Dissertation on

"Towards Safer Social Media:
Real-Time Euphemism and Toxicity Detection in Online User Interactions"

Submitted in partial fulfillment of the requirements for the award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

UE21CS461A – Capstone Project Phase - 2

Submitted by:

Chethan V	**PES2UG21CS143**
Darshan GN	**PES2UG21CS151**
Harshini Murugan	**PES2UG21CS193**
Janav Shetty	**PES2UG21CS210**

Under the guidance of

Dr. Kamatchi Priya L
Associate Professor
PES University

June - Nov 2024

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

PES UNIVERSITY

(Established under Karnataka Act No. 16 of 2013)
Electronic City, Hosur Road, Bengaluru – 560 100, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

"Towards Safer Social Media: Real-Time Euphemism and Toxicity Detection in Online User Interactions"

is a bonafide work carried out by

Chethan V	**PES2UG21CS143**
Darshan GN	**PES2UG21CS151**
Harshini Murugan	**PES2UG21CS193**
Janav Shetty	**PES2UG21CS210**

In partial fulfillment for the completion of seventh-semester Capstone Project Phase - 2 (UE21CS461A) in the Program of Study -Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period June. 2024 – Nov. 2024. It is certified that all corrections/suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 7th-semester academic requirements in respect of project work.

Signature **Dr Kamatchi Priya L** Associate Professor	Signature Dr. Sandesh B J Chairperson	Signature Dr. B K Keshavan Dean of Faculty

External Viva

Name of the Examiners	**Signature with Date**
1. _____	_____
2. _____	_____

DECLARATION

We hereby declare that the Capstone Project Phase - 2 entitled Towards Safer Social Media: Real-Time Euphemism and Toxicity Detection in Online User Interactions has been carried out by us under the guidance of Dr Kamatchi Priya L and submitted in partial fulfillment of the course requirements for the award of the degree of Bachelor of Technology in Computer Science and Engineering of PES University, Bengaluru during the academic semester June – November 2024. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

PES2UG21CS143	**Chethan V**
PES2UG21CS151	**Darshan GN**
PES2UG21CS193	**Harshini Murugan**
PES2UG21CS210	**Janav Shetty**

ACKNOWLEDGEMENT

I would like to express my gratitude to **Dr. Kamatchi Priya L**, Associate Professor Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE21CS461A - Capstone Project Phase – 2.

I am grateful to all Capstone Project Coordinators, for organizing, managing, and helping with the entire process.

I take this opportunity to thank Dr. Sandesh B J, Professor & Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support I have received from the department. I would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

I am deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro-Chancellor, PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University, and Prof. Nagarjuna Sadineni, Pro-Vice Chancellor, PES University, for providing me with various opportunities and enlightenment every step of the way. Finally, Phase 2 of the project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

The prevalence of toxic behaviour in online conversations has surged with the rise in social media usage. A significant challenge in combating this issue is the use of euphemistic language, where users incorporate symbols or modifications (e.g., *, !, @) to evade detection by existing toxicity detection models. This study focuses on developing an advanced computational solution for identifying euphemistic language using state-of-the-art Natural Language Processing (NLP) techniques. The proposed method leverages Bidirectional Encoder Representations from Transformers (BERT) and its variants to enhance the efficacy of automated detection systems.

By training the model on diverse datasets and implementing a real-time detection framework, the approach aims to achieve superior performance compared to traditional models. The solution includes designing a user-friendly interface and evaluating the system using robust performance metrics. Potential challenges such as the need for continuous model adaptation to emerging linguistic patterns are addressed. This project aspires to foster respectful online communication while laying the groundwork for advanced toxicity detection methodologies.

TABLE OF CONTENTS

LIST OF FIGURES

CHAPTER 1

INTRODUCTION

In today's world, dealing with toxic language online is a big challenge. Harmful words on social media don't just hurt people's feelings but also make online discussions unpleasant for everyone. Given the massive number of posts and comments, it's nearly impossible for moderators to catch every harmful message by hand. Furthermore, even in today's Western society, the notion of insult varied through the ages when it was either expressed in blatant vulgarism, or buried in sarcasm, or included saying things characteristic of a certain culture. This combination of several forms of toxicity definitely makes it challenging to identify and classify the harmful speech accurately.

An area of particular concern is the phenomenon of 'using honeyed words', when newcomers with ill intentions conceal themselves in districts with pretty names and falsely sound benign. Such devices may often elude normal filters as they are clearly meant to be undisclosed.

In order to solve such problems, our project employs complex Natural Language Processing (NLP) models to trace less visible types of toxic speech. Our model in particular uses the Bidirectional Long Short-Term Memory (BILSTM) model to identify euphemisms.

This is a good application for BILSTM since it is capable of recognizing the intricate context in which people express a given word and therefore, helps us to detect the embedded toxic language. In addition to BILSTM, we are employing the Roberta model to classify the reposted toxic messages as we determine their general sentiment. Through the combination of BILSTM for the detection of euphemism.

CHAPTER 2

PROBLEM DEFINITION

This research project aims to identify and analyze the euphemistic toxic language in online communication. It uses techniques of natural language processing focusing primarily on RoBERTa model and BiLSTM. Our project aims to find the euphemisms that hide toxic or harmful content using which we provide automated ways to flag such language.

Euphemisms are a kind of language that is very difficult to detect. Euphemisms are used when one wishes to avoid the direct across and use moderate forms, alternative words, subtle expressions or symbols to get across the meaning of toxic language indirectly. The past approaches have not been able to flag and accurately show the problematic language. In this context, that is our research hypothesis, whereby it rests on the premise that the state of the art technologies can assist in enhancing the detection of the euphemisms and moderation of the online communication.

The model that stands out in tackling the problem of natural language text is the RoBERTa model. The BiLSTM model deals with the euphemisms detection and replacements because of its capabilities to read backward and forward and its flexibility with different spellings.

Our project will go a step further by sieving through audio data as well so as to better the analysis in terms of improved detection of toxic language through more channels. The inclusion of both makes it possible for the model to conduct a more detailed and comprehensive analysis.

CHAPTER 3

LITERATURE REVIEW

3.1 PAPER 1

[2] The objective of this paper is to understand the impact of adversarial attack on existing neural toxicity detectores solve the issues in such cases.

By understanding the problems in existing toxicity classifiers, the research aims to enhance the robustness of the model for more accurate toxicity detection.

The methodology used creates a lexicon of the toxic token to generate realistic adversarial attacks. These attacks involve character-level perturbation and injections and non-toxic distractors token to classify the toxicity . The models used are ELMo and BERT toxicity detection. It also uses contextual denoising autoencoder (CADE) as an approach to learn robustness in character level and contextual information to improve the resilience of toxicity classifiers against adversarial attacks.

Two approaches are used to solve the problem, first involves training on synthetically noised data to enhance the model ability, second is using CADE to learn robust representation that can classifier denoise the token and improve the classification. Results show that adversarial attacks can significantly reduce the detection in the present model , with some models experiencing a drop of 50% in recall performance . The metrics used to evaluate are AUC, F1 score and recall. The study focuses on developing defenses that can withstand sophisticated adversarial attacks and improve the overall reliability of a toxicity detection system.

3.2 PAPER 2

[3] The paper introduces a new method to automatically identify multiword euphemisms that are used by fringe groups and organizations in online forums. Euphemisms are ordinary words with hidden meanings that make content moderation challenging on social media.

The researchers of this paper have divided their work into three main stages. First the quality phrases are extracted using Autophrase which is a data driven phrase mining tool. This helps in creating a list of potential candidates which are high-quality phrases that can be used for further analysis. Then the euphemistic phrase candidates are preselected by calculating the cosine similarities of the word embeddings using word2vec.This helps in filtering out the noisy candidates and identify phrases that are semantically related to the target keywords, such as drugs with context to this paper. The last step is to rank the preselected euphemistic phrase candidates using SpanBERT.

SpanBERT was designed to predict the token spans in text which helps the researchers to analyze the long textual data more effectively. Thus by combining these three stages, we achieve the automated detection of euphemistic phrases. In addition to the above features, this study also allows the detection of both known and potentially new euphemisms used online.

This enhances the efficiency and accuracy of identifying euphemisms online. By using the paper's methods such as phrase mining and SpanBERT , we plan to improve the detection of toxic language phrases in online interactions. Overall, by referencing this paper we hope to improve the content moderation efforts on digital platforms.

3.3 PAPER 3

[4] This paper talks about “Does Context Really Matter?” as the heading of the paper itself states it. So mainly in this paper the authors have focussed on two questions (a) Does context affect human judgement, and (b) Does conditioning on context improves performance of toxicity detection systems? To investigate or see whether considering context improves the performance of toxicity detection systems. Two data sets were created in this paper : (i) CAT-SMALL and (ii) CAT-LARGE.

CAT-SMALL consisted of around 250 comments from Wikipedia Talk Pages, it was divided into two groups. One group had context, the other group did not have context. Each comment was analysed for toxicity by three annotators, and scores were rounded to toxic or non toxic.

CAT-LARGE consisted of around 20,000 comments, half of them were annotated with context and the rest half of them without context. This larger dataset allowed the authors for a more detailed analysis of the impact of context on the toxicity detection.

If we look at the models and approaches used in this paper, the researchers have explored various techniques. They used RNN language models and combined the consecutive comments before using RNN. They also included contextual features for sentiment classification. These methods were used to see how context influenced the toxic comments and how well the toxicity classifiers work.

The results of the analysis revealed that context had a statistically significant effect on the perceived toxicity of the comments. Approximately 5.2% of comments showed changes in toxicity labels when the context was considered. However the research found no evidence that context actually improved the performance of toxicity classifiers.

The study concluded that context can make comments seem more or less toxic, but it did not improve any system performance. So the researchers said we need bigger datasets to understand the context’s impact on toxicity analysis better.

In summary, the research paper provides important details about how context affects toxicity detection. It also shows the complexity or challenges in accurately judging the toxicity of online comments and using context to improve automated toxicity detection systems.

