Deep Learning based Approach for Effective Hate Speech and Offensive Language Classification

Submitted by

Md. Rakibul Islam

ID: 202-35-3110

Department of Software Engineering

Daffodil International University

Supervised by

Mr. Nuruzzaman Faruqui

Assistant Professor

Department of Software Engineering

Faculty of Science and Information Technology

Daffodil International University

A thesis submitted in partial fulfilment of the requirement for the degree of B.Sc. in Software Engineering

Department of Software Engineering

DAFFODIL INTERNATIONAL UNIVERSITY

SPRING-2024

APPROVAL

APPROVAL

This thesis titled on "**Deep Learning based Approach for Effective Hate Speech and Offensive Language Classification**", submitted by **Md.Rakibul Islam (ID: 202-35-3110)** to the Department of Software Engineering, Daffodil International University has been accepted as satisfactory for the partial fulfillment of the requirements for the degree of Bachelor of Science in Software Engineering and approval as to its style and contents.

BOARD OF EXAMINERS

10/7/24

Dr. Engr. AKM Masum
Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Chairman

16/7/24

Dr. Marzia Ahmed
Assistant Professor
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 1

Md. Rajib Mia
Lecturer
Department of Software Engineering
Faculty of Science and Information Technology
Daffodil International University

Internal Examiner 2

Dr. Md. Manowarul Islam
Associate Professor
Dept. of Computer Science and Engineering
Jagannath University

External Examiner

DECLARATION

DECLARATION

I hereby declare that, this thesis titled **"Deep Learning based Approach for Effective Hate Speech and Offensive Language Classification"** is done by me under the supervision of Prof. Mr. Nuruzzaman Faruqui, Assistant Professor, Department Of Software Engineering, Daffodil International University, towards the partial fulfilments of my work. I also state that this project has not been submitted anywhere in the partial fulfilments for any degree of this or any other University. I am also declaring that neither this thesis nor any part therefore has been submitted else here for the award of Bachelor or any degree.

Certified by,

Mr. Nuruzzaman Faruqui

Assistant Professor

Department of Software Engineering

Daffodil International University

Submitted by

Md. Rakibul Islam

ID: 202-35-3110

Department of Software Engineering

Daffodil International University

ACKNOWLEDGEMENT

First and foremost, I would want to thank Almighty Allah for His divine favour, which has allowed me to finish my undergraduate thesis.

My supervisor, Mr. Nuruzzaman Faruqui, an assistant professor in the software engineering department at Daffodil International University in Dhaka, has my sincere gratitude. His profound understanding and direction in the field of "Deep Learning" were crucial to the accomplishment of this thesis. I am incredibly grateful for his unwavering empathy, academic leadership, continuous inspiration, careful oversight, helpful criticism, priceless advice, and the evaluation of multiple manuscripts that he lovingly rectified at each stage. As well as the other instructors, staff, and faculty of Daffodil International University's SWE Department, I would also like to express my profound gratitude to Dr. Imran Mahmud, the chairman of the department under the Faculty of Science and Information Technology that specialises in software engineering. Their considerate help and encouragement were essential to finishing my task.

Finally, I just want to say how much I appreciate my parents' constant love and support.

TABLE OF CONTENTS

LIST OF FIGURES

ABSTRACT

As hate speech is becoming more prevalent on social media platforms and has a negative impact on both individuals and groups, it is crucial to identify and mitigate hate speech in online environments. To solve this issue, the study suggests a method for automatically categorising tweets into three groups: Hate, Offensive, and Neither. We perform an extensive process of data collecting, preprocessing, and augmentation using a publicly available tweet dataset. Social media is used to collect initial datasets, which are then carefully cleaned to eliminate noise and unnecessary information. Synthetic data generation is used to balance the dataset to overcome the prevalent problem of class imbalance in hate speech detection placement. To generate an innovative deep learning model that will improve the high accuracy detection and classification of hate speech and abusive language. We conduct experiments to construct LSTM and Bi-LSTM models along with a transfer learning strategy based on pre-trained language models, DistilBERT and BERT. While the BERT model performs better at capturing contextual information, we focus special attention to it. The performance of the models is greatly enhanced by the addition of synthetic hate speech data. After assessment the model on test data, we achieve an accuracy of over 91 percent. Additionally, this leads to better performance in the hate speech class. use of BERT's bidirectional training approach, which improves the capacity for local contextual understanding and classification. The work also advances the area by investigating advanced techniques for producing synthetic data, which results in a more evenly group instruction dataset. This method not only increases the generalizability of the model but also provides the way to more effective management of sensitive and contextually complicated hate speech. Social media platforms and brands could more effectively regulate and lessen the negative effects of toxic content by utilising the study's strong and scalable hate speech detection technology. By shielding users from offensive words, this method fosters a more secure and friendly online community. Furthermore, the study provides a standard for subsequent research, guiding the creation of deep learning models that are more effective at detecting hate speech in a variety of languages and cultural contexts.

