A COMPARATIVE STUDY OF MACHINE LEARNING AND DEEP LEARNING METHODS FOR CYBERBULLYING DETECTION ON TWITTER

OMAR AKRAM KHALEEL AL-SAFFAR

UNIVERSITI KEBANGSAAN MALAYSIA

A COMPARATIVE STUDY OF MACHINE LEARNING AND DEEP LEARNING METHODS FOR CYBERBULLYING DETECTION ON TWITTER

OMAR AKRAM KHALEEL AL-SAFFAR

PROJECT SUBMITTED IN PARTIAL FULFILMENT FOR THE DEGREE OF MASTER OF DATA SCIENCE

FACULTY OF INFORMATION SCIENCE AND TECHNOLOGY
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI

2024

KAJIAN PERBANDINGAN KAEDAH PEMBELAJARAN MESIN DAN PEMBELAJARAN MENDALAM UNTUK PENGESANAN PEMBULIAN SIBER DI TWITTER

OMAR AKRAM KHALEEL AL-SAFFAR

PROJEK DIKEMUKAKAN SEBAGAI SEBAHAGIAN DARIPADA
PEMENUHAN SYARAT UNTUK IJAZAH SARJANA SAINS DATA

FAKULTI SAINS DAN TEKNOLOGI MAKLUMAT
UNIVERSITI KEBANGSAAN MALAYSIA
BANGI
2024

DECLARATION

I hereby declare that the work in this thesis is my own except for quotations and summaries which have been duly acknowledged.

25 July 2024

OMAR AKRAM AL-SAFFAR
P120215

ACKNOWLEDGEMENT

Foremost, I am grateful to Almighty Allah for His blessings, including the patience and good health granted throughout my master's research journey.

I deeply thank my supervisor, Dr. Lailatul Qadri Zakaria, for her steadfast guidance, support, and invaluable advice. Her prompt responses, problem-solving skills, and inspiring enthusiasm have been essential to my success.

I also express my heartfelt gratitude to my family for their spiritual support, care, and understanding, which were crucial in completing my master's degree.

My appreciation extends to the FTSM department staff at UKM, whose help and kindness greatly assisted me. I gratefully acknowledge everyone’s contributions to my studies.

Lastly, special thanks to my friends and colleagues for their supportive presence throughout this journey.

ABSTRAK

Pembulian siber di media sosial, terutamanya di platform seperti Twitter, membawa kesan psikologi dan emosi yang mendalam kepada pengguna. Pembulian siber boleh diklasifikasikan kepada pelbagai bentuk seperti berbentuk keagamaan, jantina, umur dan etnik. Cabaran dalam mengesan pembulian siber termasuk kesukaran dan keterikatan konteks bahasa yang kasar, perubahan pantas slang dan istilah baharu, serta keseimbangan antara ketepatan dan capaian untuk mengelakkan keputusan positif dan negatif palsu. Kajian ini meneliti cabaran mengesan cyberbullying di Twitter, dengan memberi fokus kepada kesukaran menganalisis bahasa slang dan tidak formal menggunakan teknik Pembelajaran Mesin (ML) sedia ada. Kajian ini menekankan keperluan untuk kejuruteraan ciri yang lebih maju dan algoritma yang lebih tepat. Kajian ini menilai keberkesanan model Pembelajaran Mesin (ML) seperti Random Forest (RF), Logistic Regression (LR), Support Vector Machine (SVM) dan model Pembelajaran Mendalam (DL) lanjutan seperti Long Short-Term Memory Networks (LSTM), Robustly Optimized BERT Approach (RoBERTa) untuk menilai keupayaan model tersebut mengenalpasti kandungan kontekstual dan semantik. Objektif utama termasuk menilai kesan kualiti data dan pemilihan fitur terhadap prestasi model dan membandingkan potensi model ML dan DL. Kajian ini menggunakan data Twitter, yang kemudiannya diproses dengan teliti untuk memastikan input berkualiti tinggi. pelbagai teknik kejuruteraan ciri termasuk Term Frequency dan Inverse Document Frequency (TF-IDF), Bag Perkataan (BoW), pembenaman perkataan, N-grams, pengekodan huruf, dan trik hashing digunakan untuk mengesan ciri-ciri bahasa pembulian siber. Keputusan eksperimen menunjukkan model ML mencapai ketepatan yang baik, dengan RF mencapai ketepatan 93.26%, (LR) pada 92.79%, dan SVM pada 92.92%. Walau bagaimanapun, model pembelajaran mendalam jauh mengatasi model ML, dengan LSTM mencapai ketepatan 94.26% dan RoBERTa mencapai ketepatan tertinggi 94.55%. Penemuan ini menunjukkan peranan penting kualiti fitur dan pemilihan model dalam meningkatkan ketepatan pengesanan. Kajian ini menyumbang kepada pengesanan pembulian siber dengan menunjukkan prestasi terbaik dari model pembelajaran mendalam dan memberikan hala tuju penyelidikan masa hadapan untuk membina sistem pengesanan yang lebih baik.

ABSTRACT

Cyberbullying on social media, especially on platforms like Twitter (now called X), presents profound psychological and emotional difficulties for users. Cyberbullying can be classified into different classes such as religious, gender, age and ethnicity. The challenges of detecting cyberbullying include the subtlety and context-dependence of abusive language, the rapid evolution of slang and new terms, and the balance between precision and recall avoiding false positives and negatives. The research addresses the challenges of detecting cyberbullying on Twitter, focusing on the difficulties of analysing slang and informal language with current ML techniques. It emphasizes the need for advanced feature engineering and more accurate algorithms. The study evaluates the effectiveness of Machin Learning (ML) models such as Random Forest (RF), Logistic Regression (LR), Support Vector Machine (SVM) and Deep Learning (DL) models such as Long Short-Term Memory Networks (LSTM), Robustly Optimized BERT Approach (RoBERTa) for their ability to capture contextual and semantic content. Key objectives include assessing the impact of preprocess data and feature representation selections on model performance and comparing ML and DL models. This research uses Twitter data, then pre-processed to ensure high-quality input. Multiple features engineering techniques, including Term Frequency and Inverse Document Frequency (TF-IDF), Bag of Words (BoW), word embeddings, N-grams, character encoding, and the hashing trick, were employed to capture the characteristics of cyberbullying language. Experimental results indicate that traditional ML models achieve reasonable accuracy, with Random Forest achieving an accuracy of 93.26%, Logistic Regression at 92.79%, and SVM at 92.92%. However, deep learning models significantly outperform traditional approaches, with LSTM achieving an accuracy of 94.26% and RoBERTa reaching the highest accuracy of 94.55%. These findings highlight the crucial role of feature quality and model development in enhancing detection accuracy. The study contributes to cyberbullying detection by showing the better performance of deep learning models and provides insights for future research to build more robust detection systems.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF ILLUSTRATIONS

LIST OF ABBREVIATIONS

DL	Deep Learning
LR	Logistic Regression
LSTM	Long Short-Term Memory Networks
ML	Machine Learning
NLP	Natural Language Processing
RF	Random Forest
RoBERTa	Robustly Optimized BERT Approach
SVM	Support Vector Machine
TF-IDF	Term Frequency-Inverse Document Frequency
BoW	Bag of Words
GloVe	Global Vectors

CHAPTER I

INTRODUCTION

1.1 Research Background

The rise of social media has become the main way people connect globally. They have evolved into active areas of engagement that people use to express themselves, their stories, and valuable information, hence enhancing discussions globally. However, the rising major issue of cyberbullying prevails in this harmonious world of digital communication where twitter has become an epitome of online communication. One of the platform's basic premises, easy and fast exchanges, thus speeds up the dissemination of toxic content, affecting people and collectives profoundly.

Cyberbullying is a significant challenge for social media users, particularly on Twitter. This behavior is characterized by certain acts of aggression, such as racism, sexism, ageism, and employing negative attitudes toward people of other beliefs. Multi-class cyberbullying detection is important because it considers these distinctions in order to offer a broad solution that helps in combating online abuse. "Some of the studies, such as Chatzakou et al., (2019), have pointed out that the nature of cyberbullying is complex, hence the need for effective detection strategies. In addition, studies by Silva et al., (2020) and Fitra Rizki et al., (2021) emphasized the chimed, psychological that's of victims, which requires the latest means to identify and deter such conduct efficiently. In response to these complex problems, the latest approaches based on NLP and ML, as depicted by Mathpati et al., (2024) and Chow et al., (2023), are found to be potential solutions. These technologies can comprehend the nature of the interactions and the context that is something that cannot be necessarily done by an ordinary person, this is why the use of technologies can improve the detection and prevention of cyberbullying. The pervasive nature of this issue is evident in many

tweets, leading to significant mental and emotional distress among victims, as noted by (Chatzakou et al., 2019). The study by Fitra Rizki et al., (2021) has shown that cyberbullying on Twitter usually has a bad outcome increased feelings of loneliness, unease, despondency, diminished self-worth, and suicide. As a result, it has a lot of work to do to stop those activities and calls for the formulation of new approaches for content analysis that could discern the subtlety of social interactions at ease (Mathpati et al., 2024; Silva et al., 2020). The Chow et al., (2023) study provides a great insight into the dangers of cyberspace and highlights the possibility of its broader usage for safety onliners.

Natural Language Processing (NLP) rises as a potent tool against the anarchy of online environments. Interestingly, exploring digital reality reveals essential stuff, evidenced by the complicated communication making in the virtual zone. Advancing from just background processing to current sentiment classification, NLP offers greater precision than before in identifying online tumult on social media. Although this progress has been steady, interpretation of slang, irony, and sometimes dynamic language use is still present alongside the need to determine the intent of the messages and the harmful or benign nature of the content. Therefore, the constant improvement of NLP will be in need in helping to foster secure online environments (Tariq et al., 2023).

This study marks a transition that extends past scholarly circles, suggesting the emergence of a widespread social initiative. It explores refined NLP and Machine Learning (ML) methodologies, such as Random Forest (RF), Logistic Regression (LR), and Support Vector Machine (SVM), with a particular emphasis on their unique ability to analyse features that aid in identifying cyberbullying. In addition, the uniqueness of each dataset is also the means to the end that is, the performance of different classifiers (Muneer et al., 2020). On the contrary, Deep Learning (DL) models or RoBERTa and LSTM which possess capabilities like context considerations and semantic nuances only scripts require intensively looked explorations to see if their effectiveness will be maintained no matter the prevailing scenarios (Tan et al., 2022; Wang et al., 2021).

Furthermore, arguably the ambiguity of the internet is a disputable issue because it provides the opportunity for the free exchange of ideas but on the other hand contributes to the spread of cyberbullying around the world. It highlights the necessity for developing NLP and ML techniques based on the intricacies of online communication, it dares to suggest NLP and ML as alternate approaches to digital safety enhancement and improvement. To achieve this objective, both traditional and state-of-the-art techniques of computing is put into place to eliminate cyberbullying hence making the internet a safer place.

1.2 Research Significance

The multiclass cyberbullying detection on Twitter requires an examination with great depth. In time surrounded by a lot of digital cases, securing the platforms is a crucial thing to do for the psychic and emotional wellness of a particular user. The work of NLP and ML is seen going beyond their current limits here. They are used especially for online bullying, where they are made to identify and neutralize cyberbullying, and social health is reached through the utilization of technology.

Accurate classification of different types of cyberbullying is essential for implementing targeted interventions, which can help mitigate the various psychological impacts on victims. This research aims to improve detection precision, contributing to safer online environments. Understanding the nuances of different types of cyberbullying is crucial, as this study highlights the importance of differentiating among them. Advanced analytical models are necessary to focus on context, sentiment, and textual nuances. This approach promotes good netiquette, teaching people how to interact respectfully online and creating a space where free expression and safety coexist.

This study shows that NLP, along with ML and DL, are essential for solving cyberbullying problems on Twitter. Addressing this issue requires ongoing and effective technological innovations to keep virtual spaces safe from harmful activities. Therefore, this task extends beyond academia and holds significant importance for society as a whole. By exploring the latest methods and technologies, the study aims to create a digital world with improved security and greater empathy.

1.3 Problem Statement

The rise of social media platforms like Twitter (now called X) has led to an increase in cyberbullying, characterized by the use of slang, informal language, and contextually complex interactions. Traditional machine learning (ML) techniques have shown limitations in effectively detecting cyberbullying due to these linguistic complexities. Studies by Chatzakou et al., (2019) and (Silva et al., 2020) highlight the challenges ML models face in understanding the nuanced language of social media texts. Similarly, Dinakar et al., (2011) emphasize the need for improved methods to handle these complexities and enhance cyberbullying detection accuracy.

The effectiveness of traditional ML algorithms such as Random Forest, Logistic Regression, and Support Vector Machine (SVM) in cyberbullying detection is a critical focus. While feature engineering techniques, such as Term Frequency-Inverse Document Frequency (TFIDF) and word embeddings, have shown potential to improve model performance (Muneer et al., 2020), there remains a need to explore more advanced and effective featues. Tariq et al., (2023) suggest that modifications in feature engineering significantly impact the performance of classical algorithms, underscoring the importance of methodological advancements.

Advanced deep learning (DL) models, such as RoBERTa and Long Short-Term Memory (LSTM) networks, have been posited to outperform traditional ML models by capturing the contextual and semantic intricacies in textual data. Studies by Tan et al., (2022) and Wang et al., (2021) demonstrate that DL models can achieve higher accuracy in cyberbullying detection, particularly with appropriate model tuning and data representation techniques. However, the performance of these models is heavily dependent on the preprocessing methods and feature extraction algorithms employed.

Critical factors influencing the performance of cyberbullying detection systems include data preprocessing and feature extraction. Mozafari et al., (2019) and Menini et al., (2019) highlight the crucial role of preprocessing and feature representation in enhancing model accuracy. Their research indicates that decisions made at these initial stages can significantly influence the performance of both ML and DL models in detecting cyberbullying.

In addressing these challenges, this study aims to evaluate the impact of traditional ML models (Random Forest, Logistic Regression, SVM) and advanced DL models (LSTM, RoBERTa) on cyberbullying detection on Twitter. By investigating various feature engineering techniques and preprocessing methods, this research seeks to improve the accuracy and robustness of cyberbullying detection systems.

1.4 Research Questions

To address challenges stated in the problem statement, this study answer three research questions as follows:

1. **RQ1:** How effective are traditional ML algorithms (Random Forest, Logistic Regression, SVM) in classifying different forms of cyberbullying in text data when applying various feature engineering techniques?

2. **RQ2:** How does the preprocess data and feature representation selections impact the performance of ML and DL models in detecting cyberbullying on social media platforms?

3. **RQ3:** Can advanced DL models (LSTM, RoBERTa) surpass traditional algorithms in understanding text's contextual and semantic intricacies to improve classification accuracy?

Therefore, this study is regarded as a top priority because of the frequent mutation of online communication practices, and, of course, the safety of online communication becomes very important. The findings generated will expand both the academic knowledge, but as well, will make the translations of pieces of knowledge into real-life applications, one of the key steps towards effective cyberbullying solutions.

1.5 Objective Of Research

This research focuses on creating advanced methods for classifying text, evaluating traditional machine learning models and exploring the capabilities of state-of-the-art

deep learning frameworks for enhanced accuracy and deeper contextual understanding. Setting the following objectives will guide the development of a whole new approach.

1. Evaluate the efficacy of traditional ML algorithms (Random Forest, Logistic Regression, SVM) in classifying diverse forms of cyberbullying within text data, focusing on the role of innovative feature engineering techniques to enhance model accuracy and interpretability.

2. To assess the impact of pre-processed data and feature representation selections on the accuracy and effectiveness of ML models in cyberbullying detection, emphasizing the preprocessing techniques and the robustness of feature representation to enhance model performance.

3. To compare and investigate the potential of advanced DL models (LSTM, RoBERTa) to surpass traditional algorithms with machine learning algorithms based on different feature representations that are used to contextual and semantic intricacies of text.

1.6 Research Scope

The research is all about multiclass cyberbullying detection on Twitter, and its contexts have delimited accordingly. This study uses "Cyberbullying Classification" dataset Jason Wang, (2020), which includes cyberbullying classes which are religion, age, gender, ethnicity and other violating classes which are other cyberbullying not considered as cyberbullying. These models were chosen for their proven efficacy in handling complex, multi-class text classification tasks, which is essential for accurately identifying various forms of cyberbullying on Twitter. Their ability to capture nuanced language patterns across different cyberbullying categories enhances the detection precision. This research uses machine learning algorithms and neural networks like Random Forest RF, Logistic Regression LR and Support Vector Machine SVM, as well as LSTM and RoBERTa models, to discover all possible factors of feature extraction and classification. Evaluation metrics like accuracy, precision, recall, and F1-score are employed in determining the performance of these models.

1.7 Research Methodology

To effectively address the research objectives and tackle the challenges of detecting cyberbullying on Twitter, this study employs a structured five-phase methodology framework, illustrated in Figure 1.1. This comprehensive approach is designed to thoroughly explore the domain, implement innovative solutions, and rigorously evaluate the outcomes. The methodology is divided into the following phases:

1. **Data Collection:** In the first of the methodology stages the main goal is collecting the data that as semantically rich and as complete as it can be and that would represent various forms of cyberbullying. Having a rich resource of Kaggle dataset dealing with the identification of patterns in texts connected to cyberbullying through the text classification approach, the process of classifying texts in the given context will be significant. The research can be methodical and labelled across different variances of harassment like religious, gender, age, and ethnicity. It reveals what type of cyberbullying are, the different methods people employ to cyberbully someone, and the devastating effects of cyberbullying.

2. **Pre-processing:** In this phase, during the process of data preparation and cleansing data, the pre-processing of data, to be able to adapt to the model's design. This method removes irrelevant elements from the data, such as external interference. But it also does address the data void to mend breakdown and has the ability to come up with an output that is error free.

3. **Feature Engineering:** This phase involves extracting and selecting features from the pre-processing phase that are crucial for effective cyberbullying detection. The selection of features is based on their relevance to the patterns and indicators of cyberbullying behaviours in the dataset.

4. **Model Application:** After the set of features is built, this phase applies ML models (such as Random Forest, Logistic Regression, and SVM) and after that advanced DL models (like LSTM and RoBERTa) to the processed.

5. **Evaluation:** The final phase involves a thorough evaluation of all models to assess their effectiveness in detecting various forms of cyberbullying. This includes using a set of evaluative metrics to compare the performance of the developed models against existing methodologies, highlighting improvements in accuracy, adaptability, and overall effectiveness.

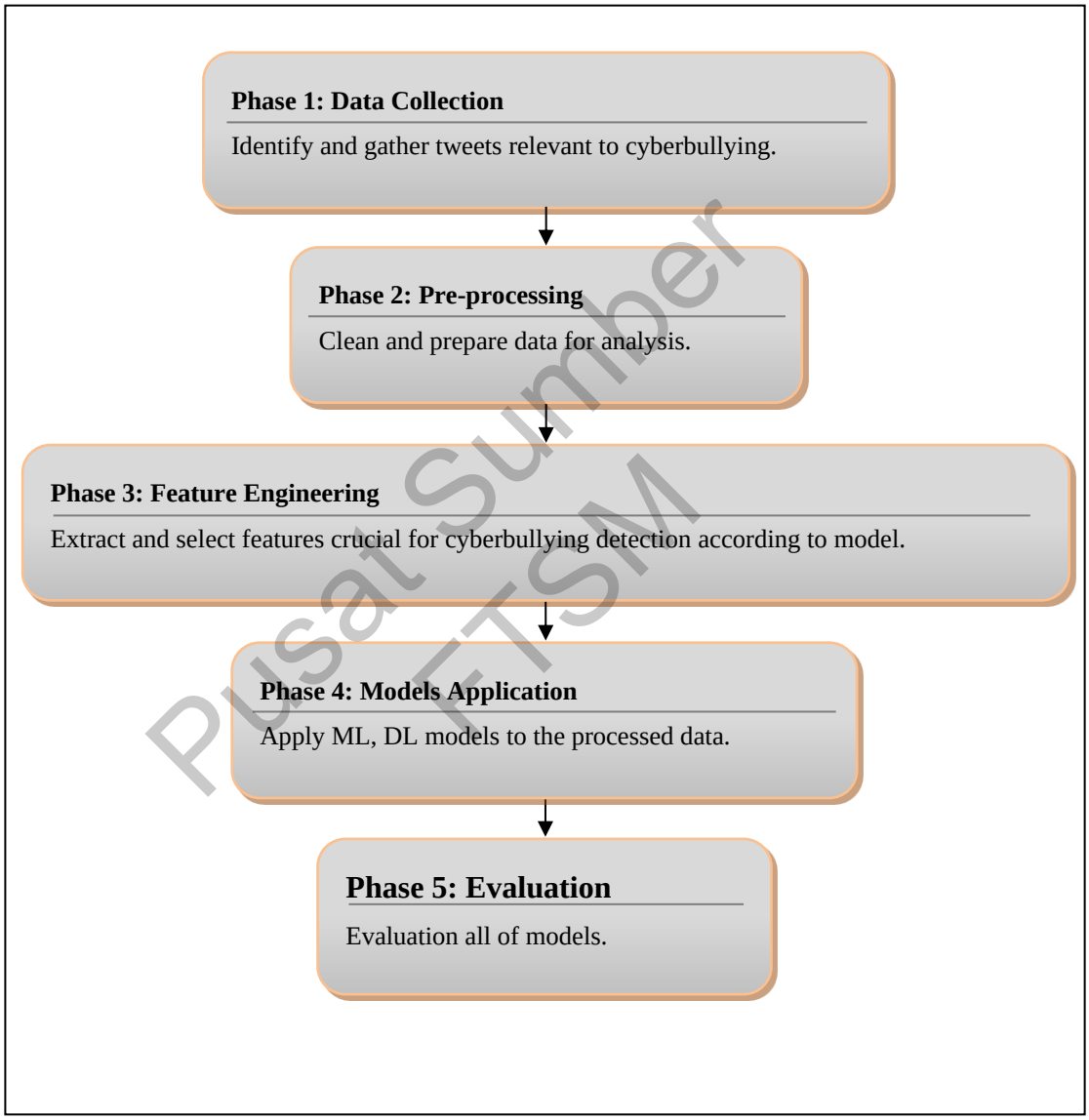

Figure 1.1 Phases of Research Design

Figure 1.1 Illustrates the structured approach from data collection through model application to the evaluation phase, highlighting the integration of ML and DL techniques in detecting cyberbullying on Twitter. The illustration representation provides a clear study's design, enhancing comprehension of the sequential and interconnected nature of the research phases.

On these grounds, the research plans to make a significant contribution to cyberbullying detection on Twitter. Each stage is vital covering the study process from the base understanding of the problem area to providing innovative solutions and their critical evaluation.

1.8 Thesis Organization

This chapter delineates the structure and organization of the thesis, guiding the reader through each subsequent chapter. Each section is designed to build upon the previous one, providing a comprehensive exploration of cyberbullying detection using NLP, ML, and DL methodologies. The narrative unfolds as follows:

Chapter 2: Literature Review - This chapter provides an in-depth summary of the academic research cyberbullying detection. In addition to this, it explores the applications of NLP, machine learning (ML), and deep learning (DL) techniques, in the area. It examines previous studies, highlighting key findings, applied methodologies, and the development of machine learning for detecting cyberbullying. The literature review data-intensive techniques, such as algorithms and feature extraction, and has been used to categorize cyberbullying based on its properties. Moreover, it reviews the pros and cons of emerging technologies and foresees their application in areas including reliability and accuracy of detection, as well as the capability to adapt to changing conditions.

Chapter 3: Methodology - Here in this chapter identifies the comprehensive methodology pursued by this study. This forms the initial step where the types of data sources to be used and the criteria of inclusion are explored. That is preceded by a through participation of the preprocessing techs used to clean the data and make it ready for analysis, which comes in place for good results. The chapter describes the feature engineering process that is applied in the discovery of the features that best correlate with cyberbullying detection. Finally, it covers building model stage during which develop both conventional ML algorithms and advanced DL models and tune them for the cyberbullying detection task.

Chapter 4: Experiments and Results for ML - This chapter concentrates on the utilization of common ML techniques, like Random Forest, Logistic Regression, and SVM. This section describes the experimental setup, which includes training and testing processes, and presents the results achieved with these models. The chapter appraises the models by metrics which include accuracy, precision, recall, and F1-score. It also studies the influence of different feature engineering methods on model performance and explains the meaning of the findings with reference to cyberbullying detection.

Chapter 5: Experiments and Results for DL - This chapter concentrates on the utilization of common DL techniques, like LSTM and RoBERTa. This section describes the experimental setup, which includes training and testing processes, and presents the results achieved with these models. The chapter appraises the models by metrics which include accuracy, precision, recall, and F1-score. It also studies the influence of different feature engineering methods on model performance and explains the meaning of the findings with reference to cyberbullying detection.

Chapter 6: Conclusion and Future Work - The conclusive chapter represents a summary comprehension of all observations, and the insights attained during the entire duration of the research. It presents the main findings, interprets them in the context of daily life, and evaluates what the study has done regarding the field of cyberbullying prevention. The level ends with recommendations for future research directions, such as the search for algorithmic improvements, better data collection methods and features engineering techniques. Besides this, it shows how this evidence could be implemented into practice domain in the form of systems against cyberbullying and its prevalence reduction on social networks.

CHAPTER II

LITERATURE REVIEW

2.1 Introduction

This chapter provides a comprehensive review of the literature related to cyberbullying classification using both traditional machine learning and advanced deep learning approaches. The structure of this chapter is meticulously designed to cover various aspects crucial to understanding and advancing cyberbullying detection methods. Section 2.2 introduces the concept of cyberbullying, explaining its definition and impact on victims. Section 2.3 delves into NLP techniques that are fundamental for analysing textual data in cyberbullying detection. Section 2.4 presents an overview of machine learning approaches, detailing different algorithms and their applications in this domain. Section 2.5 focuses on deep learning approaches, including transformer models, LSTM networks, and the RoBERTa layer, providing insights into their architectures and advantages. Section 2.6 reviews related work, comparing studies that use the same dataset and those that use different datasets to highlight various methodologies and their effectiveness. Finally, Section 2.7 summarizes the key findings and sets the stage for the subsequent chapters by synthesizing the critical points discussed in the literature.

2.2 Introduction To Cyberbullying

2.2.1 Definition And Impact

Cyberbullying is a form of bullying or harassment using electronic means, typically on social-media platforms. It can have severe psychological effects on victims, including depression, anxiety, and in extreme cases, suicidal thoughts (Patchin et al., 2010). The detection of cyberbullying is crucial for creating a safer online environment.

Cyberbullying, as defined by Albikawi, (2023), involves the use of digital technologies like social media and mobile devices to harass individuals, manifesting through offensive messages, harmful posts, and privacy violations. This prevalent issue, particularly among young people, on platforms such as Facebook and Instagram, leads to severe psychological effects, including depression and social isolation.

Dennehy et al., (2020) indicate that girls are more susceptible to cyberbullying than boys and experience greater trauma, with adolescents and teenagers being the most frequent victims across various countries based on internet usage and social-media penetration.

2.2.2 Challenges In Detection

Detecting cyberbullying poses several challenges, including the dynamic and informal nature of online language, the use of slang, abbreviations, and emojis, and the contextual understanding required to identify bullying intent (Dinakar et al., 2012). Additionally, the imbalanced nature of datasets, where non-bullying instances far outnumber bullying instances, complicates the detection process (Van Hee et al., 2018).

2.3 Natural Language Processing Techniques In Cyberbullying

Natural Language Processing (NLP) enables computers to understand and generate human language, facilitating communication between humans and machines (Basha et al., 2023). This includes functions such as speech recognition, natural language understanding and natural language generation as well as machine translation and sentiment analysis (Clark et al., 2013).

NLP plays a crucial role in the detection and prevention of cyberbullying on social media platforms; by analysing text data, NLP techniques can identify harmful and abusive language that constitutes cyberbullying (Afrifa et al., 2022). This process involves pre-processing the text to reduce noise, extracting relevant features, and applying machine learning models to classify the data accurately, NLP enables the automated detection of offensive terms and patterns associated with cyberbullying, facilitating early intervention and helping to mitigate its psychological and emotional

impacts on victims (Rahman, 2022). This technology is vital for creating safer online environments.

2.4 Machine Learning Approaches for Cyberbullying Detection

2.4.1 Feature Extraction Techniques

The use of social media including Twitter, Facebook, WhatsApp, and Instagram among has been on the increase for some past years (Krithika et al., 2020). Many messages are exchanged on these platforms, including hidden content and insulting remarks. Data used in cyberbullying can come in various formats, like text, images, and videos. Each dataset has variables known as features, though some datasets are more complex. These features are vital for analysing, predicting, and classifying data. It's clear that the features used to train models greatly impact the accuracy of any machine learning algorithm; As datasets grow and include more features, predicting based on specific features becomes increasingly difficult (Krithika et al., 2020). The quality of a dataset can be enhanced by optimizing its characteristics. This is why feature extraction is so important—it helps simplify datasets by reducing the number of features. These methods play a key role in making machine learning algorithms more efficient at detecting cyberbullying (Krithika et al., 2020).

It is important to note that using text-based features significantly improves the performance of classifiers in detecting cyberbullying, compared to using non-text-based features like images and network graphs (Goodboy et al., 2015). Different types of data offer various features that help predict cyberbullying. These features are generally categorized into content features, user features, sentiment features, and network features. (Masnizah Mohd, 2018). The methods used to extract features from a dataset depend on the type of data involved. For example, basic content features might include profanity, negativity, and subtlety as suggested by (Medhat et al., 2014). Negativity and profanity are prevalent in most cyberbullying cases (Dinakar et al., 2012). Special features can be used to identify specific labels like sexuality, intelligence, and race. As mentioned earlier, most cyberbullying is linked to text data, no matter the social media platform. This text data often includes negative sentiments, obscenities, and content related to race, physical appearance, and religion (Medhat et al., 2014).

Textual features help improve the evaluation of cyberbullying content by analysing how often improper words, special characters like question marks and exclamation points, uppercase words, smileys/emoticons, and parts of speech appear (Medhat et al., 2014). A set of indicators was used to identify cyberbullying in YouTube comments (Raisi et al., 2016). This means that for any social media site, regardless of the content posted, these can be grouped into media sessions, as explained by (Masnizah Mohd, 2018). The media session includes features like cyber aggression, profanity, network graphs, images, and linguistic elements, as described by (Hosseinmardi et al., 2015). Among the sources of data used, including Twitter, YouTube, Facebook, Instagram, Formspring, and Ask.fm, the most commonly used feature classification was content-based features (Masnizah Mohd, 2018). Sentiment-based features are the second most frequently used for identifying cyberbullying, whereas network-based features are the least utilized (Masnizah Mohd, 2018).

Feature representation is a fundamental step in NLP that involves converting text data into numerical representations that can be used by machine learning algorithms. Common techniques include:

1. **TF-IDF:** is a method that combines term frequency and inverse document frequency to evaluate the importance of a term in a dataset (Cheng et al., 2019). It uses word statistics to extract text features and vectorizes the input (Gada et al., 2021). The model focuses on common word expressions across all texts (Muneer et al., 2020). TF-IDF is a popular method in text detection, creating vectors from n-grams, words, and characters.

    a. **TF-IDF Word**: Represents the TF-IDF scores of words in a matrix.

    b. **TF-IDF N-gram**: Represents the TF-IDF scores of n-grams—combinations of (n) words— in a matrix.

    c. **TF-IDF Char**: Represents the TF-IDF scores of character-level n-grams in a matrix.

Instead of just counting word occurrences, TF-IDF weighs words by their relative frequency to avoid exaggerating common words. It highlights when a word

occurs more often in a particular statement than across the entire text corpus. This feature is useful for detecting cyberbullying (Muneer et al., 2020). The TF-IDF vocabulary is created during model training and then used again for test predictions, proving to be an effective method for text classification (Muneer et al., 2020).

The TF-IDF weight for a term in a document is mathematically represented as follows:

$$W(d,t) = TF(d,t) * log\left(\frac{N}{df(t)}\right) \quad (2.1)$$

Here, N is the number of documents, df(t) is the number of documents in the dataset containing the word t. The term TF(d,t) enhances the recall, whereas the second term enhances the word embedding accuracy (Muneer et al., 2020).

2. **Bag of Words:** The BoW method has been widely used in the field of text classification, feature representation selections, and image recognition. It represents texts as a collection of word frequencies, disregarding grammar and word order. The method simplifies documents into vectors of word occurrences, creating a histogram for each document, represents text as vectors of word frequency, capturing term frequency but not context (Qader Wisam A. et al., 2019).

3. **Word Embeddings:** Captures semantic relationships between words using models like Word2Vec, GloVe, or FastText.

    a. **Word2Vec**: The Word2Vec model heralds a substantial leap forward in NLP by transmuting words into continuous vector representations, thus empowering machines to handle text data with enhanced efficacy. Conceived by Mikolov, Chen, et al., (2013) the model bifurcates into two principal approaches: Continuous Bag of Words (CBOW) and Skip-Gram. CBOW anticipates a target word from its surrounding context, while Skip-Gram predicts the context words given a target word. These methods employ dense vector representations, preserving the intricate semantic and syntactic relationships among words. The model's efficiency is further augmented through hierarchical SoftMax and

negative sampling, which optimize the computational load by minimizing the calculations required for each word's representation (Mikolov, Chen, et al., 2013). Word2Vec has exhibited its versatility across a myriad of applications, encompassing text classification, syntactic and semantic analysis, and analogy resolution (Mikolov, Chen, et al., 2013).

Its capability to maintain contextual integrity and yield meaningful vector space mappings positions it as superior to traditional methods such as (BOW) and (TF-IDF), often falter in capturing profound semantic nuances. The adaptability and robust performance of Word2Vec in managing extensive datasets with agility underscore its pivotal role in word embeddings and natural language comprehension (Mikolov, Sutskever, et al., 2013).

b. **GloVe:** Global Vectors for Word Representation (GloVe) is an unsupervised learning algorithm used to derive word embeddings from a text corpus (Pennington et al., 2014; Raj et al., 2021). The technique involves constructing a co-occurrence matrix, which captures the frequency with which pairs of words appear together in a specified context window. This co-occurrence matrix helps to analyze the semantic relationships between terms, as words that appear in similar contexts tend to have similar meanings. For example, the words "queen" and "king" or "mother" and "woman" will have high cosine similarity in their vector representations, indicating a close semantic relationship. GloVe learns these representations from a large corpus such as Wikipedia and Gigaword unsupervised.

The core of the GloVe model is the objective function, which ensures that the dot product of the word vectors reflects the logarithm of the probability of the words co-occurring. For a word i with its vector representation wi, the objective function can be expressed as:

$$f(wi - wj, wk) = Pik / Pjk \quad (2.2)$$

where:

- w_i and w_j are the word vectors of words *i* and *j*,
- w_k is the context word vector,
- P_{ik} is the probability of word *i* co-occurring with context word *k*,
- P_{jk} is the probability of word *j* co-occurring with context word *k*,

GloVe uses the co-occurrence statistics from a large corpus to generate word vectors that capture meaningful semantic relationships, enabling various NLP tasks to leverage these rich word representations.

c. **FastText:** FastText, created by Facebook's AI Research (FAIR) lab, is a powerful and efficient tool for text classification and representation. Unlike traditional models that rely solely on whole words, FastText uses character n-grams to capture *subword* information, enhancing its ability to manage rare and out-of-vocabulary words. This results in robust word embeddings and accurate text classification even with limited data. Its efficiency makes it ideal for large-scale datasets and real-time applications, ensuring swift and reliable text processing (Joulin et al., 2016).

FastText is based on a skip-gram model and excels by considering the morphology of words, breaking them down into character n-grams. For example, the word "language" with n=3 would be divided into n-grams like 'lan', 'ang', 'ngu', 'gua', and so on. This technique allows FastText to understand the context of unknown words by decomposing them into smaller segments and matching these with known patterns from its training data (Raj et al., 2021).

4. **N-grams**: Considers sequences of 'n' words together to capture context. Recent research has highlighted the significance of N-grams, including unigrams, bigrams, trigrams, and fourgrams, in NLP tasks. Khadka, (2022) study on predicting Nepali words using N-gram models demonstrated that higher-order N-grams, such as trigrams and fourgrams, significantly enhance word prediction accuracy. This research utilized a Viterbi algorithm for decoding and found that fourgrams achieved the highest accuracy compared to bigrams and trigrams, suggesting that longer N-

gram sequences can capture more contextual information, leading to better predictive performance.

Similarly, Esther Trueman et al., (2022) applied N-grams in combination with BERT for sentiment classification of movie reviews. Their study indicated that integrating unigrams, bigrams, and trigrams within the BERT model substantially improved the classification accuracy. The researchers found that the combination of bigrams and trigrams yielded the highest accuracy, outperforming other models that utilized only single or lower-order N-grams. These findings underscore the effectiveness of using multiple N-gram levels to improve the performance of machine learning models in NLP tasks.

5. **Hashing Trick:** Converts text into a fixed-size vector using a hashing function. The feature hashing trick, also known as the hashing trick, is a crucial technique for converting text data into fixed-size vectors using a hashing function. This method is particularly effective for handling high-dimensional text data by reducing its dimensionality while preserving the sparsity of the data. In their study, Weinberger et al. demonstrated the feasibility of this approach, showing that it not only significantly reduces memory usage but also maintains high performance levels in multitask learning scenarios (Weinberger et al., 2009).

6. **Character Encoding:** Uses sequences of characters instead of words to capture more granular information. Character encoding is a fundamental aspect of text processing that involves converting characters into a standardized digital format. This process allows computers to store and manipulate text efficiently, facilitating communication across different systems and platforms. The Unicode Standard, for example, provides a comprehensive encoding scheme that supports a vast array of characters from various languages and scripts, ensuring consistency and compatibility in digital text representation. Understanding character encoding is essential for data curators and digital preservationists to manage and preserve textual data accurately, as outlined in (Erickson, 2021).

Zheng et al., (2019) conducted a comprehensive comparison of various feature representation selections methods for text classification, highlighting the importance of integrating multiple features to improve classification accuracy and generalization. Their study confirmed that combining different types of features can effectively mitigate the limitations of individual features, thus enhancing the overall performance of text classification models.

2.4.2 Classification Algorithms

Machine learning, a field that utilizes algorithms to analyse data and make decisions based on learning, encompasses various methods such as Naive Bayes, Support Vector Machine (SVM), K-Nearest Neighbour (KNN), and more for tasks like sentiment analysis (Tan et al., 2022). Supervised machine learning involves training classifiers from labelled samples, as seen in sentiment analysis and sarcasm detection tasks (Mohammad et al., 2019; Yazhou Zhang et al., 2023). Lexicon-based approaches in sentiment analysis utilize resources like SentiWordNet and statistical methods such as PMI and Chi-Square for sentiment word analysis (Torfi et al., 2020). In machine translation, traditional methods have evolved into Neural Machine Translation (NMT) based on neural networks, eliminating the need for extensive preprocessing and focusing on network structure for improved translation accuracy (Chow et al., 2023). NLP and Machine Learning are closely intertwined, showcasing the diverse applications and advancements within the machine learning domain.

Machine learning has become essential for detecting cyberbullying on social media platforms. Researchers have developed models that effectively identify bullying language patterns using features like TF-IDF, Word2Vec, and sentiment analysis by utilizing various datasets, including those from Twitter and other social networks. Classifiers such as Random Forest, Support Vector Machines, and Neural Networks have shown high accuracy and reliability in identifying cyberbullying content, demonstrating the potential of machine learning in mitigating online harassment (Alam et al., 2021; Alqahtani et al., 2024).

1. **Random Forest (RF)**

Random forest is a versatile and widely used machine learning algorithm that constructs an ensemble of decision trees to achieve more accurate and stable predictions. It is applicable for classification and regression tasks, making it a valuable tool for various machine learning applications. The bagging method combines multiple decision trees, each built from random subsets of features and data, to prevent overfitting and enhance performance. This algorithm excels in scenarios where interpretability and quick development are essential, offering easy-to-understand hyperparameters and a straightforward approach to measuring feature importance. Despite its slower prediction time when using numerous trees, the random forest remains a robust and reliable choice for many predictive modelling tasks (Dinakar et al., 2012; Van Hee et al., 2018).

2. **Logistic Regression (LR)**

Logistic regression is a supervised machine learning algorithm primarily used for binary classification tasks. It predicts the probability that an instance belongs to a particular class by analyzing the relationship between independent variables and the dependent binary variable using the sigmoid function, which maps predicted values to probabilities between 0 and 1. The logistic regression model fits an "S" shaped logistic function rather than a linear regression line, making it particularly suitable for classification problems. Depending on the type of dependent variable, logistic regression can be classified into binomial, multinomial, or ordinal types. The model's coefficients, estimated through maximum likelihood estimation, indicate the relationship between the independent variables and the log odds of the dependent variable. Despite its simplicity, logistic regression is a powerful tool for classification tasks and is valued for its interpretability and quick classification capabilities (Zaidi et al., 2023).

3. Support Vector Machine (SVM)

Support Vector Machine (SVM) is a supervised machine learning algorithm primarily used for text classification and data classification, as highlighted in various studies (Muneer et al., 2020) (Wibowo et al., 2018). SVM transforms the original feature space into a user-defined, kernel-based higher-dimensional space. A linear decision surface is constructed within this space, a linear decision surface is constructed, which helps generalize the network (Cortes et al., 1995). Conceptual challenges include finding a hyperplane that effectively generalizes data points, while technical issues involve computing high-dimensional spaces, such as constructing hyperplanes in billion-dimensional spaces for complex datasets (Cortes et al., 1995). Optimal hyperplanes, which maximize the margin between two classes, address the conceptual issue. The ratio of support vectors to training vectors indicates the generalization ability, even in infinite-dimensional spaces, with practical examples showing ratios as low as 0.03 (Cortes et al., 1995). To solve technical challenges, Cortes et al. proposed a support-vector network that rearranges the sequence of operations, allowing for efficient construction of decision surfaces (Cortes et al., 1995). SVMs are effective in binary classification, including cyberbullying detection on platforms like Twitter (Muneer et al., 2020; Wibowo et al., 2018), though they can struggle with large datasets due to language vagueness. Features of Support Vector Networks include efficient decision rule construction, universal applicability, and control over generalization factors. For non-linear data separability, kernels are used, and soft margins help manage classification errors. Figure 2.1 illustrates an SVM classifier with two features, highlighting support vectors and misclassified samples. SVMs' efficiency in binary classification is well-documented, though challenges remain as data scales (Muneer et al., 2020; Wibowo et al., 2018).

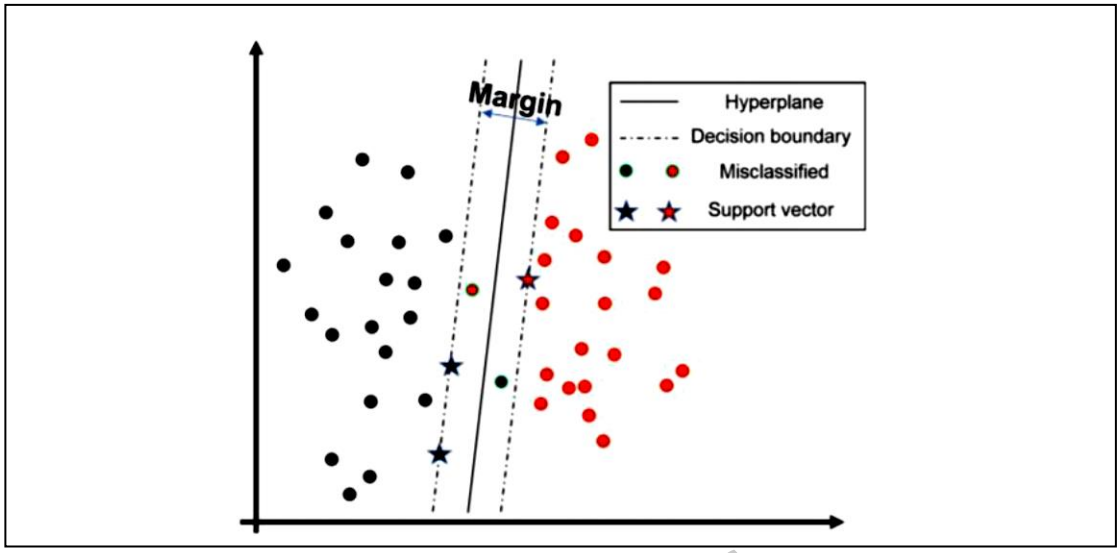

Figure 2.1 SVM Classifier (Muneer et al., 2020)

2.5 Deep Learning Approaches

2.5.1 Transformer Models

The Transformer model, introduced by (Vaswani et al., 2017) revolutionized NLP by addressing the limitations of traditional sequence-to-sequence models such as RNNs and LSTMs. Unlike these predecessors, the Transformer relies entirely on self-attention mechanisms, enabling it to process sequences in parallel rather than sequentially. This architectural shift significantly reduces training times and enhances performance on tasks involving long-range dependencies, making it a cornerstone in developing advanced language models like BERT and GPT. Rahali et al., (2023) provides an in-depth overview of various Transformer architectures, their training methods, and their advantages over traditional models in handling long-range dependencies and parallel processing.

2.5.2 Long Short-term Memory (LSTM)

LSTM, a specialized recurrent neural network (RNN) architecture, is designed to handle long-range dependencies (Stamp, 2020). As depicted in Figure 2.2 (Stamp, 2020), each state in the LSTM architecture has two lines entering and exiting: one represents the hidden state, while the other functions as a gradient during backpropagation. The key distinction between LSTM and a standard RNN is that LSTM includes feedback connections, providing two transmission states (Hochreiter et al., 1997).

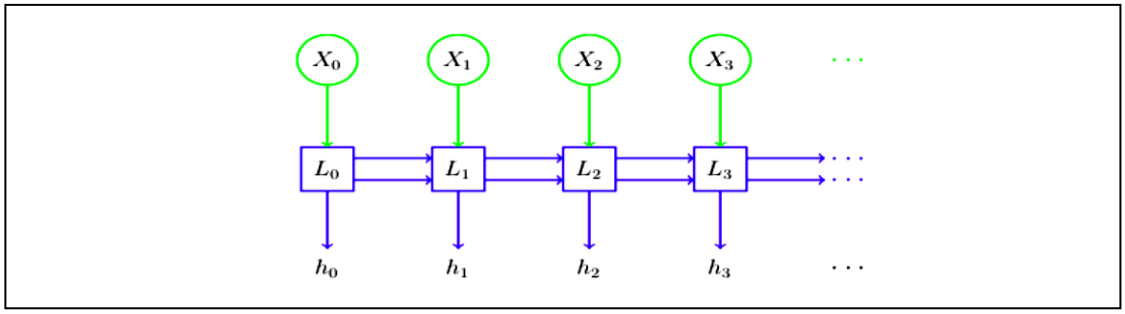

Figure 2.2 Example: LSTM (Stamp, 2020)

Figure 2.2 illustrates a LSTM network, a type of recurrent neural network (RNN) used in deep learning to address long-term dependencies and the vanishing gradient problem. It includes input nodes (Xt) for each time step, LSTM cells (Lt) that process inputs and maintain memory, and hidden states (ht) that serve as both outputs and inputs for subsequent steps. Green arrows indicate input flow into the cells, while blue arrows show hidden state transitions. LSTM cells have internal gates (input, forget, and output) that regulate data flow and enable the network to effectively learn long-term dependencies.

The LSTM architecture comprises cells, gates, and information flow. As illustrated in Figure 2.3 (Stamp, 2020), each cell in the LSTM structure includes an input gate, forget gate, output gate, and intermediate gate. Additionally, it employs sigmoid and hyperbolic tangent functions to compute gate vectors and serve as activation functions. The cells analyze dependencies among elements in the input sequence, while the gates regulate the flow of information into and out of the cell. During training, the weights of the connections between cells are calculated.

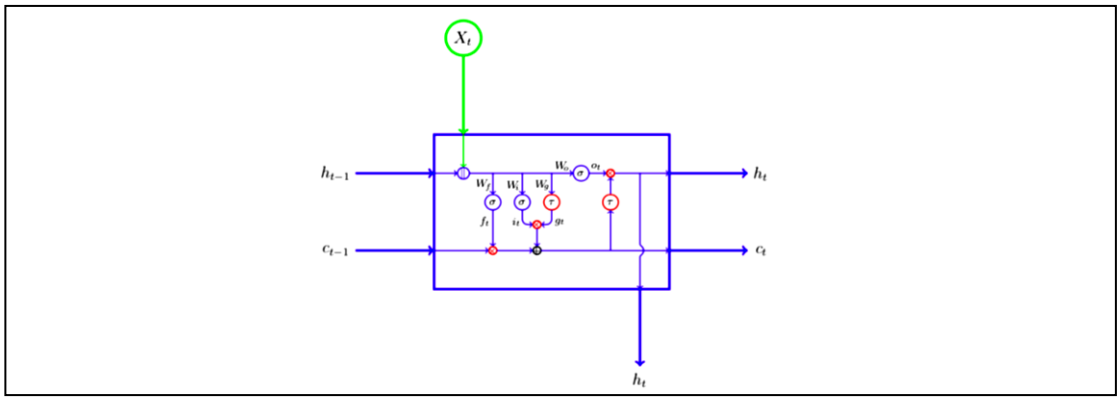

Figure 2.3 Example: One timestamp of an LSTM (Stamp, 2020)

Figure 2.3 depicts the internal workings of an LSTM cell at a single time step, showcasing how it processes inputs and manages its state through various gates. The cell receives the current input X_t and the previous hidden state h_{t-1}, and it retains information from the previous cell state c_{t-1}. The forget gate f_t determines what to discard from c_{t-1}, the input gate *it* decides what new information to add, and the output gate *ot* selects the part of the cell state c_t to output as the new hidden state h_t. The weights for each gate (W_f,W_i,W_c,W_o,W_f) are learned during training, enabling the LSTM to effectively update its cell state and capture temporal dependencies in sequential data.

LSTM, a highly successful learning model, adeptly handles both individual data points and entire data series. It has been employed in diverse domains to address various research challenges, such as classification and prediction problems. LSTM's applications span areas like sentiment analysis, speech recognition, and handwriting recognition (Hochreiter et al., 1997). Moreover, it is integral to popular products like Google Translate, Apple's Siri, and Amazon Alexa (Stamp, 2020).

2.5.3 Roberta Layer

RoBERTa, present Liu et al., (2019) is an advanced version of the BERT (Bidirectional Encoder Representations from Transformers) model developed by Facebook AI. It aims to address several limitations identified in the original BERT model by substantially improving pretraining methods. RoBERTa enhances BERT by training the model for longer durations, using larger batches of data, and on more extensive datasets. It removes the next sentence prediction objective and dynamically changes the masking patterns applied to the training data. These modifications result in significant performance improvements on various natural language understanding tasks. RoBERTa sets new state-of-the-art results on benchmarks like GLUE, RACE, and SQuAD, demonstrating its robustness and efficacy in handling complex language tasks.

RoBERTa endeavors to learn a universal language representation by pre-training on extensive unlabeled text data, subsequently fine-tuning for various downstream NLP tasks (Xu et al., 2023). Figure 2.4 illustrates the architecture of the RoBERTa model.

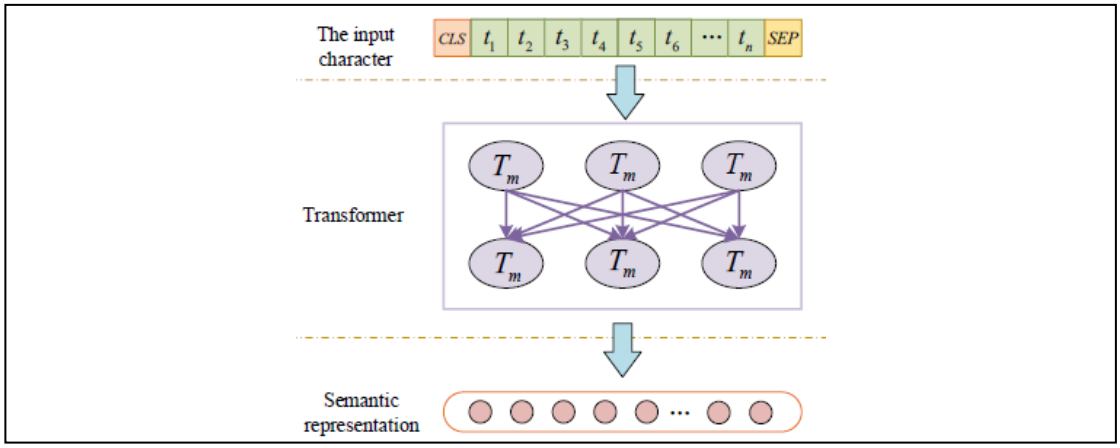

Figure 2.4 Schematic diagram of the structure of the RoBERTa (Xu et al., 2023)

Figure 2.4 depicts the architecture of a Transformer model, illustrating how it converts input characters into a semantic representation. The input sequence, including special tokens CLS and SEP, is fed into the Transformer, which consists of multiple modules (Tm). These modules use self-attention mechanisms and feed-forward networks to process the sequence, capturing relationships and dependencies between tokens. The output is a semantic representation, a vector or set of vectors that encapsulate the meaning of the input, useful for tasks like classification, translation, or summarization. The Transformer's strength lies in its ability to manage long-range dependencies and parallel processing, making it ideal for various NLP tasks.

2.6 Related Work

Several studies have compared different ML and DL approaches for the same dataset cyberbullying detection:

In recent years, detecting cyberbullying on social media has garnered significant attention from researchers, resulting in various approaches leveraging machine learning (ML) and deep learning (DL) techniques. This section discusses the contributions of four key studies that have advanced the field of cyberbullying detection through different methodologies and models.

Jason Wang, (2020) introduced the "Cyberbullying Classification" dataset on Kaggle, which analyzes cyberbullying trends across various social media platforms. During the COVID-19 pandemic.

2.6.1 Related Work On The Same Dataset

Jason Wang, (2020) present a robust framework for the detection and classification of cyberbullying on Twitter. The authors address the issue of class imbalance by employing a modified Dynamic Query Expansion (DQE) process to augment six existing Twitter cyberbullying datasets, resulting in a balanced dataset of 48,000 tweets categorized into Not Cyberbullying, Age, Ethnicity, Gender, Religion, and Other. Their methodology involves constructing a textual graph based on cosine similarities between tweet embeddings and leveraging a Graph Convolutional Network (GCN) classifier named SOSNet. Various embedding methods were tested, including BOW, TF-IDF, word2vec, GloVe, FastText, BERT, DistilBERT, and SBERT, alongside classifiers such as Logistic Regression, Naïve Bayes, KNN, SVM, XGBoost, MLP, and SOSNet. Notably, the combination of BOW with XGBoost achieved the highest accuracy and F1 score for the full dataset (Accuracy: 94.38%, F1 Score: 94.44%), while SBERT with SOSNet produced the best results on a downsized dataset of 4,000 tweets (Accuracy: 92.70%, F1 Score: 92.58%). The study demonstrates the efficacy of simple embedding methods with robust classifiers, as well as the potential of advanced embeddings and graph-based learning for cyberbullying detection.

In Mathur et al., (2023) the best results were achieved using the Random Forest classifier with 50 estimators and the Gini Index for information gain, combined with Count-Vectorizer and TF-IDF during preprocessing. The image captioning model was utilized to create descriptions for images posted on the account, which were then compared with user-written captions to identify and filter out spam tweets. This configuration yielded the highest accuracy of 94.06%, precision of 94.01%, and recall of 94.24%, outperforming other classifiers such as AdaBoost and Gradient Boosting. These results highlight the effectiveness of the Random Forest model, especially when fine-tuned with appropriate hyperparameters and preprocessing techniques.

In Alqahtani et al., (2024) the best results were achieved using ensemble methods combining Random Forest (RF), Decision Tree (DT), and XGBoost classifiers. The Stacking Classifier, which integrates these models, achieved the highest accuracy of 90.71%, with a precision of 90.85%, recall of 90.60%, and an F1 score of 90.63%.

The Voting Classifier also performed well, with an accuracy of 90.41%, precision of 90.69%, recall of 90.36%, and an F1 score of 90.45%. These results highlight the effectiveness of ensemble methods over traditional single classifiers in detecting cyberbullying on Twitter, especially when using TF-IDF with bigrams for feature extraction.

In Mahmud et al., (2022) the best results were achieved using the LightGBM classifier, outperforming other models such as XGBoost, Logistic Regression, Random Forest, and AdaBoost. LightGBM achieved the highest performance metrics with an accuracy of 85.5%, precision of 84.0%, recall of 85.0%, and an F1 score of 84.49%. The dataset used consisted of approximately 47,692 tweets categorized into six classes: age, gender, religion, ethnicity, other types of cyberbullying, and not cyberbullying. The study highlighted the effectiveness of LightGBM in detecting cyberbullying tweets, especially with text-based feature extraction using TF-IDF.

In Singh et al., (2022) the best results were achieved using the Gated Recurrent Unit (GRU) model, which outperformed other models such as Naive Bayes, Logistic Regression, SVM, Random Forest, XGBoost, and LSTM. The GRU model achieved the highest performance metrics with an accuracy of 92%, precision of 92%, recall of 92%, and an F1 score of 92%. The dataset used consisted of approximately 47,694 tweets categorized into six classes: age, gender, religion, ethnicity, other types of cyberbullying, and not cyberbullying. This study highlights the effectiveness of deep learning models, particularly GRU, in accurately detecting various forms of cyberbullying on social media platforms.

The feature representation selection process is a critical component of this study, ensuring that the models are provided with the most informative and relevant data. Features such as TF-IDF and word embeddings were chosen for their proven effectiveness in previous research. As outlined in Table 2.1, these features have consistently demonstrated their ability to enhance text classification tasks by capturing both frequency-based and semantic information. The use of these features is particularly relevant for the informal and slang-heavy language of tweets, as they can capture the nuances and context-dependent variations necessary for accurate cyberbullying

detection. Empirical support from studies like Chatzakou et al., (2019) and Silva et al., (2020) further validates this choice, ensuring a robust and reliable foundation for this research.

Table 2.1 Analysis Same Dataset of Cyberbullying Detection Models

Author/ Year	Dataset	Features	Model/ Classifier	Results
(Jason Wang, 2020)	Twitter/ 4,000 tweets (10% of the total dataset was utilized by this the study)	BOW, TF-IDF Textual Features, Semantic Similarity, Tweet Embeddings	Graph Convolutional Network (GCN), BOW+XGBoost, TF-IDF+XGBoost, SBERT+SOSNet	Accuracy: 94.38% (BOW+XGBoost), 92.70% (SBERT+SOSNet); F1-Score: 94.44% (BOW+XGBoost), 92.58% (SBERT+SOSNet)
(Singh et al., 2022)	Twitter/ 47,694 tweets	Word Embeddings	Naive Bayes, Logistic Regression, SVM, Random Forest, XGBoost, LSTM, GRU	Accuracy: (GRU) 92%; Precision: 92%; Recall: 92%; F1-Score: 92%
(Mahmud et al., 2022)	Twitter/ 47,692 tweets	TF-IDF	LightGBM, XGBoost, Logistic Regression, Random Forest, AdaBoost	Accuracy: (LightGBM) 85.5%, (XGBoost) 83.5%, (LR) 82.3%, (RF) 83.1%, (AdaBoost) 81.0%; Precision: (LightGBM) 84.0%, (XGBoost) 82.5%, (LR) 82.0%, (RF) 82.0%, (AdaBoost) 80.5%; Recall: (LightGBM) 85.0%, (XGBoost) 83.0%, (LR) 82.5%, (RF) 83.5%, (AdaBoost) 81.0%; F1-Score: (LightGBM) 84.49%, (XGBoost) 82.74%, (LR) 82.24%, (RF) 82.74%, (AdaBoost) 80.74%
(Mathur et al., 2023)	Twitter/ 47,000 tweets (*Real-Time Tweet Analyzing*)	Count-Vectorizer, TF-IDF	Random Forest	Accuracy: 94.06%; Precision: 94.01%; Recall: 94.24%; F1-Score: 94.24%
(Alqahtani et al., 2024)	Twitter/ 47,000 tweets	TF-IDF with bigrams	Random Forest, Decision Tree, XGBoost	Accuracy: (Stacking) 90.71%, (Voting) 90.41%; Precision: (Stacking) 90.85%, (Voting) 90.69%; Recall: (Stacking) 90.60%, (Voting) 90.36%; F1-Score: (Stacking) 90.63%, (Voting) 90.45%

Table 2.1 provides an overview of various studies on cyberbullying detection using machine learning techniques. Each study utilized same datasets, feature extraction methods, and classifiers to achieve varying levels of accuracy and performance.

2.6.2 Related Work On Different Dataset

In Rohini et al., (2023) the dataset includes 10,000 comments from Kaggle and 20,000 comments from various social media platforms. The features used are (BoW), Frequency (TF-IDF), and Word2Vec. The study employed Logistic Regression, Random Forest, Support Vector Machine, Naive Bayes, Decision Tree, and XGBoost classifiers, with Random Forest achieving the best results (Accuracy: 99.39%, F1-Score: 99.61%).

Kumar et al., (2022) used datasets from various social media platforms, including Twitter and Facebook. The features included BoW, TF-IDF, Word2Vec, and N-gram. The classifiers used were Naive Bayes, Support Vector Machine, Logistic Regression, Decision Tree, Random Forest, and K-nearest neighbour, with Random Forest performing the best (Accuracy: 91.153%, F-Measure: 0.898).

In Payal Budhe et al., (2023) the dataset comprised Twitter and other social media networks. The features included BoW, Skip Gram, Profanity Features, Sentiment Features, Pronouns, Demographic Features, Friends and Followers Count, Timestamp, and Location. The classifiers used were SVM, J48, Naive Bayes, Random Forest, and Social Signed Networks, with SVM achieving the best results (Accuracy: 89.75%, F1-Score: 0.886).

Islam et al., (2023) utilized 8,455 comments from WhatsApp, Facebook, Instagram, TikTok, and YouTube. The features used were TF-IDF and Word Embeddings. The classifiers included Logistic Regression, Decision Trees, Random Forest, Multinomial Naive Bayes, KNeighbors Classifier, Support Vector Machines, and Stochastic Gradient Descent, with SVM achieving the best performance (Accuracy: 90.06%, Precision: 92.60%, Recall: 84.16%, F1-Score: 88.18%).

In Alam et al., (2021) the dataset consisted of 9,093 tweets from Twitter. The features used were BoW, TF-IDF, and N-gram. The classifiers included Multinomial Naive Bayes, Logistic Regression, Decision Tree, Linear Support Vector Classifier, and ensemble methods such as Gradient Boosting, AdaBoost, Bagging, Voting Classifier, and Stacking Classifier, with the Stacking Classifier achieving the best results (Accuracy: 96%, Precision: 96.5%, Recall: 96.5%, F1-Score: 96.5%).

Afrifa et al., (2022) utilized a dataset of 16,851 tweets from Twitter. The feature extraction method used was TF-IDF. The classifiers employed were Random Forest and Support Vector Machine, with Random Forest achieving the highest accuracy (98.5%, RMSE: 0.2588, MSE: 0.0670).

Talpur et al., (2020) used 24,189 tweets from Twitter. The features included BoW, POS Tagging, and PMI-Semantic Orientation. The classifiers were Naive Bayes, K-Nearest Neighbors, Decision Tree, Random Forest, and Support Vector Machine, with Random Forest performing the best (Accuracy: 91.153%, F-Measure: 0.898).

Jain et al., (2021) employed datasets from Twitter and Wikipedia. The features used were BoW, TF-IDF, and Word2Vec. The classifiers included SVM, Logistic Regression, Random Forest, and Multi-Layer Perceptron, with SVM and TF-IDF achieving the best results (F-Measure: 0.939 for Twitter), and Neural Network achieving 92.8% accuracy for Wikipedia.

Apoorva et al., (2022) utilized a large dataset from Mendeley and found that the Gated Recurrent Unit (GRU) was the top performer, achieving an accuracy of 95.47%, precision of 0.90, recall of 0.89, and an F1-score of 0.89. "Machine Learning Based Approach for Detection of Cyberbullying Tweets" used datasets from Twitter and Kaggle, with Logistic Regression achieving the best results (accuracy: 93%, F1-score: 93%).

Shah et al., (2020) used Twitter and Kaggle datasets. The features used were TF-IDF. The classifiers included Logistic Regression, Support Vector Machine, Random Forest, Multinomial Naive Bayes, and Stochastic Gradient Descent, with Logistic Regression achieving the best performance (Accuracy: 93%, F1-Score: 93%).

Hani et al., (2019) utilized a dataset of 12,773 conversations/messages from Formspring (Kaggle). The features used were TF-IDF, Sentiment Analysis, and N-Gram. The classifiers employed were SVM and Neural Network, with the Neural Network outperforming others (Accuracy: 92.8%, F-Score: 91.9%).

Table 2.2 Analysis Different Dataset of Cyberbullying Detection Models

Author/ Year	Dataset	Features	Model/ Classifier	Results
(Rohini et al., 2023)	10,000 comments (Kaggle), 20,000 comments (social media and Kaggle)	BoW, TF-IDF, Word2Vec	Logistic Regression, Random Forest, Support Vector Machine, Naive Bayes, Decision Tree, XGBoost	Random Forest (Accuracy: 99.39%, F1-Score: 99.61%)
(Kumar et al., 2022)	Various social media platforms, including Twitter and Facebook	BoW, TF-IDF, Word2Vec, N-gram	Naive Bayes, Support Vector Machine, Logistic Regression, Decision Tree, Random Forest, K-nearest neighbor	Random Forest (Accuracy: 91.153%, F-Measure: 0.898)
(Afrifa et al., 2022)	16,851 tweets from Twitter	TF-IDF	Random Forest, Support Vector Machine	Random Forest (Accuracy: 98.5%, RMSE: 0.2588, MSE: 0.0670)
(Payal Budhe et al., 2023)	Twitter and other social media networks	BoW, Skip Gram, Profanity Features, Sentiment Features, Pronouns, Demographic Features, Friends and Followers Count, Timestamp, Location	SVM, J48, Naive Bayes, Random Forest, Social Signed Networks	SVM (Accuracy: 89.75%, F1-Score: 0.886)
(Islam et al., 2023)	8,455 comments from WhatsApp, Facebook, Instagram, TikTok, and YouTube	TF-IDF, Word Embeddings	Logistic Regression, Decision Trees, Random Forest, Multinomial Naive Bayes, KNeighbors Classifier, Support Vector Machines, Stochastic Gradient Descent	SVM (Accuracy: 90.06%, Precision: 92.60%, Recall: 84.16%, F1-Score: 88.18%)
(Alam et al., 2021)	9,093 tweets from Twitter	BoW, TF-IDF, N-gram	Multinomial Naive Bayes, Logistic Regression, Decision Tree, Linear Support Vector Classifier, Ensemble methods: Gradient Boosting, AdaBoost, Bagging, Voting Classifier, Stacking Classifier	Stacking Classifier (Accuracy: 96%, Precision: 96.5%, Recall: 96.5%, F1-Score: 96.5%)
(Talpur et al., 2020)	24,189 tweets from Twitter	BoW, POS Tagging, PMI-Semantic Orientation	Naive Bayes, K-Nearest Neighbors, Decision Tree, Random Forest, Support Vector Machine	Random Forest (Accuracy: 91.153%, F-Measure: 0.898)

To be Continue …

... Continued

(Jain et al., 2021)	Twitter and Wikipedia datasets	BoW, TF-IDF, Word2Vec	SVM, Logistic Regression, Random Forest, Multi-Layer Perceptron	SVM with TF-IDF (F-Measure: 0.939 for Twitter), Neural Network (Accuracy: 92.8% for Wikipedia)
(Apoorva et al., 2022)	159,686 comments from Mendeley	Count Vectorizer, Word Embedding	K-Nearest Neighbor, Multinomial Naive Bayes, Logistic Regression, Support Vector Machine, LSTM, Gated Recurrent Unit	Gated Recurrent Unit (Accuracy: 95.47%, Precision: 0.90, Recall: 0.89, F1-Score: 0.89)
(Shah et al., 2020)	Twitter and Kaggle datasets	TF-IDF	Logistic Regression, Support Vector Machine, Random Forest, Multinomial Naive Bayes, Stochastic Gradient Descent	Logistic Regression (Accuracy: 93%, F1-Score: 93%)
(Hani et al., 2019)	12,773 conversations/messages from Formspring (Kaggle)	TF-IDF, Sentiment Analysis, N-Gram	SVM, Neural Network	Neural Network (Accuracy: 92.8%, F-Score: 91.9%)

Table 2.2 comprehensively analyzes various studies on cyberbullying detection using machine learning techniques. Each study utilized different datasets, feature extraction methods, and classifiers to achieve varying levels of accuracy and performance.

These studies collectively highlight advancements in cyberbullying detection, sophisticated machine learning and deep learning techniques. By employing diverse methodologies, ranging from feature-based text classification to ensemble and deep learning models, these works contribute significantly to enhancing the accuracy and robustness of automated cyberbullying detection systems. The integration of innovative approaches, such as attention mechanisms and graph-based models, further underscores the potential of these technologies in addressing the evolving nature of cyberbullying on social media platforms.

2.7 Summary

This chapter provides a comprehensive overview of the current state of cyberbullying detection, highlighting its definition, impact, and the challenges involved. The literature review delves into the complexities of detecting cyberbullying, emphasizing the role of NLP and machine learning techniques. It explores various feature extraction methods such as TF-IDF, Word2Vec, and GloVe, etc., alongside advanced machine learning classifiers including Random Forest, Logistic Regression, and Support Vector Machine. Additionally, it reviews the application of deep learning models like LSTM and RoBERTa. Comparative studies using different datasets underscore the effectiveness of various methodologies in improving detection accuracy. The integration of these sophisticated techniques showcases the ongoing advancements and the critical role of technology in mitigating the psychological impacts of cyberbullying, thereby contributing to creating safer online environments.

CHAPTER III

METHODOLOGY

3.1 Introduction

The elaborate strategy implemented in curbing cyberbullying detection through standard machine learning and advanced deep learning methods has been articulated in this chapter. The methodology is structured into distinct phases, each critical to the development of an effective cyberbullying detection application. Section 3.2 starts with the research design. It, therefore, has a complete survey about the steps followed, starting from the data collection to the model evaluation. Section 3.3 presents the dataset used in this study and provides insights on the diversity of the dataset. Section 3.4 describes techniques for normalizing data and preprocessing, which are crucial for the strategic preparation of the dataset. Section 3.5 discusses the feature engineering techniques to bolster the model performance through efficient representation of the textual data. Section 3.6 provides comprehensive discussion to tackle the application of both machine learning and deep learning models in classification. Section 3.7 discuss the evaluation metrics used during the research, which is centred around the objective of identifying and categorizing cyberbullying occurrences on online platforms.

3.2 Research Design

The research design encapsulates a structured approach across five primary phases, each crucial for developing an effective cyberbullying detection system. The workflow begins with data collection, followed by data normalization, feature engineering, model application, and concludes with a thorough evaluation of the models' performance. This design is structured to meet the study’s objectives by applying both traditional machine learning and advanced deep learning models to a dataset of cyberbullying instances on Twitter. As shown below.

The first phase of the iterative design process encompasses the compilation of a broad dataset, which should include tweets with different orientations and show the dynamic nature of Twitter communication. The next is, in phase two, to thoroughly prepare the processed data for use in further research studies. Throughout the third phase, advanced feature engineering techniques such as TF-IDF, word embeddings, n-grams, BOW, Character Encoding, and Hashing Vectorization are employed in order to filter out the relevant features for the given text data to be extracted. The fourth phase is of utmost importance, where there is a choice of models that range from classics such as RF, LR, and SVM to advanced models, namely LSTM and RoBERTa to be trained. The culmination of this process happens in the fifth phase if the efficiency of models is compared against key performance indicators such as accuracy, precision, recall, and the F1 measure to determine their ability to correctly identify cyberbullying in various forms. Each of these phases is critical, serving as a building block in the complex structure of this research design, which is illustrated as follows:

1. **Phase 1 (Cyberbullying Tweets Dataset)**: This initial phase includes collection, representation and description of the dataset that will deal with research questions, namely, how data diversity is ensured in machine and deep learning for Twitter analysis.

2. **Phase 2 (Data Normalization):** Preparation of the dataset through various normalization techniques, with the data standardization and reduction of noises proceeds aiming to make the data suitable for further analysis and processing.

3. **Phase 3 (Feature Engineering):** This phase involves the identification and extraction of selection of textual features such as TF-IDF, word embedding, n-grams, hashing vectorization and character encoding.

4. **Phase 4 (Models Application):** In this step, there can be a diverse number of models, such as RF, LR and SVM, and they can be more advanced like LSTM and RoBERTa.

5. **Phase 5 (Evaluation):** The last phase is assessment of models using several metrics to determine their effectiveness, it is taking in the comparison of performance of traditional machine learning algorithms and advanced deep learning approaches, where it shed lights on the appropriateness in real-world scenarios.

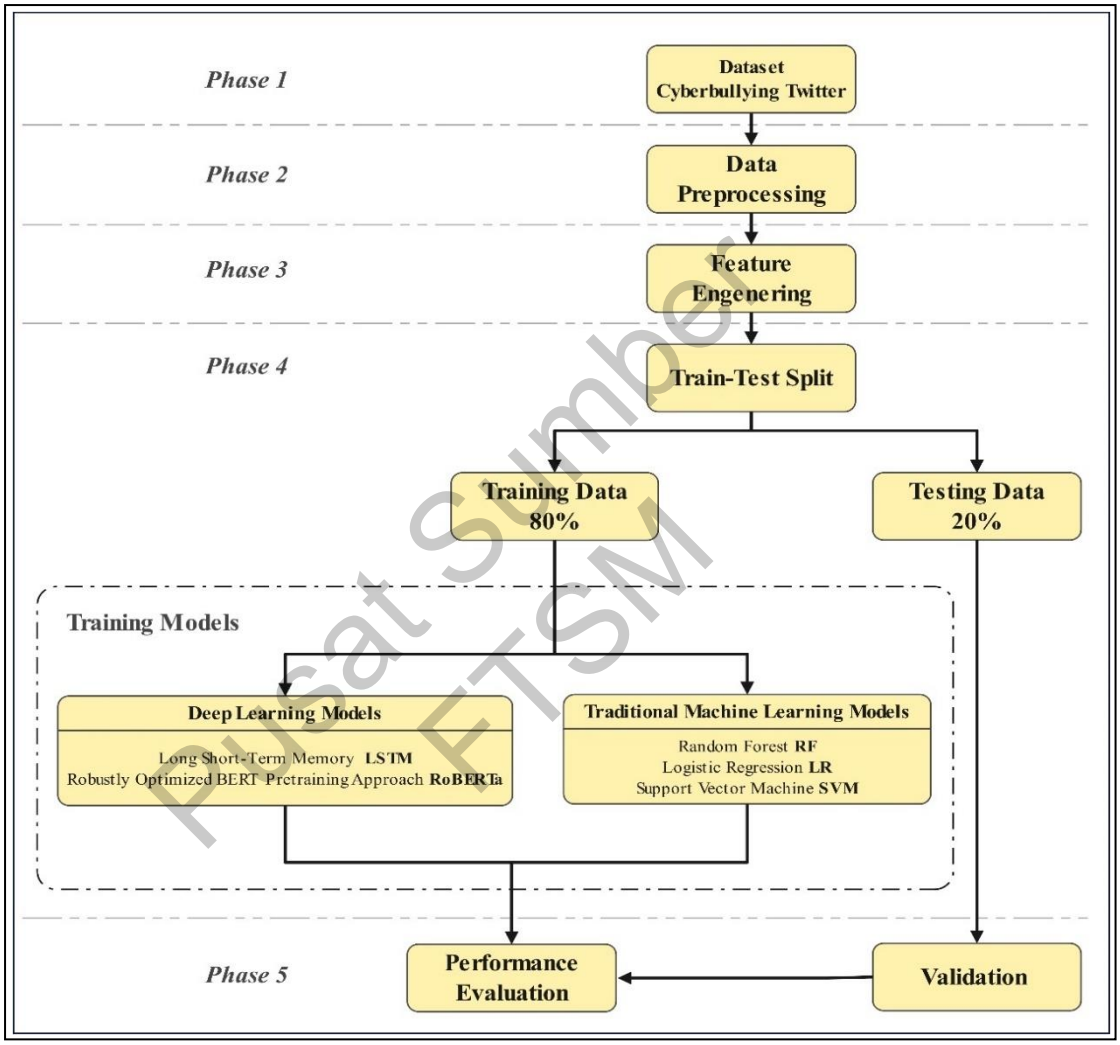

Figure 3.1 Phases of Research Methodology

Figure 3.1 visually summarizes these phases, illustrating the comprehensive approach adopted in this study.

3.3 Cyberbullying Tweets Dataset

Jason Wang, (2020) presented a dataset that is under examination and available on Kaggle entitled "Cyberbullying Classification". It comprises textual data collected to

scrutinize cyberbullying trends across diverse social media platforms, with entries categorized into multiple forms of cyberbullying or non-cyberbullying activities. Highlighted during the COVID-19 pandemic —specifically noted by UNICEF on April 15, 2020—this dataset gains prominence due to the rise in cyberbullying incidents correlating with increased digital engagement and reduced in-person interactions. These figures are scary, as 36.5% of middle and high school students have felt cyberbullied, and 87% have observed cyberbullying. The consequences of these actions are far-reaching, resulting in decreased academic performance and mental health conditions, including depression and suicidal thoughts. The Cyberbullying Tweets Dataset is highly relevant for research due to its compilation during the COVID-19 pandemic, characterized by heightened online activity and an increase in cyberbullying (*Cyberbullying Classification*, n.d.). The dataset's wide annotations of the different types of cyberbullying, like religion, age, gender, and ethnicity, make it possible to create algorithms that can detect the specific types of cyberbullying and thus solve the problem. Besides, its representation of every class equally is the barrier to the biased machine learning models towards the most common categories. Thus, the algorithms will work equally on all types of cyberbullying.

Table 3.1 shows some examples of different categories of cyberbullying. Corpus is drawn from around 47,692 tweets distributed into six classes, which were properly assigned with corresponding labels based on the nature of the content they represent.

Table 3.1 Cyberbullying Examples

Label	**Example Tweet Content**	**Category Description**
Religion	"Your religious beliefs are so backward #ignorant"	Cyberbullying based on religion
Age	"Wow, you act like such a kid. Grow up! #immature"	Cyberbullying targeting age
Gender	"He thinks he's tough, but acts like a girl #weak"	Cyberbullying related to gender
Ethnicity	"I bet you can't even speak proper English #stereotype"	Cyberbullying targeting ethnicity
Not_cyberbullying	"Just enjoyed a great day at the beach with friends #happy"	Non-cyberbullying content
Other_cyberbullying	"You're so ugly you should be banned from posting pics #mean"	Other forms of cyberbullying

Table 3.2 shows the distribution of classes within the dataset. It indeed presents an even spread of cyberbullying groups. The structure of the cyberbullying dataset is as follows: Text: This column contains the tweet text. Label: This column categorizes each tweet, indicating whether it is considered cyberbullying or not.

Table 3.2 Class Distribution Analysis

Label	Count	Percentage of Total
Religion	7,998	16.8%
Age	7,992	16.8%
Gender	7,973	16.7%
Ethnicity	7,961	16.7%
Not_cyberbullying	7,945	16.6%
Other_cyberbullying	7,823	16.4%
Total	47,692	100%

Figure 3.2 Snippet of the dataset displays the type of cyberbullying [Religious, Gender, Age, Ethnicity, Not Cyberbullying, and Other Cyberbullying].

```
1 df.sample(10, random_state=42)
```

	text	label
40362	@Goree_JuhssGuns hahaha he ain't even worth my...	ethnicity
15019	RT @hsaymssik: Sucks to have the smile wiped o...	gender
46321	Just a reminder, it's absolutely disgusting to...	ethnicity
23927	RT @BuzzFeedUK: When you accidentally open you...	other_cyberbullying
1640	Loving the look of the fritters! #mkr	not_cyberbullying
46681	Has 2 interesting events last year involving r...	ethnicity
40139	NIGGERS is the real way you dumb fuck	ethnicity
12668	things that AREN'T jokes - rape - sexism - rac...	gender
36459	You were bullied? If so, I'm sorry that happen...	age
8442	Haha did you watch big brother?, "Zankie" was ...	gender

Figure 3.2 Snippet of the Cyberbullying Dataset

To ensure a comprehensive understanding of the Cyberbullying Dataset, Figure 3.3 presents a visualization generated to illustrate the distribution of labels.

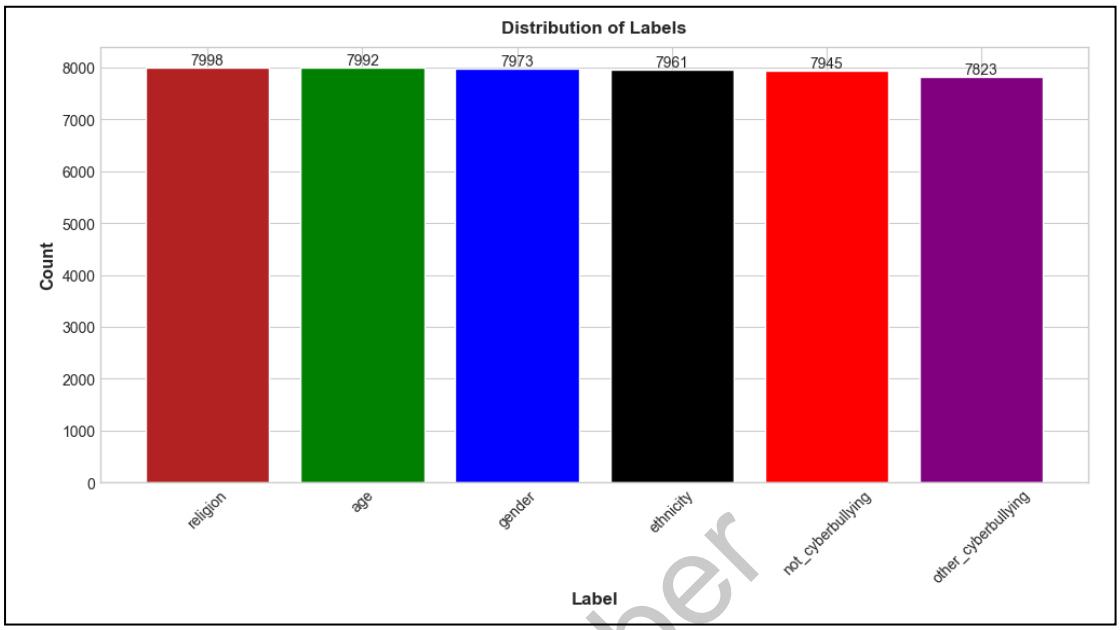

Figure 3.3 Distribution of labels

This dataset is an essential part of the study since it underlies the creation and advanced assessment of the models needed in the fight against cyberbullying on social media platforms.

3.4 Data Normalization

Normalizing data in text analysis, especially for social network tracking, is vital for establishing the pattern recognition and classification process by ML and DL due to the direct effect of inaccuracies in the text on the result of model training. To ensure that the model identifies cyberbullying effectively, the implements normalization steps such as emoji removal, entity stripping, hashtag cleaning, contraction expansion, and non-English filtering are implemented to cleanse the data textually. The main goal of these normalization processes is to sustain the quality and relevance of the dataset which ultimately forms the basis of proficient model training and, which, in turn, leads to the effectiveness of cyberbullying detection systems. Subsequent subsections provide a more detailed exploration of these tasks.

3.4.1 Basic Text Cleaning

Basic text cleaning is the foundational step in data preprocessing, where text data undergoes essential standardization and purification processes. These actions are designed to significantly improve the quality and effectiveness of subsequent data analyses. By ensuring that the text is clean and uniformly formatted, this step helps pave the way for more accurate and insightful analytical results.

a. Emoji Removal

Emoji removal, a technique for isolating and discarding emojis from text datasets, aims to purify the analytical focus on textual communications, excluding graphic interferences.

b. Stopword Elimination (Stopword removal)

Stopword Elimination, these common words that are believed to unimportant for the analysis are taken out of the text data. Such as stopwords are often characterize by articles, conjunctions, prepositions, and pronouns, for example 'the', 'and', 'a', 'of' and 'in'. The reason for this stopwords is to cut the dataset size down and emphasize meaningful words that probably have more effect in the requested activity.

c. Entity Stripping

Entity stripping stands for deleting the major details like names, locations and dates from the textual data; this is one of the essential steps in handling data. These actions are designed mainly for the purpose of protecting privacy as well as improving data analysis with the aid of more generalized content. The specific personal data and location specific information are removed, thanks to this method, and as a result the core of the text is preserved, and the story is told accurately at the same time.

d. URL Shortener Removal

Clarity and security of online interactions derive from the eradication of URL shorteners. Because such an operation provides the ability to detect and correct

shortened URLs to complete ones, it is very important from the perspective of data representation standardization. Primarily, it beefs up security features, specifically, thwarting phishing scams, hence, every hyperlink of this email marketing message is updated and readable.

e. Whitespace Normalization

Whitespace normalization, an essential part of text formatting, is the way to manually set up space with a certain distance between words and characters in order to improve its readability and look. Cutting unnecessary spaces, tabs, and line breaks, and normalizing the spacing between the words and sentences were robust actions that unified the appearance of the text across all the domains of the document or dataset.

3.4.2 Text Standardization

Text standardization normalizes dissimilar textual data to a consistent format which is effortlessly easily interpretable and analyzed by humans and machines. Through the process of fixing spelling mistakes, using only one lexicon and abbreviation, and unified date and number formats, the data present within dataset will be the same quality and format standards.

a. Case Normalization

In the field of text standardization case normalization is a specific technique, the purpose of which is to convert all text characters to a standard case - either to lowercase or to uppercase. This technique is frequently used as a pre-process to reduce data inconsistency and to make the subsequent tasks like searching, sorting or indexing more effective.

b. Language Filtering

Language filtering, a key tool for data management, deals with the procedure of checking and dividing the text according to the languages in which it is written. This, in turn, filters out texts that fail to meet the required grammatical criteria and

as well as ensures the linguistic consistency of the remaining data. These standards are inescapable for particularly strict guideline adherence cases such as machine translation, content moderation, and the case of precisely targeting marketing initiatives.

c. Elongated Word Normalization

Normalized elongation correction is one major approach in text analytics which mostly deals with the modification of deliberately elongated words or the ones that are usually seen in informal digital communicating platforms like text messages and social media posts. In this selection, words possessing unnecessary character duplications are isolated. For example, the word "soooo" would be converted to "so", and then it would be revived to its original form.

d. Character Filtering

Character filtering emerges as a pivotal practice within the realm of text processing, focusing sharply on the elimination of specific undesirable characters—chiefly non-alphanumeric and special symbols—from textual data. This operation is instrumental in purging the data of redundant elements that might otherwise muddle or complicate the analysis process.

e. Language Standardization

Language standardization involves creating guidelines for written and spoken communication to ensure consistency and clarity. This process helps bridge linguistic gaps and supports the development of language policies for education and government. For example, standardizing tweets by converting "b4" to "before" and "gonna" to "going to" makes them easier to understand and more uniform.

f. Removal of Short and Non-Informative Text

In a bid to reduce redundancy and improve text quality, unending string of short and non-significant text should be eliminated because it does not add value to a text. Hence, the text will be mainly composed of meaningful words which will be devoid

of unnecessary phrases thus they will enrich the remaining sections rather than retaining them. This technique, however, not only purifies the text but it can also communicate meaningful messages to a large target pool, thus heightening the overall quality and effectiveness of the content.

g. Short Tweet Removal

Short Tweet Removal is an improved data cleaning method used in twitter data analysis which identifies, and then removes tweets that are significantly rather short and devoid of clarity from the dataset. This way of thinking classifies tweets in terms of content and meaning thus the dataset binomially enhances the analysis of that information. The concentration of research on heavier tweets allows the analysts to extract more powerful and interpretable information, which generates high-level analytics in the end.

h. Short Words Removal

Short Word Removal which is a crucial step of text preprocessing where words with less than specific number of characters that usually ranges from two to three are discarded from the text. This approach focuses on what may be referred to analytical contexts as less consequential words such as 'an', 'in" and 'at'.

i. Repeated Punctuation Removal

Punctuation Removal, which is one of the efficient processes of text processing, deals with the elimination of superfluous punctuation symbols (e.g. multiple question and exclamation marks) from text documents. Through this, the desired outcome such as uniformity of the document, higher readability, and professionalism is achieved. This approach is also of great significance for such things as sentiment analysis.

3.4.3 Semantic and Structural Preparation

The structural and semantic preparation is equivalent to processing text, meanwhile it constructs and polish the information which gives a better understanding of data

analysis. Based on organizing sentences, paragraphs and documents logically, and by adding semantic improvements like synonym replacement and context enrichment, it is the purpose of this process to achieve the dataset that interprets information easily, but actually optimized through computing models.

a. Hashtag Normalization

In the context of natural language technology, normalization of hashtags stands out as the major operation performed on the hashtags used in social media to be able to understand them. The presented technique of disentangling a hashtag's words with the help of extra space (#SocialMedia becomes "social media) plays a significant role in the process of meaning extraction and makes text more manageable to interpret. Its prime function is to take care of the coherency of the keywords in the text, i.e. they should contribute positively to the semantic value instead of losing the clarity of the text.

b. Text Lemmatization

One of the important NLP techniques is tokenization into lemmas. To merge the meaning concepts participants of that process study thoroughly not only the context but also the morphological properties of the each of a word. Unlike the primitive technique in which affixes are removed and nothing is left but an inaccurate root word that lacks lexical context, lemmatization ensures that the derived root word correctly represents all the lexical forms meanings of the stem. In such way this well-established procedure is essential for improving the accuracy and capability of different NLP applications by removing the hassle of morphological variations behind language.

c. Tokenization and Vectorization

Tokenization and vectorization are the key text processing components of NLP. First, the tokenization function piece texts down into discrete terms, known as tokens, which simplifies handling big text data. Being that vectorization assigns each token a different format by techniques such as one-hot encoding, TF-IDF, and

word embeddings. These numerical representations are an integral part of machine learning algorithms, allowing them to evaluate and analyse text data with great complexity.

3.4.4 Data Integrity and Enhancement

Data integrity is vital during the data life cycle to preserve the exactitude, consistency, and integrity of the data so that it maintains the unaltered form and becomes the representation of information. Rather than that, data enhancement, however, is a process focused on raising the level of the provided information by, for example, improving the quality of the elements of the given data through processes like cleansing and formatting, and, by means of integration, combining other sources of information. These processes on the other hand which clear the data with errors and keep it authentic also versions and enrich the value, making it more attractive and insightful for careful decision making.

a. Deduplication and Missing Value Handling

Deduplication is a data cleaning process that mercilessly scrutinizes datasets so as to remove any identical records making every distinct record present. That is of the essence for the preservation of the desired quality and efficiency of large databases. In addition, missing value handling resorts to imputation as a way of replacement and deletion of incomplete entries which are important for the statistics of the whole dataset.

Table 3.3 Example of Data Before and After Normalization

Type of Normalize	Raw Tweet	After Normalize	Label
Emoji Removal	"Can't believe this! 😡"	"Can't believe this!"	age
Stopword Elimination	"She is the best at everything, honestly."	"She best everything, honestly."	gender
Entity Stripping	"@user Check out this link! http://example.com"	"Check out this link!"	ethnicity
URL Shortener Removal	"Amazing article here https://bit.ly/abc123"	"Amazing article here"	other_cyberbullying

To be Continue ...

... Continued

Whitespace Normalization	"What are you doing? "	"What are you doing?"	not_cyberbullying
Case Normalization	"Just Watched an AMAZING movie!"	"just watched an amazing movie!"	religion
Language Filtering	"Just watched un film incroyable!"	[Removed if non-English]	age
Elongated Word Normalization	"That was soooo amazing!!!!"	"That was so amazing!!!"	gender
Character Filtering	"Hello!!!@@@***"	"Hello!!!"	religion
Language Standardization	"I can't believe it, it's totes amazeballs!"	"I cannot believe it, it is totally amazing!"	other_cyberbullying
Removal of Short and Non-Informative Text	"Hi!"	[Removed]	not_cyberbullying
Short Tweet Removal	"Wow"	[Removed]	not_cyberbullying
Short Words Removal	"In the a an the he"	"In"	other_cyberbullying
Repeated Punctuation Removal	"What?!?!!!"	"What?"	not_cyberbullying
Hashtag Normalization	"Loving this #SunnyDay"	"Loving this Sunny Day"	age
Deduplication and Missing Value Handling	Duplicate entry of "Cyberbullying is a concern."	Single entry of "Cyberbullying is a concern."	gender
Text Lemmatization	"The cars were driven faster by the drivers"	"The car be drive fast by the driver"	ethnicity
Tokenization and Vectorization	"Cyberbullying is a growing concern."	["cyberbullying", "is", "a", "growing", "concern"]	gender

Table 3.3 Examples of Cyberbullying Before and After Normalization - This table shows how the transformation of tweet data through assorted normalization steps, demonstrating the effectiveness of each processing technique.

3.4.5 Addressing Ambiguous Class

The removal of ambiguously defined classes like "other_cyberbullying" from a dataset addresses several issues in machine learning model training: class imbalance, which biases the model towards overrepresented classes, and overfitting, where models memorize anomalies rather than learn to generalize. By eliminating such classes,

training can focus on specific, well-defined categories, thereby improving accuracy and generalization capabilities of models on new, unseen data. This strategic method is based on techniques such as resampling and data augmentation to get rid of class imbalance, thus, the model will have better performance on the classes that are more representative (Khan et al., 2024; Wu, 2023).

3.4.6 Data Augmentation Methodology

The *SynonymAug* function that is the keyword module of the *nlpaug* library utilizes the WordNet database to replace synonyms with the words of texts. This aspect of text enhancement is very important in the development of datasets, which are needed for machine learning projects. An augmentation process is initiated with the creation of a synonym augmenter: consum=naz= nuclear option. SynonymAug(aug_src='wordnet'). Finally, the function called augment_data is created, which needs dataflow, a class label, and an augmenter together with the target sample count which are the inputs. This version is exactly for the predefined group and makes the text to be changed for the new textual entities. The amplification goes on until the level of the desired sample number is reached, thus the class representation becomes the same. The size of the sampling is changed to be in accordance with the class scale of augmentation of the classes to that of the biggest class of augmentation in the original dataset. The correction of this factor is vital since it is the source of the representation of classes being fair and the solution of the initial class imbalance (Putu Widiarta Nandana Githa et al., 2024).

Rationale for Employing Synonym Replacement (Li et al., 2024):

a. **Preservation of Semantic Integrity:** The synonym substitution approach proves to be a quite delicate technique of text augmentation, as it prevents the change of the original content meaning and context. This feature of the model makes it an effective tool where content privacy is very crucial, for example, to detect cyberbullying.

b. **Enhancement of Model Robustness:** This method increases the lexical richness of the dataset, which is then used to train models to recognize and interpret different ways of conveying the same meaning. Such flexibility is the key to successful accuracy and reliability in classification tasks which is significant.

c. **Addressing Class Imbalance:** Data augmentation was employed after preprocessing and cleaning to address the loss of data from various categories caused by these processes. Augmentation was utilized to ensure all categories remained balanced before initiating the training process.

The problem of class imbalance in cyberbullying datasets was solved by employing data augmentation techniques with SynonymAug function as a substitute for synonyms. Texts are replaced with words from WordNet and the meaning of the content is kept intact but very careful. For classes that are underrepresented, synthetic samples are generated; that allows them to increase to the count which is equal to that of the prevalent class. This equilibrium is extremely pivotal for teaching the machine learning models to predict and deal with different types of cyberbullying equally. This will thus enable the models to handle all the cyberbullying categories with consistent accuracy and fairness. Table 3.6 the detailed view on the initial class distribution, count after normalization, the required augmentation, and Final Count for each class.

Table 3.4 Dataset Class Distribution and Augmentation Requirements

Category	Initial Count	After Normalization	Required Augmentation	Final Count
Religion	7,998	7906	0	7906
Age	7,992	7830	76	7906
Ethnicity	7,973	7418	488	7906
Gender	7,961	7227	679	7906
Not Cyberbullying	7,945	5983	1923	7906

Table 3.4 shows the detailed view on the initial class distribution, count after normalization, the required augmentation, and Final Count for each class.

a. Handling Class Imbalance

Class weights are designed to be used during the training to mitigate the class imbalance. Such weights may be assigned to different learning algorithms for more emphasis on minority classes during the learning phase.

Table 3.5 Class Distribution Ratios at Different Stages of Data Processing.

Category	Initial Distribution Ratio	After Normalize Distribution Ratio	After Augmentation Distribution Ratio
Religion	7998 / 47692 ≈ 0.168	7906 / 41554 ≈ 0.190	7906 / 47463 ≈ 0.167
Age	7992 / 47692 ≈ 0.168	7830 / 41554 ≈ 0.188	7906 / 47463 ≈ 0.167
Ethnicity	7973 / 47692 ≈ 0.167	7418 / 41554 ≈ 0.179	7906 / 47463 ≈ 0.167
Gender	7961 / 47692 ≈ 0.167	7227 / 41554 ≈ 0.174	7906 / 47463 ≈ 0.167
Not Cyberbullying	7945 / 47692 ≈ 0.167	5983 / 41554 ≈ 0.144	7906 / 47463 ≈ 0.167

Table 3.5 Presents a visual overview of curriculum adjustments in the dataset made to balance the learning and training system.

b. Label Encoding

To enhance the processing of cyberbullying categories within ML models, text labels are converted into numerical forms through label encoding. This method assigns a unique integer to each cyberbullying category, facilitating more efficient computation and accurate analysis. Label encoding simplifies the model's task by standardizing category representations, thereby ensuring effective and consistent performance across various analytical processes. Figure 3.4 provides a visual representation of the label encoding process used for the cyberbullying classes within the dataset, demonstrating how categorical text labels are efficiently converted into numerical identifiers necessary for computational analysis.

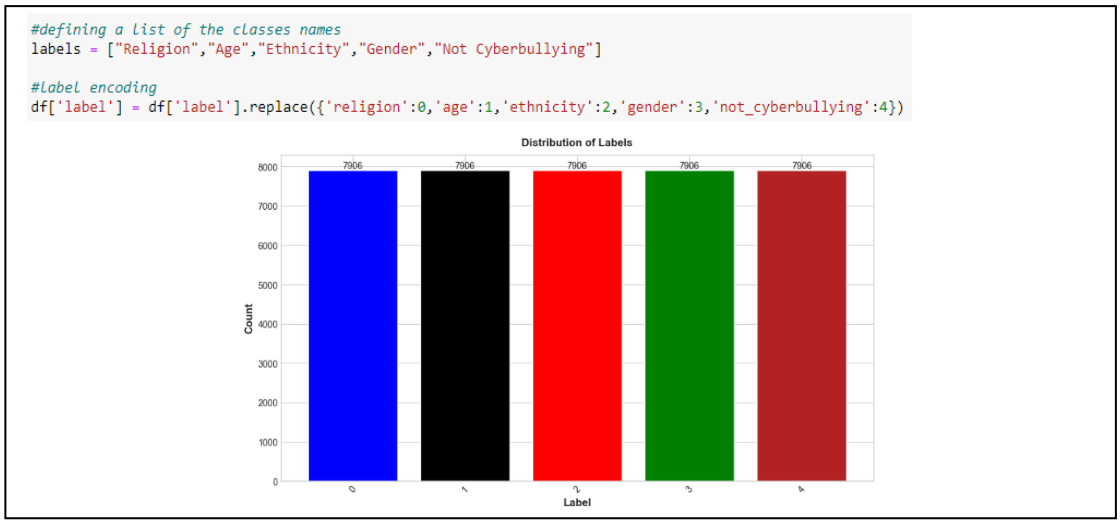

Figure 3.4 Visual Representation of Label Encoding in Cyberbullying Classes

3.4.7 Visualization and Assessment

***Wordcloud* Visualization:** The main purpose of word clouds is highlighting the most popular terms found in the particular form of cyberbullying. These could cause any given category of online harassment to be interpreted in a clearer and more in-depth way. On the following *wordcloud* diagrams which represent each class of cyberbullying, —namely not_cyberbullying, gender—related, religious, age—related, and ethnicity—based bullying—one it is possible to notice such a difference in vocabulary in these categories. These visualizations are helpful as a qualitative measure of the dataset's semantic landscape:

1. The not_cyberbullying *wordcloud* prominently features neutral and broad-spectrum terms like "people," "school," and the abbreviation "RT" for retweet, indicating general social media chatter.

2. Gender—related cyberbullying is captured through a *wordcloud* where slurs and pejorative terms associated with gender and sexual orientation are prevalent, such as "gay" and "bitch," revealing the targeted nature of harassment.

3. The religious category cloud is interspersed with words that highlight religious identities and affiliations, including "Muslim" and "Christian," along with provocative words like "terrorist," suggesting a context of religious discrimination.

4. The *wordcloud* for age—related cyberbullying clusters around terms indicative of school life and youth, such as "girl," "school," and "bullied," reflecting a context where minors are often targeted.

5. Lastly, the ethnicity—based bullying cloud is, unfortunately, dominated by racial epithets and derogatory language like "nigger," "white," "racist", laying bare the reality of racially motivated cyberbullying.

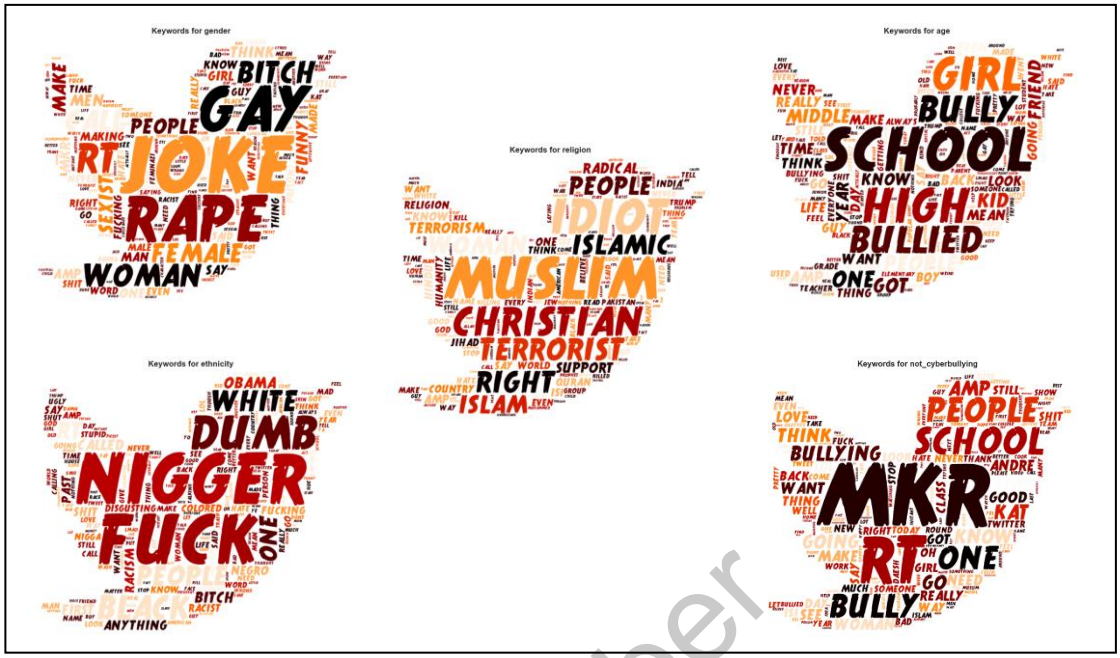

Figure 3.5 The *wordcloud* classes

Figure 3.5 displays the wordcloud for this category, with words that might be specific to [Religious, Gender, Age, Ethnicity and Not cyberbullying] bigotry or slurs dominating the visualization.

3.5 Feature Engineering

Feature engineering is a crucial step in the data preparation process for machine learning. It involves constructing suitable features from raw data, which can significantly enhance predictive performance. The process typically includes brainstorming potential features, creating new features, and refining them as needed to improve the model's accuracy and efficiency (Chia et al., 2021).

In the development of NLP Machin learning models, especially for identifying cyberbullying on social media platforms like Twitter, the feature engineering step is crucial. The effectiveness of these models largely depends on the representation of mass text data, which will be explored through specific techniques in this context.

In the domain of cyberbullying detection, feature representation is crucial for the performance of both machine learning and deep learning models. Traditional machine learning models, such as Random Forest, Logistic Regression, and Support

Vector Machines, rely heavily on features like TF-IDF and word embeddings. These features are engineered to represent the complex and informal language of tweets. However, traditional models often struggle with the intricacies of social media text, which can impact their accuracy. Advanced deep learning models, such as RoBERTa and LSTM, are designed to capture the contextual and semantic nuances of text. These models have shown superior performance in detecting cyberbullying. The effectiveness of these models depends significantly on the quality of data preprocessing and the selection of feature representation methods. This study emphasizes the importance of meticulous feature engineering and preprocessing to enhance model performance, providing a comparative analysis of traditional and deep learning approaches.

The selection of feature representations is crucial for the performance of deep learning models in tasks like cyberbullying detection. In this study, standard feature representations such as Word2Vec, GloVe, BERT, and Sentence-BERT were chosen for their distinct advantages. Word2Vec and GloVe are known for their simplicity and speed, effectively capturing semantic relationships between words. They were selected for their robustness and efficiency, showing high accuracy and F1 scores in preliminary experiments. However, they lack contextual sensitivity, producing the same vector for a word regardless of its usage. To address this, BERT was included due to its ability to understand context by considering both left and right contexts of words. Despite being computationally intensive, BERT's state-of-the-art performance justified its use. Sentence-BERT was also selected for its ability to generate semantically meaningful sentence embeddings, which are crucial for understanding the intent of entire tweets, providing a good balance between performance and computational efficiency.

Feature representation selections; In the study of cyberbullying detection on Twitter, the choice of feature representations such as Term Frequency-Inverse Document Frequency (TF-IDF), Bag of Words (BoW), Word Embeddings (Word2Vec, GloVe, FastText), N-grams (Unigrams, Bigrams, Trigrams, N-grams(1-2), N-grams(1-4)), Hashing Trick, and Character Encoding is well justified due to their robust performance in capturing the semantic and syntactic nuances of textual data. These methods have been extensively validated in natural language processing (NLP) and text classification tasks, demonstrating superior effectiveness in representing textual content for machine learning and deep learning models (Yin Zhang et al., 2010). Specifically,

Word Embeddings like Word2Vec, GloVe, and FastText are renowned for their ability to capture semantic relationships and contextual information, which are crucial in identifying the subtle and often context-dependent nature of cyberbullying language (Bojanowski et al., 2016; Mikolov, Chen, et al., 2013; Pennington et al., 2014).

Conversely, the study opted not to include user profile features, such as user demographics or account metadata, due to several considerations. Firstly, the focus of the research was on analysing the textual content of tweets, aligning with the core objective of understanding language use in cyberbullying. Incorporating user profile features might introduce bias and privacy concerns, as well as complicating the feature space with potentially less relevant information for the linguistic analysis (Waseem et al., 2016).

Additionally, previous studies have shown that while user profile features can provide supplementary context, they are often less effective in isolation compared to robust text-based feature representations (Davidson et al., 2017). Therefore, prioritizing advanced textual feature representations over user profile features ensures a more direct and ethical approach to cyberbullying detection on social media platforms.

In this study, the selection of machine learning models—Random Forest (RF), Logistic Regression (LR), and Support Vector Machine (SVM)—along with deep learning models—Long Short-Term Memory (LSTM) and RoBERTa—is driven by their demonstrated efficacy in the domain of cyberbullying detection and text classification tasks. Random Forest is favored for its ability to handle large datasets with higher dimensionality and its robustness against overfitting, as highlighted by (Alam et al., 2021) and (Ali et al., 2020). Logistic Regression is selected for its simplicity and interpretability, providing a baseline for comparison while efficiently handling binary and multi-class classification problems (Zaidi et al., 2023). SVM is known for its effectiveness in high-dimensional spaces and its capacity to handle nonlinear boundaries through kernel trick application, as originally outlined by (Cortes et al., 1995)and further supported by studies such as (Fitra Rizki et al., 2021b). On the deep learning front, LSTM is particularly adept at capturing temporal dependencies and contextual information in sequential data, making it suitable for nuanced text analysis tasks, as demonstrated by (Gada et al., 2021) and the foundational work by (Hochreiter

et al., 1997). RoBERTa, a transformer-based model, leverages extensive pretraining on diverse corpora to excel in understanding the intricacies of human language, thereby improving detection accuracy for cyberbullying instances, as evidenced by (Alrowais et al., 2024; Liu et al., 2019). This combination of ML and DL models allows for a comprehensive approach to cyberbullying detection, balancing interpretability and performance across varying data characteristics and classification complexities (Afrifa et al., 2022; Alqahtani et al., 2024). The features utilized in this study are outlined below:

3.5.1 TF-IDF Vectorization

The text-to-vector transformation process with TF-IDF technique is applied. It adjusts the significance level of each word by its relevance across the documents. This leads to an emphasis on the infrequent words over and above the frequent ones thereby emphasizing the unique words. The utilized *TfidfVectorizer* of Scikit-learn has just 500 feature limits to maximize the effectiveness of the machine learning performed. Efficacy of this transformation is proved and established by the final shapes of the transformed data visuals, in which it is clear that the text data had been reduced to the exact number of features required, resulting in enhanced computational efficiency.

3.5.2 Word Embeddings Techniques

1. **Word2Vec:** Word2Vec is deployed for the purpose of representation of relative meanings which is one of the most difficult aspects of the subtle semantic contexts of cyberbullying in text. The application process involves converting the tokenized texts to the already developed models. Furthermore, the size of each vector is dynamically estimated based on the number of words specific for each text to the fourth root. In particular, the Word2Vec model is built up with the help of the Gensim library on the tokenized training data, where processed embeddings can preserve the semantic aspects that are later recombined in order to assemble the sentence-level embeddings. These vectors are well suited for classificational purposes hence in detection of indictors of cyberbullying that may be imbedded in the communication.

2. **GloVe (Global Vectors):** GloVe has the ability to apply the global statistic property of a corpus to create vectors for words. Glove can capture conditional meanings as well as words that are frequently used in the cyberbullying context. In this research, GloVe embeddings are pre-trained and are read from an external file. Each sentence in the training and test sets is expressed as a vector through averaging of the embeddings of the words that make it up. As a result of this method, even such refined semantic relations in the text matter and can be taken into account in detecting cyberbullying.

3. **FastText:** Having been invented by Facebook, FastText, which is an extension to the concept of Word2Vec, is the foundation of any modern-day social media analysis that has to deal with slang, typos, and concatenated words. Through FastText's ability to analyse the subwords or n-grams of characters within a word, it can even generate more precise word representations for unconventional words that are not very well-known or are newly coined words which is quite typical in the fast-moving linguistic context of Twitter-like platforms. For the study, FastText with zero embeddings for out-of-vocabulary words or by using padding for uniform feature size across different texts will be used.

3.5.3 N-Gram Features

N-gram features represent certain text sequences into the set lengths, such as unigrams (singlewords) or n-grams of more extended sequences. Prominent markers like this are essential for the identification of the contextual relationships between words in texts, which allows the model to have a capability of discerning patterns pertinent to the cyberbullying detection.

In the context of the cyberbullying detection project, the n-gram features are derived by the CountVectorizer from Scikit-learn being configured to compile and vectorize up to four words concurrently (four-grams). The process is as follows:

1. **Unigrams to Four-grams Extraction:**

   i. **Unigrams:** The simplest form of n-grams, capturing individual word usage, is extracted using CountVectorizer set to ngram_range = (1, 1).

   ii. **Bigrams**: Sequences of two words are extracted to capture immediate word-to-word relationships, using ngram_range = (2, 2).

   iii. **Trigrams:** Three-word sequences are extracted to gain deeper contextual insights, set with ngram_range = (3, 3).

   iv. **Four-grams**: The extraction of four-word sequences provides the most context within the constraints set, using ngram_range = (4, 4).

Each configuration limits the features to a maximum of 500 to maintain computational efficiency and focus on the most important ones.

2. **Vectorization:**

Each level of n-gram features is transformed into a dense matrix representation, where each row corresponds to a document and each column represents an n-gram feature. This transformation is crucial for feeding textual data into machine learning models that require numerical input.

3. **Dataset Transformation Shapes:**

The outcome is in the form of arrays with the shapes holding the number of sample and the extracted features. This shape is one of the crucial components for maintaining the input dimensions on the constant level throughout several stages of the machine learning pipelines.

3.5.4 Advanced Transformations

- **Hashing Vectorization:** Hashing vectorization is the main mechanism in agglutinative text data compression. The feature of this vectorization is numeric formatting of data (assuming that predefined vocabulary is not used). Here the HashingVectorizer class will process the operation, this later will transform the text into feature vectors of 500 size, if specified. The procedure operates on the principles of a hash function that utilizes a non-linear mapping which converts the textual information with the help of a high-dimensional character space. By using this technique, the processing of the big data can get faster because less memory space is needed which may consequently encourage the data process. In the example, by hash function vectorization, the machine learning algorithm can look smoothly through a vast dataset, with features extraction still being computationally reliable even for a huge dataset.

- **Character Encoding:** The process of character encoding within this project is to have each character in the text transformed to its corresponding ASCII. This part is crucial to get the syntax of the text in the detail. Another part of the method entails padding or truncation of the encoded values to the fixed length, equal to the maximum length seen in both the training and testing sets, hence, the input size for the models will always be the same. The application of n-grams improves the recognition of patterns that depend on the character composition, which is an essential feature for identifying cyberbullying behaviours that employ slight text modifications not detectable by traditional word and token-based techniques.

3.5.5 Comprehensive Features (Combination of N-Grams)

1. **N-Grams (1-4):** The vectorization of all n-grams from one to four words, in turn, creates a dense and comprehensive feature set containing linguistic clues and structures from a large spectrum of the scale. Along with this characteristic, the feature set is crucial for the models that depend on syntax and context within the text.

2. **N-Grams (1-2):** Simultaneously, the following feature extraction considers single-words and bigrams. Whereas the former captures the word usage of an individual context for an entity and the latter records the consecutive words relationship, the combination is the most advantageous.

The vectorized n-grams are turned into arrays which form the basis for the cyberbullying detection algorithms. The transformed datasets are carefully shaped and then checked to ensure their dimensions accurately represent the textual data in the transformed space.

3.5.6 LSTM Features

LSTM models utilize several key features for processing and analyzing sequential data, particularly in the context of cyberbullying detection:

1. **Embedding Dimensions**: High-dimensional embeddings provide a nuanced understanding of input features. In the study, embedding dimensions (e.g., 128 and 200) were used to capture complex patterns in the text data.

2. **Hidden Units**: A larger number of hidden units (e.g., 256) in the LSTM layers help in capturing more detailed information from the input sequences.

3. **Input Gates, Forget Gates, and Output Gates**: These gates regulate the flow of information, allowing the LSTM to effectively manage long-term dependencies in the data.

4. **Activation Functions**: Sigmoid and hyperbolic tangent functions are used to compute gate vectors and work as activation functions within the LSTM cells.

3.5.7 RoBERTa Features

RoBERTa, an advanced version of the BERT model, employs several enhancements in its architecture and training methodology:

1. **Pre-trained Embeddings**: RoBERTa uses embeddings that are pre-trained on large-scale unlabeled text data. This helps in capturing rich semantic information and contextual nuances from the text.

2. **Dynamic Masking**: During training, RoBERTa dynamically changes the masking patterns applied to the training data, which improves the model's robustness and performance.

3. **Fine-Tuning on Specific Tasks**: After pre-training, RoBERTa is fine-tuned on specific downstream tasks, such as cyberbullying detection, to enhance its performance on these tasks.

4. **Attention Mechanisms**: RoBERTa leverages the self-attention mechanisms from the Transformer architecture to effectively process and understand the input sequences.

3.6 Models Application

During this phase of the study, machine learning and deep learning models are going to be utilized to train, adjust, and modify the final tool, all of which should be able to detect cases of cyberbullying that have been shared on Twitter. Different models which are Random Forests (RF), Logistic Regression (LR), and Support Vector Machines (SVM), as well as more complex neural networks such as LSTM and RoBERTa models are applied.

3.6.1 Traditional Machine Learning Models (ML)

1. **Random Forest (RF):** Based on text data as a source of information and combines TF-IDF score and n-grams into its features with its balanced ability in dealing with dataset. The proposed procedure of the training machine uses a grid search with a pattern structure and determines the model effectiveness by accuracy, precision, recall, F1-score indicators. Model Curve and Complexity Curve, which act as a result of a different number of trees' model behaviour is presenting the visualize of the model complexity level.

2. **Logistic Regression (LR):** Utilized for its efficiency and interpretability, Logistic Regression in this project is configured to explore different regularization techniques to prevent overfitting. The model undergoes fine-tuning through various solver and penalty configurations to enhance its effectiveness in detecting cyberbullying. Its performance is carefully evaluated and visually represented in plots to monitor the progression of learning outcomes across different iterations.

3. **Support Vector Machines (SVM):** This model is chosen because it has the superior ability to traverse the high-dimensional spaces usually involved in text classification tasks. SVM work with different kernel functions to do the best possible fit of both linear and non-linear relationship in the dataset. In the training process, the employ a thorough investigation of the choice of C parameter and the gamma values to fine-tune the decision boundaries priming the margin classification model.

When the Random Forest, Logistic Regression, and SVM are tested for effectiveness in the prediction of multiclass cyberbullying, the validation results confirm their effectiveness. The current academic research has proven that these algorithms are very accurate, precise, and recallable in distinguishing the different levels of cyberbullying, which shows that they are very useful in real life (Ali et al., 2020; Islam et al., 2023; Muzakir et al., 2022).

Such models are widely applauded for their ability to process and study data, which has complex and layered data structures that normally prevail in the social media. Also, these models are known to be very adaptable to the unique features of cyberbullying data such as textual, metadata and network dynamics. In a nutshell, Random Forest performs well with large volumes of data by integration its model as an assembly of decision trees sensitizing the different factors revolving around cyberbullying. Logistic Regression gives us the chance to find out the probability of certain features that can be related to cyberbullying classification and this is very helpful for the interpretive analysis. The SVM texture which is one of its strong appropriations has the skill to unearth intricate patterns within the text allowing it to identify subtle cyberbullying cues with relative ease. Cumulatively, the utilization of these models in prior research that proved their application affords rational basis for their continued use

in related settings. As a result, the new study is enabled to build and build on proven frameworks and to utilize as benchmarks, standard outcomes thus eliminating the confusions of model selection and enhances the rigor in the analysis of cyberbullying research.

3.6.2 Deep Learning Models (DL)

1. **LSTM:** This model outperforms in situations where contextual information of the sequence proves of great significance. The employ the LSTM approach to deal with the embedding of words, with dependencies along the sets of sentences considered. Adjustments are being done to the layers, nodes and dropout rates after the learning stage to improve its performance, with progress showcased through accuracy and loss curves.

2. **RoBERTa:** Considered one of the leading models in natural language processing, RoBERTa is particularly well-suited for cyberbullying detection. This is due to its enhanced pre-training process on a larger dataset and longer sequences compared to BERT, resulting in better performance in language understanding tasks. Transfer learning is a key aspect of RoBERTa, starting with a pre-trained language model which is then fine-tuned with domain-specific data to improve detection accuracy for specific categories of cyberbullying. The training process involves various learning rates and batch sizes, and during the evaluation phase, standard classification metrics and confusion matrices are employed to assess the model's effectiveness. While BERT is also highly effective, RoBERTa's improvements in training strategies and data utilization make it more adept for this particular task (Liu et al., 2019).

Many recent studies have demonstrated the effectiveness of LSTM and RoBERTa in detecting cyberbullying in multi-class scenarios. These deep learning models, renowned for their applicability to sequential and textual data processing, are particularly noted for their ability to model complex language structures necessary for identifying and categorizing cyberbullying content (Alrowais et al., 2024; Faraj et al., 2024) .

Each model is designed to be a combination of the best of the strengths therefore, to increase the effectiveness and the scalability factors of the online cyberbullying detection solution. The design of the tests and the validation of the models are necessary to ensure that the models are indeed effective in terms of the accuracy and the reliability on realistic situations, hence the results obtained will be convincing proving the application of machine learning and deep learning techniques in real life scenarios.

3.7 Evaluation

The assessment phase of the project is absolutely essential for achieving the goal of recognizing cyberbullying cases through effective machine learning models. This phase will include performing necessary conditions and conducting experiments with various quantitative metrics and techniques to determine model performance under different conditions and configurations.

During the final phase of the study, which is the evaluation stage, the ways to measure the true effect of the machine learning models that were applied for cyberbullying detections are very important. In this step, different performance metrics as well as quantitative works will be used to see that the optimum status of each model can be reached under different conditions and parameter settings.

3.7.1 Evaluation Techniques

Performance Metrics: The study employs a comprehensive suite of performance metrics to evaluate the efficacy of ML models in detecting cyberbullying. These metrics include:

1. **Accuracy:** Measures the overall correctness of the model across all the classes, providing a general indication of performance.

Referring to the Equation 3.1 below, Accuracy Equation

$$Accuracy = \frac{\text{Number of correct predictions}}{\text{Total number of predictions}} \tag{3.1}$$

2. **Precision (Positive Predictive Value):** Evaluates the model's ability to identify only relevant instances as positive, crucial for minimizing false positives.

   Referring to the Equation 3.2 below, Precision Equation

$$Precision = \frac{True\ Positives}{True\ Positives + False\ Positives} \tag{3.2}$$

3. **Recall (Sensitivity or True Positive Rate):** Assesses the model's effectiveness in identifying all actual positives, essential for ensuring no instance of cyberbullying is overlooked.

   Referring to the Equation 3.3 below, Recall Equation

$$Recall = \frac{True\ Positives}{True\ Positives + False\ Negatives} \tag{3.3}$$

4. **F1-Score:** Harmonic mean of precision and recall, offering a balance between the two, especially useful in scenarios with uneven class distributions.

   Referring to the Equation 3.4 below, F1-Score Equation

$$F1\ Score = 2 \times \frac{Precision \times Recall}{Precision \times Recall} \tag{3.4}$$

Additional metrics used are:

Macro-Average Metrics: Computes the average scores (accuracy, precision, recall, F1-score) across all classes, ensuring equal weight to each class, beneficial in imbalanced datasets.

Referring to the Equation 3.5 below, Macro Average Accuracy Equation

$$Macro\ Average\ Accuracy = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{True\ Positives_i + True\ Negatives_i}{Total\ Population_i}\right) \tag{3.5}$$

Referring to the Equation 3.6 below, Macro Average Precision Equation

$$Macro\ Average\ Precision = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{True\ Positives_i}{True\ Positives_i + False\ Positives_i}\right) \quad (3.6)$$

Referring to the Equation 3.7 below, Macro Average Recall Equation

$$Macro\ Average\ Recall = \frac{1}{N}\sum_{i=1}^{N}\left(\frac{True\ Positives_i}{True\ Positives_i + False\ Negatives_i}\right) \quad (3.7)$$

Referring to the Equation 3.8 below, Macro Average F1-Score Equation

$$Macro\ Average\ F1-Score = \frac{1}{N}\sum_{i=1}^{N}\left(2 \times \frac{Precision_i \times Recall_i}{Precision_i + Recall_i}\right) \quad (3.8)$$

Where Precision $_i$, Recall $_i$, are the Precision and Recall for class i.

3.7.2 Application Across Algorithms

1. **Hyperparameter Tuning:** In the course of evaluating the Hyperparameter tuning and fine-tuning in the various ML and deep learning models, it is understood to be a strategically planned implementation aimed at achieving enhanced cyberbullying detection. For the traditional machine learning models, for example, Random Forest, Logistic Regression, and SVM, hyperparameter tuning is a critical activity of respective parameter settings searching and selecting process management by GridSearchCV tool, including *n_estimators*, C, and kernel type, where lack of which may cause model accuracy and overfitting difficulties. However, for deep learning models like LSTM and RoBERTa, the attention moves towards fine tuning, where challenges for the later stages are set on optimal hyperparameter settings to normalizes the models to the peculiar features of the dataset concerned. Here contrasts over the techniques of traditional and deep required learning diverge which underlines the intrinsic differences in an algorithm searching for the best parameters contrasts with fine-tuning the pre-conceived conditions to achieve higher efficiency on individual tasks.

2. **Cross-validation:** The implementation of cross-validation methods in machine learning algorithms is an imperious approach for the improvement of model's credibility as well as accuracy in the area of social media platforms cyberbullying detection. When working with traditional machine learning models such as Random Forest, Logistic Regression and Support Vector Machines, cross-validation is a popular approach to hyperparameter optimization. Implementation of GridSearchCV, with its characteristic of covering a wide range of configurations, is another technique that is used to optimize the model by measuring the performances on the base of metrics such as accuracy and ensure that the model generalizes well on the unseen data.

On the other side, in deep learning models like LSTM and RoBERTa, cross-validation is rarely applied due to their high computational costs and the complicated data structure, these kinds of methods handle. Particularly, LSTM and RoBERTa make use of the alternative approaches, including the split validation and training, to prevent overfitting and comply with the limitations of the context that large datasets bring in addition to the nature of the data which is sequential. Based, therefore, on these factors, namely, the model structure and the computing capacity available, the type of cross-validation or its alternatives is mostly chosen. This emphasizes a case-by-case methodology toward model training and validation in machine learning.

Each of the model types has its own merits and demerits and therefore, should be accorded the due considerations. This is to achieve the optimum partition for a given application.

3.8 Summary

This chapter outlines the detailed approach applied to the cyberbullying detection problem solving through the conventional machine learning and the deep learning techniques. The methodology is broken down into the several phases, which start with the design of the research and the collection of data, thus, the cyberbullying instances on Twitter are presented in a very diversified and representative way. This is followed by the application of data normalization and preprocessing techniques to the dataset for the analysis, which includes removing emojis, stopping repeated words, and text

standardization. TF-IDF, word embeddings, and n-grams among other methods of feature engineering are used to boost the textual data representation.

The chapter also delves into the use of various models, which are mostly the traditional machine learning algorithms like Random Forest (RF), Logistic Regression (LR), and Support Vector Machines (SVM) but also the advanced ones like LSTM and RoBERTa. Each model, accordingly, to its performance metrics, like accuracy, precision, recall, and F1-score, is trained and tested in order to make sure that it can detect and categorize the cyberbullying content successfully. The last step is the detailed assessment that will be done by using these metrics to see which model has the better performance and also to check if they are applicable in the real world. This article is intended to create a structured way of the study that will result in the production of the reliable and robust cyberbullying detection systems.

CHAPTER IV

EXPERIMENTS AND RESULTS FOR MACHIN LEARNING

4.1 Introduction

This chapter discusses experiments and the findings of the extensive literature on machine learning (ML) in detecting cyberbullying. The objective is to compare results attained with model sorts and feature extraction techniques to decide how proper perception of the diverse circumstances can be categorized as cyberbullying and recognized with the assistance of machine learning algorithms on social media networks efficiently. The chapter is divided into 4 sections where a significant aspect of the execution and results is highlighted. Section 4.2 describes the technological context of the experiment setting: such as programming languages and tools of different versions and hardware characteristics. Section 4.3 looks at how the feature extraction process can be utilized to transform the text data into vector representation. Section 4.4 discusses the findings and analysis section marked by the designation of various models based on the performance criteria. Finally, section 4.5 provides a brief discussion and main contributions from the experiments.

4.2 Experiment Details

The Machine Learning experimental setup is carried out on Jupyter notebook and Python. A proper fit is ensured by the Anaconda Navigator integration that encourages easy running of Jupyter Notebooks directly in a web browser for future analysis and representation.

The hardware specifications for the study comprise NVIDIA GeForce RTX 4080 LAPTOP GPU, 13th Generation Intel Core i9-13950HX processor, 12GB of RAM expandable to 64GB, and 1TB of storage; These rich computing power and

storage capacity will be enough for ML models training, including complicated data processing.

The experiments used Python with key libraries such as Scikit-learn for machine learning models, and *gensim* for handling some of the text embeddings. Each model was evaluated using a variety of text feature extractions to determine which methods yield the best performance in terms of accuracy, precision, recall, and F1-score. Table 4.1 provides a list of the primary libraries and tools utilized in the experiments:

Table 4.1 Primary Libraries and Tools

Package	Description
NumPy and Pandas	Employed for data manipulation and numerical computations.
Scikit-learn	Used for implementing machine learning algorithms.
Gensim	Utilized for managing word embeddings like Word2Vec and FastText.
NLTK	Employed for text processing and feature extraction.
Matplotlib	Used for generating visualizations of the model performances.
Lpaug Augmenter	This module from the *nlpaug* library provides functionalities to augment textual data using natural NLP techniques
wordcloud	Used for generating a visual representation of word frequency in text data. It creates a "word cloud" from text, which highlights the most frequently occurring words in a visually appealing format.
Transformers	Provides state-of-the-art general-purpose architectures for NLP, including BERT, GPT-2, T5, etc. It allows for easy use of these models for text classification, generation, and translation.
PyTorch	Also known as PyTorch, this is a library for machine learning that provides flexibility and speed in building deep learning models, essential for handling the computational aspects of neural networks.

4.3 Discussion On Feature Extraction

Feature engineering is the process of converting raw data using methods that make data more effective to apply when modelling machine learning algorithms, which is indeed the most important thing when it comes to the success of such algorithms. As covered in the work adopted several feature engineering techniques which are supported to process a variety of text data. Each feature extraction technique was selected for its ability to capture different aspects of the text data:

1. TF-IDF (Term Frequency-Inverse Document Frequency): Measures the importance of words in the text relative to their frequency across all documents, TF-IDF helps highlight significant words that are more informative for classification.
2. Bag of Words (BoW): Represents text by the frequency of each word, BoW is simple and effective for capturing basic text patterns.
3. Word Embeddings (Word2Vec, GloVe, FastText): Converts words into dense vectors that capture semantic relationships, Word embeddings provide richer contextual understanding and capture semantic similarities between words.
4. N-grams (Unigrams, Bigrams, Trigrams, N-grams(1-2), N-grams(1-4)): Captures sequences of words to understand context, N-grams preserve some context and are useful for detecting phrases and co-occurrence patterns.
5. Hashing Trick: Maps text data into fixed-size vectors using a hashing function, reduces dimensionality and manages large vocabularies efficiently.
6. Character Encoding: Captures stylistic and syntactic features at the character level, Useful for detecting stylistic nuances and specific patterns related to cyberbullying.

The data cleaning and preprocessing steps applied in this research represents a complete solution for getting textual data ready for machine learning applications. The method begins with the initialization of essential tools like lemmatizers and stopwords, and then it consists of a set of functions, the main purpose of which is to clean-up the text data. These functions actively deal with several functions, such as eliminating emoji, URLs, non-English characters, numbers, and punctuation marks; transforming contractions into words; lemmatization and short or elongated words filtering. The cleansed text is later used to generate a structured dataset by importing raw data, applying the cleaning functions, and saving the cleaned output.

Additionally, the pre-processing step involves handling duplicates, checking class distribution, and augmenting data using synonym replacement to address class imbalances. These pre-processing steps help create a noise-free, balanced dataset with sufficient linguistic features for effective model training and evaluation. Visualizing the distribution before and after augmentation enhances the transparency and interpretability of the data preprocessing pipeline.

4.3.1 TF-IDF Feature Vectorization

TF-IDF is a widely used method to convert text into numerical values that highlight important words in a document relative to their occurrence in the corpus. It helps in reducing the impact of commonly used words that carry less information. The shapes (31624, 500) for training and (7906, 500) for testing indicate that each document is represented by a 500-dimensional vector. This dimension is limited to the most significant 500 words in the dataset, making it a compact and informative representation.

4.3.2 Bag-of-Words (BOW)

BoW represents text by the occurrence of words without considering order or context. It is a simple and effective method for text classification tasks. The shapes (31624, 500) for training and (7906, 500) show that each document is represented by a 500-dimensional vector based on word frequency. This straightforward representation is often effective for basic text analysis and classification tasks.

4.3.3 Word Embeddings (Word2Vec, GloVe, FastText)

Word embeddings capture semantic meanings of words by converting them into vectors such that words with similar meanings have a similar representation. This research observes three different types of embedding models as follows:

1. **Word2Vec**: The Word2Vec model was customized to the dataset by training on the tokenized tweets. The decision to utilize a relatively small vector size of 14 was strategically made to balance the need for capturing essential contextual information while maintaining computational efficiency. Smaller vector dimensions are advantageous in reducing computational complexity and memory usage, which is particularly beneficial when dealing with large datasets. Despite the reduced dimensionality, the optimized Word2Vec model retains sufficient capacity to discern subtle contextual clues.

The shapes (31624, 14) for training and (7906, 14) for testing indicate that each document is represented by the mean of its word vectors, with an optimal vector size of 14 derived from the unique words in the dataset. This compact representation effectively captures the semantic nuances of the words while ensuring that the model remains computationally efficient and scalable for large datasets. The choice of a 14-dimensional vector strikes a balance between preserving critical contextual information and minimizing resource usage, making it a practical solution for real-world applications.

2. **GloVe (Global Vectors for Word Representation):** Constructs word embeddings by leveraging statistical information from the entire corpus, capturing both local and global contexts. Utilized pre-trained embeddings which likely include a broader context from a larger corpus, resulting in vectors of size 200. This can be advantageous for capturing subtleties in language that are not represented in the training data alone.

GloVe embeddings, sourced from a Stanford NLP pre-trained model and loaded from 'embeddings/glove.twitter.27B/glove.twitter.27B.200d.txt' (GloVe: Global Vectors for Word Representation, n.d.). The shapes (31624, 200) for training and (7906, 200) for testing indicate that each document is represented by a 200-dimensional vector. This representation, derived from averaging the word vectors, effectively captures the sentiment and thematic nuances of tweets. The choice of a 200-dimensional vector is based on its ability to provide a detailed yet efficient representation of text.

3. **FastText:** Similar to GloVe, but also considers *subword* information, which can be particularly useful for social media text where misspellings and slang are common. The vectors of size 300 ensure a rich representation.

FastText holds the complexities of social media language, including slang and misspellings, by embedding subwords. The embeddings, loaded from 'fasttext/cc.en.300.vec/cc.en.300.vec' (Word vectors for 157 languages · fastText, n.d.). The shapes (31624, 300) for training and (7906, 300) for testing indicate that

each document is represented by a 300-dimensional vector. These vectors are derived by averaging the word embeddings, ensuring a standardized and consistent representation of tweets. This rich representation captures the intricate details of social media language, enhancing the model's ability to detect cyberbullying. The choice of 300 dimensions balances the need for a detailed representation with computational efficiency, making it well-suited for processing large datasets with complex linguistic features.

In cases where memory efficiency is a problem, GloVe and FastText models, tackle large vocabularies and embedding matrices, because they allow the loading of just the needed components and not the whole model. The use of pre-trained models GloVe and FastText model models able to adapt changes in social media environment especially new words and slang are captured properly.

4.3.4 N-grams (Unigrams, Bigrams, Trigrams, Four-grams):

The challenge to specify phrases and regional features in the text, which matter a lot when one interprets the intentions behind that tweet. Each n-gram technique concentrates on some aspects of the word combination lengths, and 500 words have been used as the maximum set to ensure a good balance between the detail and the computation time. These functions assist, thus, to find certain patterns of language that are relevant for cyberbullying.

1. **Combination of Unigrams and Four-grams (N-grams 1-4):** This complete building was primarily used for extracting features such as individual words, duo-words and sequence of four words. Furthermore, with a setting of max features to 500, the ensure that all the most relevant and vital phrases are captured, while balancing between extremely detailed and lengthy textual representation and the computational efficiency. This versatile range of N-grams is a feature that expedites the recognition and understanding of many linguistics tools, ranging from simple to complex.

2. **Combination of Unigrams and Bigrams (N-grams 1-2):** The combination of unigram and bigram record each word, its surrounding words, sentence structure and

semantic relations. It happens when the sentence structure and semantic relations are used as a base for the construction of a model that also catches wide vocabulary and recognizes specific two-word conjunctions. These conjunctions might be very important for understanding cyberbullying.

Expanding n-grams will thus bring more context into the model. However, it will also increase number of features, and hence, brings the question of computational overhead and, probably, overfitting if underestimated. In certain cases, restricting five of the sets in the program to 500 features might lead to some information missing. Common words may also be important, but they are removed due to low frequency in a given text.

4.3.5 Feature Hashing (Hashing Trick)

Feature Hashing reduces dimensionality and computational complexity, while BoW counts the frequency of words. Both methods produce a fixed-size representation of text (500 features), ensuring that the models are not overwhelmed by the high dimensionality typically associated with raw text data. The *HashingVectorizer* is particularly useful in environments with limited computational resources because it does not require a two-pass fitting and transformation process, unlike other vectorizers.

The shapes (31624, 500) for training and (7906, 500) indicate that each document is represented by a 500-dimensional vector, with features mapped using a hash function.

4.3.6 Character Encoding

Character encoding converts each character to its ASCII value, capturing the exact textual representation. This method is useful for tasks sensitive to spelling and punctuation. The shapes (31624, 173) for training and (7906, 173) for testing indicate that each document is represented by a sequence of 173-character encodings. This length was determined based on the longest text in the dataset, ensuring that each document, regardless of its original length, is truncated or padded to this fixed size. By using a fixed length, the model can process the text data consistently. This

representation effectively captures detailed textual information, allowing the model to consider the exact character composition of the text, which is crucial for tasks where precise textual details matter.

All feature representation techniques used in this study were designed to limit the feature space to 500 dimensions. This decision was crucial for managing memory usage and computational speed, particularly important when scaling to large datasets or deploying in real-time systems. Techniques like TF-IDF and n-grams can produce sparse matrices, which might affect model performance. Sparse representations were handled efficiently to ensure they did not negatively impact the training process.

Table 4.2 The Best Hyperparameters for Random Forest by Features

Feature Extraction Type	**Training Set Size**	**Testing Set Size**
TF-IDF	(31624, 500)	(7906, 500)
Feature BoW	(31624, 500)	(7906, 500)
Word Embeddings - Word2Vec	(31624, 14)	(7906, 14)
Word Embeddings - GloVe	(31624, 200)	(7906, 200)
Word Embeddings - FastText	(31624, 300)	(7906, 300)
Feature Hashing (Hashing Trick)	(31624, 500)	(7906, 500)
Character Encoding	(31624, 173)	(7906, 173)
Unigrams	(31624, 500)	(7906, 500)
Bigrams	(31624, 500)	(7906, 500)
Trigrams	(31624, 500)	(7906, 500)
Four-grams	(31624, 500)	(7906, 500)
N-grams (1-4)	(31624, 500)	(7906, 500)
N-grams (1-2)	(31624, 500)	(7906, 500)

Table 4.2 states the best hyperparameters for the Random Forest algorithm when used with different feature extraction methods. The dataset is segmented into training and testing sets, with the same sizes for each feature type. Basically, TF-IDF, BOW, and the others use a 500-dimensional feature space, while word embeddings such as Word2Vec, GloVe, and FastText have different dimensions (14, 200, and 300, respectively). This detailed comparison of the Random Forest algorithm with various

features techniques shows that the model is able to adapt to different textual representations, thus proving its robustness.

4.4 Results And Discussion

The experimental results provide a comprehensive view of how different machine learning models perform with various feature extraction methods under a range of hyperparameter settings.

4.4.1 Random Forest (RF)

The comprehensive evaluation of the Random Forest classifiers trained on diverse text feature types reveals significant insights into model performance, feature effectiveness, and the influence of hyperparameters. Below is a synthesized analysis across different metrics and feature types. Table 4.3 The comparative Performance Metrics of the Random Forest Algorithm by features:

Table 4.3 Comparison Results for the Random Forest by Features

Features	Accuracy	Precision	Recall	F1- score
tfidf	**0.9326**	**0.9358**	**0.9326**	**0.9335**
bow	**0.9321**	**0.9348**	**0.9321**	**0.9329**
word2vec	0.8883	0.8890	0.8883	0.8880
glove	0.8749	0.8796	0.8749	0.8756
fasttext	0.8745	0.8792	0.8745	0.8749
hashing	0.9209	0.9243	0.9210	0.9219
char_enc	0.4929	0.5019	0.4929	0.4860
unigrams	**0.9321**	**0.9348**	**0.9321**	**0.9329**
bigrams	0.7425	0.8282	0.7425	0.7556
trigrams	0.4540	0.8105	0.4540	0.4489
fourgrams	0.3052	0.8447	0.3052	0.2485
n-grams (1-4)	0.9317	0.9341	0.9317	0.9325
n-grams (1-2)	0.9309	0.9335	0.9309	0.9317

Table 4.3 shows the performance of the Random Forest algorithm for different features, thereby highlighting the relative efficiency of each feature. TF-IDF and present the highest accuracy, precision, recall, and F1-score. Character encoding and n-

gram features, especially trigrams and fourgrams, exhibit a lower success rate, proving that these techniques have some shortcomings for this task. The variability of these results clearly shows the need of choosing the right feature extraction methods to get the best model performance. Figure 4.1 represents the data in Table 4.2 by highlighting significant performance discrepancies between different features, making it easier to digest the Random Forest algorithm's response to different textual representations.

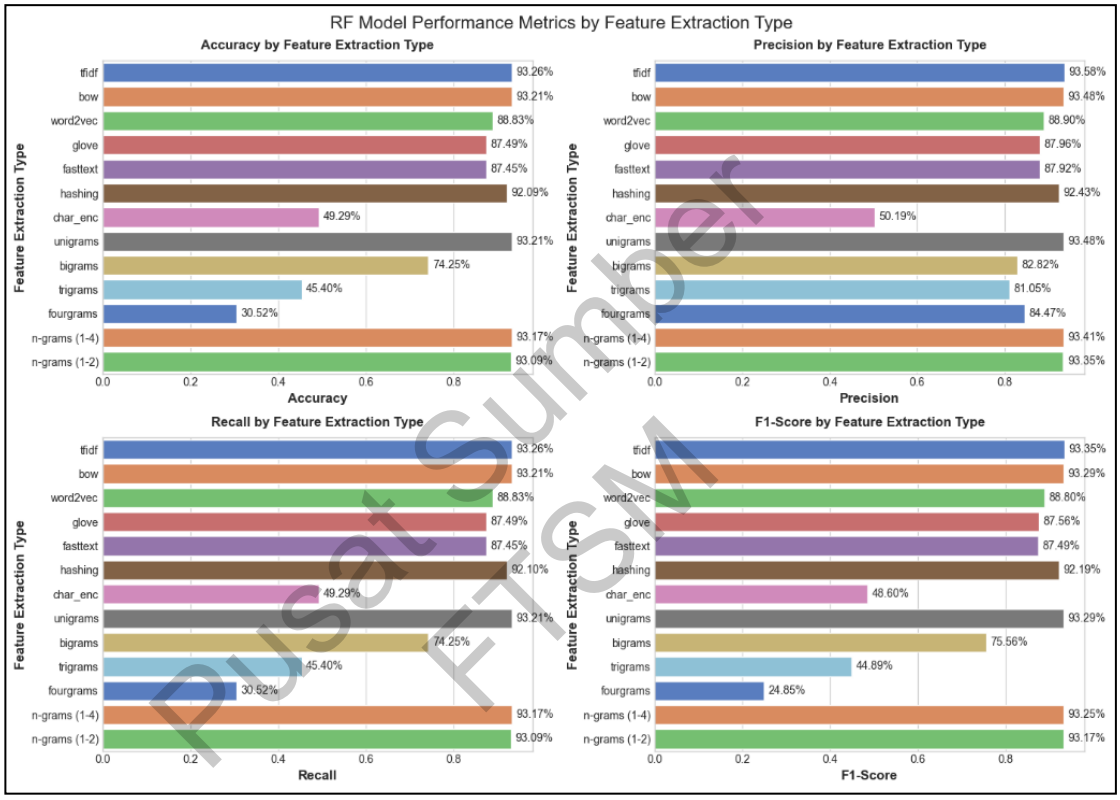

Figure 4.1 Comparison Random Forest Performance by Features

Figure 4.1 present the performance metrics of the Random Forest (RF) model across various feature extraction types, evaluating accuracy, precision, recall, and F1-score. TF-IDF and BoW consistently achieve the highest performance across all metrics, with TF-IDF leading slightly (accuracy: 93.26%, precision: 93.58%, recall: 93.26%, F1-score: 93.35%). FastText and hashing also show strong performance, with hashing notably achieving high precision 92.43%.

Conversely, character encoding (char_enc) and higher-order n-grams (trigrams and fourgrams) perform poorly, with fourgrams having the lowest accuracy (30.52%) and F1-score (24.85%). The discrepancies indicate that simpler feature extraction

methods like TF-IDF, BOW, and unigrams are more effective for the RF model due to their ability to capture essential text information without excessive dimensionality. On the other hand, elaborate measures such as Contextual encoding of characters and n>grams up to a certain value can aggravate the noise level and are likely to increase dimensionality, thus reducing the likelihood of a good model. Based on this analysis, RF models are better off when simple and efficient feature extraction methods are employed, thereby giving improving classification results.

All the features reveal a broad spectrum of performance depending on the chosen measure with the simple and combined features showing the highest overall performance while fourth order n-grams and char_enc achieve the worst performance out of all the tested models. Here's an in-depth look into why these variations occur:

1. **High-Performance Features:** Features like TF-IDF, n-grams (unigrams to ngrams (1-2), and BoW demonstrated high accuracy, precision, recall, and F1-scores, often surpassing 93%. These methods effectively capture key textual information, which helps in accurately classifying texts.

2. **Moderate-Performance Features:** With Word2Vec, GloVe and FastText, they achieve average results, with the accuracy ranging from 87-88%. While these features may be important for capturing semantic relationships between the entities, they are possibly not isomorphic to the exact nuances needed by the classification function without the extra depth offered by network architectures or context.

3. **Low-Performance Features:** There was a decrease in the precision and recall in both classes when using higher mode n-grams such as trigrams and four-grams, and in encoding characters. This drop is attributed solely on the high dimensionality and the sparsity of these features which probably overfit to train data and underperformed on unseen data.

Confusion Matrix Analysis: The confusion matrix for the Random Forest model with TF-IDF features demonstrates strong classification accuracy across several categories, as evidenced by the prominent diagonal values. These values highlight the

model's effective use of TF-IDF to capture term importance, which is essential for differentiating between various types of text content. However, the model exhibits a tendency to mislabel certain categories as 'Not Bullying', indicating a possible bias towards this class. This could be due to class imbalance, where 'Not Bullying' instances are more prevalent, leading the model to favour this majority class.

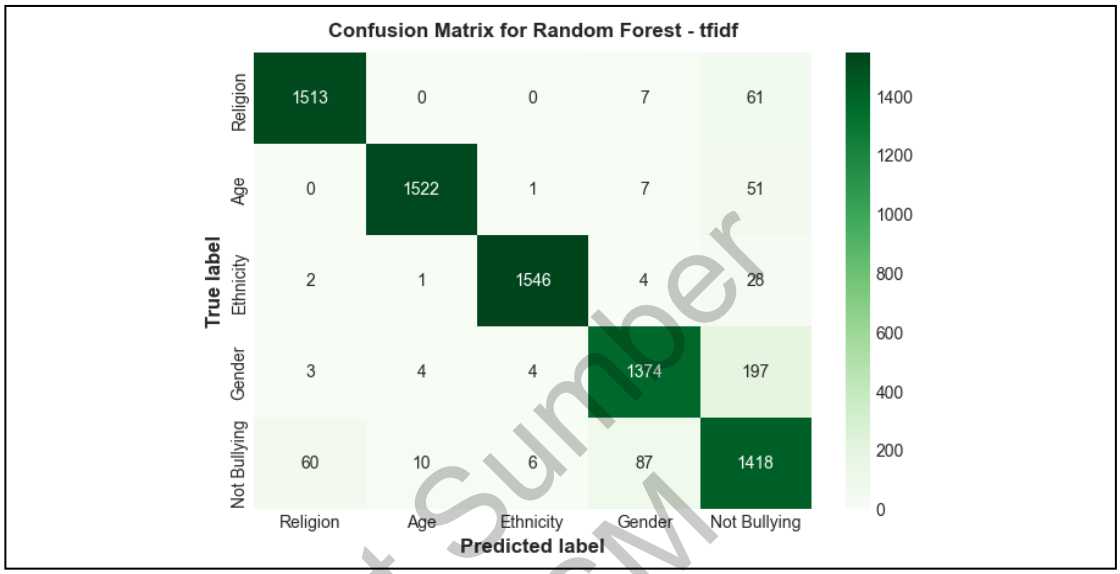

Figure 4.2 Confusion Matrix for Random Forest - TF-IDF

Figure 4.2 shows the confusion matrix for the Random Forest model using TF-IDF features, the number of correct and incorrect predictions by categories Religion, Age, Ethnicity, Gender, and Not Bullying for Random Forest - TF-IDF. The figure demonstrates the model's performance in classifying various cyberbullying types and non-bullying instances. The model shows high accuracy in predicting 'Religion', 'Age', and 'Ethnicity' categories with very few misclassifications. For 'Religion', there are 1513 correct predictions and 61 misclassifications as 'Not Bullying'. 'Age' has 1522 correct predictions with minor misclassifications. 'Ethnicity' has 1546 correct predictions with minimal errors. The 'Gender' category has more misclassifications, with 1374 correct predictions and 197 instances misclassified as 'Not Bullying'. The 'Not Bullying' category has 1418 correct predictions but includes some misclassifications spread across other categories, with the highest confusion being with 'Gender'.

The model performs well across most categories, though there is notable confusion between 'Gender' and 'Not Bullying' labels, suggesting room for improvement in distinguishing these categories.

Hyperparameter Grid and Cross-validation: The hyperparameter grid for Random Forests (*n_estimators*, *max_depth*, *min_samples_split*, *min_samples_leaf*) significantly influenced model performance. For instance, the best models typically used higher numbers of trees (*n_estimators* around 200-300) and no limit on *max_depth*, which suggests a complex decision boundary. Cross-validation was used to ensure the robustness and generalizability of the models, which is crucial in avoiding overfitting and underfitting.

Table 4.4 The Best Hyperparameters for Random Forest by Features

Features	n_estimators	Criterion	Max Depth	Min Samples Split	Min Samples Leaf	Max Features
tfidf	300	gini	None	2	1	sqrt
bow	300	gini	None	5	1	sqrt
word2vec	300	gini	None	2	1	sqrt
glove	300	gini	None	2	1	sqrt
fasttext	300	gini	30	2	1	sqrt
hashing	200	gini	None	2	1	sqrt
char_enc	300	gini	30	2	1	sqrt
unigrams	300	gini	None	5	1	sqrt
bigrams	200	gini	None	10	1	sqrt
trigrams	200	gini	None	5	1	sqrt
fourgrams	200	gini	None	2	1	sqrt
ngrams (1-4)	300	gini	None	5	1	sqrt
ngrams (1-2)	200	gini	None	5	1	sqrt

In the Table 4.4 there are presented the summarizes the best hyperparameters for Random Forest models across different feature extraction types. Most feature types, such as TF-IDF, BOW, Word2Vec, GloVe, and unigrams, perform best with 300 estimators, the 'gini' criterion, no maximum depth, minimum samples split of 2, a minimum samples leaf of 1, and 'sqrt' for max features. Some exceptions include FastText and character encoding, which both have a max depth of 30, and hashing,

bigrams, trigrams, and fourgrams, which use 200 estimators. Bigrams and trigrams also have higher minimum samples split values of 10 and 5, respectively. This indicates that while a general set of hyperparameters works well for most feature types, specific adjustments are needed for certain feature extractions to optimize Random Forest performance.

Consistency, for a lot of feature types, the hyperparameters are generally the same, which shows that sometimes there exist that settings work very well with different text representations. Variations in Depth and Split, Feature types with the specified max depth such as fasttext and char_enc evidently imply these representations are best obtained with more controlled tree growth. In opposition, other style of dress may not limit the depth at all. Balanced Min Samples Split, having a larger min_split of 2 or 3 for bow, unigrams, bigrams, and ngrams prevents the overfitting issue by making sure that there are enough samples for the splits.

Specific settings, like increasing the number of estimators for 'tfidf' and 'bow' or adjusting 'Max Depth' for 'fasttext' and 'char_enc', reflect how feature complexity and algorithmic needs vary across different types of data representation.

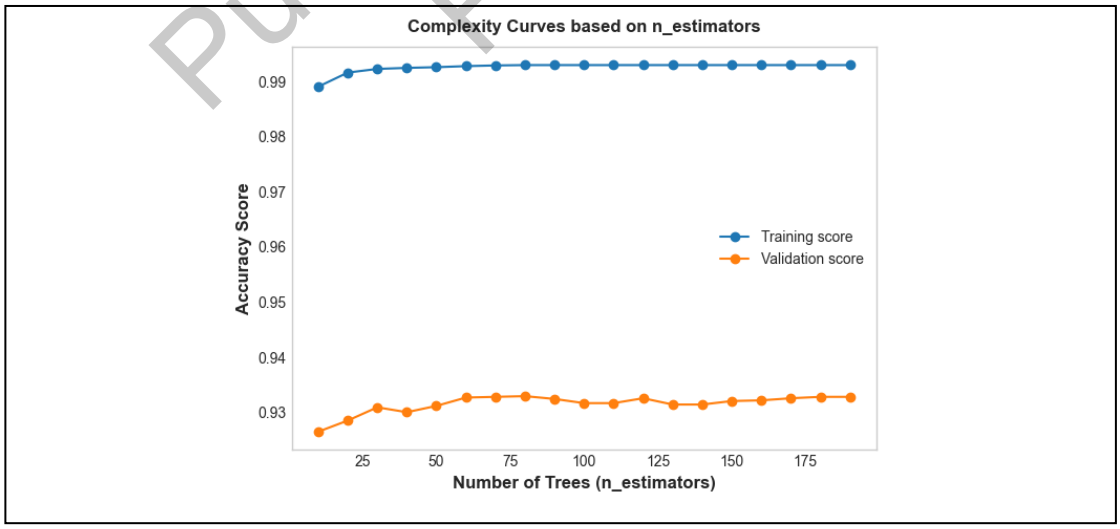

Figure 4.3 The Number of Trees (*n_estimators*)

Figure 4.3 The graph represents the accuracy of the Random Forest - TF-IDF classifier changes with the number of trees (n_estimators) in the model. The provided complexity curves depict the accuracy scores for training and validation datasets based

on the number of trees (n_estimators) in a Random Forest model. The training accuracy remains consistently high, close to 1.0, indicating the model's strong capability to fit the training data. The validation accuracy starts around 0.93 and shows a slight upward trend, stabilizing near 0.94 as the number of trees increases. This suggests that adding more trees improves generalization up to a point, after which the gains plateau. The gap between training and validation accuracy indicates a well-regularized model with minimal overfitting, balancing complexity and performance effectively.

Learning Curves: Learning curves for high-performing features demonstrated good convergence between training and validation scores, indicating well-trained models. For features with lower performance, the learning curves often showed a gap between training and validation scores, suggesting overfitting. The training accuracy starts off high and remains stable, indicating that the model can easily learn from the training data. The validation accuracy increases with more training examples, narrowing the gap between itself and the training accuracy, which suggests improving generalization as the model is trained on more data. Training loss is minimal, maintaining near zero throughout different training sizes. The validation loss decreases as more data is provided, converging closer to the training loss.

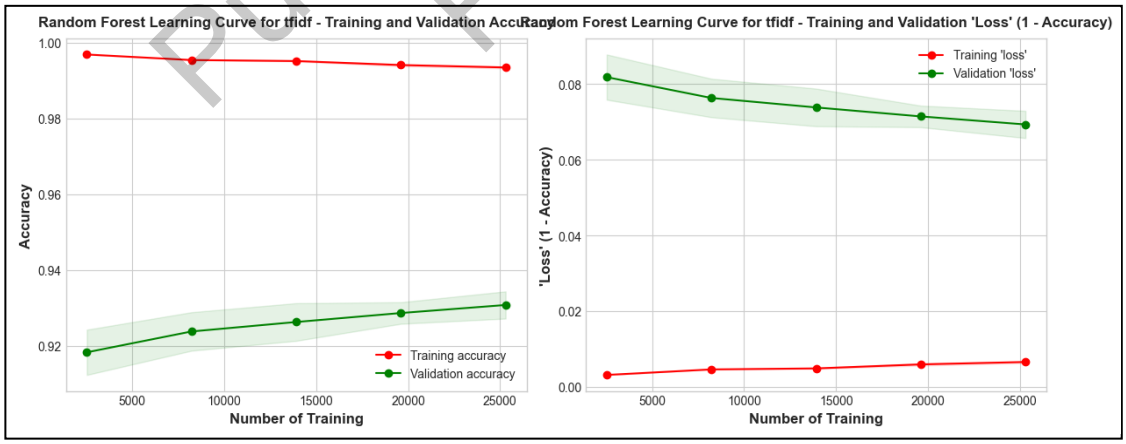

Figure 4.4 The Learning Curve and Loss Curve for Random Forest

Figure 4.4 presents the learning curves for the Random Forest model using TF-IDF features display training and validation accuracy and loss as functions of the number of training samples. The left plot shows that training accuracy remains high and stable, close to 1.0, indicating the model fits the training data well. Validation accuracy

steadily increases with more training samples, approaching 0.94, suggesting improved generalization.

The right plot illustrates the corresponding training and validation loss. Training loss remains low, reinforcing the model's strong fit to the training data. Validation loss decreases as more data is added, indicating that the model benefits from additional data, reducing overfitting and improving its performance on unseen data. These trends justify the effectiveness of using TF-IDF with Random Forest, highlighting that increasing the training dataset size enhances the model's ability to generalize, thus improving overall performance.

4.4.2 Logistic Regression (LR)

The comprehensive evaluation of the Logistic Regression classifiers trained on different features reveals significant insights into model performance, feature effectiveness, and the influence of hyperparameters. Table 4.5 Shows comparative performance metrics of the Logistic Regression:

Table 4.5 Comparison Results for the Logistic Regression by Features

Features	Accuracy	Precision	Recall	F1- Measure
tfidf	0.9241	0.9263	0.9241	0.9247
bow	**0.9279**	**0.9312**	**0.9279**	**0.9286**
word2vec	0.8071	0.8177	0.8071	0.8051
glove	0.8620	0.8610	0.8620	0.8613
fasttext	0.9002	0.9016	0.9002	0.9004
hashing	0.8764	0.8807	0.8764	0.8777
char_enc	0.3835	0.3767	0.3835	0.3716
unigrams	**0.9279**	**0.9312**	**0.9279**	**0.9286**
bigrams	0.7455	0.8331	0.7455	0.7591
trigrams	0.4555	0.8184	0.4555	0.4501
fourgrams	0.3056	0.8447	0.3056	0.2490
n-grams (1-4)	**0.9263**	**0.9303**	**0.9263**	**0.9271**
n-grams (1-2)	0.9260	0.9300	0.9260	0.9269

Table 4.5 provides a comparison of logistic regression performance across various feature extraction methods. The BoW and unigrams achieved the highest accuracy (0.9279) and F1-score (0.9286). TF-IDF and n-grams (1-4) also performed well with accuracy around 0.9260-0.9263. Word2Vec and GloVe showed moderate performance with accuracy of 0.8071 and 0.8620 respectively, while FastText had a slightly higher accuracy of 0.9002. The Hashing method achieved a respectable accuracy of 0.8764. However, character encoding and higher-order n-grams (trigrams, fourgrams) showed significantly lower performance, with accuracies ranging from 0.3056 to 0.4555. This analysis highlights that simpler features like BoW and unigrams are highly effective for this task, while complex methods like character encoding and higher-order n-grams are less suitable for logistic regression in this context. Figure 4.5 summarized in Table 4.4 displays the highlights the effectiveness of the Logistic Regression algorithm's feature combinations:

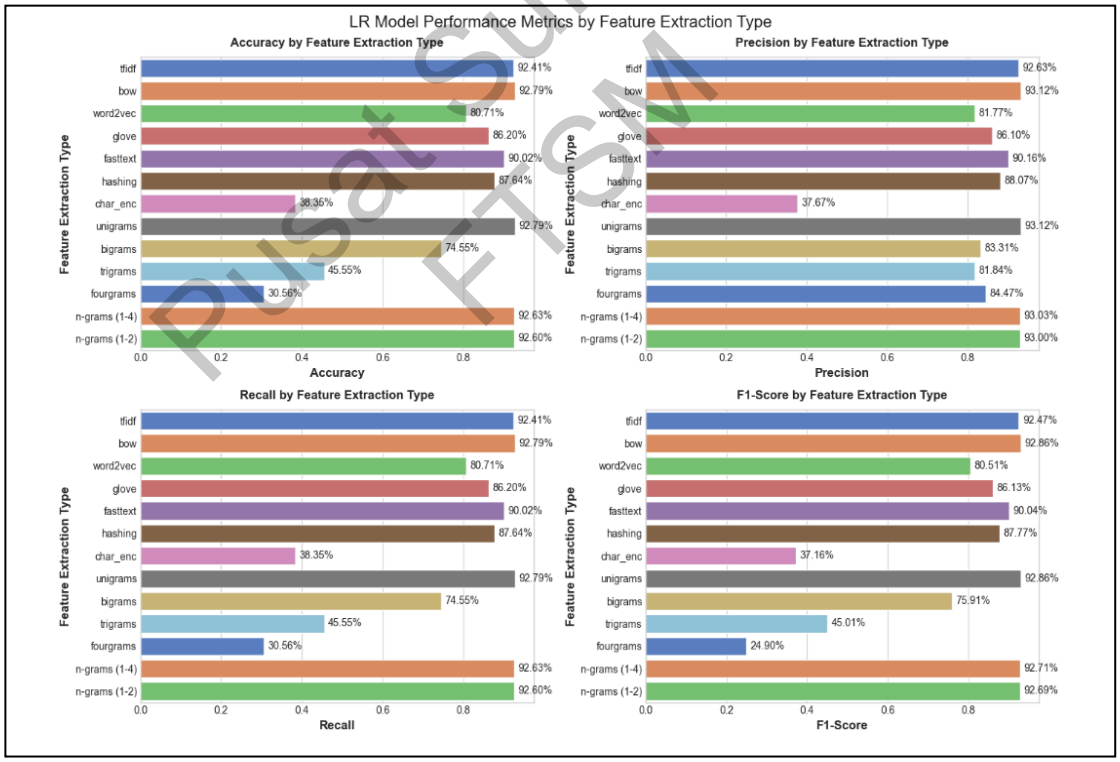

Figure 4.5 Comparison Logistic Regression Performance by Features

Figure 4.5 present the performance metrics of the Logistic Regression (LR) including accuracy, precision, recall, and F1-score. TF-IDF and BOW/unigrams show the highest performance across all metrics, with TF-IDF leading slightly (accuracy:

92.41%, precision: 92.63%, recall: 92.41%, F1-score: 92.47%). FastText and hashing also perform well, particularly in precision and F1-score.

Conversely, character encoding (char_enc) and higher-order n-grams (trigrams and fourgrams) exhibit poor performance, with fourgrams having the lowest accuracy (30.56%) and F1-score (24.90%). This suggests that simpler feature extraction methods like TF-IDF and BOW/unigrams are more effective for LR models, as they balance dimensionality and feature informativeness without overwhelming the model. Complex methods such as character encoding and higher-order n-grams may introduce noise and high dimensionality, leading to decreased model performance. This analysis underscores the efficiency of straightforward feature extraction techniques for LR in text classification tasks.

The analysis of various text feature extraction techniques using logistic regression reveals notable differences in performance metrics across different methods. Among these, the 'bow' (Bag of Words) and 'ngrams' (including unigrams) models exhibited the highest performance. Here's an in-depth look into why these variations occur:

1. **High-Performance Features:** TF-IDF, BOW, Unigrams, Ngrams(1-4), Ngrams(1-2) These features generally provided high accuracy, precision, recall, and F1-score across evaluations. Their effectiveness can be attributed to the comprehensive representation of text data they provide, capturing both the frequency and the context of words in documents. The application of n-gram models up to trigrams helps in preserving sequential information, which is crucial for understanding the semantic structure of texts.

2. **Moderate-Performance Features:** Word2Vec, GloVe, FastText These embeddings capture semantic meanings and have shown moderate performance. Their performance depends heavily on the dataset's alignment with the training corpus of the embeddings. For instance, embeddings trained on generic sources may not perform as well on domain-specific tasks without fine-tuning.