3.4 PAPER 4

[19] This paper talks about using Masked Language Model (MLM) for Euphemism Detection. It introduces the concept of self-supervision into the task of detection and classification of Euphemism. Self supervision of this task helps reduce the manual efforts required to define the euphemisms and also deals with problem of subjectivity that is introduced due to human judgement. The authors have divided their research to focus mainly on two tasks- euphemism detection and euphemism identification. Euphemism detection is the algorithm that takes a set of forbidden words as input and tries to generate the possible variations of the words, basically euphemisms. These set of words generated may signify the same meaning as the words in the forbidden set, and is called Candidate set. The candidate set of terms is used by the model to find new euphemisms and detect
forbidden words. The second task is to take a single word that is considered to be euphemistic and find out its meaning. Both the tasks are applied in succession, like a pipeline, by the moderators to discover new euphemisms and their meanings. The authors use Masked Language Models and other self-supervised techniques for accomplishing these tasks. In this paper, the approach used is to consider an input sentence and mask the term that is likely to be an euphemism. Based on the context of its occurrence and the rest of the sentence, the meaning of the masked term is predicted.

The classification of a euphemistic term is highly dependent on the context of the sentence as the word considered may or may not be used in a veiled sense. The paper uses Masked Language Model twice- once to filter out the masked sentences and then to detect hidden meanings in the masked terms.

For euphemism identification task, it uses two classifiers- coarse classifier, to filter out the sentences and multi-class classifier, to identify the meaning of the masked terms. The model proposed in this paper can detect previously unknown euphemisms, which makes it more effective for content moderation without manual efforts. It proposes a model with self-supervision but the model needs domain-specific training. The paper considers precision at k as the evaluation metric.

CHAPTER 4

PROJECT REQUIREMENTS SPECIFICATION

4.1 Functional Requirements

- Data Input:
    - The system should be able to accept text data in various formats like text and audio.
    - Preprocessing functionalities should be implemented to clean and prepare the text data (that is, tokenization, removal of punctuation, normalization).
- Toxicity Classification Model:
    - The system implements a pre-trained RoBERT model for the toxicity classification.
    - The selected RoBERTa model is fine-tuned on a labeled toxicity dataset.
- Euphemism Detection:
    - Identification of euphemisms within the processed input data.
- Model Output:
    - It is necessary that the system provides a clear output that indicates euphemism detection in the text data.
    - This can include highlighting euphemistic words/phrases or providing a classification label or flag.

4.2 External Interface Requirements

4.2.1 User Interfaces

- Required screen formats with GUI standards for styles: A clean and intuitive interface featuring input boxes for users to enter text or audio.
- Screen layout and standard function: The priority is implicity and usability of the layout, which includes proper labeling of the input fields and buttons. The results, if any, of the toxicity analysis are displayed prominently with appropriate metrics or visualizations.
- Relative timing of inputs and outputs: Inputs (text or audio) will be submitted by users. The system will then process the input and provide toxicity analysis results within a reasonable timeframe, typically within a few seconds to maintain user engagement
- Availability of some form of programmable function key: The application may include shortcut keys for common functions such as submitting the input or accessing help, enhancing user convenience and efficiency.
- Error messages: The interface will handle errors, providing informative messages to users in case of invalid inputs or technical issues.

4.2.2 Hardware Requirements

- Protocols:

  The application communicates with the user device via the HTTP/HTTPS protocols, while communication with external APIs- the RoBERTa and BiLSTM models-is carried out as per the respective protocols supported by those APIs, typically either HTTP or HTTPS.

- Hardware Dependencies:

  It requires minimal hardware, as all the processing will happen on the server, hence user devices only need to run a modern web browser to access the web version. The server should be reasonably powerful to allow many requests to be handled concurrently at peak time with an adequate response time back to users.

4.3 Software Requirements

- **Programming Languages**:
  - Python (for backend, machine learning models)
  - HTML, CSS, JavaScript (for frontend)
- **Web Framework**:
  - Flask (for web server)
  - Flask-SocketIO (for real-time message broadcasting
- **Machine Learning Libraries**:
  - PyTorch (for loading and running the RoBERTa toxicity detection model)
  - TensorFlow or PyTorch (for training and running the euphemism replacement model)
  - Transformers (for loading the RoBERTa model from Hugging Face)
  - Scikit-learn (for label encoding and model evaluation)
- **Data Processing Libraries**:
  - Pandas (for data handling)
  - Numpy (for numerical operations)
- **Frontend Libraries**:
  - Bootstrap (for responsive layout and styling)
  - Socket.IO (JavaScript client library for real-time communication)
- **Environment**:
  - Python Virtual Environment (to manage dependencies)

4.4 Non Functional Requirements

4.4.1 Performance Requirement

- Large language model interface.
- Large language model interaction with BERT model with a good average response time per query
- Quality attribute: speed and efficiency.

4.4.2 Scalability

- Data Volume Scalability: The system must scale to accommodate growing data and ensure optimal performance as the dataset expands.
- Quality Attribute: Scalability, performance

4.4.3 Reliability, Availability and fault tolerance

- The system should handle faults gracefully, ensuring continued functionality in the presence of unforeseen errors or disruption.
- Quality Attribute: Robustness, reliability

4.4.4 Resource Utilization

- Objective : The system should maintain optimal CPU and memory utilization to prevent resource saturation and ensure stability
- Quality Attribute: Resource efficiency, Stability, database query response time, database queries should be executed within a time limit to maintain efficient data retrieval and processing, Database performance, efficiency.