Keywords: Deep learning, hate speech detection; offensive language detection; synthetic data; transformer-based model; BERT;

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

This chapter will delve into the study by providing an overview of the context, specifically focusing on hate speech in online environment and its severity for businesses and society. We will cover the need of this research, we will outline what motivates us and how in urgent situation an automatic system for identifying harmful content safekeeping could overcome limitations inherent to traditional ways. The main research question is whether deep learning-based approach can be exploited to improve hate speech detection. We will highlight the research gap saying that few of powerful methods are available to be used and most small and imbalanced dataset urancesleting style. The novelty of our research goals to construct and validate a model that will generate an incremental step increase in accuracy scores for both diagnosis, addressing these gaps. This research aims at testing the performance of our models on a social media evaluation, as well as assessing its generalisation capabilities and practical feasibility across several settings. The closure of the association between this study and my thesis will hence establish a benchmark for understanding hate speech detection from both technical and practical perspectives which can be helpful to guide the research community in digital communication & safety in general.

1.2 Background of the Study

In this era of digital communication and ads, Companies are more concerned about the hate speech content online. Hate Speech:Messages that contain abusive language or expletives, are derogatory, degrading and disrespectful about any individual other than the user. Brands can control what they post to their websites and social media, but not what users say about them in comments or mentions on those said platforms [1]. We have seen brand marketing roles morph over the past decade into Community Manager, or Social Content Managers reflecting a need to listen and monitor how our brands are being talked about online. Historically, these roles have been around community management (commenting back to your audience), real-time response and post-for-persuasion around brand content. This demand has led to rise of social monitoring platforms like Olapic, Meltwater and Stackla that were designed to increase online engagement, brand advocacy etc [2].

But the rise of hate speech has expanded brand marketing's remit. The Community Managers and positions alike who fulfil similar roles in our times must keep a vigilant watch on the digital channels of their brand to ensure no introduction of hate speech. Identifying hate speech or offensive language can be a labour-intensive task without the correct tools, often time-consuming and resource-heavy. Community Managers often lack the capacity to track all brand-specific mentions for hate speech in a comprehensive way. This could require more resources to keep pace with increasing demand for digital content control. But of course, these are the same humans who need enough energy to get themselves out of bed in the morning and contain inherent flaws— bad handwriting, reading abilities that lag after 100 pages, personal feelings about what things are a stressor — it would be inappropriate at best to consider them completely reliable for determining whether or not hate speech has occurred. By using artificial intelligence (AI), this work is done automatically, minimizing labor and enhancing the efficiency and accuracy of detection[2,3].

This is an algorithm that by using it you can automatically detect hate speech in Twitter, help brands to manage the risk of extraction and notes like black stains on their image, save time will manual monitoring e prevent acts of hatred. The rise of social media has allowed for more significant communication between individuals, now leading all the way to the public expression of thoughts. Probably the most universally powerful one being, "Words are perhaps the only thing in this world that is magical". Either way, they are words that can be used build up or destroyed [5]. Words are things, I am convinced. They get in your Wallpaper; they get into you Rugs. As social media popped up, it made people outspoken as they could voice their thoughts and opinions on global platforms about anything or anyone. While this transparency has popularized free speech in social networking sites (SNS), it also lead to what is often dubbed at an open environment where toxic content can thrive, like hate speech or offensive language [3].

The EU defines hate speech

Such as any public act that could incite violence or hatred toward a group of persons or towards an individual on the grounds of their race, place of origin, colour, religion or descent [4].

Hate speech and offensive language on the internet and via social media have, however, grown exponentially. These toxic contents are different by how subjective their attack is. Different countries have different legislative norms to deal with this]-->[4].

The US, by contrast, holds hate speech to be protected under free-speech laws, with other countries enacting their own laws banning any form of it. