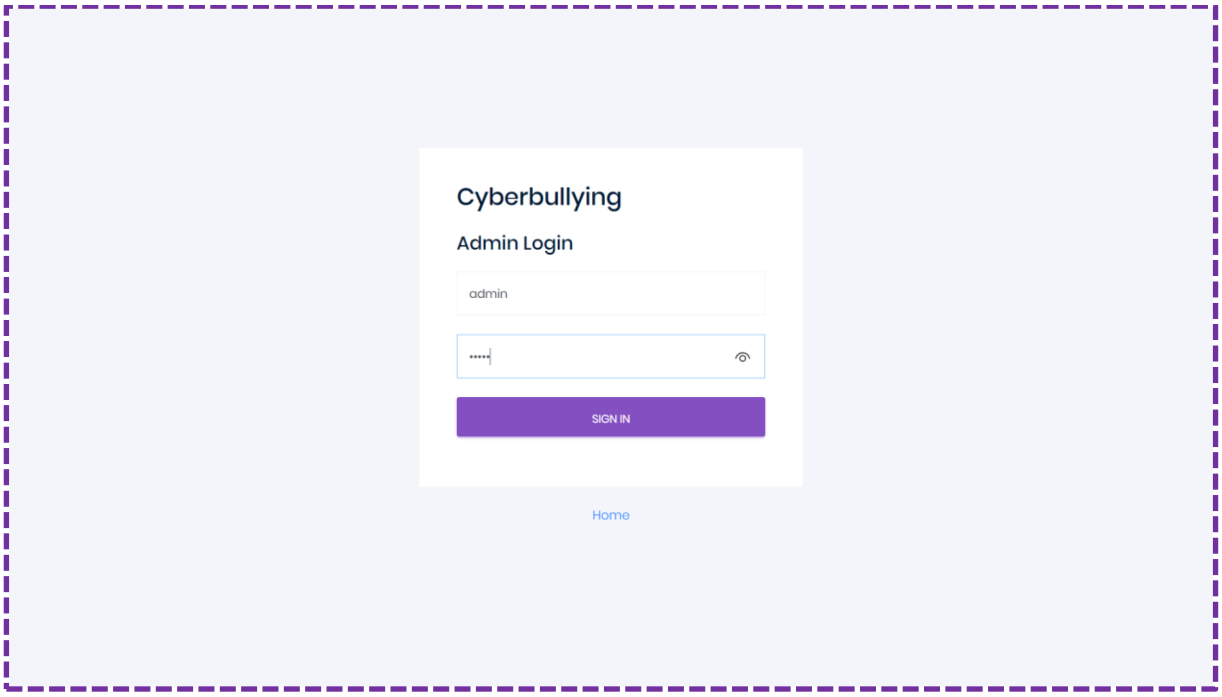
```
mycursor = mydb.cursor()
mycursor.execute("SELECT max(id)+1 FROM user_post")
maxid = mycursor.fetchone()[0]
if maxid is None:
maxid=1
if file.filename == '':
flash('No Select file')
#return redirect(request.url)
if file:
fname = "P"+str(maxid)+file.filename
file_name = secure_filename(fname)
file.save(os.path.join(app.config['UPLOAD_FOLDER']+"/comments/", file_name))
today = date.today()
rdate = today.strftime("%d-%m-%Y")
cursor2 = mydb.cursor()
'''loc = ("dataset.xlsx")
# To open Workbook
wb = xlrd.open_workbook(loc)
sheet = wb.sheet_by_index(0)
nr=sheet.nrows
i=0
while i<nr:
#print(sheet.cell_value(i, 0))
dd=sheet.cell_value(i, 0)
if post.find(dd) != -1:
break
i+=1'''
x=0
deptype=0
f1=open("static/test.txt","r")
dat=f1.read()
f1.close()
dat1=dat.split("|")
for rd in dat1:
```

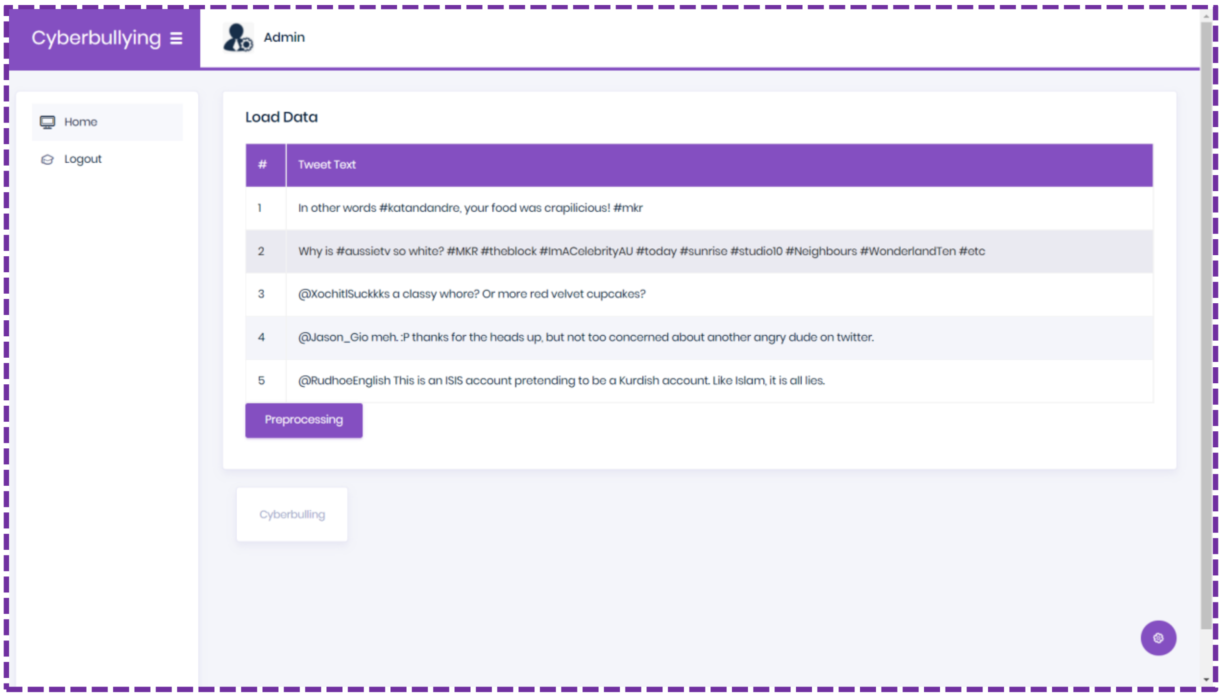
```
t1=post
t2=rd.strip()
if t2 in t1:
act="yes"
st=1
x+=1
break
if cnt==2:
act="warn"
elif cnt>=3:
pcursor2 = mydb.cursor()
pcursor2.execute('update register set dstatus=1 where uname = %s', (uname, ))
mydb.commit()
act="block"
def prediction():
cursor = mydb.cursor()
cursor.execute('SELECT * FROM register where dstatus=0')
data = cursor.fetchall()
cursor.execute('SELECT * FROM register where dstatus=1')
data2 = cursor.fetchall()
cursor.execute('SELECT count(*) FROM user_post')
cnt = cursor.fetchone()[0]
cursor.execute('SELECT count(*) FROM user_post where status=0')
cnt2 = cursor.fetchone()[0]
cursor.execute('SELECT count(*) FROM user_post where status=1')
cnt3 = cursor.fetchone()[0]
per_hu=0
per_bot=0
if cnt2>0:
per_hu=(cnt2/cnt)*100
else:
per_hu=0
if cnt3>0:
per_bot=(cnt3/cnt)*100
```

```
else:
per_bot=0
dat=['Not Cyberbullying','Cyberbullying']
dat1=[per_hu,per_bot]
courses = dat #list(data.keys())
values = dat1 #list(data.values())
fig = plt.figure(figsize = (10, 5))
# creating the bar plot
plt.bar(courses, values, color ='maroon',
width = 0.4)
plt.xlabel("Prediction")
plt.ylabel("Percentage")
plt.title("")
fn="result.png"
plt.savefig('static/chart/'+fn)
#plt.close()
plt.clf()
```

CHAPTER 7

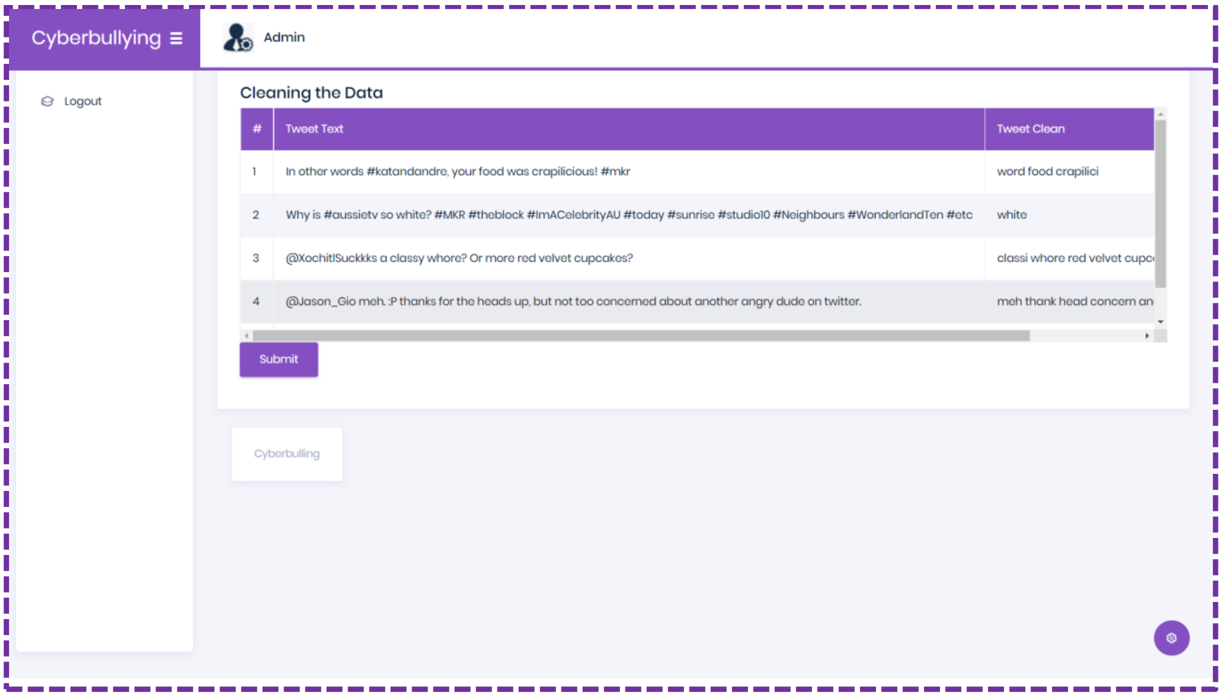
SCREEN LAYOUTS





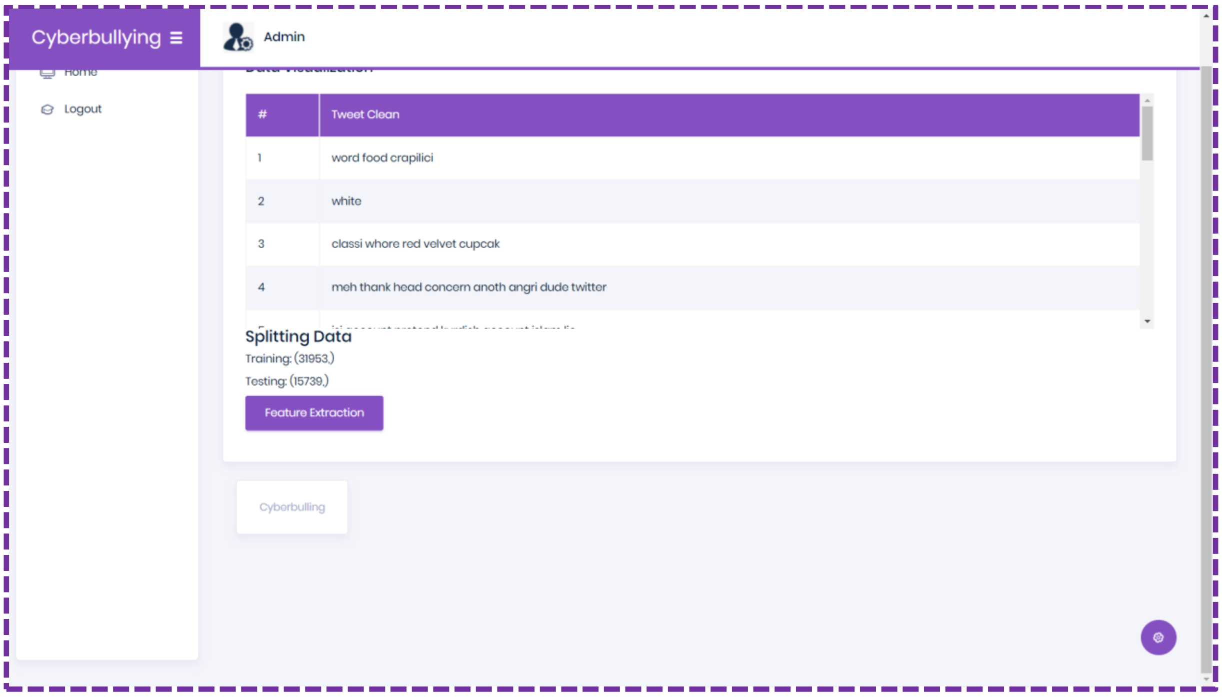
Load Data

#	Tweet Text
1	In other words #katandandre, your food was crapilicious! #mkr
2	Why is #aussietv so white? #MKR #theblock #ImACelebrityAU #today #sunrise #studio10 #Neighbours #WonderlandTen #etc
3	@XochitlSuckkks a classy whore? Or more red velvet cupcakes?
4	@Jason_Gio meh. :P thanks for the heads up, but not too concerned about another angry dude on twitter.
5	@RudhoeEnglish This is an ISIS account pretending to be a Kurdish account. Like Islam, it is all lies.



Cleaning the Data

#	Tweet Text	Tweet Clean
1	In other words #katandandre, your food was crapilicious! #mkr	word food crapilici
2	Why is #aussietv so white? #MKR #theblock #ImACelebrityAU #today #sunrise #studio10 #Neighbours #WonderlandTen #etc	white
3	@XochitlSuckkks a classy whore? Or more red velvet cupcakes?	classi whore red velvet cupc
4	@Jason_Gio meh. :P thanks for the heads up, but not too concerned about another angry dude on twitter.	meh thank head concern an

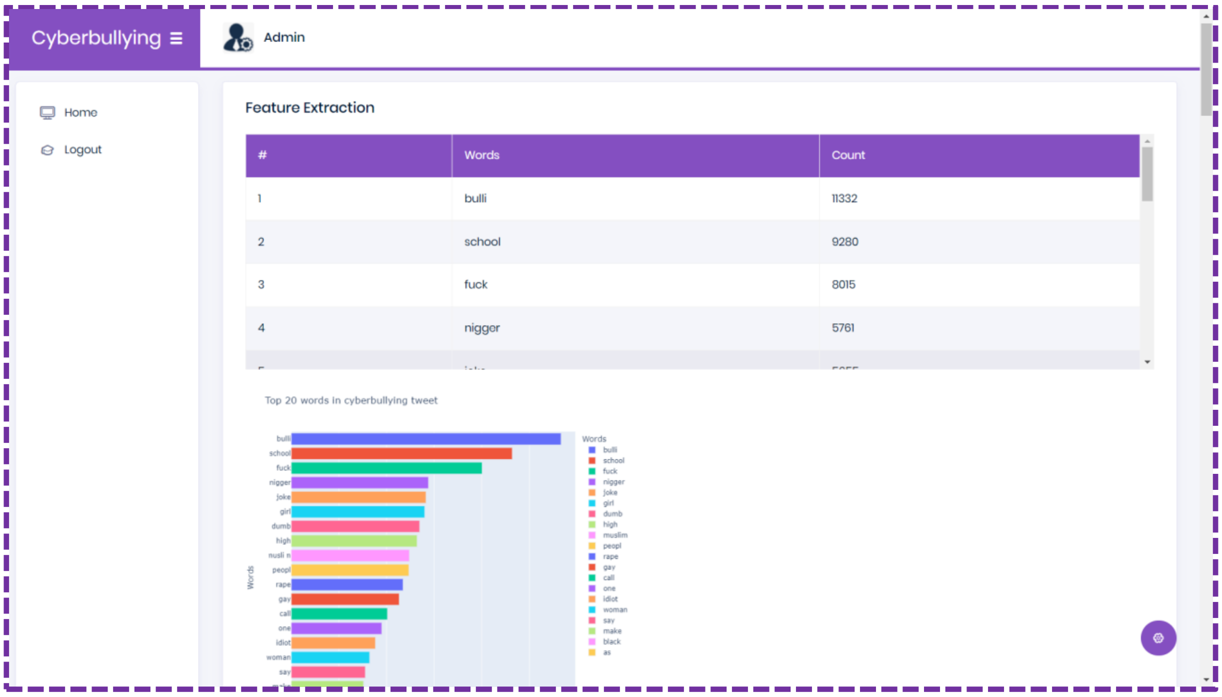


#	Tweet Clean
1	word food crapilici
2	white
3	classi whore red velvet cupcak
4	meh thank head concern anoth angri dude twitter

Splitting Data

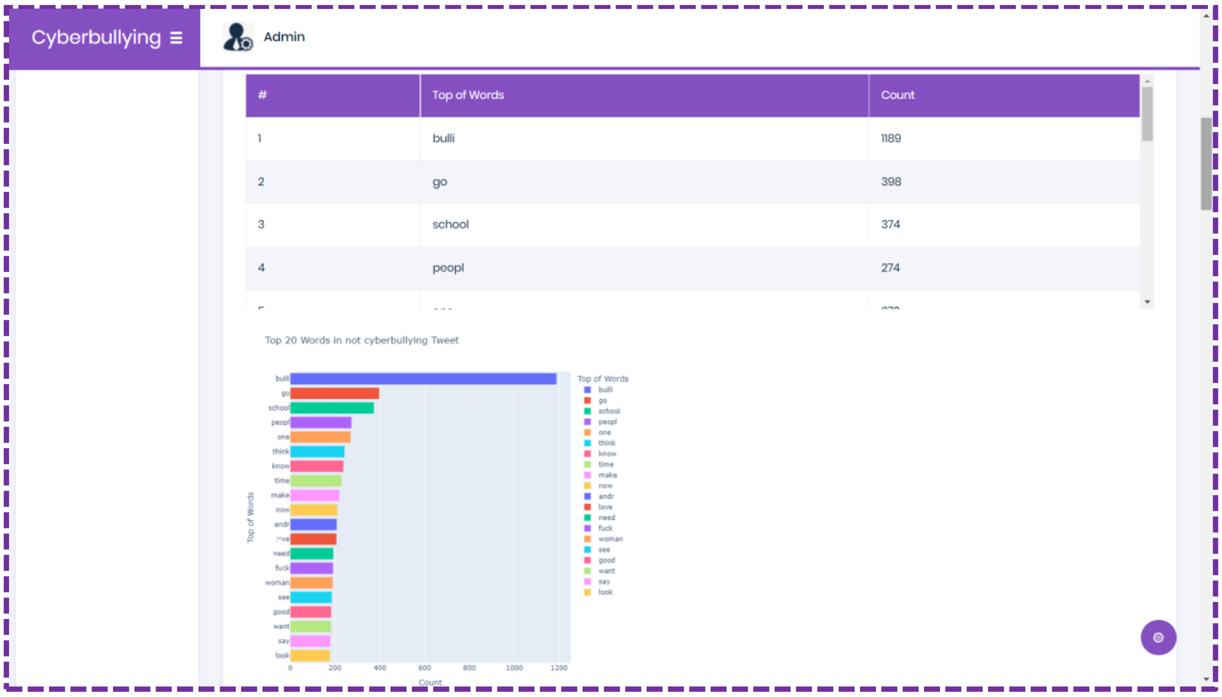
Training: (31953,)

Testing: (15739,)

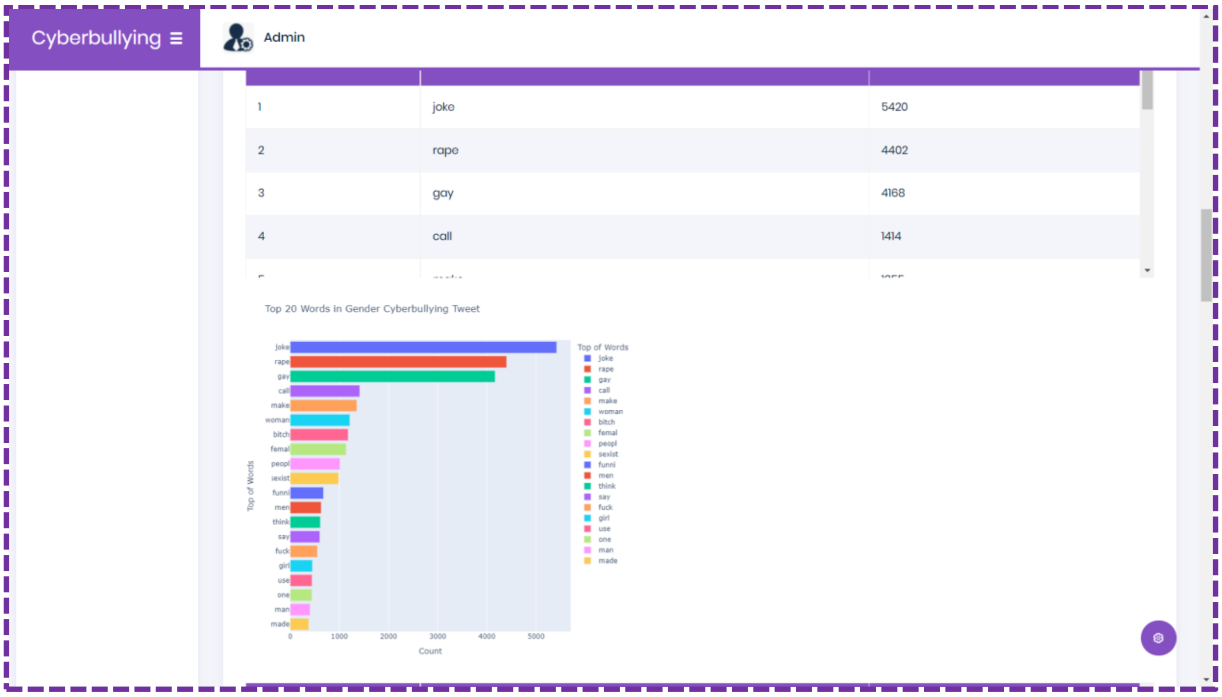


Feature Extraction

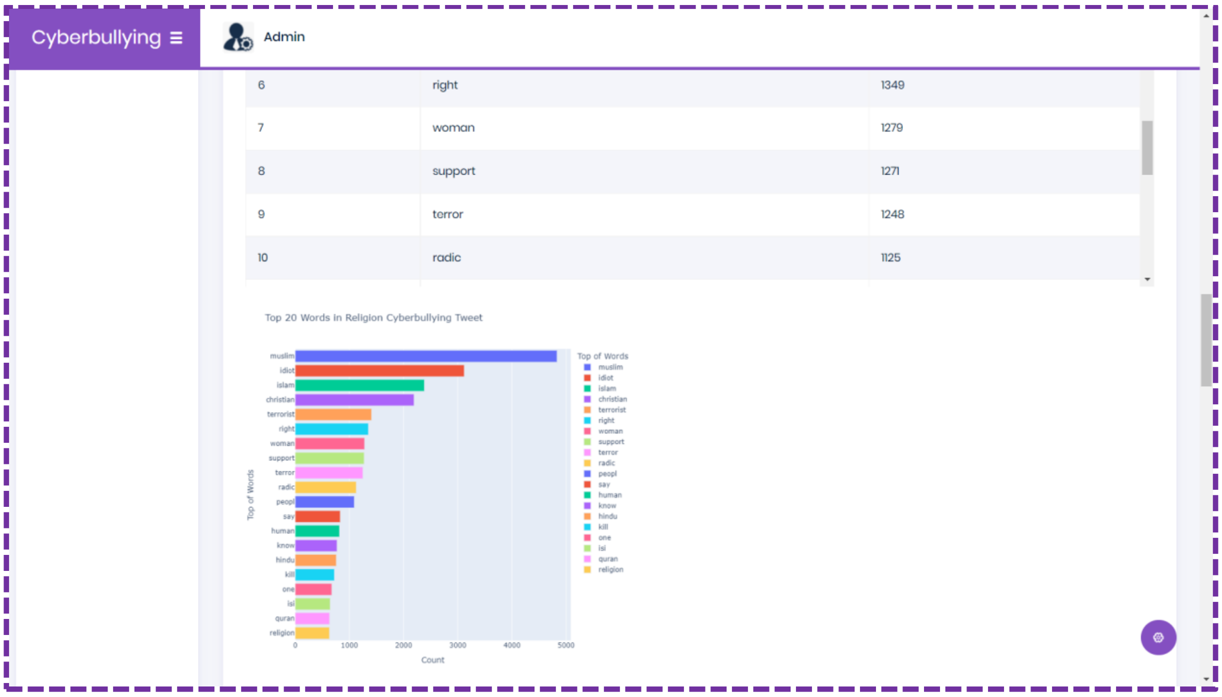
#	Words	Count
1	bulli	11332
2	school	9280
3	fuck	8015
4	nigger	5761



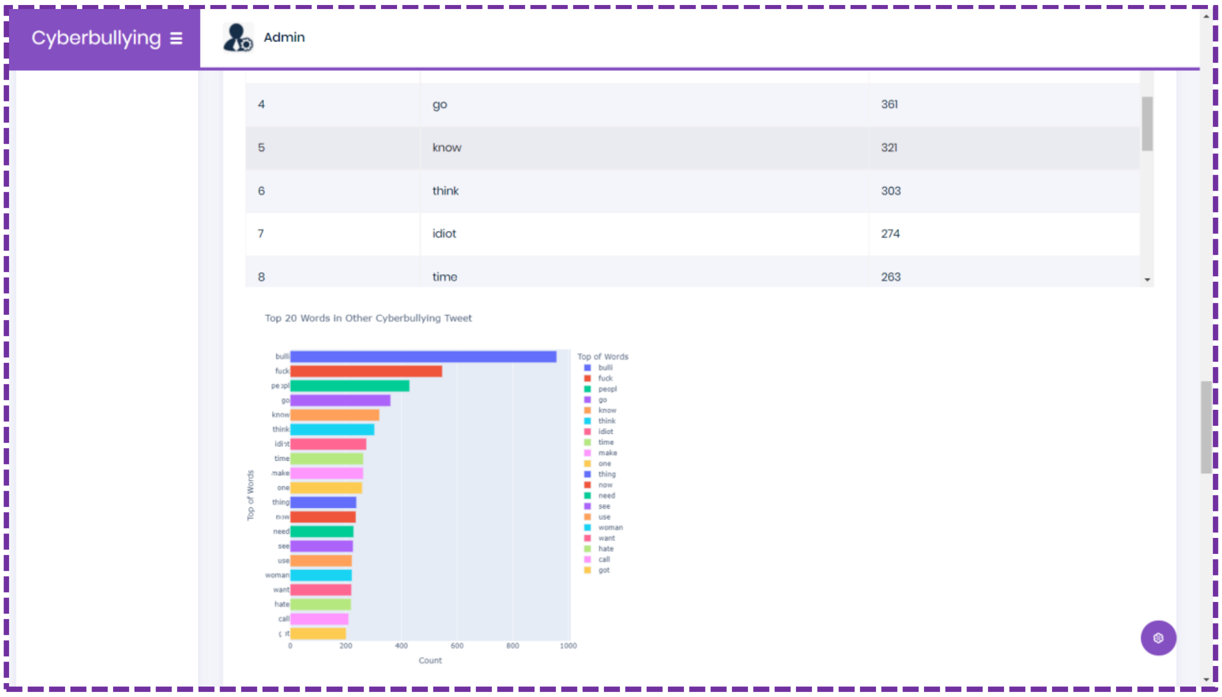
#	Top of Words	Count
1	bulli	1189
2	go	398
3	school	374
4	peopl	274



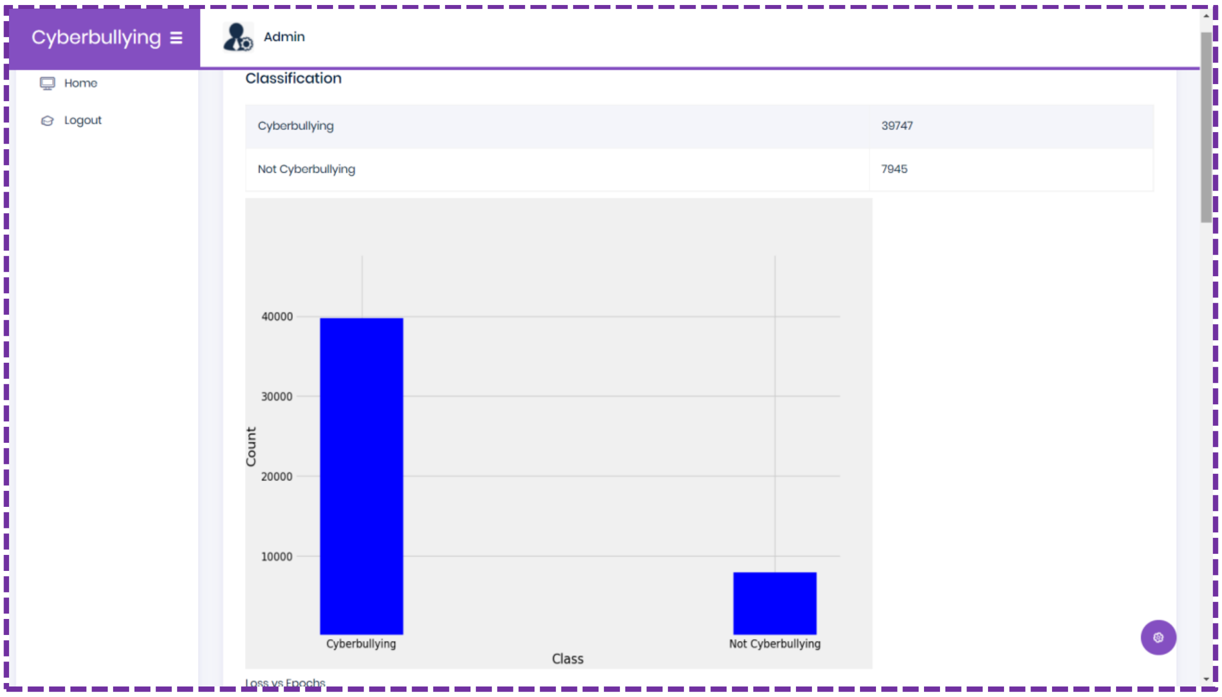
1	joke	5420
2	rape	4402
3	gay	4168
4	call	1414



6	right	1349
7	woman	1279
8	support	1271
9	terror	1248
10	radic	1125

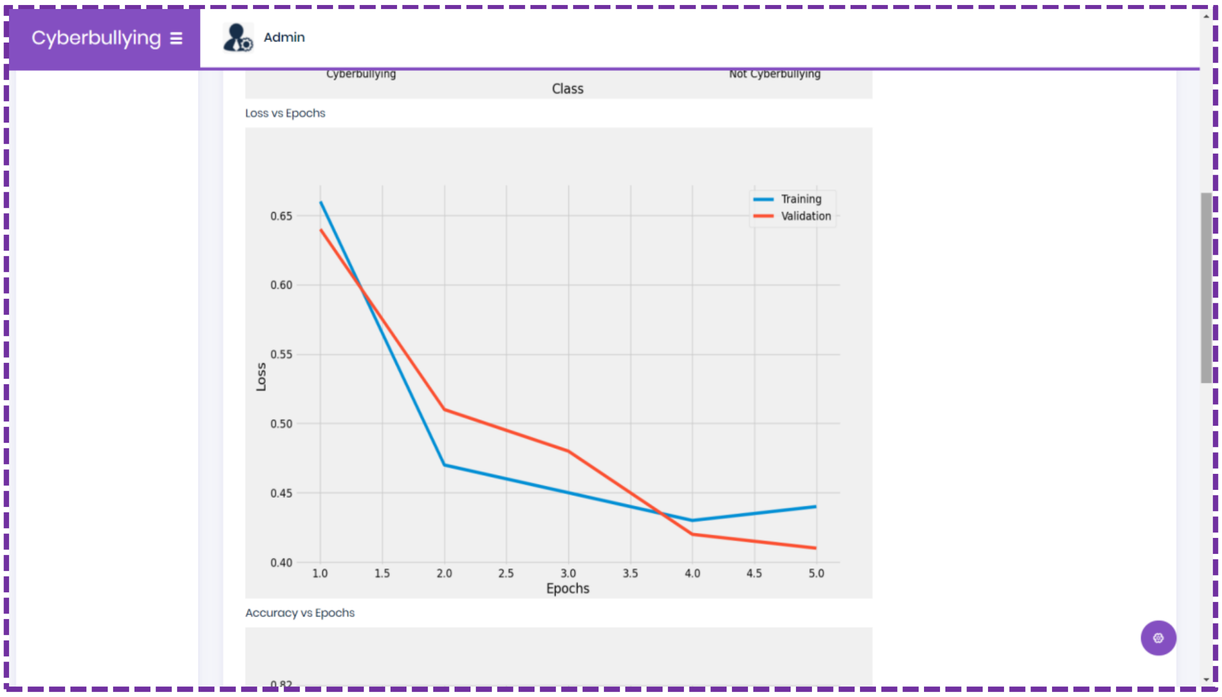


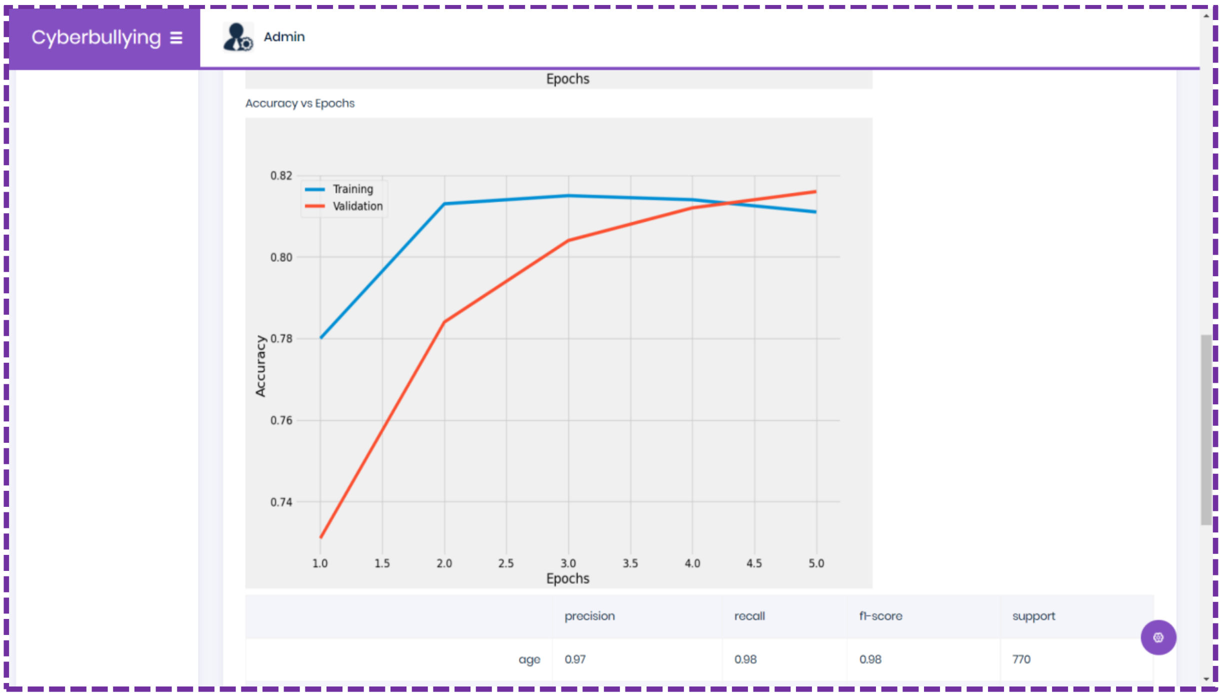
4	go	361
5	know	321
6	think	303
7	idiot	274
8	time	263



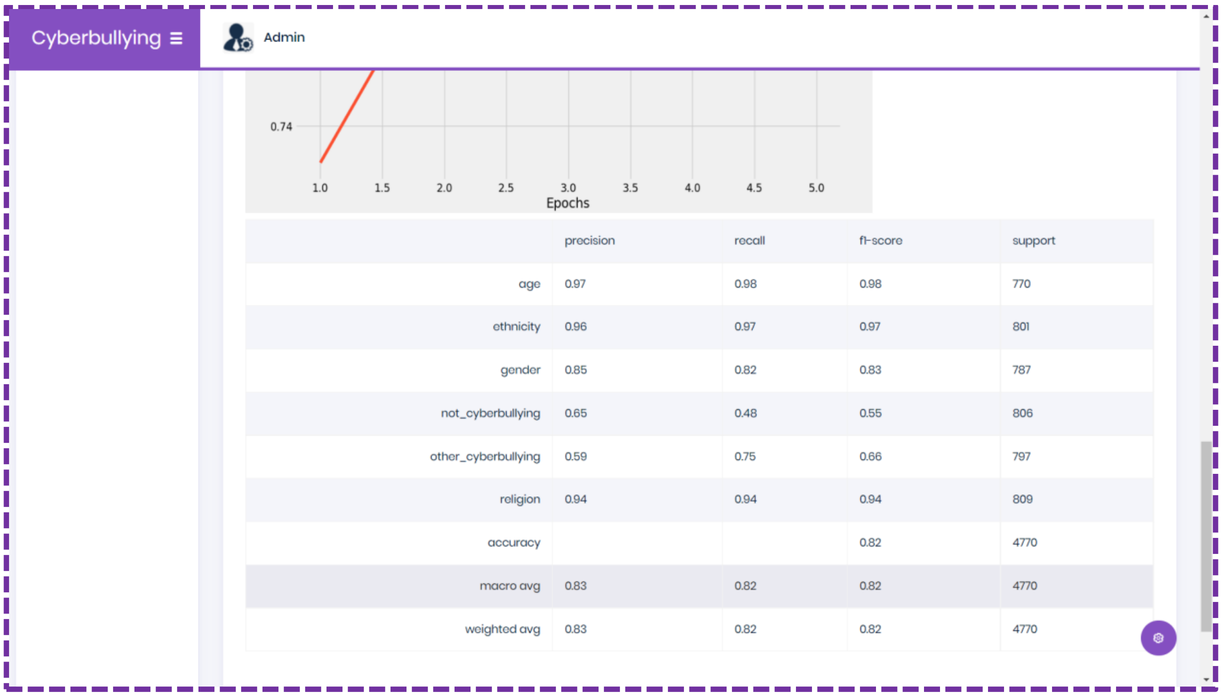
Classification

Cyberbullying	39747
Not Cyberbullying	7945



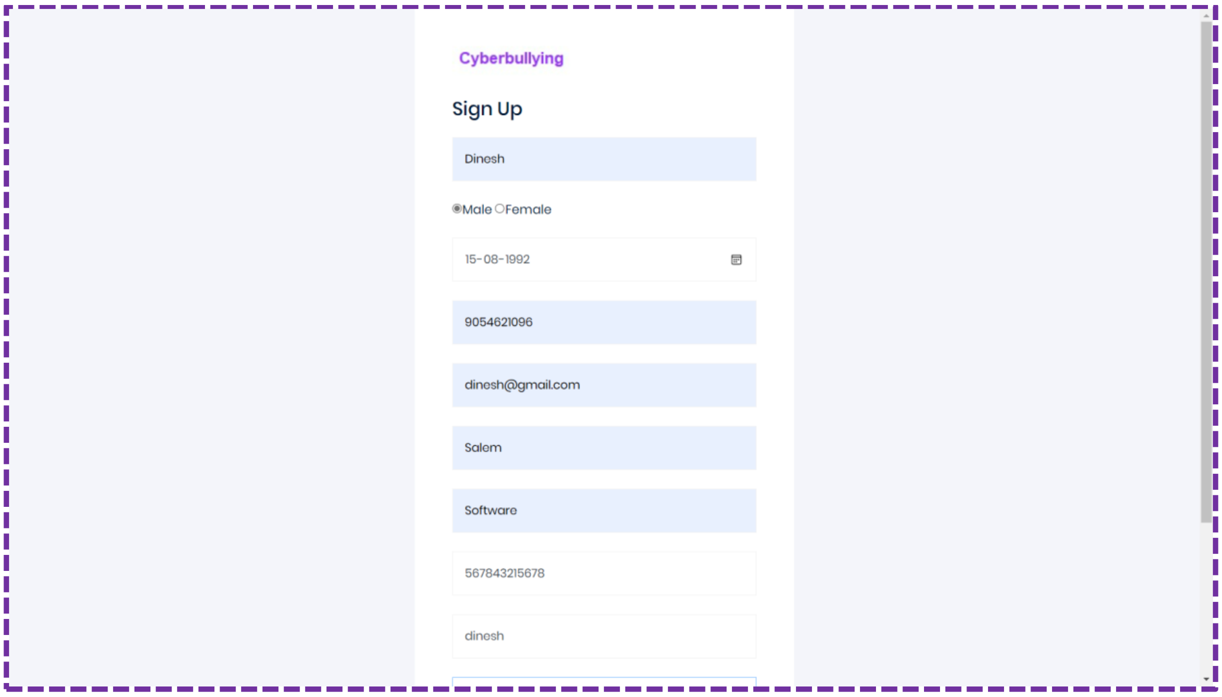


	precision	recall	f1-score	support
age	0.97	0.98	0.98	770



Cyberbullying ≡ Admin

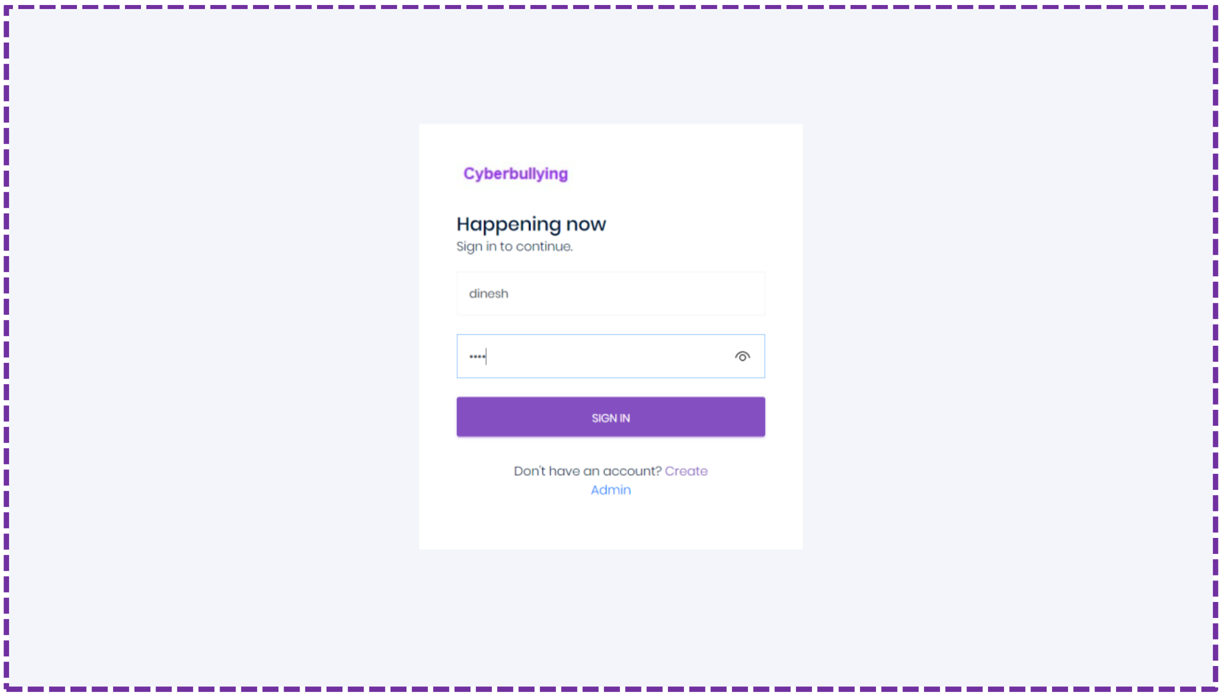
	precision	recall	f1-score	support
age	0.97	0.98	0.98	770
ethnicity	0.96	0.97	0.97	801
gender	0.85	0.82	0.83	787
not_cyberbullying	0.65	0.48	0.55	806
other_cyberbullying	0.59	0.75	0.66	797
religion	0.94	0.94	0.94	809
accuracy			0.82	4770
macro avg	0.83	0.82	0.82	4770
weighted avg	0.83	0.82	0.82	4770

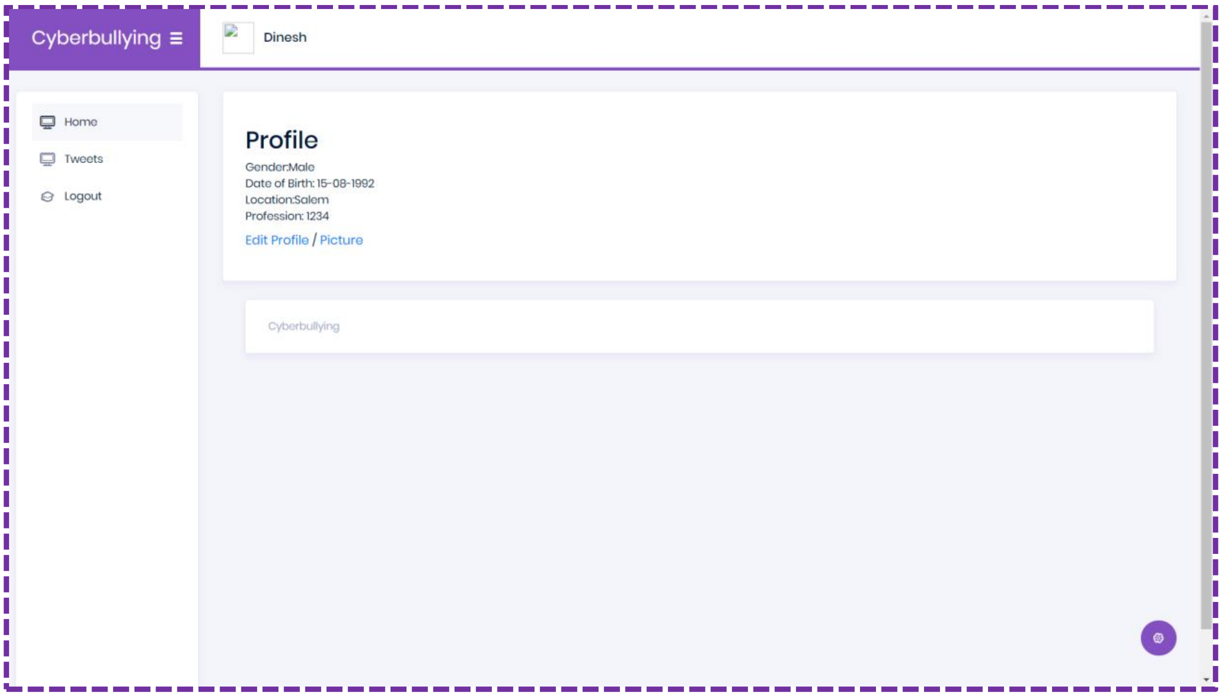


Cyberbullying

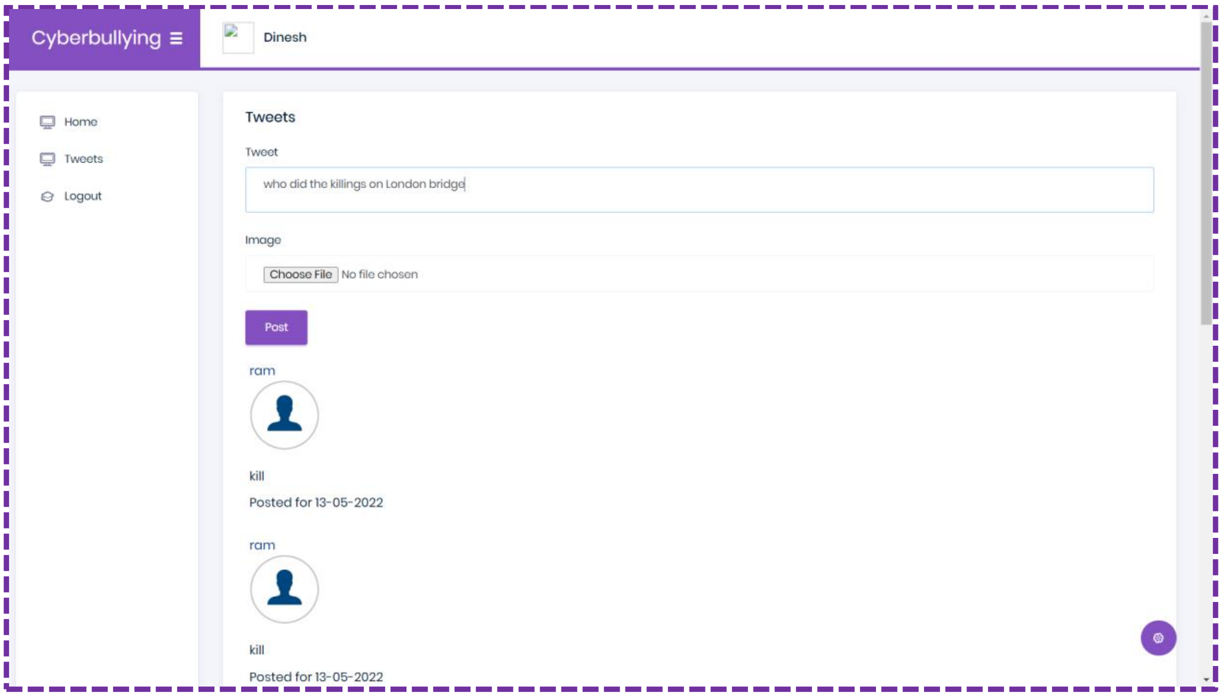
Sign Up

Dinesh
[✓ Male] Female
15-08-1992
9054621096
dinesh@gmail.com
Salem
Software
567843215678
dinesh





Gender:Male
Date of Birth: 15-08-1992
Location:Salem
Profession: 1234



Cyberbullying ≡

Dinesh

Home

Tweets

Logout

Tweets

Tweet

who did the killings on London bridge

Image

Choose File No file chosen

Post

ram

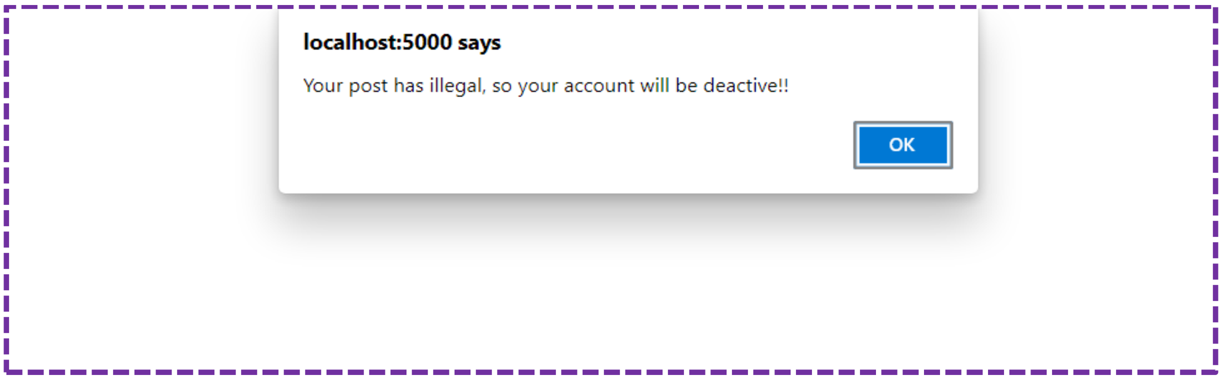
kill

Posted for 13-05-2022

ram

kill

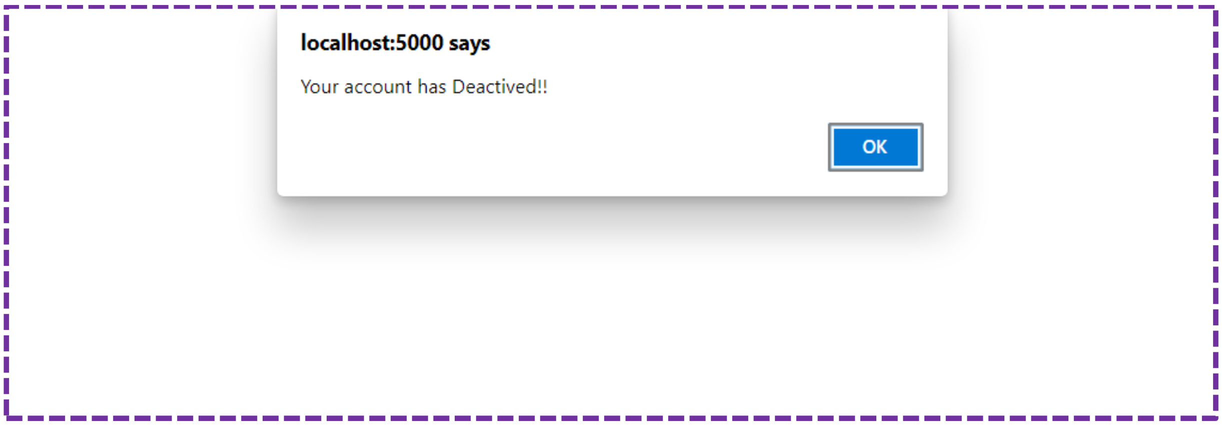
Posted for 13-05-2022



localhost:5000 says

Your post has illegal, so your account will be deactive!!

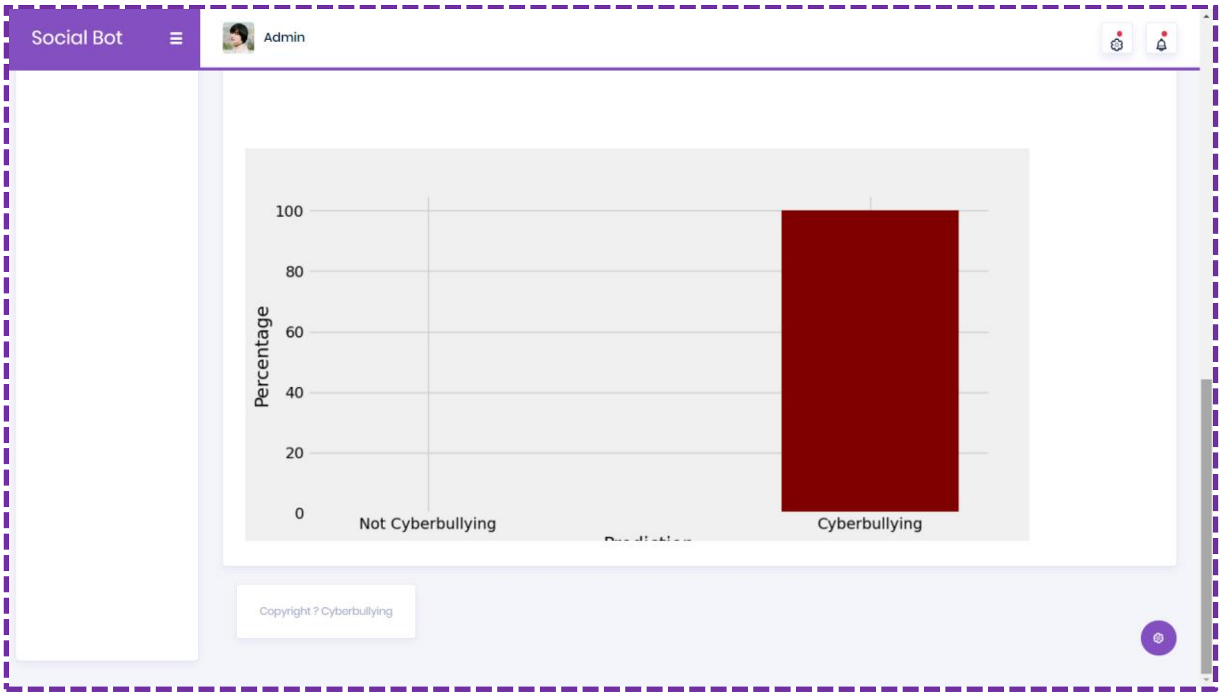
OK



localhost:5000 says

Your account has Deactived!!

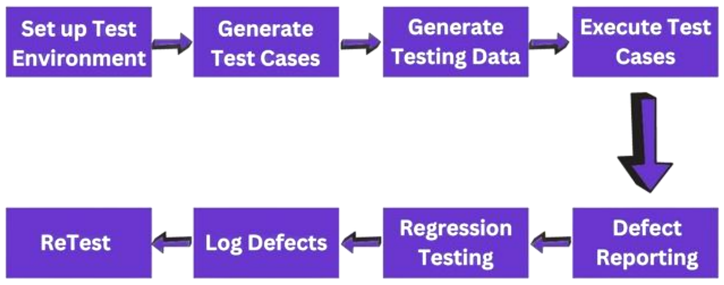
OK


Social Bot
Admin
100
80
60
40
20
0
Percentage
Not Cyberbullying
Cyberbullying
Copyright ? Cyberbullying

CHAPTER 8
SYSTEM TESTING AND IMPLEMENTATION

8.1. SYSTEM TESTING

In this phase of methodology, testing was carried out on the several application modules. Different kind of testing was done on the modules which are described in the following sections. Generally, tests were done against functional and non-functional requirements of the application following the test cases. Testing the application again and again helped it to become a reliable and stable system.



Usability Testing

This was done to determine the usability of the application that was developed. This helped to check whether the application would be easy to use or what pitfalls would the users come through. This was used to determine whether the application is user friendly. It was used to ascertain whether a new user can easily understand the application even before interacting with it so much. The major things checked were: the system flow from one page to another, whether the entry points, icons and words used were functional, visible and easily understood by user.

Functional Testing

Functional Testing is defined as a type of testing which verifies that each function of the software application operates in conformance with the requirement specification. This testing mainly involves black box testing and it is not concerned about the source code of the application. Functional tests were done based on different kind of features and modules of the application and observed that whether the features are met actual project objectives and the modules are hundred percent functional. Functional tests, as shown in the following Table-1 to Table-5, were done based on use cases to determine success or failure of the system implementation and design. For each use case, testing measures were set with results being considered successful or unsuccessful. Below are the tables which are showing some of the major test cases along with their respective test results.

8.1.1. TEST CASES

1. **Test Case ID:** TC001
    - **Input:** A non-bullying tweet "Enjoying a peaceful day."
    - **Expected Result:** Classification result: 0 (No bullying)
    - **Actual Result:** 0 (No bullying)
    - **Status:** Pass
2. **Test Case ID:** TC002
    - **Input:** A bullying tweet "You are so stupid!"
    - **Expected Result:** Classification result: 1 (Bullying)
    - **Actual Result:** 1 (Bullying)
    - **Status:** Pass
3. **Test Case ID:** TC003
    - **Input:** A neutral tweet with emojis "Feeling happy today! ☺"
    - **Expected Result:** Classification result: 0 (No bullying)
    - **Actual Result:** 0 (No bullying)
    - **Status:** Pass
4. **Test Case ID:** TC004
    - **Input:** A tweet with a URL "Check out this interesting article: [URL]"
    - **Expected Result:** Preprocessing removes the URL, classification result: 0 (No bullying)
    - **Actual Result:** 0 (No bullying)
    - **Status:** Pass
5. **Test Case ID:** TC005
    - **Input:** An abusive tweet with misspelled words "U r so dum!"
    - **Expected Result:** Classification result: 1 (Bullying)
    - **Actual Result:** 1 (Bullying)
    - **Status:** Pass
6. **Test Case ID:** TC006
    - **Input:** Empty tweet
    - **Expected Result:** Classification result: 0 (No bullying)
    - **Actual Result:** 0 (No bullying)
    - **Status:** Pass

7. **Test Case ID:** TC007
    - **Input:** Long tweet exceeding character limit
    - **Expected Result:** Proper handling and classification result: 1 (Bullying)
    - **Actual Result:** 1 (Bullying)
    - **Status:** Pass
8. **Test Case ID:** TC008
    - **Input:** User registration with valid credentials
    - **Expected Result:** Successful registration and login
    - **Actual Result:** Successful registration and login
    - **Status:** Pass
9. **Test Case ID:** TC009
    - **Input:** User login with incorrect credentials
    - **Expected Result:** Unsuccessful login, error message displayed
    - **Actual Result:** Unsuccessful login, error message displayed
    - **Status:** Pass
10. **Test Case ID:** TC010
    - **Input:** Blocking a user who posted a bullying tweet
    - **Expected Result:** User is blocked, and the bullying tweet is hidden
    - **Actual Result:** User is blocked, and the bullying tweet is hidden
    - **Status:** Pass

8.1.2. TEST REPORT

Introduction

The Cyberbullying Detection Framework Test was conducted to evaluate the functionality, reliability, and performance of the system designed to Detect, Block, and Prevent Cyberbullying tweets and bully users in Social Networking Sites using LSTM - CNN Algorithms.

Test Objective

The primary objective of this testing was to ensure that the Cyberbullying Detection Framework functions as intended, accurately classifies tweets, and effectively takes preventive actions against cyberbullying incidents.

Test Scope

The testing scope covered the following aspects:

- Cyberbullying tweet classification accuracy.
- Proper functioning of user registration and authentication.
- Handling and processing of various types of tweets (with emojis, URLs, misspelled words, etc.).
- User blocking and preventive measures against cyberbullying.

Test Environment

- **Programming Language:** Python
- **Web Framework:** Flask
- **Deep Learning Libraries:** TensorFlow
- **Database:** MySQL
- **Frontend Framework:** Bootstrap
- **Test Data:** Diverse set of tweets including non-bullying, bullying, and edge cases.
- **Test Browser:** Chrome or Firefox

Test Results

The testing process was executed systematically, and the results are as follows:

- **Cyberbullying Detection Accuracy:** The LSTM - CNN model demonstrated high accuracy in classifying cyberbullying tweets. A diverse set of test cases resulted in accurate and expected classifications.
- **User Authentication:** User registration and authentication processes were successful. Invalid credentials resulted in appropriate error messages, and the system handled authentication securely.

- **Tweet Processing:** The system effectively processed tweets with emojis, URLs, misspelled words, and other variations. Preprocessing steps were observed to remove unwanted elements successfully.
- **User Blocking and Preventive Measures:** Blocking users and implementing preventive measures against cyberbullying, such as hiding or warning about offensive content, were successfully executed.

Test Conclusion

The Cyberbullying Detection Framework has demonstrated robust functionality and reliability in detecting, blocking, and preventing cyberbullying incidents. The system meets the specified requirements and objectives, providing a promising solution for creating a safer online environment. The test results indicate that the framework is ready for deployment with high confidence in its ability to address cyberbullying issues effectively.

8.2. SYSTEM IMPLEMENTATION

System implementation is the phase in the system development life cycle (SDLC) where the designed system is turned into a working system. It involves the actual construction, coding, testing, deployment, and maintenance of the software or information system. The goal is to ensure that the system is operational, meets user requirements, and aligns with the organization's objectives. Below are the key steps and considerations involved in the system implementation phase:

- **Environment Setup**

To begin the system implementation, establish the necessary development environment. Install Python, Flask, and the required libraries for deep learning algorithms such as TensorFlow or PyTorch for LSTM-CNN. Additionally, incorporate natural language processing (NLP) packages like NLTK or spaCy, and set up MySQL for database interactions. For a polished front-end, include Bootstrap in the project.

- **LSTM-CNN Model**

Implement the LSTM-CNN model responsible for detecting cyberbullying tweets. Train the model using appropriate cyberbullying data and save the trained model for later use in the application.

- **Database Integration**

Integrate MySQL with Flask to handle user data and other cyberbullying-related information. This integration ensures efficient data storage and retrieval within the system.

- **Frontend Design**

Create visually appealing and user-friendly templates using HTML, CSS, and Bootstrap. Design forms for user registration, login, and tweet posting, making the user interface intuitive.

- **Backend Implementation**

Implement the backend logic within Flask routes. This includes handling user requests, processing tweets, and leveraging the LSTM-CNN model for cyberbullying detection. Ensure smooth communication between the frontend and backend components.

- **Testing**

Thoroughly test the system to ensure the correct functioning of all implemented functionalities. Conduct testing with various scenarios to validate the effectiveness of the cyberbullying detection and prevention mechanisms.

- **Deployment**

Deploy the system on a server, utilizing WampServer or another preferred web server. Ensure proper configuration and security measures to protect user data and maintain system integrity.

- **Maintenance**

Establish a regular maintenance routine to address bugs, enhance system performance, and introduce new features. This ongoing process ensures the system remains effective and up-to-date.

- **Documentation**

Document all aspects of the implementation, including system architecture, functionalities, and usage instructions. Comprehensive documentation aids in future reference and troubleshooting.

- **User Training**

Provide user training, particularly for administrators responsible for handling cyberbullying incidents. Ensure users understand the system's functionalities and can navigate it effectivelz.

CHAPTER 9
CONCLUSION AND FUTURE ENHANCEMENT

9.1. CONCLUSION

In conclusion, the project focused on developing a robust framework to tackle the pervasive issue of cyberbullying within social networking sites. The integration of advanced LSTM-CNN algorithms and Natural Language Processing (NLP) techniques, implemented through a Flask web application, has yielded significant achievements. The framework demonstrated notable success in effectively detecting and classifying cyberbullying tweets, showcasing its potential to contribute significantly to creating safer online environments. The user interface, developed using Flask and Bootstrap, stands out for its user-friendliness, offering a seamless and intuitive experience. One of the standout features is the dynamic action handling capability of the system. From user blocking to preventive measures and warning emails, administrators have been empowered to respond promptly to cyberbullying incidents. Additionally, robust user authentication mechanisms have been implemented, enhancing the overall security of the platform and safeguarding user data. The project has made significant strides in laying the foundation for a safer online environment. By addressing the complex issue of cyberbullying, the framework provides immediate value and stands as a testament to the commitment to fostering respectful online interactions. The journey doesn't end here; it continues with a dedication to refining and optimizing the framework to ensure its effectiveness in the ever-evolving digital landscape.

9.2. FUTURE ENHANCEMENT

For the present, the bot works for Twitter, so it can be extended to various other social media platforms like Instagram, Reedit, etc. Currently, only texts are classified for twitter content, classifying image, videos could be an addition. A report tracking feature could be added along with a cross-platform Mobile / Desktop application (Progressive Web App) for the Admin. This model could be implemented for many languages like French, Spanish, Russian, etc. along with India languages like Hindi, Gujarati, etc

CHAPTER 10
BIBILOGRAPHY

10.1 REFERENCES

1. A. S. Srinath, H. Johnson, G. G. Dagher and M. Long, "BullyNet: Unmasking cyberbullies on social networks", IEEE Trans. Computat. Social Syst., vol. 8, no. 2, pp. 332-344, Apr. 2021.
2. Z. L. Chia, M. Ptaszynski, F. Masui, G. Leliwa and M. Wroczynski, "Machine learning and feature engineering-based study into sarcasm and irony classification with application to cyberbullying detection", Inf. Process. Manage., vol. 58, no. 4, Jul. 2021.
3. N. Yuvaraj, K. Srihari, G. Dhiman, K. Somasundaram, A. Sharma, S. Rajeskannan, et al., "Nature-inspired-based approach for automated cyberbullying classification on multimedia social networking", Math. Problems Eng., vol. 2021, pp. 1-12, Feb. 2021.
4. R. R. Dalvi, S. B. Chavan and A. Halbe, "Detecting a Twitter cyberbullying using machine learning", Ann. Romanian Soc. Cell Biol., vol. 25, no. 4, pp. 16307-16315, 2021.
5. N. Yuvaraj, V. Chang, B. Gobinathan, A. Pinagapani, S. Kannan, G. Dhiman, et al., "Automatic detection of cyberbullying using multi-feature based artificial intelligence with deep decision tree classification", Comput. Electr. Eng., vol. 92, Jun. 2021.
6. A. Al-Hassan and H. Al-Dossari, "Detection of hate speech in Arabic tweets using deep learning", Multimedia Syst., Jan. 2021.
7. Y. Fang, S. Yang, B. Zhao and C. Huang, "Cyberbullying detection in social networks using bi-GRU with self-attention mechanism", Information, vol. 12, no. 4, pp. 171, Apr. 2021.
8. B. A. Talpur and D. O'Sullivan, "Multi-class imbalance in text classification: A feature engineering approach to detect cyberbullying in Twitter", Informatics, vol. 7, no. 4, pp. 52, Nov. 2020.
9. A. Agarwal, A. S. Chivukula, M. H. Bhuyan, T. Jan, B. Narayan and M. Prasad, "Identification and classification of cyberbullying posts: A recurrent neural network approach using under-sampling and class weighting" in Neural Information Processing, Cham, Switzerland:Springer, vol. 1333, pp. 113-120, 2020.
10. C. Iwendi, G. Srivastava, S. Khan and P. K. R. Maddikunta, "Cyberbullying detection solutions based on deep learning architectures", Multimedia Syst., 2020.

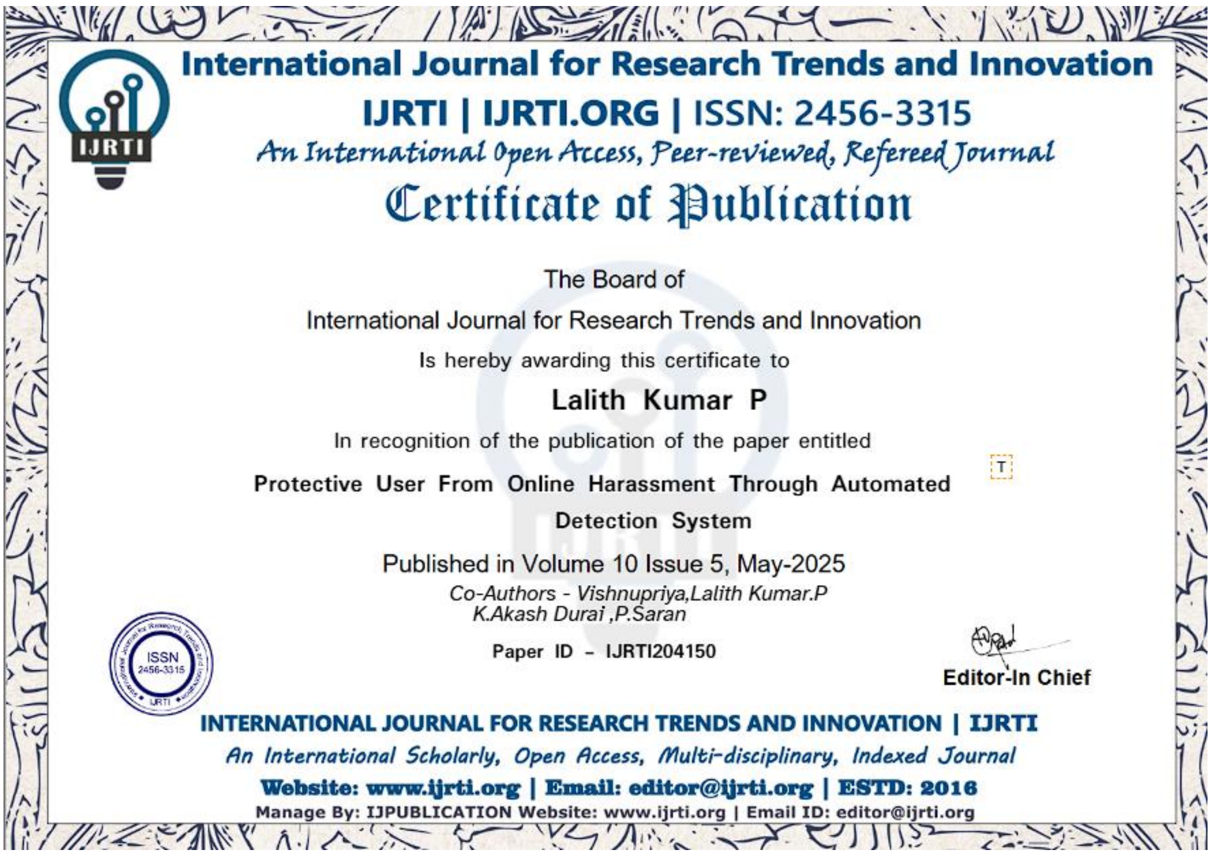
11. L. Cheng, J. Li, Y. N. Silva, D. L. Hall, and H. Liu, "XBully: Cyberbullying detection within a multi-modal context," in Proc. 12th ACM Int. Conf. Web Search Data Mining, Jan. 2019, pp. 339–347.
12. C. Van Hee et al., "Automatic Detection of Cyberbullying in Social Media Text." 2018.
13. M. Rezvan, S. Shekarpour, L. Balasuriya, K. Thirunarayan, V. Shalin, and A. Sheth, "A Quality Type-aware Annotated Corpus and Lexicon for Harassment Research," pp. 33–36, 2018.
14. A. H. Alduailej and M. B. Khan, "The challenge of cyberbullying and its automatic detection in Arabic text," 2017 Int. Conf. Comput. Appl. ICCA 2017, pp. 389–394, 2017.
15. A. Power, A. Keane, B. Nolan, and B. O. Neill, "A lexical database for public textual cyberbullying detection," Rev. Lenguas Para Fines Específicos, vol. 2, pp. 157–186, 2017.
16. M. Drahošová and P. Balco, "ScienceDirect The analysis of advantages and disadvantages of use of social media the analysis of advantages and disadvantages of use of social media in European Union in European Union," Procedia Comput. Sci., vol. 109, pp. 1005–1009, 2017.
17. R. Zhao, A. Zhou, and K. Mao, "Automatic detection of cyberbullying on social networks based on bullying features," in Proc. 17th Int. Conf. Distrib. Comput. Netw., Jan. 2016, pp. 1–6.
18. V. K. Singh, Q. Huang, and P. K. Atrey, "Cyberbullying detection using probabilistic socio-textual information fusion," in Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining (ASONAM), Aug. 2016, pp. 884–887.
19. A. Squicciarini, S. Rajtmajer, Y. Liu, and C. Griffin, "Identification and characterization of cyberbullying dynamics in an online social network," in Proc. IEEE/ACM Int. Conf. Adv. Social Netw. Anal. Mining, Aug. 2015, pp. 280–285.
20. P. Galán-García, J. G. De La Puerta, C. L. Gómez, I. Santos, and P. G. Bringas, "Supervised machine learning for the detection of troll profiles in Twitter social network: Application to a real case of cyberbullying," Logic J. IGPL, vol. 24, no. 1, pp. 42–53, 2015.

10.2. BOOK REFERENCES

1. "Deep Learning" by Ian Goodfellow - Comprehensive guide to fundamental deep learning concepts, including neural networks.
2. "Natural Language Processing in Action" by Lane, Howard, and Hapke - Practical insights into natural language processing techniques.
3. "Flask Web Development" by Miguel Grinberg - A hands-on guide for building web applications using Flask.
4. "MySQL Cookbook" by Paul DuBois - Practical solutions and examples for working with MySQL databases.
5. "Bootstrap 4 in Action" by Ray Villalobos - A practical guide for creating responsive web interfaces using Bootstrap.
6. "Recurrent Neural Networks in Python" by Daniel Slater - Focuses on understanding and implementing recurrent neural networks, relevant for LSTM.
7. "Convolutional Neural Networks in Python" by Anthony Williams - Practical insights into building and understanding CNNs.
8. "Natural Language Processing with Python" by Steven Bird, Ewan Klein, and Edward Loper - Covers NLP fundamentals using Python.
9. "Web Scraping with Python" by Ryan Mitchell - Useful for data collection from online sources.
10. "Machine Learning Yearning" by Andrew Ng - Practical insights and strategies for machine learning projects.

10.3. WEB REFERENCES

1. TensorFlow Documentation: https://www.tensorflow.org/
2. PyTorch Documentation: https://pytorch.org/
3. NLTK Documentation: https://www.nltk.org/
4. Flask Documentation: https://flask.palletsprojects.com/
5. MySQL Documentation: https://dev.mysql.com/doc/
6. Bootstrap Documentation:
   https://getbootstrap.com/docs/4.0/getting-started/introduction/
7. W3Schools - HTML Tutorial: https://www.w3schools.com/html/
8. W3Schools - CSS Tutorial: https://www.w3schools.com/css/
9. W3Schools - JavaScript Tutorial: https://www.w3schools.com/js/
10. Kaggle - Datasets for NLP: https://www.kaggle.com/datasets?tags=18524-nlp



International Journal for Research Trends and Innovation

IJRTI | IJRTI.ORG | ISSN: 2456-3315

An International Open Access, Peer-reviewed, Refereed Journal

Certificate of Publication

The Board of

International Journal for Research Trends and Innovation

Is hereby awarding this certificate to

Lalith Kumar P

In recognition of the publication of the paper entitled

Protective User From Online Harassment Through Automated Detection System

Published in Volume 10 Issue 5, May-2025

Co-Authors - Vishnupriya,Lalith Kumar.P
K.Akash Durai ,P.Saran

Paper ID - IJRTI204150

ISSN 2456-3315 IJRTI

[illegible]

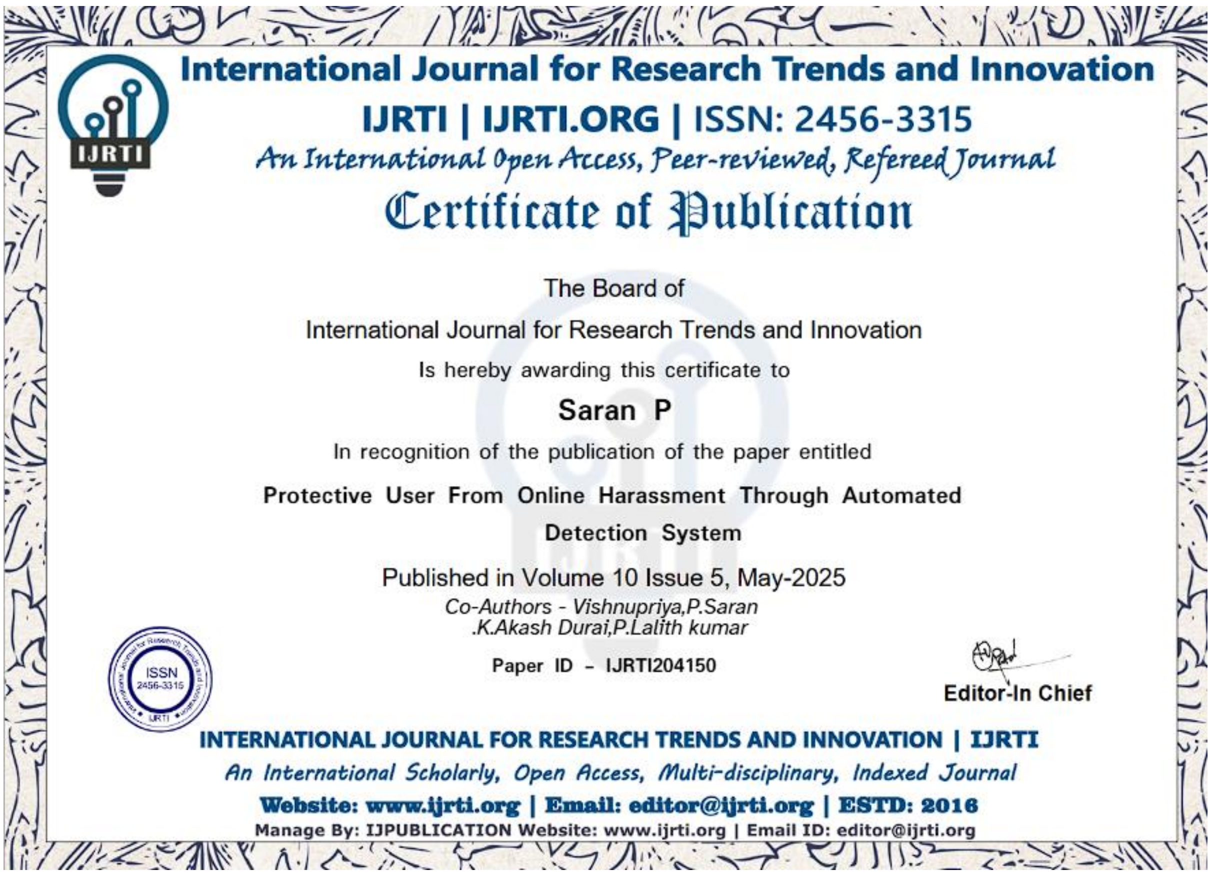
Editor In Chief

INTERNATIONAL JOURNAL FOR RESEARCH TRENDS AND INNOVATION | IJRTI

An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal

Website: www.ijrti.org | Email: editor@ijrti.org | ESTD: 2016

Manage By: IJPUBLICATION Website: www.ijrti.org | Email ID: editor@ijrti.org



International Journal for Research Trends and Innovation

IJRTI | IJRTI.ORG | ISSN: 2456-3315

An International Open Access, Peer-reviewed, Refereed Journal

Certificate of Publication

The Board of

International Journal for Research Trends and Innovation

Is hereby awarding this certificate to

Saran P

In recognition of the publication of the paper entitled

Protective User From Online Harassment Through Automated Detection System

Published in Volume 10 Issue 5, May-2025

Co-Authors - Vishnupriya,P.Saran .K.Akash Durai,P.Lalith kumar

ISSN 2456-3315 IJRTI

Paper ID - IJRTI204150

[illegible]

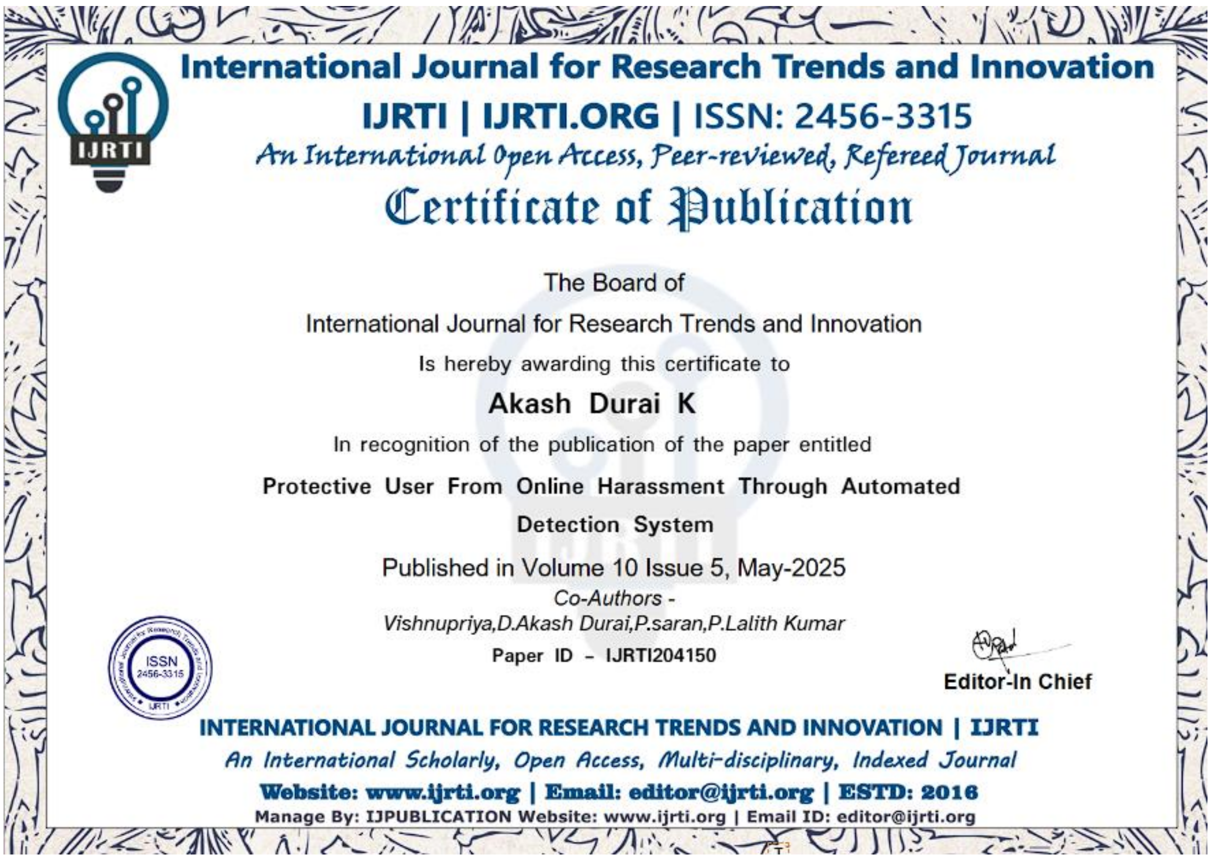
Editor-In Chief

INTERNATIONAL JOURNAL FOR RESEARCH TRENDS AND INNOVATION | IJRTI

An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal

Website: www.ijrti.org | Email: editor@ijrti.org | ESTD: 2016

Manage By: IJPUBLICATION Website: www.ijrti.org | Email ID: editor@ijrti.org



International Journal for Research Trends and Innovation

IJRTI | IJRTI.ORG | ISSN: 2456-3315

An International Open Access, Peer-reviewed, Refereed Journal

Certificate of Publication

The Board of

International Journal for Research Trends and Innovation

Is hereby awarding this certificate to

Akash Durai K

In recognition of the publication of the paper entitled

Protective User From Online Harassment Through Automated Detection System

Published in Volume 10 Issue 5, May-2025

Co-Authors -

Vishnupriya,D.Akash Durai,P.saran,P.Lalith Kumar

Paper ID - IJRTI204150

ISSN 2456-3315 IJRTI

[illegible]

Editor-In Chief

INTERNATIONAL JOURNAL FOR RESEARCH TRENDS AND INNOVATION | IJRTI

An International Scholarly, Open Access, Multi-disciplinary, Indexed Journal

Website: www.ijrti.org | Email: editor@ijrti.org | ESTD: 2016

Manage By: IJPUBLICATION Website: www.ijrti.org | Email ID: editor@ijrti.org