CHAPTER 5

SYSTEM DESIGN

5.1 Architecture Diagrams

5.1.1 High Level System Design

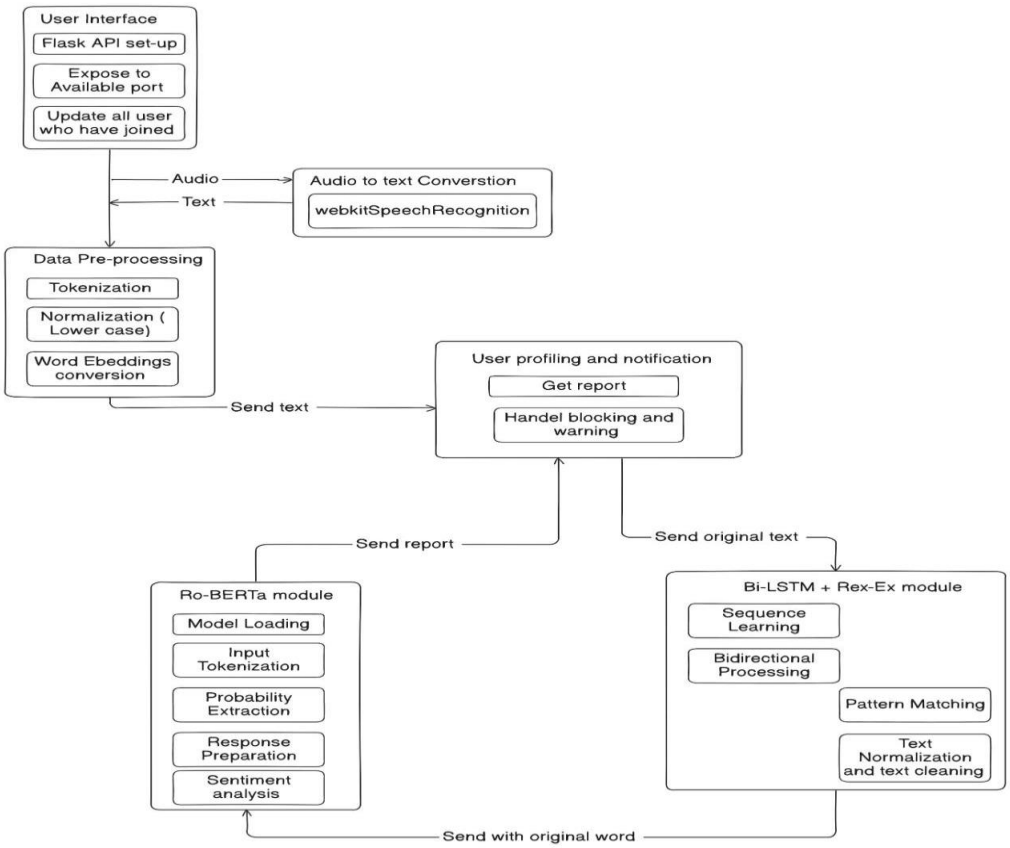

Fig.5.1.1 High Level System Design

5.1.2 Master Class Diagram

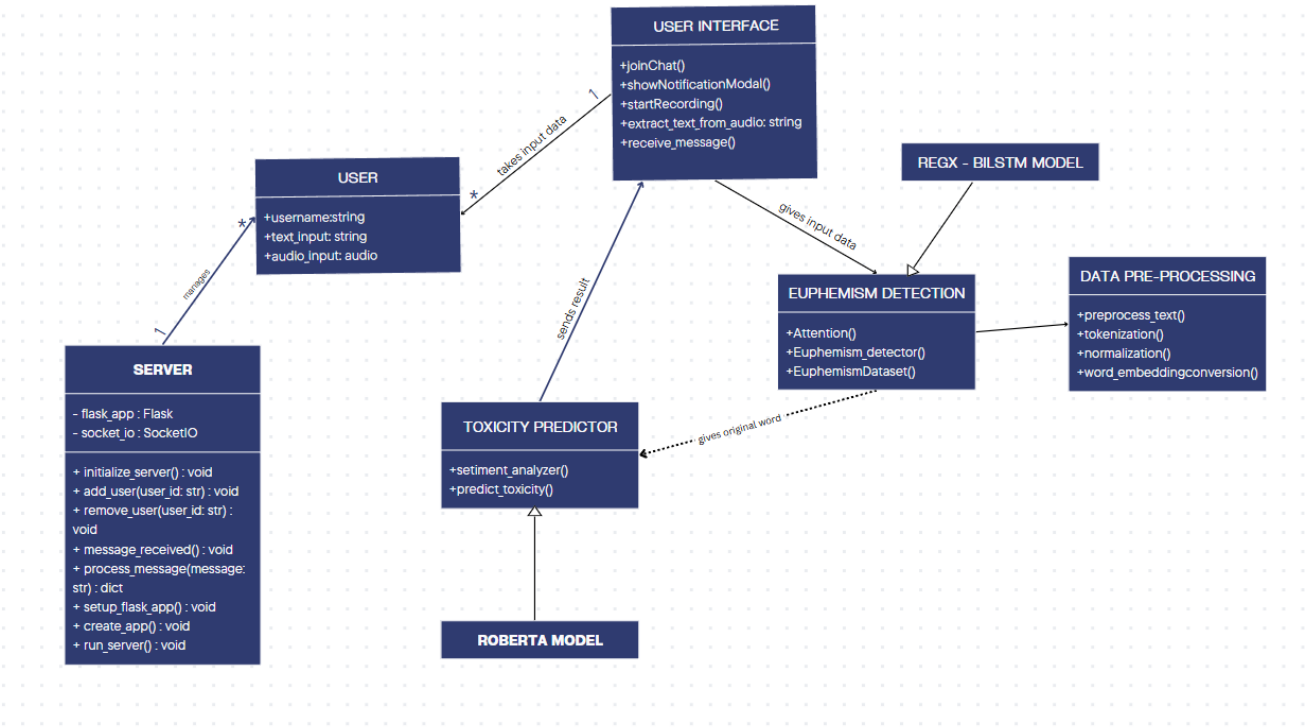

Fig.5.1.2 Master Class Diagram

5.1.3 Architecture Diagram

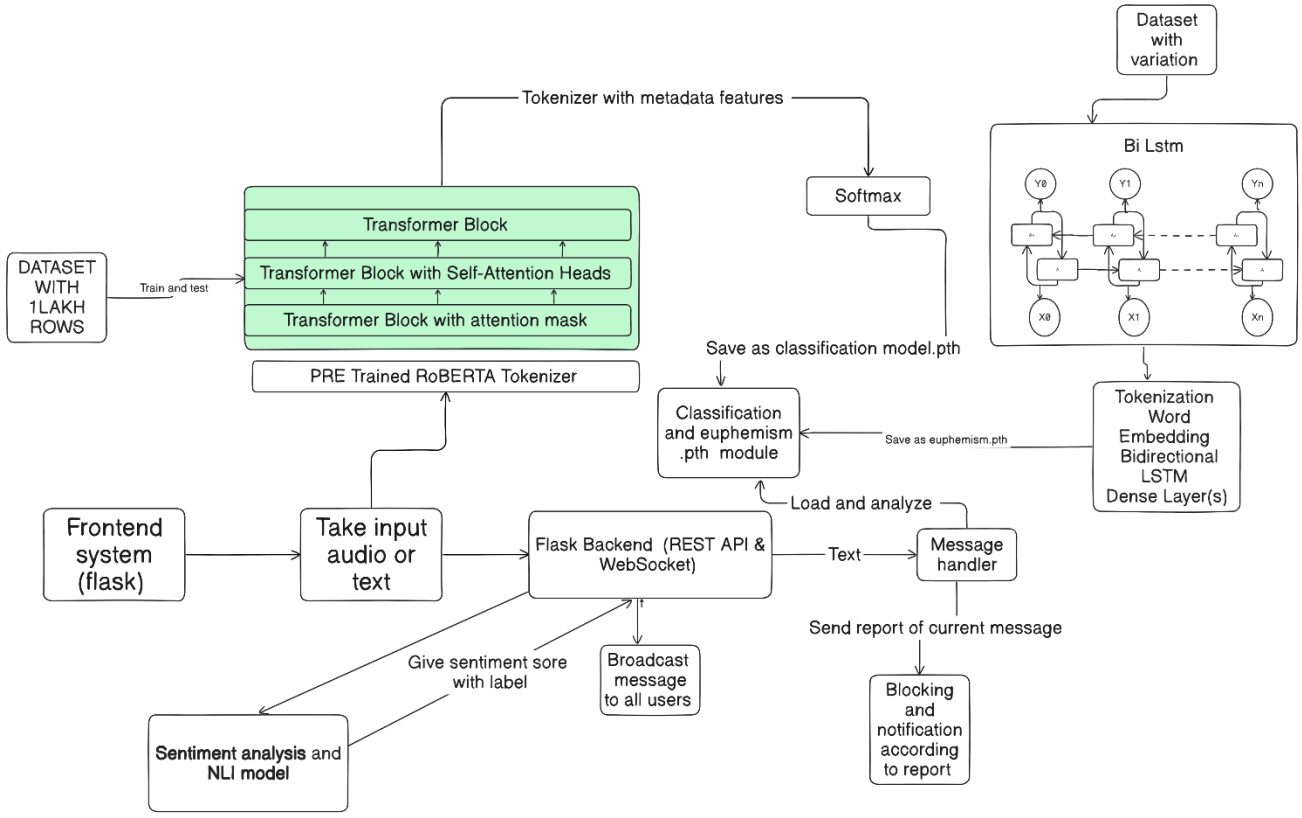

Fig.5.1.3 Architectural Diagram

5.1.4 External Interfaces

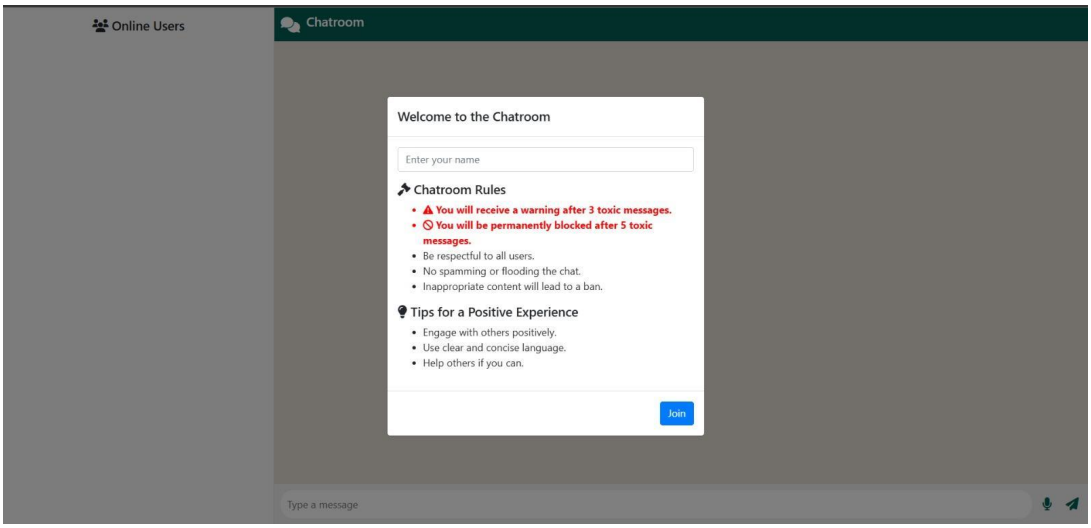

Fig.5.1.4.1 Chatroom rules

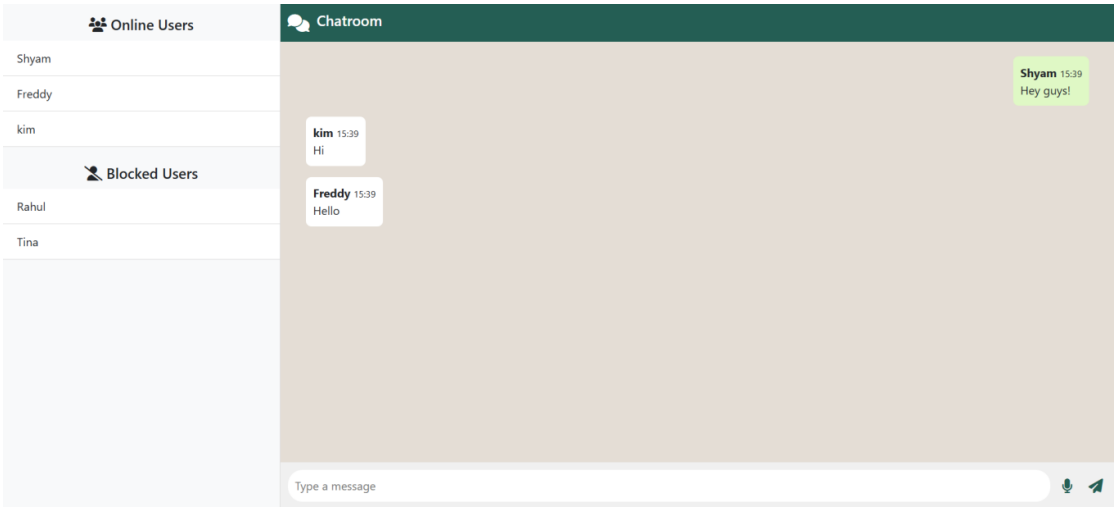

Fig.5.1.4.2 Chatroom interface

5.2 Design Considerations

5.2.1 Design Goals

- **Speed:** The system needs to respond fast to keep the chat conversations smooth. This way the users won't have to wait or face any delays when chatting in real-time.
- **Scalability:** The system should be made so that new features can be added without much trouble. It should also handle more users as the number increases without any problems.
- **Accuracy:** We use good models like RoBERTa to detect toxic language correctly. The system can find subtle insults and understand the context to identify harmful content accurately.
- **Portability:** The platform should work on different operating systems and web browsers. This makes sure that all users have the same experience, no matter what device they use.

5.2.2 Architecture Choice

Our project uses a client-server setup. The client is a web based interface that anyone can access in a web browser. The server does routine work in the back end, for example, scanning texts for the presence of linguistic aggressiveness or managing users' accounts.

- **Client:** We finalized the web page's design in HTML, CSS with bootstrap integration and JavaScript. There is also a chat room on the platform which allows users to send and receive messages conveniently.
- **Server:** Flask and Flask-socketio are used for the backend of the system. It connects users to one another, sends messages to everyone's screen, and includes the feature of measures against aggression.

- **Toxicity Detection:** For us at the server, there is the roBERTa model developed by Hugging Face Transformers. Such tools allow us to implement effective automatic content moderation quickly and accurately.

