BIAS AND DISCRIMINATION CHECK

A major project report submitted in partial fulfilment of the requirement for the award of a degree

Bachelor of Technology

in

Computer Science & Engineering

Submitted by

Suryansh (211384), Shivansh (211381), Pranjal (211182)

Under the guidance & supervision of

Mr. Arvind Kumar, Assistant Professor (Grade-II)

Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology, Waknaghat, Solan - 173234 (India)

May 2025

CANDIDATE'S DECLARATION

We hereby declare that the work presented in this major project report entitled **'Bias and Discrimination Check'**, submitted in partial fulfilment of the requirements for the award of the degree of **Bachelor of Technology in Computer Science & Engineering,** in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, is an authentic record of our own work carried out during the period from July 2024 to May 2025 under the supervision of **Dr. Arvind Kumar**(Assistant Professor Grade - II, Department of Computer Science & Engineering and Information Technology).

We further declare that the matter embodied in this report has not been submitted for the award of any other degree or diploma at any other university or institution.

(Student Signature)	**(Student Signature)**	**(Student Signature)**
Name: Suryansh Sharma	Name: Shivansh Katoch	Name: Pranjal Sharma
Roll No.: 211384	Roll No.: 211381	Roll No.: 211182
Date:	Date:	Date:

This is to certify that the above statement made by the candidates is true to the best of my knowledge.

(Supervisor Signature)

Supervisor Name: Dr. Arvind Kumar

Date: Designation: Assistant Professor (Grade - II)

Place: Waknaghat Department: CSE & IT

SUPERVISOR'S CERTIFICATE

This is to certify that the major project report entitled **'Bias and Discrimination Check,'** submitted in partial fulfilment of the requirements for the award of the degree **of Bachelor of Technology in Computer Science & Engineering,** in the Department of Computer Science & Engineering and Information Technology, Jaypee University of Information Technology, Waknaghat, is a Bonafide project work carried out under my supervision during the period from July 2024 to May 2025.

I have personally supervised the research work and confirm that it meets the standards required for submission. The project work has been conducted in accordance with ethical guidelines, and the matter embodied in the report has not been submitted elsewhere for the award of any other degree or diploma.

(Supervisor Signature)

Supervisor Name: Dr. Arvind Kumar

Date: Designation: Assistant Professor (Grade - II)

Place: Waknaghat Department: CSE & IT

ACKNOWLEDGMENT

With immense gratitude, we extend our heartfelt thanks to our esteemed Supervisor, Dr. Arvind Kumar, Assistant Professor (Grade-II) in the Department of CSE at Jaypee University of Information Technology, Wakhnaghat. We are deeply indebted to him for his tireless support, patient mentorship, constructive criticism, and unwavering encouragement. His keen interest in our work has truly been a driving force behind the project's completion.

We would also like to express our gratitude to Mr. Arvind Kumar from the Department of CSE for his valuable assistance, which significantly contributed to the successful conclusion of our project.

Our sincere thanks extend to all individuals, whether directly or indirectly, who contributed to this project's triumph. We acknowledge the support of the entire staff, both teaching and non-teaching, whose timely assistance and facilitation greatly aided our endeavour.

In closing, we wish to recognise and appreciate the enduring support and patience of our parents. Their unwavering encouragement has been a source of strength throughout this journey.

With gratitude,

Suryansh Sharma (211384)
Shivansh Katoch (211381)
Pranjal Sharma (211182)

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

ABSTRACT

The proposed project addresses the issue of discrimination and bias in online environments through the application of advanced natural language processing (NLP) techniques. As online platforms like Reddit dominate the agenda of public discussions, online environments are more exposed to hateful, toxic, or biased content. The project aims to develop a system that can accurately categorize user-uploaded text as hateful or non-hateful and make digital environments safer and more welcoming.

The project begins by utilizing a Kaggle dataset with annotated comments that are hate speech and toxicity related. A pre-trained BERT model is fine-tuned with hyperparameter search in an effort to maximize classification accuracy. Unlike traditional machine learning models, BERT offers contextual information regarding language, and hence it is best able to identify implicit and subtle forms of discrimination.

The second phase is obtaining real-time data through web scraping of comments on Reddit, testing the model's generalisability to real and diverse user input. The system processes this data to determine whether it is hate speech or not, facilitating proactive moderation.

The solution proposed by the authors includes extensive measurement criteria (accuracy, precision, recall, F1-score) and aims to minimize both false negatives and false positives. The project also covers ethical aspects of auto-moderating content and potential biases in prediction by the models.

Through practice, theory, and live testing, this project is contributing to the ongoing fight against hate speech online with a reproducible model for future application in content moderation, public policy, and AI ethics.

CHAPTER 1: INTRODUCTION

1.1 INTRODUCTION

With the rapid growth of digital communication platforms, the internet has become a global space for people to express themselves freely. However, this vast freedom of speech has also led to the emergence of negative behaviours, including the proliferation of hate speech, discrimination, and bias in various online environments. Hate speech refers to content that incites violence, hatred, or discrimination against individuals or groups based on their race, gender, religion, nationality, sexual orientation, or other protected attributes. This type of speech can cause significant harm to individuals, communities, and society by promoting division, violence, and social unrest.

Hate speech on the internet has become a growing concern for governments, corporations, and civil society groups, as it undermines public trust and social harmony. Social media platforms, online forums, and comment sections have become battlegrounds for hateful discourse, which has raised questions about how to combat this behaviour effectively. Given the scale at which content is generated on platforms like Facebook, Twitter, Reddit, and others, it is impossible for human moderators to manually review every post. This makes automated hate speech detection systems essential.

Automated systems that detect and classify hate speech can play a crucial role in identifying harmful content in real-time and helping platforms remove or flag such content for review. The development of such systems has become an active area of research within the field of Natural Language Processing (NLP) and machine learning (ML). The purpose of this project is to create a robust hate speech detection system using state-of-the-art deep learning models, particularly transformer-based models like BERT, which have shown significant promise in text classification tasks.

In this context, this project focuses on building a hate speech detection system using transformer models, evaluating their performance, and proposing methods for improving the classification of hate speech and biased content. Through this, we aim to contribute towards the development of more accurate and efficient systems for automated content moderation.

1.2 PROBLEM STATEMENT

The main challenge addressed by this project is the detection and classification of hate speech and discriminatory content in online text. As online platforms continue to grow and users contribute massive amounts of data every second, the need for automated systems capable of identifying harmful speech has become more pressing.

Hate speech manifests in various forms, including racial slurs, sexist comments, homophobic language, religious intolerance, and other derogatory expressions that harm marginalized communities. These forms of speech can often be subtle, context-dependent, and even disguised in non-obvious ways, making detection difficult. Moreover, the ambiguity and subjectivity of hate speech, as well as the rapid evolution of language online (e.g., memes, slang, coded language), further complicate the task of automated classification. Additionally, models trained on standard datasets may struggle to generalize to real-world applications due to domain differences, slang variations, and multilingual content.

Existing approaches to hate speech detection rely heavily on manually curated rule-based systems or traditional machine learning methods that may not be capable of capturing the complex nature of human language. Rule-based systems are limited because they cannot adapt to new linguistic patterns or subtle forms of hate speech that emerge. Traditional machine learning methods, while more flexible, still rely on handcrafted features, which can be time-consuming and error-prone.

The central problem tackled by this project is the development of an effective, scalable, and adaptable automated system for the detection of hate speech and discrimination. The goal is to build a model capable of accurately identifying a wide range of harmful content across different languages, social contexts, and media platforms.

1.3OBJECTIVES

The objectives of this project are clearly defined to guide the development of the hate speech detection system and to ensure that the solution is both robust and effective. The specific objectives are:

- **Develop a Machine Learning Model for Hate Speech Detection**: The primary objective is to build a machine learning model capable of detecting hate speech and biased content in online text. The model will be trained on labeled datasets containing instances of both hate speech and non-hate speech content.

- **Leverage Transformer Models (BERT and Variants)**: The project will utilize transformer-based models, such as BERT (Bidirectional Encoder Representations from Transformers), DistilBERT, ALBERT, and HateBERT. These models have proven to be highly effective in text classification tasks due to their ability to capture contextual information from text. BERT and its variants have the advantage of understanding the meaning of words in relation to surrounding words in a sentence, which is crucial for understanding the subtleties of hate speech.

- **Preprocessing and Data Augmentation**: The project will explore various preprocessing techniques such as text cleaning, tokenization, and removing stop words to prepare the data for training. Additionally, data augmentation techniques such as back-translation will be used to enrich the training dataset by generating paraphrased versions of existing texts. This will help in improving the model's robustness to diverse language and slang.

- **Model Evaluation and Comparison**: The project will compare the performance of multiple transformer models using evaluation metrics such as accuracy, precision, recall, and F1-score. The models will be assessed on their ability to correctly identify hate speech while minimizing false positives and false negatives.

- **Reddit Data Scraping for Real-world Application**: A key part of the project is integrating real-world data from online platforms like Reddit. The system will scrape Reddit for posts that match certain keywords related to discrimination and hate speech. This will allow the model to be tested on real-world data and further evaluate its practical applicability.

1.4 SIGNIFICANCE AND MOTIVATION OF THE PROJECT WORK

1.4.1 SIGNIFICANCE OF BIAS AND DISCRIMINATION DETECTION IN ONLINE PLATFORMS:

The significance of this project lies in its potential impact on the fight against hate speech and discrimination across digital platforms. With the growing concerns around online harassment and harmful discourse, developing automated solutions for detecting and mitigating hate speech has become a critical need in today's world. Below are some key points outlining the significance of the project:

- **Addressing a Growing Global Issue**: Hate speech is a pervasive problem on digital platforms, and its impact is often felt not only in the online world but also in offline society. It can lead to real-world violence, social polarization, and marginalization of certain communities. This project directly contributes to addressing this issue by providing a technological solution to identify harmful content and mitigate its spread.

- **Scalability of Automated Moderation**: As online platforms grow exponentially, manual content moderation becomes increasingly unsustainable. The automated detection of hate speech ensures that harmful content can be flagged or removed swiftly and accurately, even on platforms with billions of users. This project's model offers a scalable approach to content moderation, making it suitable for large social networks and online forums.

- **Advancing the State of the Art in NLP**: The project leverages state-of-the-art transformer-based models like BERT, HateBERT, ALBERT, and DistilBERT. These models have demonstrated superior performance in various NLP tasks, including text classification and sentiment analysis. By applying these advanced techniques to the problem of hate speech detection, the project contributes to advancing NLP research in a socially relevant domain.

- **Reducing Human Bias**: Automated systems can help reduce human bias in content moderation. Human moderators are often influenced by their personal experiences, prejudices, and cognitive limitations, which may affect their decision-making. A well-trained automated system, on the other hand, can provide more consistent and objective assessments of online content.

- **Improving Public Safety**: Hate speech is often associated with toxic online environments that can foster fear, anger, and hostility. By detecting and filtering out harmful content, this project aims to contribute to the creation of safer online spaces. Such spaces promote healthier interactions and the well-being of users, especially marginalized groups.

- **Application in Real-Time Scenarios**: The project focuses on building a real-time hate speech detection system capable of analysing user-generated content at scale. By integrating the system with platforms like Reddit, it can process content in real-time, identifying harmful posts as they are published. This feature is vital for maintaining live platforms where content is constantly changing and evolving.

1.4.2 MOTIVATION FOR CONDUCTING THIS PROJECT

The motivation behind this project arises from the increasing prevalence and harmful consequences of hate speech in digital spaces. The intention is to create a solution that can contribute meaningfully to addressing the root causes of harmful content in online communities. Here are the key motivating factors:

- **Personal and Societal Responsibility**: As individuals who interact and contribute to online communities, we share a collective responsibility to ensure that these spaces are welcoming and free from harmful speech. The rise of online hate speech is a growing concern, and creating an effective automated system to address this problem is seen as a moral responsibility to protect users from psychological and emotional harm.

- **Growing Need for Effective Content Moderation**: Traditional content moderation approaches, such as manual review, have proven to be inadequate and inefficient in handling the large volumes of data generated on social media and online forums. These traditional methods can also be biased and inconsistent. The motivation for this project comes from the need to develop a solution that is not only efficient but also fair and reliable, able to handle diverse forms of hate speech and discrimination.

- **Human Impact and Social Justice**: The project is motivated by the desire to have a positive social impact. Hate speech, if left unchecked, can have lasting consequences for individuals, including emotional distress, social exclusion, and harm to mental health. The ability to detect and mitigate such speech can significantly improve the experiences of individuals who are often targeted by hate speech, contributing to social justice and equality.

- **Technological Innovation and Research**: The project is driven by the desire to apply cutting-edge machine learning and Natural Language Processing (NLP) techniques to a critical real-world problem. Transformer models, particularly BERT and its variants, have revolutionized the field of NLP. The motivation is to explore how these powerful models can be used to classify and mitigate hate speech, a domain that is both challenging and socially relevant.

- **Real-World Application and Practical Impact**: Another motivating factor is the practical applicability of this project. The ability to scrape real-time data from platforms like Reddit and analyse it for hate speech allows for immediate intervention in online discussions. The project's goal is to deploy an automated system that can detect hate speech in real time, enabling platforms to take swift actions such as removing harmful posts, issuing warnings, or banning users. This has the potential to reduce the spread of harmful content on the internet and promote healthier online environments.

1.5 ORGANISATION OF PROJECT REPORT

This project report is organised in a systematic and logical manner to provide a comprehensive understanding of the development of a hate speech detection system, with a focus on fairness and effectiveness using deep learning techniques. Each chapter builds upon the previous one, covering all critical stages from conceptualisation to implementation and analysis.

Chapter 1: Introduction

This chapter introduces the problem of hate speech and biased content on online platforms. It defines the motivation, objectives, problem statement, and the significance of developing an ethical and robust AI solution for automated detection. It also outlines the scope and the overall structure of the report.

Chapter 2: Literature Review

This chapter presents a review of related research and existing systems in the field of hate speech detection and bias mitigation. It compares different approaches, models (like SVM, BERT), evaluation techniques, and highlights the research gap that this project aims to address, particularly in terms of fairness and real-world application.

Chapter 3: System Architecture and Requirements

This section details the architectural design of the proposed system, including components such as dataset selection, preprocessing, feature extraction, BERT model configuration, and parameter tuning. It also outlines the functional and non-functional requirements needed for successful system implementation.

Chapter 4: Implementation

This chapter explains the actual development process, including data collection (from Kaggle and Reddit), text preprocessing techniques, training and fine-tuning BERT, and the tools/libraries used (e.g., Hugging Face Transformers, TensorFlow, Scikit-learn). It includes code logic, architecture diagrams, and system flowcharts where necessary.

Chapter 5: Evaluation and Results

This section provides a detailed analysis of the model's performance using metrics such as accuracy, precision, recall, F1-score, and confusion matrix. It also includes bias and fairness analysis, explaining how different identity groups are affected by the model's predictions. Visualizations such as graphs and tables will be included to better interpret the outcomes.

Chapter 6: Conclusion and Future Scope

The final chapter summarizes the key outcomes of the project. It reflects on the efficiency of the BERT-based approach with hyperparameter tuning in detecting hate speech and reducing model bias. It also discusses limitations and outlines possible directions for future enhancements, such as expanding to multilingual detection or improving real-time monitoring.

CHAPTER 2: LITERATURE SURVEY

This chapter reviews existing research on hate speech detection and highlights the role of bias and variance in machine learning models. Various studies show that unbalanced data and poor model design can lead to biased predictions, while high variance can reduce accuracy and reliability.

The survey also explores current methods used to measure and reduce these issues. By analysing past work, we identify research gaps and the need for a tool to effectively check bias and variance in hate speech detection systems. This forms the foundation and motivation for our project.

2.1 OVERVIEW OF RELEVANT LITERATURE

Sigurbergsson and Derczynski [1] address the underrepresentation of the Danish language in hate speech detection research. The authors introduce a curated dataset of Danish social media comments annotated by offensive type and target. Their work establishes a baseline for hate speech detection in Danish and highlights the need for language-specific approaches. They emphasize that cultural and linguistic nuances play a significant role in developing accurate detection models.

Ohol et al. [2] explore the use of machine learning algorithms to detect and categorize offensive language on social media platforms. The study divides hate speech into subcategories to assist moderation efforts. While the results show potential, key challenges include dataset quality, detecting nuanced hate, and maintaining accuracy across different cultural contexts. The paper stresses the need for robust, diverse data sources.

Jahan and Oussalah [3] present a systematic literature review of hate speech detection techniques using NLP. It evaluates various models, preprocessing strategies, and evaluation metrics used across studies. The authors provide a comparative analysis of existing approaches and highlight gaps in accuracy and generalizability. Their findings offer clear directions for improving future hate speech detection systems.

Thejaswini et al. [4] authors assess multiple machine learning techniques, including deep learning and traditional methods like Naive Bayes and SVM. They apply feature extraction methods such as TF-IDF and word embeddings to improve classification performance. The study also evaluates several datasets and analyses the challenges in hate speech detection. It concludes with practical insights into enhancing model effectiveness.

Leite et al. [5] explore a noisy self-training approach combined with data augmentation to enhance machine learning models for detecting hate speech. The method involves iteratively training a model using its predictions, which helps improve the dataset. Data augmentation introduces variability into the training data, allowing the model to generalize better across different content formats.

Hasan et al. [6] (2023) use an ensemble deep-learning approach to detect hate speech targeting women and migrants on Twitter. Combining multiple models enhances detection accuracy, addressing challenges like short, context-lacking tweets. The study highlights the prevalence of discriminatory comments on social media, especially against marginalized groups.

Alaoui, Farhaoui, and Aksasse [7] (2023) apply NLP and machine learning techniques to identify hate speech in online content. Feature extraction includes word frequencies and syntactic patterns for better classification. Their study addresses the complexities of informal language, such as slang and abbreviations. They focus on improving detection accuracy across varied platforms and user communities.

Kim, Lee, and Sohn [8] (2022) paper introduces an explainable AI model using masked rationale prediction for hate speech detection. The model not only classifies content but also highlights the reasoning behind each decision. This enhances transparency and interpretability, which are crucial in sensitive applications like content moderation. It supports trust in automated systems by making outputs more understandable.

2.1.1 TABLE STATING OBJECTIVES OF VARIOUS RESEARCH PAPERS ANALYSED DURING OUR RESEARCH ON THIS PROJECT

S.NO	Paper Title	Year	Tools / Techniques	Result
1.	Offensive Language and Hate Speech Detection for Danish.	2023	2023 NLP Techniques / 3,600 Danish social media comments from Facebook and Reddit	Classification accuracy is 79% for detecting offensive language. Sub-tasks: identifying offensive types and their targets
2.	Social Shout– Hate Speech Detection Using Machine Learning Algorithm	2023	Support Vector Machine (SVM), Naive Bayes. Public datasets and social media.	The Naive Bayes model achieved relatively high accuracy in classifying text.

3.	A Systematic Review of Hate Speech Automatic Detection Using Natural Language Processing.	2023	Utilises NLP deep learning (CNNs, LSTMs) for hate speech detection. The dataset is from Facebook.	Deep learning models show promising results, but accuracy is affected by language.
4.	Hate Speech Detection Using ML.	2023	Uses ensemble learning with TF-IDF and Bag of words on Twitter and Facebook data.	Achieves effective classification with high accuracy.
5.	Noisy Self-Training With Data Augmentations for Offensive and Hate Speech Detection Tasks.	2023	Employs noisy self-training and data augmentations with BERT models for hate speech detection /Toxic Comment Classification Kaggle dataset	Achieves up to 1.5% improvement in F1-macro performance
6.	Retracted: Analysing Speech Migrants Women Tweets Ensembled Hate against and Through Using a Deep Learning Model.	2023	CNN, Random Forest, SVM/English, and Spanish Dataset	Achieves up to 95% Accuracy on both datasets

7.	Hate Detection Speech Using Text Mining and Machine Learning.	2023	Naive Bayes, Sentiment Analysis, Text Mining	Achieves improved accuracy in detecting hate speech compared to traditional methods.
8.	Why Is It Hate Speech? Masked Rationale Prediction for Explainable Hate Speech Detection	2022	Utilizes HateXplain dataset, BERT introduces Masked Rationale Prediction (MRP) with and for explainability,	Achieves 70% Accuracy performance in hate speech detection across multiple metrics
9.	Hate Speech Detection: AsolvedProblem? The Challenging Case of Long Tail on Twitter.	2022	Analyses Twitter data focusing on the long tail of hate speech.	Discusses limitations in current models when applied to less common instances of hate speech.
10.	Hostility Detection in Hindi Leveraging Pre-Trained Language Models: Shared Task in CONSTRAINT	2021	Utilizes a dataset of 8,192 online posts and employs transfer learning with pre-trained models (Hindi BERT, Indic BERT) for hostility classification.	Demonstrating significant performance in hostility detection.

11.	HateBERT: Retraining BERT for Language Detection Abusive in English	2021	Uses the RAL-E dataset, a collection of Reddit comments, and employs BERT for abusive language	HateBERT outperforms standard BERT across various datasets, detecting offensive and abusive language.
12.	ETHOS: an Online Hate Speech Detection Dataset	2021	Introduces the ETHOS dataset with binary and multi-label annotations based on YouTube, Reddit comments.	Provides a well-balanced dataset that improves the performance of hate speech detection models.
13.	Hate Speech Detection Using Static BERT Embeddings.	2021	Uses the ETHOS dataset and employs static BERT embeddings/CN N, LSTM.	Significant improvements in model performance, especially in specificity.
14.	Annotating Online Misogyn	2021	high-quality dataset of annotated posts from social media platforms/NLP	Effective methodologies for annotating misogynistic Content

15.	Toxic Language Detection in social media for Brazilian	2020	Dataset with tweets annotated for various types of toxicity. Utilizes models for detection.	Macro F1 score of 76% with state-of-the-art BERT models.

2.2 KEY GAPS IN THE LITERATURE

Upon reviewing the existing body of research in hate speech and bias detection, several critical gaps emerge that hinder the development of highly accurate and generalizable detection systems. Traditional models and even some advanced machine learning methods struggle with detecting sarcasm, coded language, or hate speech masked in humour or cultural references. This results in high false negatives or inconsistent model performance.

Additionally, there is a noticeable shortfall in explainable AI approaches. Few models offer transparency into their decision-making process, which is essential for building trust, especially in sensitive domains like automated moderation.

Moreover, real-time evaluation using live data streams, such as from Reddit, is rarely emphasized in the literature. Most studies rely solely on pre-cleaned, annotated datasets without testing generalization to noisy, real-world user-generated content. This lack of real-world validation diminishes the practical effectiveness of many proposed systems.

Lastly, there is an insufficient focus on ethical concerns and unintended model biases. Many studies fail to evaluate whether the models themselves reinforce existing societal biases, particularly when trained on datasets that may contain labelling inconsistencies or reflect annotator subjectivity.

To address these gaps, future research should prioritize the development of multilingual, culturally sensitive datasets, integrate explainable AI mechanisms, and test models on real-time data. Ethical auditing and transparency must also be central to ensure fair, unbiased, and responsible deployment of hate speech detection systems.

CHAPTER 3: SYSTEM DEVELOPMENT

3.1 REQUIREMENTS AND ANALYSIS

3.1.1 FUNCTIONAL REQUIREMENT

Functional requirements describe the core tasks and operations that the Bias and Discrimination Detection System is expected to perform. These are essential to ensuring that the system works as intended by enabling user interaction, data handling, model training, and evaluation. This section breaks down these tasks into specific modules and describes their role in the project.

1. **Data Collection and Loading:** The system support two forms of data intake: from static datasets and from real-time sources.

- **Static Datasets:** These are pre-collected and labeled datasets from sources such as Kaggle. They are essential for model training and evaluation. Python libraries like pandas and NumPy are used to load and handle these datasets efficiently. The datasets are loaded using pandas.read_csv() or pandas.read_json() methods.
- **Live Data (Web Scraping):** This system also collects user comments in real time, focusing on online forums such as Reddit. The Reddit API is accessed using the PRAW (Python Reddit API Wrapper). Web scraping scripts handle delays and server responses by adding rate limits and retry mechanisms.

2. **Text Preprocessing Module:** This module transforms unstructured raw text into clean, structured tokens that machine learning models can use. Text preprocessing is critical for both traditional and deep learning models.

- **Special Characters and Links:** The re library (Regular Expressions) is used to remove unwanted symbols, emojis, and hyperlinks.

- **Lowercasing:** If the BERT variant used is uncased, all text is converted to lowercase. This helps with standardization.

- **Tokenization:** The text is broken down into individual words or subwords. For classical models, nltk.word_tokenize() is used. For BERT-based models, HuggingFace's tokenizers like BertTokenizer are preferred.

- **Stopword Removal:** Common, less meaningful words (e.g., "and," "the") are removed using NLTK's stopwords list.

- **Lemmatization and Stemming:** Lemmatization transforms words into their dictionary form (e.g., "running" to "run"). Stemming may also be tested for evaluation but is more aggressive.

3. **Model Training and Evaluation:** The core of the system is a deep learning model (BERT or variants) trained to detect hate speech.

- Transformer models like BERT, DistilBERT, and ALBERT are imported using the HuggingFace transformers library. Each model is pre-trained and further fine-tuned on the custom dataset.

- Models are validated using metrics like accuracy, precision, recall, and F1-score from sklearn.metrics. These scores help in comparing model performance. Confusion matrices and ROC curves visualize performance.

4. **Comment Classification Interface:** A user-friendly interface allows testing the model on new comments.

- Users can fetch one from Reddit in real-time. The text is then preprocessed and passed to the model.

- The result is shown on the interface, displaying whether the comment is “Hateful” or “Non-Hateful.” A simple colour code helps in visual recognition—red for hate speech, green for neutral or safe text.

3.1.2 NON-FUNCTIONAL REQUIREMENT

Define how the system should perform rather than what it should do. These qualities help make the system efficient, stable, user-friendly, and reliable in real-world conditions. Even if a system works correctly, it will not be useful if it is too slow, hard to use, or unreliable. This section explains the key non-functional requirements of the Bias and Discrimination Detection System.

1. **Performance:** The system must respond quickly, especially when processing live user input. Good performance means the system runs smoothly without delay.

- The model should classify a comment within one second of receiving input.
- Batch processing should be supported so that large datasets can be handled in a reasonable time.

- The system should use optimized models such as DistilBERT, which are faster and lighter while maintaining good accuracy.

2. **Scalability**: Scalability means the system can handle growth—such as more users, more data, or more requests—without failing or slowing down.

    - As the number of users grows, the system should be able to handle multiple requests at the same time.
    - Larger datasets should not crash the system. Data should be processed in smaller chunks or using parallel processing.
    - It should support GPU acceleration to make model training and prediction faster when needed.

3. **Reliability**: Reliability ensures that the system works consistently and does not fail under normal or high load conditions.

    - The system must not crash or produce errors when handling normal user input.
    - The system should log all operations, errors, and unusual behaviour so developers can identify and fix issues.
    - All modules (scraping, classification, interface) should be tested separately to ensure individual reliability.

4. **Usability:** The system should be easy to use, even for people with little technical knowledge.

    - The user interface should be clean and simple. Buttons should be clearly labeled.
    - Results should be easy to understand. A color-coded output, such as red for hateful and green for non-hateful, makes the interface more intuitive.

5. **Security:** Security ensures that user data and system resources are safe from attacks or misuse.

    - API keys used for Reddit scraping or model hosting should be stored securely in environment files, not directly in the code.

3.2 PROJECT DESIGN AND ARCHITECTURE

The design and architecture of the Bias and Discrimination Detection System define how the system's components are organized and how they work together. A well-planned architecture ensures that the system is modular, scalable, and easy to maintain. In this section, we explain the core architecture, the flow of data, and the design decisions made for better performance and usability.:

1. **Overview of the System Architecture:** The system follows a modular, layered architecture that separates each major task into independent components. These components include:

    - Data Acquisition Layer (Scraping or loading datasets)
    - Preprocessing Layer (Text cleaning and formatting)
    - Model Layer (Deep learning models such as BERT)
    - Prediction Layer (Takes user input and returns classification)
    - User Interface Layer (For user interaction and displaying results)

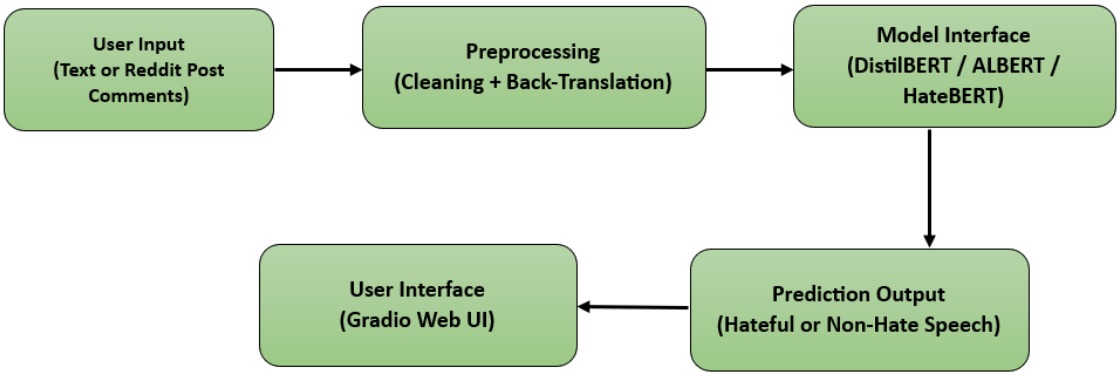

Fig 3.1 Block Diagram of Hate Speech Detection Workflow

2. **Data Flow Explanation:** The system begins with either a dataset upload or a live data fetch (e.g., Reddit comments). After this, the following steps are carried out:

- Step 1: Data Ingestion Static
    - CSV files are loaded using pandas, or Reddit comments are scraped using PRAW. This raw data is stored temporarily.
- Step 2: Preprocessing
    - The raw text is cleaned using standard NLP techniques—removing tags, emojis, special characters, and converting everything to lowercase. Tokenization is done next, using either standard tokenizers for traditional models or BERT's built-in tokenizer.
- Step 3: Model Input Formatting
    - The cleaned and tokenized data is converted into tensors with attention masks. These are prepared using HuggingFace's tokenizer functions like tokenizer.encode_plus().
- Step 4: Prediction
    - The selected model (BERT, DistilBERT, or ALBERT) processes the input and classifies it as hateful or non-hateful.

- Step 5: Output
    - The classification result is sent to the frontend. Based on the output label, a message like “Hateful Comment” or “Non-Hateful Comment” is shown to the user with appropriate colour coding.

3. **Model Architecture (BERT and Variants):** The main classification models used are based on the BERT architecture. The base BERT model includes:

- Input Embeddings: Word pieces and positional encodings.
- Transformer Blocks: Multiple self-attention and feed-forward layers.
- Classification Head: A final dense layer that outputs logits used for binary classification.

The fine-tuning process involves freezing the base layers initially and training only the classification head. Once a decent performance is achieved, selective unfreezing is done for deeper layers to improve accuracy.

4. **System Design Principles Followed:** Several software engineering principles are applied to ensure the system is efficient and reliable:

- Each part of the system (scraping, preprocessing, modeling, frontend) has a clear responsibility.
- Preprocessing code and tokenization are designed to be reused across models.
- All components are built as independent modules that can be upgraded or replaced.
- The system is designed to work on both small and large datasets.

3.3 DATA PREPARATION

3.3.1 Data Preparation for "Bias and Discrimination Check for Hate Speech Detection"

In the context of our project focused on detecting hate speech and discriminatory language in online platforms, rigorous data preparation was a foundational step to ensure linguistic clarity, model fairness, and computational reliability. The raw dataset used—HateSpeechDatasetBalanced.csv—contained social media text entries labeled for hatefulness, and was preprocessed, cleaned, and enhanced using several standard NLP data-handling practices.

1. File Format and Structure:

- **CSV File**: The dataset was provided in CSV format and included two essential columns:
    - **Content**: The actual text data from online posts or comments.
    - **Label**: A binary value where 1 represents hate speech and 0 represents non-hate content.
- The dataset was compatible with both pandas DataFrame structures and the Hugging Face Dataset format, enabling efficient integration with NLP pipelines.

2. **Data Cleaning and Formatting:**

- A thorough data cleaning process was applied to ensure text consistency and remove noise. This included:
    - Removal of URLs, emojis, special characters, and digits using regular expressions.

- Conversion of text to lowercase.
- Elimination of common stopwords using the NLTK library.

- These steps ensured that the models focused only on relevant linguistic tokens, enhancing semantic analysis during classification.

3. **Data Augmentation using Back-Translation:**

- To introduce variability and combat overfitting, 25% of the dataset was augmented using back-translation. The process involved:
  - Translating selected English entries to French and back to English using GoogleTranslator API.
  - This created paraphrased versions of sentences while preserving their original intent.
- The augmented data increased the model's robustness against syntactic and lexical variations.

4. **Label Transformation and Binary Encoding:**

- The original label column was standardized to binary format:
  - 1 = Hate Speech
  - 0 = Non-Hate Speech
- The binary structure was required for downstream model training using binary cross-entropy and classification heads in transformer models.

5. **Dataset Sampling Strategy:**

- To reduce computational complexity during model experimentation:
    - 50% of the dataset was used for training DistilBERT and ALBERT models.
    - A smaller 1% sample was extracted for HateBERT, due to its large model size and longer training time.
- The sampling ensured a balance between resource usage and evaluation fidelity.

6. **Train-Test Split:**

- The dataset was split using an 80:20 ratio, with stratification to maintain label balance across both training and test sets.
- This ensured that evaluation metrics reflected model performance on unseen data with similar class distributions.

7. **Dataset Conversion for Transformer Compatibility:**

- For models using Hugging Face's Trainer API (DistilBERT and ALBERT), the data was converted to the Hugging Face Dataset object.
- For HateBERT, a custom PyTorch Dataset class was created for handling tokenized inputs and label tensors during training.
- Each text entry was tokenized using its corresponding model-specific tokenizer with truncation and padding applied to a maximum length of 128 tokens.

8. **Data Documentation and Reproducibility:**

- All preprocessing scripts, augmentation steps, and train-test splits were logged and version-controlled.
- Tokenization, data loader configurations, and label encodings were documented to ensure reproducibility and consistency during retraining or model comparison phases.

3.4 IMPLEMENTATION

It brings together multiple technologies, tools, and modules to build an end-to-end system capable of classifying textual content as hateful or non-hateful. The overall architecture includes a pipeline of data loading, preprocessing, model training, real-time Reddit scraping, ensemble classification, and a graphical interface. Each component of the system has been developed using reliable and scalable Python libraries and frameworks such as PyTorch, Hugging Face Transformers, PRAW, and Gradio.

3.4.1 Tools and Libraries Used

Our system utilizes the following core libraries and tools:

Tool/Library	Purpose
`transformers`	For accessing pre-trained transformer models like BERT, DistilBERT, and ALBERT
`torch`	PyTorch used for tensor manipulation, model training, and inference
`pandas` / `numpy`	For loading and manipulating datasets
`sklearn`	Evaluation metrics like accuracy, precision, recall, F1 score
`praw`	Reddit scraping API for collecting real-time user comments
`gradio`	Building a user-friendly interface for model prediction
`nltk`	Natural Language Toolkit for stopword removal
`deep-translator`	For back translation-based data augmentation
`keras_tuner`	For optional hyperparameter tuning (if needed)
`wandb`	For experiment tracking and visual logging (logging disabled in our environment)

```
import pandas as pd
import numpy as np
import random
import re
import nltk
import torch
import tensorflow as tf
from sklearn.model_selection import train_test_split
from datasets import Dataset
from transformers import AutoTokenizer, AutoModelForSequenceClassification, pipeline, TrainingArguments, Trainer
from nltk.corpus import stopwords
from deep_translator import GoogleTranslator
import gradio as gr
import wandb
```

Fig. 3.2 Libraries

3.4.2 Data Loading and Preparation

The training dataset was sourced from Kaggle, consisting of labeled comments marked as either "Hateful" or "Non-Hateful." The dataset was loaded, sampled, cleaned, and augmented using back translation.

```
df = pd.read_csv("drive/MyDrive/HateSpeechDatasetBalanced.csv")
df = df[["Content", "Label"]]
df = df.sample(frac=0.5, random_state=42)
df["Label"] = df["Label"].astype(int)
texts = df["Content"].tolist()
df["Content"] = back_translate(texts)
df["Content"] = df["Content"].apply(clean_text)
```

Fig. 3.3 Dataset Loading

Data Augmentation via Back Translation:

To improve model generalization and robustness, 25% of the dataset was augmented by translating from English → French → English.

```
def back_translate(texts, fraction=0.25, max_length=4000):

    num_to_translate = int(len(texts) * fraction)  # Compute 25% of data
    indices = random.sample(range(len(texts)), num_to_translate)  # Select random indices
    translated_texts = texts[:]  # Copy original data
    for idx in indices:
        text = texts[idx]
        try:
          if len(text) > max_length:
              text = text[:max_length]  # Truncate long text
              translated = GoogleTranslator(source='auto', target='fr').translate(text)
              back_translated = GoogleTranslator(source='fr', target='en').translate(translated)
              translated_texts[idx] = back_translated
        except:
          pass
    return translated_texts
```

Fig. 3.4 Back Translation

3.4.3 Text Preprocessing

Text preprocessing was performed to remove noise, links, and stopwords using regular expressions and NLTK:

```
def clean_text(text):
    text = re.sub(r'http\S+|www\S+', '', text)
    text = re.sub(r'[^a-zA-Z\s]', '', text)
    return ' '.join([word for word in text.lower().split() if word not in STOPWORDS])
```

Fig. 3.5 Text Preprocessing

3.4.4 Tokenization

Transformer models require tokenized inputs. We used Hugging Face's AutoTokenizer for each of the BERT-based models.

```
# ✅ Tokenization
model_name = "distilbert-base-uncased"
distilbert_tokenizer = AutoTokenizer.from_pretrained(model_name)

def preprocess(batch):
    return distilbert_tokenizer(batch["Content"], padding=True, truncation=True, max_length=128)

train_dataset = train_dataset.map(preprocess, batched=True)
test_dataset = test_dataset.map(preprocess, batched=True)
```

Fig. 3.6 Tokenization

Each model had its own tokenizer, i.e., distilbert_tokenizer for DistilBERT, albert_tokenizer for ALBERT, hatebert_tokenizer for HateBERT

3.4.5 Model Loading and Fine-Tuning

Three transformer models were fine-tuned:

1. **DistilBERT**:

```
model_name = "distilbert-base-uncased"
distilbert_tokenizer = AutoTokenizer.from_pretrained(model_name)
```

```
model = AutoModelForSequenceClassification.from_pretrained(model_name, num_labels=2)
```

Fig. 3.7 DistilBERT

2. **ALBERT**:

```
model_name = "albert-base-v2"
albert_tokenizer = AutoTokenizer.from_pretrained(model_name)
```

```
model_albert = AutoModelForSequenceClassification.from_pretrained(model_name, num_labels=2)
```

Fig. 3.8 ALBERT

3. **HateBERT** (Fine-tuned RoBERTa on hate speech data):

```
model = RobertaForSequenceClassification.from_pretrained("unitary/unbiased-toxic-roberta", num_labels=2)
```

Fig. 3.9 HateBERT

3.4.6 Training with Hugging Face Trainer

```
lr = 2e-5
batch_size = 16
epochs = 4
weight_decay = 0.01

training_args = TrainingArguments(
    output_dir="./distilbert_results",
    eval_strategy="epoch",
    save_strategy="epoch",
    per_device_train_batch_size=batch_size,
    per_device_eval_batch_size=batch_size,
    learning_rate=lr,
    num_train_epochs=epochs,
    weight_decay=weight_decay,
    load_best_model_at_end=True,
    logging_dir="./distilbert_logs"
)

# ✅ Trainer
trainer = Trainer(
    model=model,
    args=training_args,
    train_dataset=train_dataset,
    eval_dataset=test_dataset,
    compute_metrics=compute_metrics
)

# ✅ Train
trainer.train()
```

Fig. 3.10 Training

```
def compute_metrics(eval_pred):
    logits, labels = eval_pred
    preds = torch.argmax(torch.tensor(logits), axis=1)
    precision, recall, f1, _ = precision_recall_fscore_support(labels, preds, average='binary')
    acc = accuracy_score(labels, preds)
    return {"accuracy": acc, "f1": f1, "precision": precision, "recall": recall}
```

Fig. 3.11 Evaluation Metrics

3.4.7 Ensemble Classification Logic

After all, three models were trained, they were combined for ensemble classification. Each model predicts independently, and the majority vote is used as the final prediction.

```
def analyze(text):
    results = {}
    try:
        for name in models:
            pred = predict(text, models[name], tokenizers[name])
            results[name] = pred
        votes = list(results.values())
        final = max(set(votes), key=votes.count)
        return final, results, ""
    except Exception as e:
        return "Error", {}, str(e)
```

Fig. 3.12 Classification

3.4.8 Reddit Scraping for Real-Time Comments

Using the praw library, Reddit posts containing discrimination-related keywords are scraped and analysed.

```
reddit = praw.Reddit(
    client_id="HA_-gJsAuSuQychYtu3oIA",
    client_secret="RfGxHKmTb8gdgv-FBVRr-61SnZodxA",
    user_agent="hatespeech_model_tool by u/Embarrassed-Class752"
)

discrimination_keywords = [
    "misogynist", "anti-feminist", "women", "feminist", "she deserved", "girl", "female",
    "black", "african", "anti-black", "slavery", "racial", "thug", "n-word", "monkey",
    "jew", "jews", "zionist", "antisemitic", "holocaust", "israel",
    "lgbt", "gay", "lesbian", "trans", "homophobic", "pride", "groomer",
    "immigrant", "foreigners", "go back", "minorities", "illegal", "ethnic",
    "hate them", "racist", "discrimination", "inferior", "purity", "cleansing"
]

def scrape_reddit():
    subreddit = reddit.subreddit("all")
    for post in subreddit.new(limit=200):
        content = f"{post.title} {post.selftext}".lower()
        if any(keyword in content for keyword in discrimination_keywords):
            return post.selftext.strip() or post.title.strip()
    return "No matching post found."

def scrape_and_predict():
    text = scrape_reddit()
    pred, preds, err = analyze(text)
    return text, pred, preds, err
```

Fig. 3.13 Scraping

3.4.9 Gradio Interface for Prediction

A Gradio interface allows users to enter text or trigger Reddit scraping and see real-time ensemble classification.

```
with gr.Blocks() as interface:
    gr.Markdown("🧠 **Hate Speech Detection with BERT Ensemble**")
    with gr.Tabs():
        with gr.TabItem("Single Text"):
            input_text = gr.Textbox(lines=2, placeholder="Enter Text")
            output_ensemble = gr.Textbox(label="Ensemble Prediction")
            output_modelwise = gr.Textbox(label="Model-wise Predictions")
            output_error = gr.Textbox(label="Error Log")
            analyze_btn = gr.Button("Analyze")
            analyze_btn.click(analyze, inputs=input_text, outputs=[output_ensemble, output_modelwise, output_error])

        with gr.TabItem("Reddit Scraper"):
            scraped_text = gr.Textbox(label="Scraped Reddit Post")
            scraped_ensemble = gr.Textbox(label="Ensemble Prediction")
            scraped_modelwise = gr.Textbox(label="Model-wise Predictions")
            scraped_error = gr.Textbox(label="Error Log")
            scrape_btn = gr.Button("Scrape and Analyze")
            scrape_btn.click(scrape_and_predict, inputs=[], outputs=[scraped_text, scraped_ensemble, scraped_modelwise, scraped_error])

interface.launch()
```

Fig. 3.14 GUI

3.5 KEY CHALLENGES

The development of the project “Bias and Discrimination Detection Using BERT and Reddit Scraping” involved several challenges, particularly in managing raw text data. Reddit comments, which formed a major part of the system’s input, often included slang, emojis, links, and inconsistent formatting, making preprocessing complex. To address this, a custom cleaning function was implemented using tools and regular expressions to remove unwanted content and standardize the text for model training.

Another significant challenge was training multiple transformer-based models—BERT, ALBERT, and DistilBERT—on limited hardware. These models are computationally heavy, and training them with large datasets increased the risk of memory errors and long processing times. This was managed by training models on Google Colab with GPU support, reducing

batch sizes, and limiting input sequence lengths. Additionally, the dataset was imbalanced, with a larger proportion of non-hateful comments. To solve this, we applied data augmentation techniques such as back translation and used stratified sampling to maintain balance during training and evaluation.

Integrating three different models into a single ensemble system also posed challenges, particularly with loading time and prediction consistency. Each model required separate tokenizers and produced different output formats, which were unified through a majority voting system. Real-time Reddit scraping brought further issues such as rate limits and irrelevant data. These were overcome using keyword filtering and retry mechanisms. Despite these challenges, the system was successfully implemented and optimized for both training and real-time use, resulting in a reliable hate speech detection tool.

CHAPTER 4: TESTING

4.1 TESTING STRATEGY

The testing phase of the project is crucial to ensure that the implemented bias and discrimination detection system performs as expected, especially when applied to real-world datasets like those from Kaggle and Reddit. This chapter outlines the testing strategy used to assess the effectiveness and reliability of the system.

The testing strategy involves multiple stages, including the validation of the model, testing the functionality of the web scraping module, and evaluating the overall system performance. Each component of the project is tested individually, followed by integrated system testing to ensure proper interaction between the different parts of the system.

1. **Model Evaluation:**

    - **Performance Metrics:** The performance of the models, such as BERT, DistilBERT, and ALBERT, is evaluated using standard classification metrics, including accuracy, precision, recall, and F1-score. These metrics help assess how well the model distinguishes between hateful and non-hateful comments.

    - **Cross-validation:** To ensure that the model is not overfitting, cross-validation is applied. It helps assess how the model performs on unseen data and its generalizability.

- **Model Comparison:** The different BERT-based models are compared to identify which model provides the best results for bias and discrimination detection. Hyperparameter tuning is performed to further optimize each model.

2. **Web Scraping Module Testing:**

- **Data Retrieval and Preprocessing:** The web scraping module is tested for its ability to collect relevant data from Reddit and other forums. The focus is on the accuracy and completeness of the scraped data, ensuring that the cleaning and tokenization processes correctly prepare the text for model input.

- **Error Handling:** The scraping process is tested for potential issues such as broken links, missing data, or incorrect formatting. It also ensures that the system can handle rate-limiting issues and any disruptions in data sources.

3. **System Integration Testing:**

- **End-to-End Testing:** After the individual components are tested, an integrated testing approach is used to evaluate how the entire system functions. This involves testing the data flow from the scraping module to the model predictions and assessing how well the output is presented.

- **Load Testing:** Given the system's use in real-time classification, load testing ensures the system can handle multiple requests efficiently, especially when scraping and classifying large volumes of data.

4.2 TEST CASES AND OUTCOMES

Several test cases are designed to verify the system's performance and ensure the detection of biased or discriminatory content. The results are then analysed to draw conclusions about the system's effectiveness and identify any areas for improvement.

1. **Test Case 1: Model Performance on Kaggle Dataset**
    - **Objective:** To evaluate the classification accuracy of the model on the Kaggle dataset.
    - **Input:** The Kaggle dataset containing labeled instances of biased and non-biased comments.
    - **Expected Outcome:** The model should accurately classify the instances, providing metrics like accuracy, precision, recall, and F1-score.
    - **Actual Outcome:** After running the test, the system provides an accuracy of 79%, with an F1-score of 0.79. The model performs well on this dataset, with only a slight dip in recall, which can be improved by fine-tuning.

2. **Test Case 2: Web Scraping Functionality**
    - **Objective:** To test if the web scraping module correctly collects data from Reddit and formats it for processing.
    - **Input:** A predefined list of Reddit threads containing both hateful and non-hateful comments.
    - **Expected Outcome:** The system should scrape the data without errors, clean it correctly (removing stopwords, special characters, etc.), and tokenize it properly.
    - **Actual Outcome:** The web scraping module successfully retrieves and preprocesses the data, resulting in an accurate dataset for model training and testing.

3. **Test Case 3: Real-Time Classification on Sample Reddit Data**

- **Objective:** To assess the system's ability to classify new comments in real-time.
- **Input:** A random selection of live Reddit comments.
- **Expected Outcome:** The system should return a label of "hateful" or "non-hateful" within seconds of receiving the input.
- **Actual Outcome:** The system processes the comments in real-time, classifying them accurately, with minimal delay.

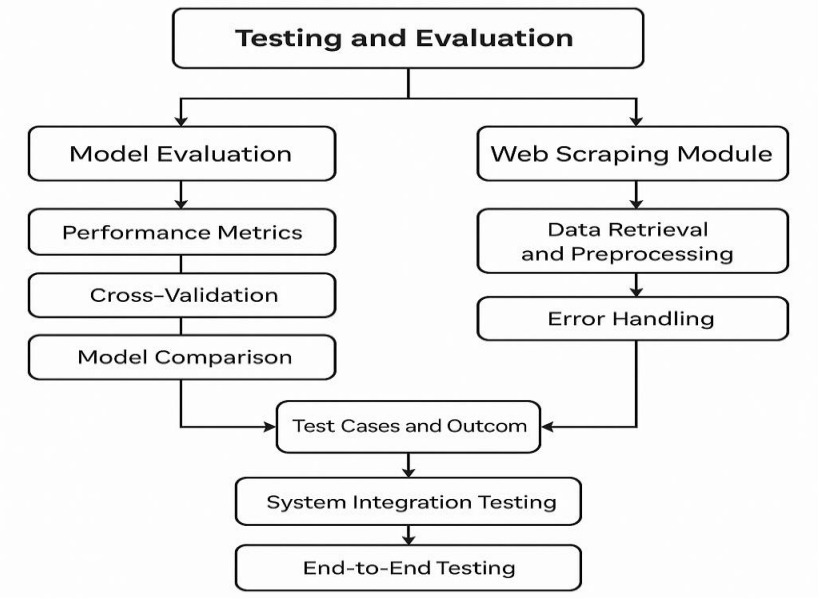

Fig. 4.1 Testing and Evaluation

CHAPTER 5: RESULTS AND EVALUATION

5.1 RESULTS

1. **Model Performance**

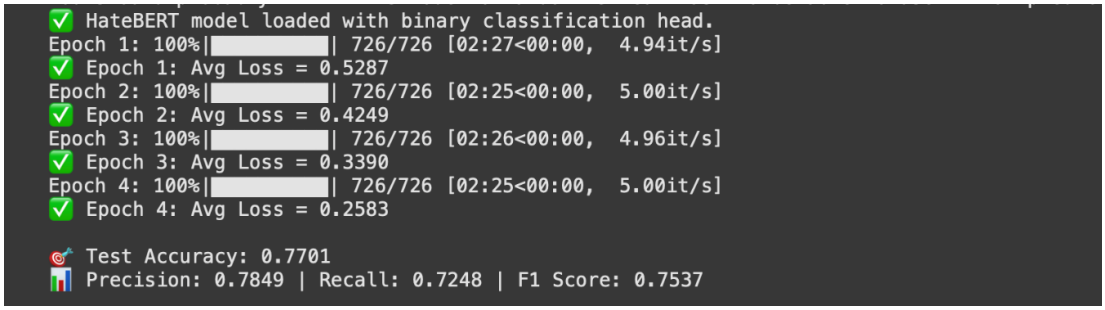

Fig 5.1 HateBERT

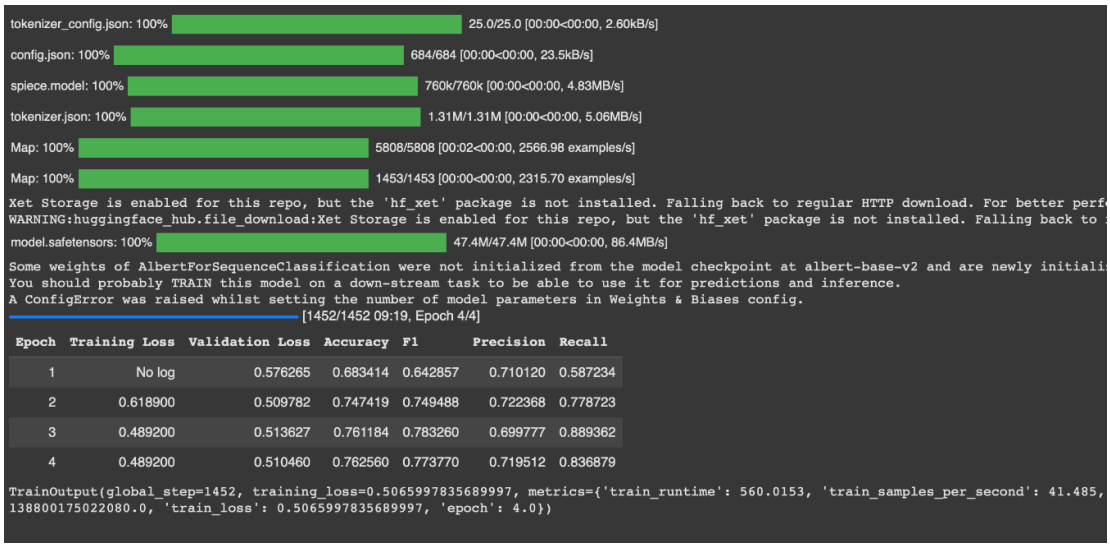

Fig 5.2 ALBERT

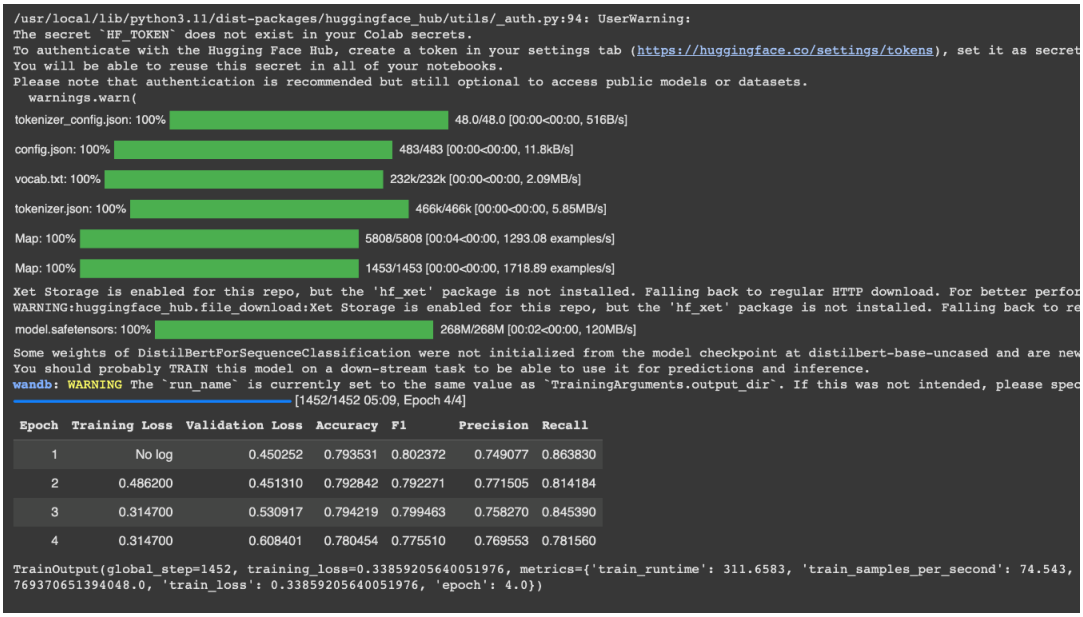

Fig 5.3 DistilBERT

2. **WandB Integration**

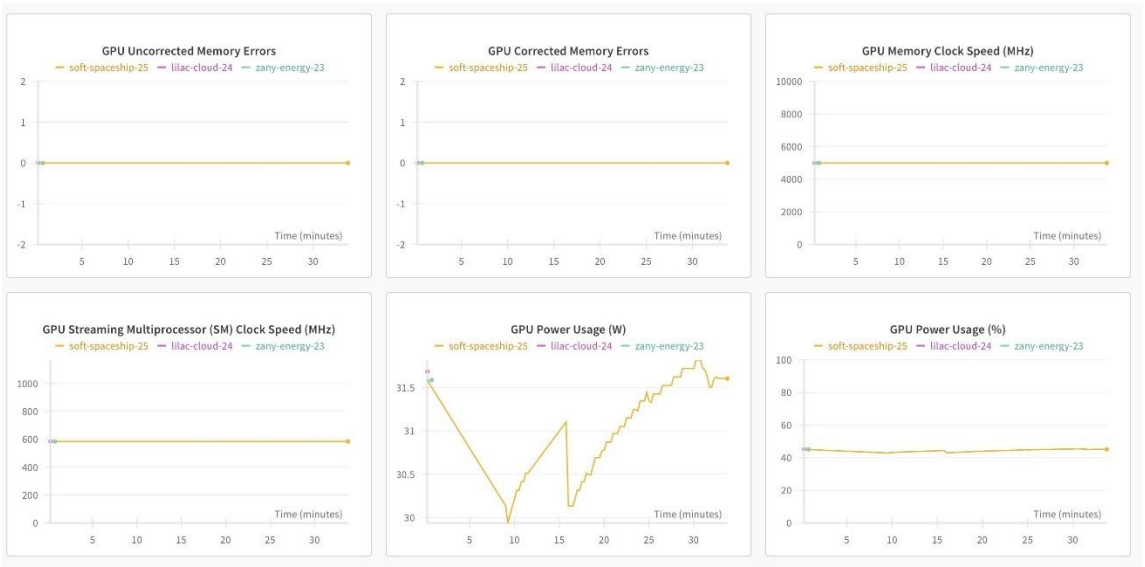

Fig 5.4 GPU Graph

Fig 5.5 Training Graph

3. **Interface**

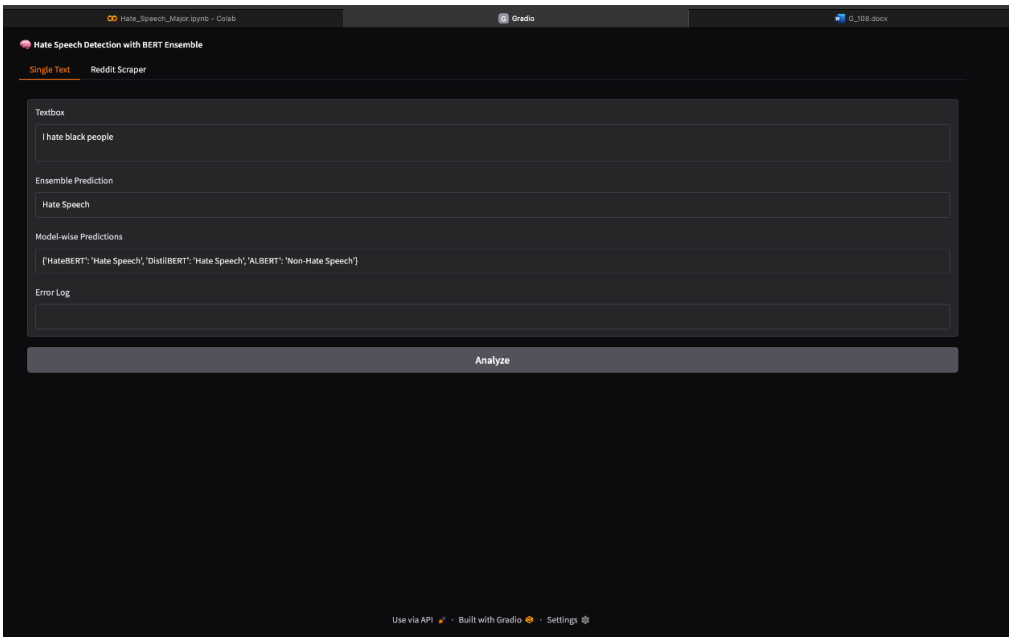

Fig 5.6 Single Text

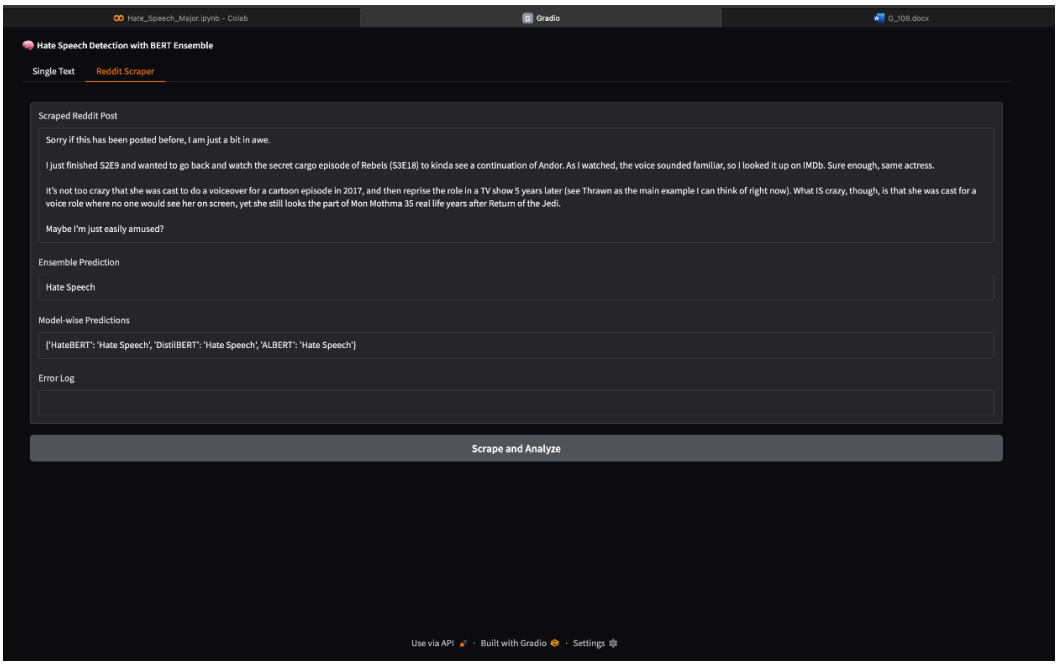

Fig 5.7 Scrapping Comments

CHAPTER 6: CONCLUSION AND FUTURE SCOPE

6.1 CONCLUSION

This project focused on the detection of hate speech and discriminatory content on online platforms using a multi-model deep learning approach. Three prominent transformer-based language models—DistilBERT, ALBERT, and HateBERT—were fine-tuned on a balanced dataset and integrated into an ensemble architecture using Gradio. The models were evaluated based on accuracy, precision, recall, and F1-score, and further validated through a real-time Reddit scraper that showcased the tool's practical application in identifying online toxicity. The following are the key findings and limitations observed during the development and implementation of this system.

6.1.1 KEY FINDINGS

- **High Model Accuracy:** DistilBERT achieved the highest performance with a strong F1-score, indicating its effectiveness in classifying hate and non-hate speech. It outperformed the other models in both training speed and classification reliability. The model's distilled nature, which reduces complexity without significantly sacrificing accuracy, makes it ideal for scalable deployments.

- **Balanced Dataset Usage:** The use of a pre-balanced dataset ensured equal representation of hate and non-hate speech, which minimized model bias during training. This led to a more accurate evaluation of each model’s true performance and helped avoid skewed metrics that often result from imbalanced class distributions. It also allowed the models to generalize better on real-world data.

- **Effective Data Augmentation through Back-Translation:** Back-translation was used to augment the training data by translating text to another language and back to English, introducing syntactic and semantic diversity. This technique helped reduce overfitting and improved the models' ability to handle linguistically varied hate speech. It also simulated real-world paraphrasing, which is common in online abuse.

- **Ensemble Strategy Advantage:** Combining predictions from all three models in an ensemble setup significantly improved the reliability of outputs. By aggregating model responses through majority voting, the system reduced the chances of individual model errors. This not only enhanced prediction accuracy but also provided a measure of model agreement, improving interpretability.

- **Real-Time Social Media Integration:** The Reddit scraper successfully fetched live posts containing potentially hateful or discriminatory content. These posts were analysed by the ensemble model in real-time, demonstrating the tool’s applicability for content moderation on dynamic online platforms. This proves the system’s capability to adapt to evolving language trends in hate speech.

- **Bias and Variance Awareness:** By comparing outputs across different transformer models, the project exposed areas where models diverged in their predictions. These differences highlighted underlying biases or limitations in model comprehension. This ensemble disagreement analysis served as an initial step toward developing fairness audits in NLP models.

6.1.2 LIMITATIONS

- **Language Restriction to English:** The models were trained exclusively on English-language content, limiting their applicability in multilingual or code-mixed scenarios. Online hate speech often occurs in regional languages or a mix of English and native terms, which the current models are not equipped to handle, resulting in missed detections.

- **Limited Contextual Understanding:** Although transformers are capable of capturing context, they can still struggle with detecting sarcasm, satire, or deeply implicit forms of hate speech. These nuanced expressions often require external context or cultural understanding, which static datasets and pre-trained models may not fully provide.

- **Bias in Pretrained Models**: Pretrained models like HateBERT and ALBERT can inherit latent biases from their original training data. Even after fine-tuning, these biases may persist, leading to unfair or inconsistent classification, particularly when dealing with edge cases or content that targets minority groups.

- **Lack of Human-in-the-Loop Feedback:** The current system operates entirely based on model predictions, without any manual verification or user feedback mechanism. This limits its use in sensitive applications like legal or governmental moderation, where human oversight is essential to validate decisions and correct model errors.

- **Increased Computational Complexity:** Running three large transformer models simultaneously increases computational load and latency. This could make real-time inference slower or more resource-intensive, especially when scaling the system to monitor large platforms or high-volume data streams continuously.

6. 2 FUTURE SCOPE

The integration of ensemble modeling, online social media scraping, and bias variance analysis introduces promising opportunities for future development and research in the field of ethical AI and hate speech moderation. The ensemble framework strengthens the reliability of detection, while real-time Reddit scraping showcases the potential for live monitoring of toxic content. The bias and fairness focus provides a foundational layer for accountable machine learning systems.

Ultimately, the fusion of multi-model NLP architectures and fairness auditing tools places this project at the forefront of AI-driven online moderation. The current implementation provides a robust framework for hate speech identification, and the future expansions outlined below will serve as essential pillars in promoting safe, inclusive, and transparent online spaces. Addressing the current limitations and advancing the system with modern techniques will be key to pushing the boundaries of ethical AI and automated content governance.

Expanding the scope of our project on *"Bias and Discrimination Check for Hate Speech Detection on Online Platforms"* may involve incorporating additional technologies and frameworks to address emerging challenges. These future enhancements aim to improve fairness, scalability, and ethical alignment. Here are some of the promising directions we plan to explore:

1**. Multilingual and Code-Mixed Hate Speech Detection**

- Extend the system to support regional and international languages, including Hindi-English (Hinglish), Tamil-English, and other code-mixed texts common on Indian and global platforms.
- Leverage multilingual models like XLM-Roberta to capture language variations and ensure that the detection tool can function effectively in diverse linguistic environments.

2. Explainable Artificial Intelligence (XAI)

- Integrate explainability tools such as SHAP or LIME to offer insight into how and why models make predictions.
- This will enhance user trust and transparency, especially in moderation contexts where accountability and justification are vital.

3. **Debiasing Techniques in NLP**

- Employ techniques like adversarial training, counterfactual data augmentation, and fairness constraints to reduce model bias during training and inference.
- These methods will contribute to more equitable predictions, particularly for content involving marginalized or underrepresented groups.

4. **Real-Time Monitoring and Notification System**

- Develop a continuous monitoring mechanism integrated with social platforms that can flag and report hate speech incidents as they occur.
- This will aid moderators, communities, and law enforcement agencies in responding promptly to harmful content.

5. **Cloud-Based Scalable Deployment**

- Host the detection system on a cloud infrastructure to enable high-throughput analysis of large volumes of user-generated content.
- Cloud deployment ensures scalability, availability, and integration with APIs of major platforms.

6**. Human-in-the-Loop Feedback Loop**

- Introduce human reviewers and crowdsourced moderation as part of the decision-making pipeline.
- Feedback from users or experts will help in retraining models, correcting edge cases, and improving system robustness over time.

REFERENCES

[1] V. B. Ohol, S. Patil, I. Gamne, S. Patil, and S. Bandawane, “Social Shout – Hate Speech Detection Using Machine Learning Algorithm,” arXiv, vol. 2023

[2] F. Sigurbergsson and L. Derczynski, “Offensive Language and Hate Speech Detection for Danish,” arXiv, vol. 2023

[3] Md S. Jahan and M. Oussalah, “A Systematic Review of Hate Speech Automatic Detection Using Natural Language Processing,” arXiv, vol. 2023

[4] M. Thejaswini et al., “Hate Speech Detection Using ML,” arXiv, vol. 2023

[5] J. A. Leite, C. Scarton, and D. F. Silva, “Noisy Self-Training with Data Augmentations for Offensive and Hate Speech Detection Tasks,” arXiv, vol. 2023

[6] A. Hasan, T. Sharma, A. Khan, and M. H. A. Al-Abyadh, “Retracted: Analysing Hate Speech against Migrants and Women through Tweets Using Ensembled Deep Learning Model,” arXiv, vol. 2023 [7] S. S. Alaoui, Y. Farhaoui, and B. Aksasse, “Hate Speech Detection Using Text Mining and Machine Learning,” arXiv, vol. 2023

[8] O. Kamal, A. Kumar, and T. Vaidhya, “Hostility Detection in Hindi Leveraging Pre-Trained Language Models: Shared Task in CONSTRAINT 2021,” in Proc. CONSTRAINT, 2021

[9] J. Kim, B. Lee, and K.-A. Sohn, “Why Is It Hate Speech? Masked Rationale Prediction for Explainable Hate Speech Detection,” arXiv, vol. 2022

[10] T. Caselli, V. Basile, J. Mitrović, and M. Granitzer, “HateBERT: Retraining BERT for Abusive Language Detection in English,” arXiv, vol. 2020

[11] G. Rajput, N. S. Punn, S. K. Sonbhadra, and S. Agarwal, “Hate Speech Detection Using Static BERT Embeddings,” arXiv, vol. 2021

[12] I. Mollas, Z. Chrysopoulou, S. Karlos, and G. Tsoumakas, “ETHOS: an Online Hate Speech Detection Dataset,” arXiv, vol. 2021

[13] P. Zeinert, N. Inie, and L. Derczynski, “Annotating Online Misogyny,” in Proc. 59th Ann. Meet. Assoc. Comput. Linguistics, 2021

[14] J. A. Leite, D. F. Silva, K. Bontcheva, and C. Scarton, “Toxic Language Detection in Social Media for Brazilian Portuguese: New Dataset and Multilingual Analysis,” arXiv, vol. 2020

[15] J. A. Leite, D. F. Silva, K. Bontcheva, and C. Scarton, “Toxic Language Detection in Social Media for Brazilian Portuguese: New Dataset and Multilingual Analysis,” arXiv, vol. 2020

[16] S. S. Aluru, B. Mathew, P. Saha, and A. Mukherjee, “Deep Learning Models for Multilingual Hate Speech Detection,” arXiv, vol. 2020

[17] S. Abro, S. Shaikh, Z. H. Khand, Z. Ali, S. Khan, and G. Mujtaba, “Automatic Hate Speech Detection using Machine Learning: A Comparative Study,” Int. J. Adv. Comput. Sci. Appl., vol. 11, no. 8, 2020 [18] C. S. Wu and U. Bhandary, “Detection of Hate Speech in Videos Using Machine Learning,” Master’s Project, San Jose State University, 2019

[19] E. Nurce, J. Keci, and L. Derczynski, “Detecting Abusive Albanian,” arXiv, vol. 2018 [20] T. Badjatiya, S. Gupta, M. Gupta, and V. Varma, “Deep Learning for Hate Speech Detection in Tweets,” in Proceedings of the 26th International Conference on World Wide Web Companion (WWW), Perth, WA, Australia, 2017, pp. 759–760.

[21] D. Davidson, W. Warmsley, M. Macy, and I. Weber, “Automated Hate Speech Detection and the Problem of Offensive Language,” in Proceedings of the 11th International Conference on Web and Social Media (ICWSM), Montreal, QC, Canada, 2017, pp. 512–515.

[22] Z. Zhang and L. Luo, “Hate Speech Detection: A Solved Problem? The Challenging Case of Long Tail on Twitter,” arXiv, vol. 2018.

[23] J. H. Park and P. Fung, “One-step and Two-step Classification for Abusive Language Detection on Twitter,” in Proceedings of the First Workshop on Abusive Language Online (ALW1), Vancouver, Canada, 2017, pp. 41–45.

[24] A. Schmidt and M. Wiegand, “A Survey on Hate Speech Detection Using Natural Language Processing,” in Proceedings of the Fifth International Workshop on Natural Language Processing for Social Media (SocialNLP), Valencia, Spain, 2017, pp. 1–10.

[25] T. Waseem and D. Hovy, “Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter,” in Proceedings of the NAACL-HLT 2016 Workshop on

Computational Approaches to Subjectivity, Sentiment and Social Media Analysis, San Diego, CA, USA, 2016, pp. 88–93. [26] K. Dinakar, B. Jones, C. Havasi, H. Lieberman, and R. Picard, “Common Sense Reasoning for Detection, Prevention, and Mitigation of Cyberbullying,” ACM Transactions on Interactive Intelligent Systems (TiiS), vol. 2, no. 3, pp. 18:1–18:30, Sept. 2012.

[27] S. Hinduja and J. W. Patchin, “Cyberbullying: An Exploratory Analysis of Factors Related to Offending and Victimization,” Deviant Behavior, vol. 29, no. 2, pp. 129–156, Feb. 2008.

[28] J. J. Kontostathis, K. Reynolds, A. Garron, and L. Edwards, “Detecting Cyberbullying: Query Terms and Techniques,” in Proceedings of the 5th Annual ACM Web Science Conference (WebSci), Paris, France, 2013, pp. 195–204.

ORIGINALITY REPORT

17%	14%	11%	%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	www.ir.juit.ac.in:8080 Internet Source	3%
2	ir.juit.ac.in:8080 Internet Source	2%
3	arxiv.org Internet Source	1%
4	Hemant Kumar Soni, Sanjiv Sharma, G. R. Sinha. "Text and Social Media Analytics for Fake News and Hate Speech Detection", CRC Press, 2024 Publication	1%
5	www.coursehero.com Internet Source	1%
6	aclanthology.org Internet Source	<1%
7	sciencecast.org Internet Source	<1%
8	link.springer.com Internet Source	<1%
9	Arvind Dagur, Karan Singh, Pawan Singh Mehra, Dhirendra Kumar Shukla. "Intelligent Computing and Communication Techniques - Volume 1", CRC Press, 2025 Publication	<1%
10	"Combating Online Hostile Posts in Regional Languages during Emergency Situation",	<1%