Pros

- **Scalability:** The server is able to interact with a big number of users at once. This makes it easier for us to include additional users as more bear clients join us.
- **Centralized Processing**: All the toxicity detection modules are provided by the server. This makes it more user friendly as the most sophisticated techniques can be applied as all users get the same treatment.
- **Security:** Since all the activities are performed by the server, this allows a very high level of security measures such as data encryption and user authentication. This ensures confidentiality of the data.
- **Ease of Maintenance**: The website as it appears on the server could be updated and repaired. Thus, the management of the system is made less complex.

Cons

- **Latency:** Sending messages to the server can cause delays. This might make the chat feel slow when you're talking.
- **Server Dependency:** Our chat app needs the server to work. If the server is down or slow, the chat won’t work properly.
- **Increased Complexity:** Having both the client and server makes the project harder. We need to manage both parts carefully.

5.2.3 General Constraints, Assumptions and Dependencies

- **Computational Resources:** The server needs enough power to check messages quickly, especially when many people are using the chat. So we use GPUs to help make this faster.
- **Data Availability:** The system relies on the availability of a comprehensive dataset for training the toxicity detection models including variations and euphemisms of toxic words.
- **External Libraries:** We use Flask, Flask-SocketIO, and Hugging Face Transformers. We make sure they work well together and update them when needed.
- **Evaluation Metrics:** We check how well the system works by looking at precision, recall, F1-score, and how fast it processes messages. This helps us ensure everything is working right.
- **User Privacy:** We follow privacy rules when handling user data. We keep personal information safe and prevent others from accessing it without permission.

5.3 Design Details

5.3.1 Novelty:-

- **Multi-Modal Input:** Our chat system permits the users to send text messages and records voice messages as well. When a person sends a voice message, there is a conversion to the text form which is done automatically. The output of the voice is then passed through the algorithm that looks for the presence of any offensive words.
- **Euphemism Handling:** It is very good in terms of evading detection due to the presence of toxic words that are not straightforward. It can also intercept diversions, possibly directed towards insults, such as use of characters in place of letters or intentional misspellings. Therefore, we are able to uncover all types of hateful remarks.

- **Contextual Understanding:** Models such as RoBERTa are used to interpret the content that the people are angry and saying. This is important for the accuracy of the system in determining whether a message is hateful or not which reduces the chances of errors.
- **Real-Time Feedback**: When the message is being written by a user and the system determines that it is offensive, the user will receive a pop up message at once when the system detects such a case. This serves to enhance the safe and respectful chat for everyone involved.

5.3.2 Interoperability

- **Integration with Web Platforms:** Created for easy incorporation into current web browsers, making it possible to run on different platforms with no need for extra installations of application software.
- **API Compatibility:** Server-side toxicity detection can be made available through APIs thus making it possible to integrate with other applications or services requiring toxicity analysis.
- **Modular Design:** The modular components of the system allow for easy integration into diverse content management systems and social networks enhancing its efficiency in different organizations.

5.3.3 Performance

- **Response Time:** Aimed at performing message processing and message analysis within a low latency and ensuring that there are no delays while communication takes place for real-time messages with high efficiency in aggressive behaviour assessment.
- **Scalability:** Designed to contain increased loads in the future by adding more servers horizontally or vertically by enhancing the capability of the servers, enabling them to maintain the same level of performance as the user volume increases.

- **Resource Optimization:** Optimal use of computable resources, including usage of GPU acceleration to reduce final models to a minimum efficiency level without wasting resources.

5.3.4 Maintainability

- **Modularity:** Such design enables the substitution of any parts including but not limited to the toxicity detection module and user management without having to sacrifice the seamlessness of the entire system.
- **Comprehensive Documentation**: The code is accompanied by comprehensive documentation including but not limited to system structure diagrams, a description of the components and their functions, user manuals, thus making system maintenance less challenging.
- **Code Quality:** There is a high standard of coding discipline that is invariably followed which translates into writing tidy, structured as well as well documented code thereby minimizing future interventions and potential bugs.

5.3.5 Portability

- **Platform Independence:** As the application is built on the foundation of the languages CSS, HTML, Javascript, and Python, it is platform independent and works perfectly on any operating system which includes Windows, macOS, and Linux among many others.
- **Browser Compatibility:** Allows more compatibility with the frequent web browsers which include Google Chrome, Mozilla Firefox, Safari, Microsoft Edge so that end users do not have a different experience depending on the browser.
- **Deployment Flexibility:** Server may be deployed on different architectures, may be cloud-based such as AWS or Azure, or may be on-premise giving an opportunity to deploy that matches the business needs.

5.3.6 Reusability

- **Component Reusability:** The prototype has components that can be reused in other applications, such as the toxicity detection module and the user management system, with the possibility of adding more features without too much alteration in the primary design.
- **Protocol Standards:** Employs a standard communication protocol (for example - websocket using Socket.IO), which means that the system is capable of interacting with other applications, or services which are using the protocol in the same or compatible form.
- **Adaptable Architecture:** The client-server structure is resizable and can be altered for different usage scenarios, e.g. when additional platforms are integrated or when further scalability is needed in order to support a broader user base.

CHAPTER 6

PROPOSED METHODOLOGY

The main strategy is based on integrating and utilizing several modern-aided Natural Language Processing (which is referred to as NLP) models for the comprehensive detection and classification of not so polite language. The whole pipeline consists of two models- for detection and remediation of euphemisms and for classification of toxicity.

For this application, we utilize Bidirectional Long Short-Term Memory (BiLSTM) as it is capable of handling characters for structural work. We train our model with self built dataset, for the purpose of detecting any kind of “euphemism” in a particular message that is meant to be masked for the ease of bypassing the traditional methods of detection of toxicity focused on-euphemisms. Then, we apply sentiment analysis for the output of the model of euphemism. After the detection and modification of euphemisms and the analysis of the five sentiments, the replacement message is subjected to classification of toxicity.

An intuitive Flask interface has been developed and presented as a complete package, in order for the users to input either written text or to voice text and receive results of the analysis within seconds. There is also an interface incorporate warnings, flagging and blocking the user when toxic characteristics persist. This simple method of implementing the whole process makes it possible to detect and classify toxic content very quickly and at the same time making it convenient for user moderation.

CHAPTER 7

IMPLEMENTATION AND PSEUDOCODE

In this section, we discuss the aimed implementation of our model and its unique features including assessing the sentiment of utterances, which we deem as the most important aspect, our algorithm is designed to recognize and address both the subtle euphemizations and also the aggressive ones.

7.1 Message Processing and Preprocessing

In the first stage, the user's input message is accepted and then undergoes preprocessing. Preprocessing involves a few simple procedures such as noise removal, segmenting the text into lexical units (tokenization), and formatting tree structure of the text to meet requirements of our models for toxicity detection and euphemism detection.

Pseudocode:

FUNCTION preprocess_message(message):

- Convert message to lowercase
- Remove any unnecessary punctuation and whitespace
- Tokenize the message into individual words or tokens
- Return the cleaned and tokenized message

7.2 Detecting Euphemisms

Once the content undergoes preprocessing, the system proceeds to evaluate the message in order to search for euphemisms or variations of even the most toxic words. Rather than mapping the euphemism directly to its corresponding toxic word, the model takes a more sophisticated approach in detecting and interpreting euphemistic language.

The system combines the content of the tokenized message, a pre-existing language model like RoBERTa, and a collection of euphemisms they have created themselves. This means that the model can automatically detect when euphemisms are used, that is, those words or phrases that may not be exact toxic terms, but are likely to be euphemisms. In this way, the system detects indirect toxicity without changing the core message.

If the model sees an opportunity to use a euphemism, it turns to the surrounding words and the previous knowledge from the training data to decide whether there is any toxic intent behind the apathetic-sounding word. This step in the process guarantees accuracy in flagging euphemisms without overly simplistic and often incorrect word-to-word mapping.

Pseudocode:

```
FUNCTION detect_euphemisms (tokens, euphemism_model, context_window):
    FOR each token IN tokens:
        - Extract surrounding context using context_window
        - Use euphemism_model to analyze token within context
        IF euphemism_model classifies token AS euphemistic:
            - Mark token AS potential euphemism
RETURN tokens (marked with potential euphemisms)
```

Explanation:

- tokens: The parsed output of the message in words for analysis purposes.
- euphemism_model: A language model that has been trained and further notes subtle to moderate abuse being applied in a different context.
- context_window: An additional offset given on both sides of each token to allow more words for the model for information. This helps the model in determining whether a term is used in a euphemistic manner.

The model patterns do not work directly as some substitutive patterns which replace the euphemism with a more aggressive word. The contrary is true though in that the words which are otherwise neutral are used in a toss to replace an aggressive connotation but only in certain contexts and based on the way that specific word was seen in practice. This allows for a more nuanced and complex understanding of euphemisms, and circumvents the need to overly depend on direct correspondences.

7.3 Sentiment Analysis

In order to avoid any potentially miscommunication, the message, which has been preprocessed, is sent for sentiment evaluation. In the case of a positive or a neutral sentiment, the message can be skipped to avoid unnecessary toxic content analysis.

Pseudocode:

FUNCTION perform_sentiment_analysis(message):

- Use a pre-trained sentiment analysis model to analyze the message
- IF sentiment is POSITIVE or NEUTRAL:
  - RETURN "non-toxic"

ELSE:

- RETURN "potentially toxic"

Explanation:

- This stage assists in the elimination of extreme toxicity of language that is used unnecessarily but for positive or neutral purpose in the non-toxic messages.

7.4 Predicting Toxicity

The central aim of the system is in evaluating the toxicity of the message. Now, after the euphemism detection and the desperate attempts to filter the sentiment of the transformed message, the model makes predictions about the message's toxicity. The model distinguishes abuse types and gives them some probability such as "insult" ,"threat" or "obscenity."

Pseudocode:

FUNCTION predict_toxicity(modified_message):

- Encode the modified_message using the tokenizer for the chosen model
- Pass the encoded message through the toxicity model
- Obtain probability scores for each toxicity category
- IF any category probability exceeds the threshold:
    - Flag the message as toxic
    - RETURN list of toxic categories detected

ELSE:

- RETURN "non-toxic"

Explanation:

- **modified_message:** The message from which all euphemisms have been edited out.
- The model applies the probability to these thresholds and classifies the messages according to the relevant toxicity category.

7.5 Generating and Logging the Report

When a message is classified as toxic, the system issues a complaint report on the type of toxicity which is detected, any euphemisms which were found, and the sentiment in general. This report is kept for the sake of moderation or techniques and investigation. Also, if a user maintains sending bulk messages which are found to be toxic, they may be included at the blocked users list.

Pseudocode:

FUNCTION generate_report(message, toxicity_categories,euphemeisms , sentiment):

- Create a report object containing
  - Original message
  - List of detected toxicity categories
  - Any identified euphemisms
  - Sentiment result

  RETURN report

7.6 Updating Blocked Users List

Any user who sends a number of messages containing toxicity which is above the threshold definitely goes into the blocked users list. This is to be sure that users who are on the practice of sending the messages that are harmful, cannot have any further interactions.

Pseudocode:

FUNCTION update_blocked_users(user, report_count):

- IF report_count >= blocking_threshold:
- Add user to blocked_users list
- Notify the user that they have been blocked
- RETURN blocked_users list

Summary of Implementation

The above pseudocode describes the key components of our toxicity detection system:

- **Message Preprocessing:** The stage where the message is subjected to cleaning and tokenization.
- **Euphemism Detection:** Replacing euphemisms with the base toxic words.
- **Sentiment Analysis**: Filtering non-toxic messages based on sentiment.
- **Toxicity Prediction:** Using a deep learning model to categorize toxic language.
- **Report Generation:** Creating detailed logs for toxic messages.
- **User Blocking:** Managing users based on repeated toxic behavior.

This structured approach ensures that our model not only detects direct toxic language but also identifies indirect toxicity through euphemisms. By combining these methods, we enhance the model's robustness and contribute to more accurate, nuanced toxicity detection.

CHAPTER 8

RESULTS AND DISCUSSION

8.1 User Interaction and Feedback:

- The interface of the chat application is built in a way that allows the users to familiarise themselves with any rules, especially the dos and don’ts, at the earliest chance possible. As shown in figure one showing the initial login screen, users are made aware of chatroom rules that govern the behavior including the restriction of hate messages, so that they engage in respectful communication.

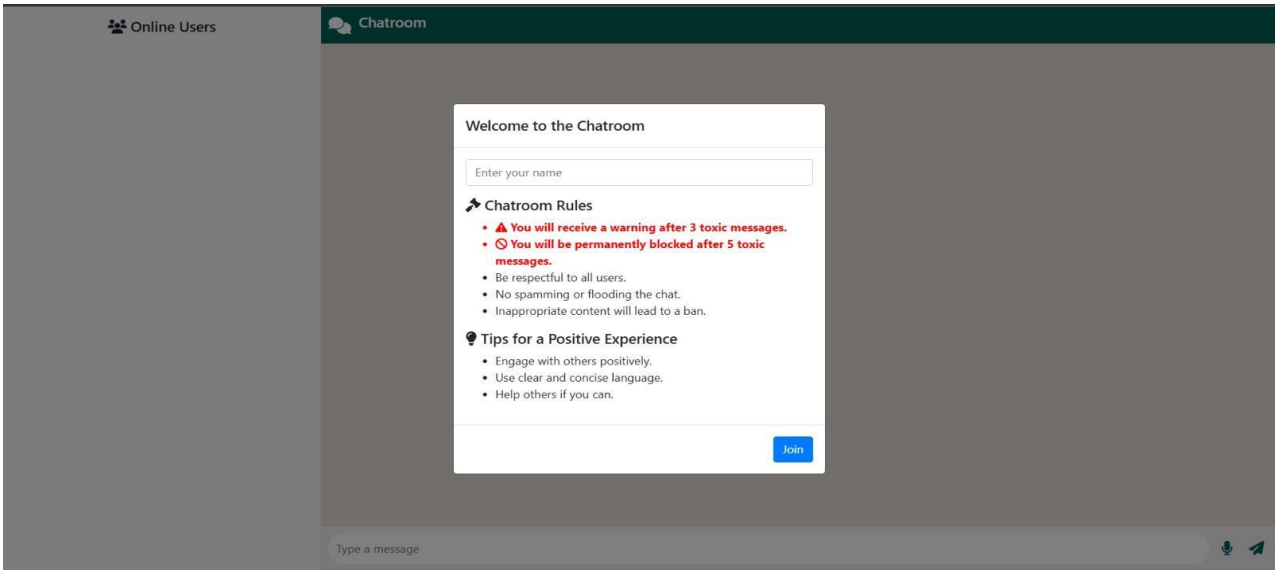

Fig.8.1.1 Chatroom rules

- When a person joins the chat, they are given a notification and online members can see the name of this person.

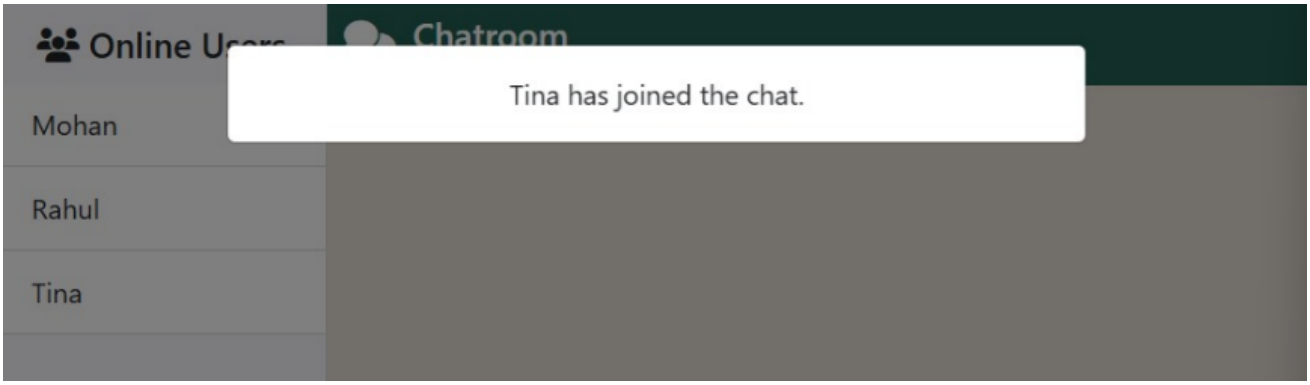

Fig.8.1.2 Notification of user joining

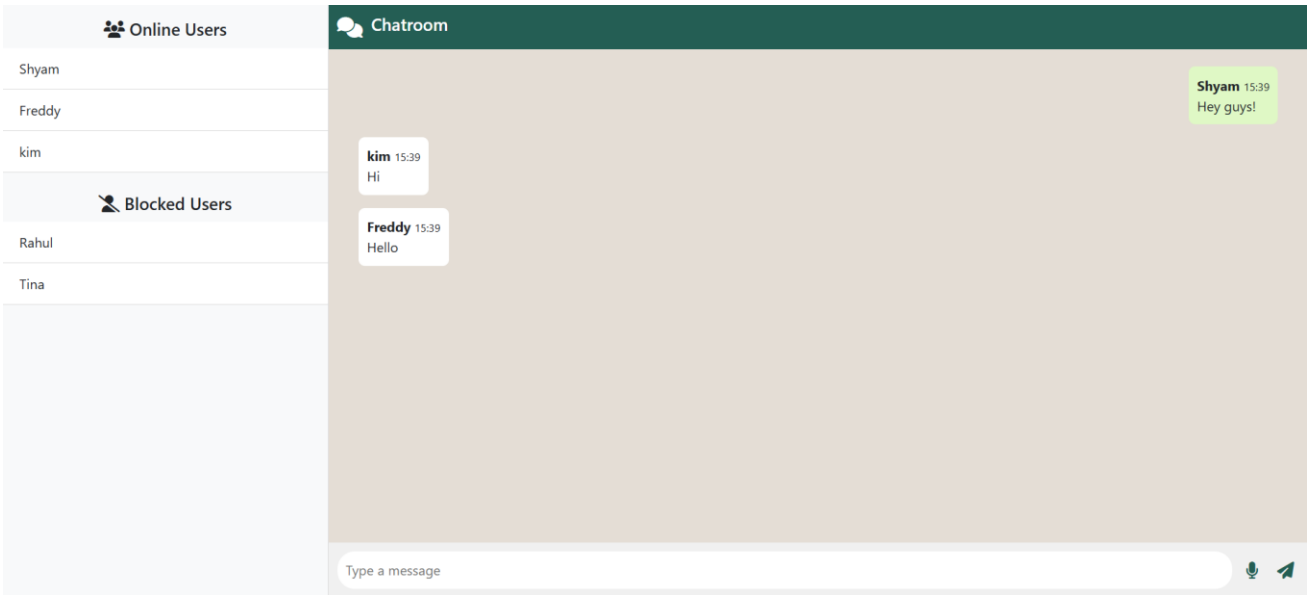

Fig.8.1.3 Chatroom interface

8.2 Toxicity Detection and Warnings:

- By employing a fine-tuned RoBERTa model, system classifies the incoming messages in multiple dimensions such as ‘threat’, ‘identity hate’, ‘insult’ etc. When the user sends a toxic message, the user is warned.

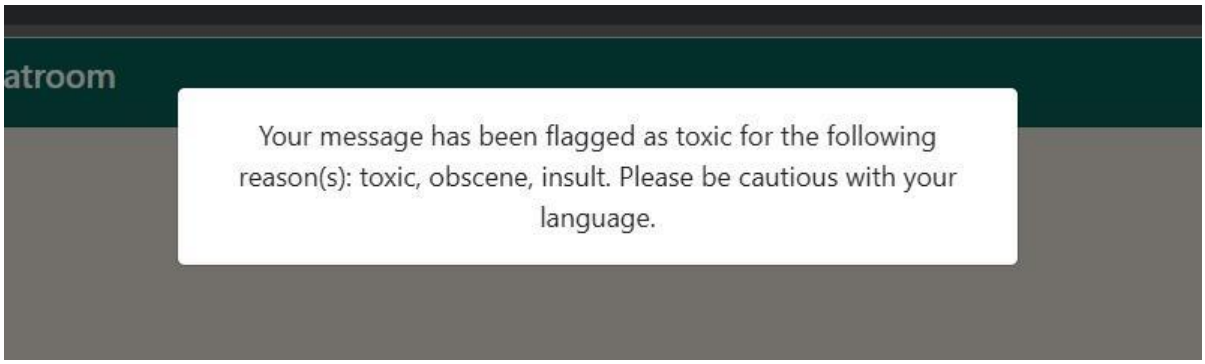

Fig.8.2.1 Warning notification

- After three warnings, the user is warned about being blocked and after the fifth toxic message, the user is completely blocked. This automated system of warning and banning is of great importance in any chatroom since it prevents the overuse of toxicity in conversations.

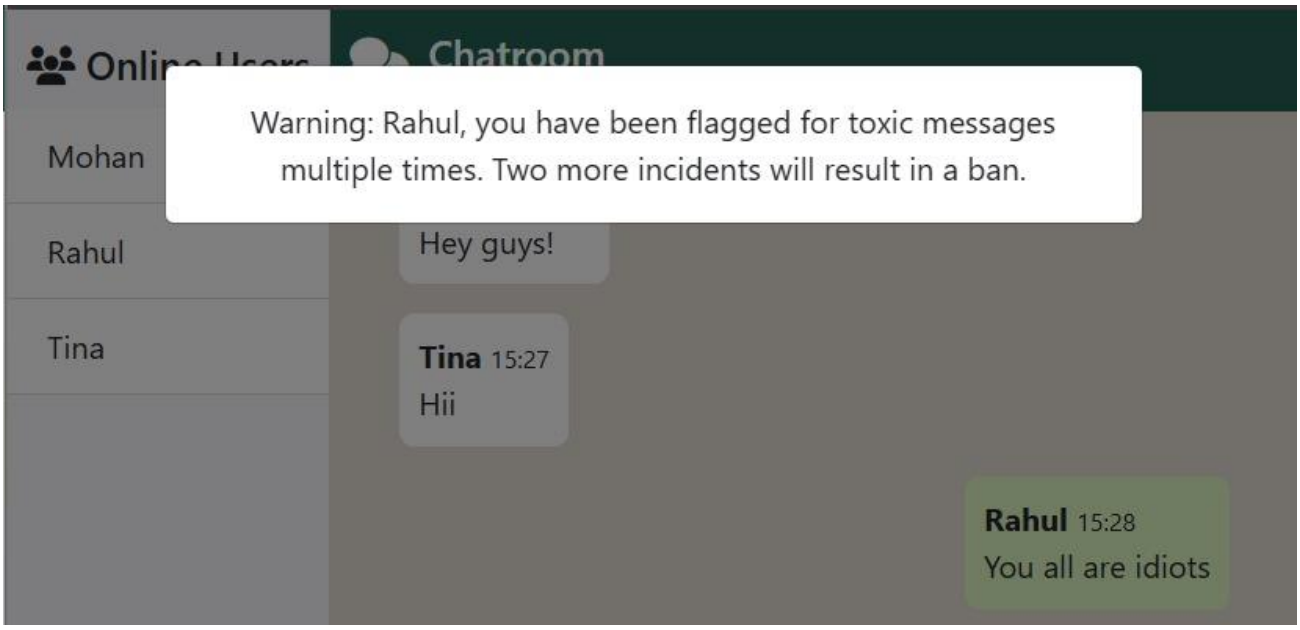

Fig.8.2.2 Notification of warning after 3 toxic messages

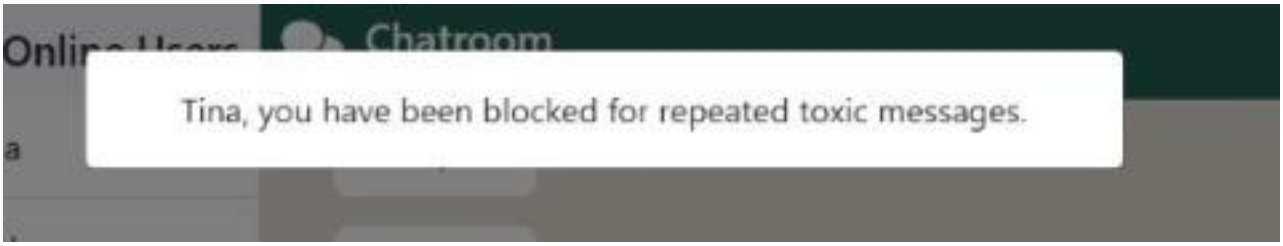

Fig.8.2.3 Notification of blocking

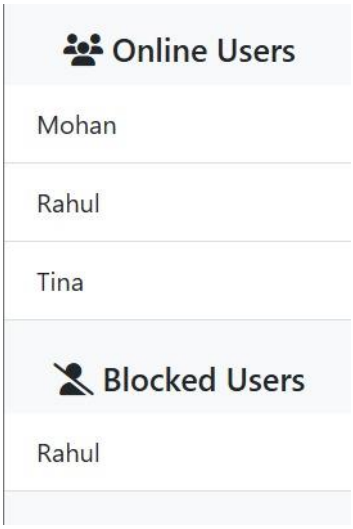

Fig.8.2.4 List of blocked users

- In the experiments conducted, the RoBERTa model proved effective in the classification of offensive and abusive language which helped in the containment of toxicity in the chat.

8.3 Euphemism Replacement Mechanism:

To increase toxicity detection even further, the system also included a BiLSTM model which detect euphemisms to harmful speech, and replaces them with plain language. This is an important and healthy addition to the system, as it allows the system to detect toxic content that is very slight and might have easily gone undetected. For instance, ‘offensive’ slangs are not used but rather their straight forms are used, which assists the roberta model in understanding the intent of the speaker. This replacement serves a dual purpose in that it helps in the detection of toxicity and also helps the users learn the language that is not acceptable aiding in self-regulation.

input message	detected euphemism	replace with	sentiment analysis	model classification	Note
You are not an idiot	None	None	Positive	Not toxic	The sentence negates the insult, making the overall sentiment positive, and no toxicity is detected.
You're not stupid, but you're acting like one.	None	None	Neutral	Not toxic	The sentence contains an implied insult but the overall sentiment is neutral and not directly offensive.
You're a great person, but you still act like a fool.	None	None	Neutral Toxic	Toxic	While the phrase "a great person" is positive, the latter part ("act like a fool") is toxic, triggering the classification as toxic.
What the f****k man	f****k	fuck	Negative	Toxic	"f****k" is identified as a euphemism and replaced with "fuck," triggering a toxic classification due to its offensive nature.
You're a great person, but you're acting like a b****rd	b****rd	bastard	Negative	Toxic	"b****rd" is replaced with "bastard," and the negative sentiment and insult result in toxicity classification.

Fig.8.3 Table of results

8.4 Effects of Sentiment Analysis:

The use of sentiment analysis on DistilBERT creates an additional layer of context in relation to predictions of toxicity. Sentiment analysis categorizes the emotional content of a text that translates into positive, neutral, or negative sentiments before the process of toxicity checks is implemented. In most cases, messages that carry a positive or a neutral sentiment do not carry out toxicity checks hence avoiding unnecessary flags on harmless messages. This has mitigated the cases of false alarms and enhanced the efficiency of the process of determining properly the toxicity levels of messages which made it possible to control harassment while still allowing good interactions.

8.5 Models of Comparison

The deployed models show high precision on various tasks, confirming the appropriateness of each model chosen.

- Of the models tested for toxicity classification, the RoBERTa model registered the highest accuracy of 0.98, better than BERT (0.92) and LSTM (0.96), suggesting that the design of RoBERTa is optimal when toxic language is stratified to many classes.

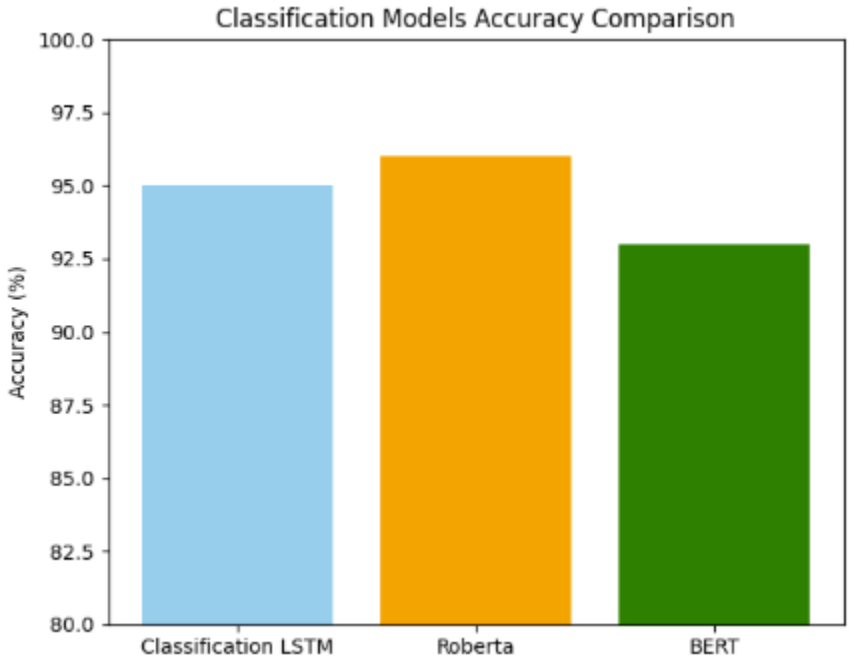

Fig.8.5.1 Toxicity classification models

- For euphemism detection, the optimized BiLSTM model achieved an accuracy of 0.99, which is higher than the original ByT5 model (0.84) and a regular LSTM model (0.89) as well.

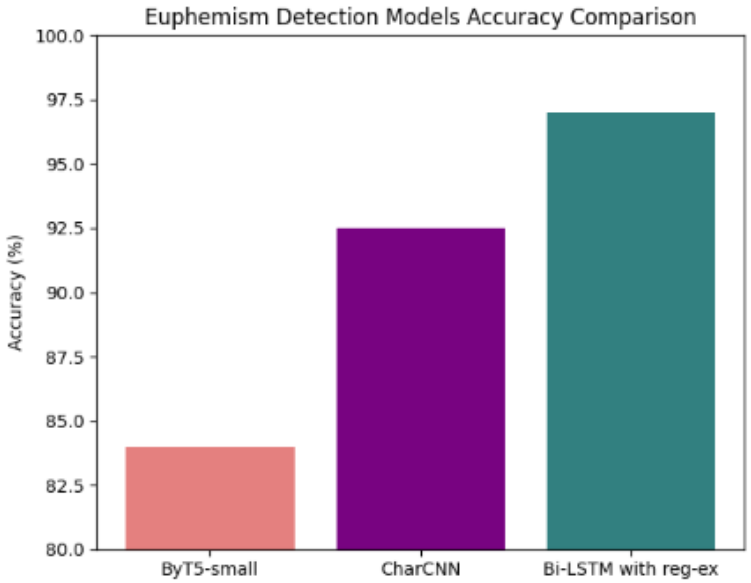

Fig.8.5.2 Euphemism detection models

- Such results support the use of the BiLSTM model in dealing with
- euphemisms, due to its two–layered structure enabling it to recognize and analyze complex patterns of language usage.

CHAPTER 9

CONCLUSION AND FUTURE WORK

The project combines different models to ensure detection of toxicity and the definitive use of euphemisms, through the use of attention techniques, a BILSTM model for the euphemism replacement, sentiment analysis using a DistilBERT model and a multi label classification of toxicity using a fine-tuned RoBERTa. As it stands, the combination of these models seeks to ensure there is a proper coverage of harmful language detection and handling techniques without steering from the overall viewpoints communicated across the interactions.

The RoBERTa model achieves high accuracy in classifying toxic categories such as threat, identity hate assumptions amongst others whereas the BiLSTM model identifies and replaces of euphemism which enhances the reliability of the detection greatly. The model's functional comprehension of language is also further enriched by the integration of sentiment analysis into DistilBERT which comprehends the essence of the messages and therefore provides more context to toxicity detections. Sampled this way, the layered model architecture is a good reminder of how deep learning can play a critical part in understanding language in its many different aspects and shades, paving the way for further developments in the nature of engaging moderation for content and detection of language in its responsible context.

Future work may include broadening the scope of this project by employing cross language and cross dialect moderation techniques of toxicity, to effectively address censorship of diverse languages. A more robust model of euphemism detection may possibly provide the user with options, thus enabling the user to tailor his or her comment and provide the ability to do so in real time, allowing for greater flexibility in expression.

REFERENCES

[1] A. Vaswani et al., “Attention Is All You Need,” arXiv, Jun. 12, 2017.https://arxiv.org/abs/1706.03762

[2] Kurita, A. Belova, and A. Anastasopoulos, “Towards Robust Toxic Content Classification,” arXiv:1912.06872 [cs], Dec. 2019, Available: https://arxiv.org/abs/1912.06872

[3] W. Zhu and S. Bhat, “Euphemistic Phrase Detection by Masked Language Model,” arXiv.org, 2021. https://arxiv.org/abs/2109.04666 (accessed Nov. 26, 2024).

[4] J. Pavlopoulos, J. Sorensen, L. Dixon, N. Thain, and I. Androutsopoulos, “Toxicity Detection: Does Context Really Matter?,” arXiv.org, Jun. 01, 2020. https://arxiv.org/abs/2006.00998

[5] “RoBERTa,” huggingface.co. https://huggingface.co/docs/transformers/en/model_doc/roberta

[6] K. Team, “Keras documentation: RoBERTa,” Keras.io, 2024. https://keras.io/api/keras_nlp/models/roberta/ (accessed Nov. 26, 2024).

[7] “PyTorch,” www.pytorch.org. https://pytorch.org/hub/pytorch_fairseq_roberta/

[8] N. Chakrabarty, “A Machine Learning Approach to Comment Toxicity Classification,” arXiv.org, 2019. https://arxiv.org/abs/1903.06765 (accessed Nov. 26, 2024).

[9] A. Sheth, V. L. Shalin, and U. Kursuncu, “Defining and detecting toxicity on social media: context and knowledge are key,” Neurocomputing, vol. 490, Dec. 2021, doi: https://doi.org/10.1016/j.neucom.2021.11.095.

[10] M. Taleb, A. Hamza, M. Zouitni, Nabil Burmani, Said Lafkiar, and Noureddine En-Nahnahi, “Detection of toxicity in social media based on Natural Language Processing methods,” May 2022, doi: https://doi.org/10.1109/iscv54655.2022.9806096.

[11] M. M. Chowdhury, M. U. Ahmed, and M. Alam, "A deep learning approach for early detection of social media toxic behavior," *IEEE Access*, vol. 9, pp. 119378–119388, 2021. [Online]. Available: https://doi.org/10.1109/ACCESS.2021.3085306.

[12] A. Alsharef, K. Aggarwal, Sonia, D. Koundal, H. Alyami, and D. Ameyed, "An Automated Toxicity Classification on Social Media Using LSTM and Word Embedding," Computational Intelligence and Neuroscience, vol. 2022, p. e8467349, Feb. 2022, doi: https://doi.org/10.1155/2022/8467349

[13] B. van Aken, J. Risch, R. Krestel, and A. Löser, "Challenges for Toxic Comment Classification: An In-Depth Error Analysis," arXiv:1809.07572 [cs], Sep. 2018, Available: https://arxiv.org/abs/1809.07572

[14] A. Garlapati, N. Malisetty, and G. Narayanan, "Classification of Toxicity in Comments using NLP and LSTM," 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), Mar. 2022, doi: https://doi.org/10.1109/icaccs54159.2022.9785067.

[15] K. S. Ashok, K. A. Ashok, and S. M. B. Naseem, "A Neuro-NLP Induced Deep Learning Model Developed Towards Comment Based Toxicity Prediction," IEEE Xplore, Dec. 01, 2022. https://ieeexplore.ieee.org/document/10039597

[16] A. Bonetti, M. Martínez-Sober, J. C. Torres, J. M. Vega, S. Pellerin, and J. Vila-Francés, "Comparison between Machine Learning and Deep Learning Approaches for the Detection of Toxic Comments on Social Networks," Applied Sciences, vol. 13, no. 10, p. 6038, Jan. 2023, doi: https://doi.org/10.3390/app13106038

[17] J. Liu, Mahesh Kumar Nandwana, Janne Pylkkönen, Hannes Heikinheimo, and M. McGuire, "Enhancing Multilingual Voice Toxicity Detection with Speech-Text Alignment," Interspeech 2022, pp. 4298–4302, Sep. 2024, doi: https://doi.org/10.21437/interspeech.2024-1228

[18] R. Jadhav, N. Agarwal, Srushti Shevate, Chinmayee Sawakare, Piyush Parakh, and Snehankit Khandare, "Cyber Bullying and Toxicity Detection Using Machine Learning," Jun. 2023, doi: https://doi.org/10.1109/icpcsn58827.2023.00017

[19] W. Zhu et al., "Self-Supervised Euphemism Detection and Identification for Content Moderation," IEEE Xplore, May 01, 2021. https://ieeexplore.ieee.org/stamp/stamp.jsp?tp=&arnumber=9519422

Appendix A: Definitions, Acronyms and Abbreviations

BERT: Bidirectional Encoder Representations from Transformers, a pre-trained natural language processing model developed by Google.

Euphemism: A mild or indirect word or expression used in place of a more direct or harsh one, often to avoid offense or discomfort.

Context Analysis: The process of examining the surrounding circumstances, environment, or background information to understand the meaning and implications of a situation or communication.

LSTM: Long Short-Term Memory, a type of recurrent neural network (RNN) architecture designed to model sequential data and maintain long-term dependencies.

Toxicity: The quality of being harmful, poisonous, or detrimental, especially in the context of social media, referring to content that is offensive, abusive, or harmful to others.

Towards Safer Social Media:
Real-Time Euphemism and Toxicity Detection in Online User Interactions

Dr. Kamatchi Priya L
Associate Professor
Department of computer science
PES University
Bangalore, 560010 India
priyal@pes.edu

Chethan V
Department of computer science
PES University
Bangalore, 560010 India
pes2202101345@pesu.pes.edu

Harshini Murugan
Department of computer science
PES University
Bangalore, 560010 India
pes2202101330@pesu.pes.edu

Darshan G N
Department of computer science
PES University
Bangalore, 560010 India
pes2202101353@pesu.pes.edu

Janav Shetty
Department of computer science
PES University
Bangalore, 560010 India
pes2202101320@pesu.pes.edu

Abstract **-** With the rise in social media platforms and usage, incidents of toxic language have significantly increased, creating a hostile online environment. Toxic language detection is crucial not only for social media but also across various digital platforms, such as forums, gaming communities, and workplace communication tools, where offensive language can harm user experience. Existing models for toxicity detection face pitfalls in handling euphemisms and contextual variations, often resulting in false positives and reduced adaptability. Our proposed model addresses these challenges by integrating a pre-trained model, character-level embeddings , and a layer with attention, specially designed to detect nuanced toxic language and euphemisms more efficiently. This model demonstrated enhanced accuracy and robustness, reducing false positives and improving detection rate. Comparative analysis with existing models highlights its superior adaptability, making it a versatile tool for moderating toxic language across diverse online spaces.

I. INTRODUCTION

The growing prevalence of toxic language across digital platforms, from social media to various other online platforms.

Effective moderation of toxic language is essential to fostering positive online interactions and safeguarding user experience. Conducting this research is vital, as traditional detection systems have struggled to keep pace with evolving language patterns and widespread euphemisms in toxic messaging, which often allows harmful content to bypass filters.

Exiting toxicity detection models, such as BERT-based classifiers and logistic regression approaches, often face substantial limitations when confronted with nuanced toxic languages.

For instance, BERT-based toxicity classifiers, though advanced, have shown vulnerabilities to euphemisms and subtle variations, which contribute to a high false-positive rate, reaching approximately 20% in specific cases [2]. Additionally, these models exhibit a notable drop in recall, about 30-35%, when dealing with adversarial attacks like character perturbations or token distractions, which reduce their effectiveness in filtering disguised toxic content[2]. On the other hand, simpler methods like logistic regression with TF-IDF, while computationally less demanding, lack contextual awareness necessary for nuanced toxicity detection and have demonstrated accuracy rates as low as 60% on datasets with complex toxic expression [3]. These limitations highlight the need for more sophisticated models that can interpret and detect evolving toxic language patterns efficiently.

To address these gaps, we propose a model that builds on **RoBERTa** with a **BiLSTM** layer enhanced by character-

level embeddings and attention mechanisms. Our models are designed to incorporate both semantic and character-level insights, aiming to improve both the precision and recall of toxic language detection, especially in identifying euphemisms and contextually ambiguous terms. The proposed model demonstrates superior performance over BERT and SpanBERT, which excel in contextual understanding but struggle with subtle euphemistic expressions. Additionally, it outperforms classification LSTM, which, while accurate, suffers from overfitting issues. Our approach effectively captures the nuances of toxic language, making robust solutions for moderating harmful content across diverse digital platforms.

	Model	Advantages
1	Roberta	String contextual understanding
2	BERT	Effective in capturing general context
3	Classification LSTM	High accuracy but prone to overfitting

Table 1 : Model of comparison for classification

	Model	Advantages
1	By T5-small	Good baseline performance
2	Char CNN	Effective for detecting euphemism
3	Bi-LSTM with reg-ex	High accuracy in euphemism detection

` Table 2 : Model of comparison for euphemism

The research progresses through systematic stages, including dataset creation, model architecture implementation, and model evaluation, We have curated a comprehensive dataset sourced from a publicly available at kaggle challenge, comprising over 100,000 rows of diverse textual data. This dataset includes a wide range of examples illustrating various forms of toxic language

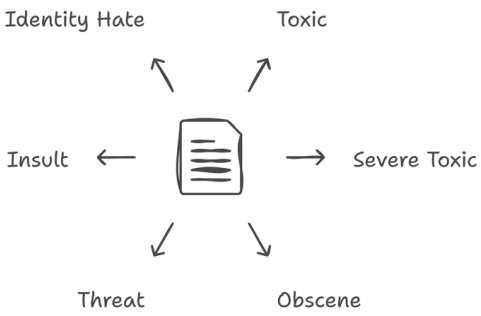

Fig 1 : Forms of toxicity

For euphemism detection, we specifically curated the dataset using common offensive words and their variations, ensuring a rich representation of euphemistic expressions. Following the dataset preparation, we developed and fine-tuned our Roberta-BiLSTM architecture. We conducted comparative evaluations with baseline models like BERT and Span BERT, measuring accuracy, false-positive rates, and interpretability. This iterative process enabled us to systematically assess and refine our model, culminating in a robust tool capable of detecting various forms of toxic language effectively across varied digital platforms.

II. Literature review

.

In designing our model for robust toxicity detection, we draw upon key research insights from several foundational works. Each of these studies has influenced distinct aspects of our model architecture, particularly with respect to attention mechanisms, contextual understanding, and the handling of euphemistic language.

The paper **[1]** introduces transformer architecture, a foundational model that replaces recurrent layers with a self-attention mechanism to capture long-range dependencies efficiently. Our model leverages the attention layer that enhances the contextual encoding of toxic language. By using attention mechanisms, our model can efficiently weigh the importance of words within a sentence, which is particularly valuable for detecting nuanced toxic expressions that may be context-dependent.

The study **[2]** directly addresses the challenges of adversarial and noisy inputs in toxicity detection. This paper's insights into handling character-level perturbations and noisy inputs in toxicity detection. This paper's insights into handling character-level

perturbations and adversarial language inform our model's robustness. We implement character-level embeddings and utilize preprocessing strategies to enhance our model's resilience to subtle textual modifications, building on the methodologies outlined in this work.

Finally, the study **[4]** investigates the significance of context in toxicity detection. Based on their findings, we integrate a BiLSTM layer to enhance contextual understanding in our model. This layer, combined with attention mechanisms, allows our model to differentiate between toxic and non-toxic contexts more effectively, particularly when words alone may not indicate toxicity.

III . Proposed Model

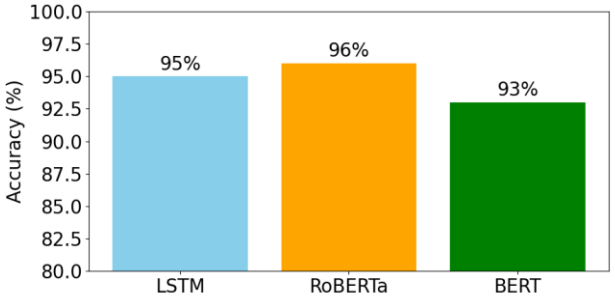

Fig 2 : Classification comparison graph

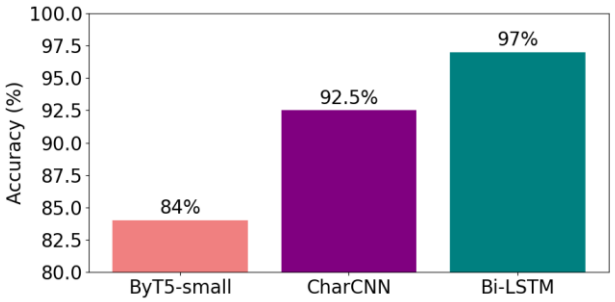

Fig 3 : Euphemism comparison graph

References

[1] A. Vaswani et al., "Attention Is All You Need," arXiv, Jun. 12, 2017.https://arxiv.org/abs/1706.03762

[2] Kurita, A. Belova, and A. Anastasopoulos, "Towards Robust Toxic Content Classification," arXiv:1912.06872 [cs], Dec. 2019, Available: https://arxiv.org/abs/1912.06872

[3] W. Zhu and S. Bhat, "Euphemistic Phrase Detection by Masked Language Model," arXiv.org, 2021. https://arxiv.org/abs/2109.04666 (accessed Nov. 26, 2024).

[4] J. Pavlopoulos, J. Sorensen, L. Dixon, N. Thain, and I. Androutsopoulos, "Toxicity Detection: Does Context Really Matter?," arXiv.org, Jun. 01, 2020. https://arxiv.org/abs/2006.00998

[5] "RoBERTa," huggingface.co. https://huggingface.co/docs/transformers/en/model_doc/roberta

[6] K. Team, "Keras documentation: RoBERTa," Keras.io, 2024. https://keras.io/api/keras_nlp/models/roberta/ (accessed Nov. 26, 2024).

[7] "PyTorch," www.pytorch.org. https://pytorch.org/hub/pytorch_fairseq_roberta/

[8] N. Chakrabarty, "A Machine Learning Approach to Comment Toxicity Classification," arXiv.org, 2019. https://arxiv.org/abs/1903.06765 (accessed Nov. 26, 2024).

[9] A. Sheth, V. L. Shalin, and U. Kursuncu, "Defining and detecting toxicity on social media: context and knowledge are key," Neurocomputing, vol. 490, Dec. 2021, doi: https://doi.org/10.1016/j.neucom.2021.11.095.

[10] M. Taleb, A. Hamza, M. Zouitni, Nabil Burmani, Said Lafkiar, and Noureddine En-Nahnahi, "Detection of toxicity in social media based on Natural Language Processing methods," May 2022, doi: https://doi.org/10.1109/iscv54655.2022.9806096.

[11] M. M. Chowdhury, M. U. Ahmed, and M. Alam, "A deep learning approach for early detection of social media toxic behavior," IEEE Access, vol. 9, pp. 119378–119388, 2021. [Online]. Available: https://doi.org/10.1109/ACCESS.2021.3085306.

[12] A. Alsharef, K. Aggarwal, Sonia, D. Koundal, H. Alyami, and D. Ameyed, "An Automated Toxicity Classification on Social Media Using LSTM and Word Embedding," Computational Intelligence and Neuroscience, vol. 2022, p. e8467349, Feb. 2022, doi: https://doi.org/10.1155/2022/8467349.

[13] B. van Aken, J. Risch, R. Krestel, and A. Löser, "Challenges for Toxic Comment Classification: An In-Depth Error Analysis," arXiv:1809.07572 [cs], Sep. 2018, Available: https://arxiv.org/abs/1809.07572

[14] A. Garlapati, N. Malisetty, and G. Narayanan, "Classification of Toxicity in Comments using NLP and LSTM," 2022 8th International Conference on Advanced Computing and Communication Systems (ICACCS), Mar. 2022, doi: https://doi.org/10.1109/icaccs54159.2022.9785067.

[15] K. S. Ashok, K. A. Ashok, and S. M. B. Naseem, "A Neuro-NLP Induced Deep Learning Model Developed Towards Comment Based Toxicity Prediction," IEEE Xplore, Dec. 01, 2022. https://ieeexplore.ieee.org/document/10039597

[16] A. Bonetti, M. Martínez-Sober, J. C. Torres, J. M. Vega, S. Pellerin, and J. Vila-Francés, "Comparison between Machine Learning and Deep Learning Approaches for the Detection of Toxic Comments on Social Networks," Applied Sciences, vol. 13, no. 10, p. 6038, Jan. 2023, doi: https://doi.org/10.3390/app13106038

[17] J. Liu, Mahesh Kumar Nandwana, Janne Pylkkönen, Hannes Heikinheimo, and M. McGuire, "Enhancing Multilingual Voice Toxicity Detection with Speech-Text Alignment," Interspeech 2022, pp. 4298–4302, Sep. 2024, doi: https://doi.org/10.21437/interspeech.2024-1228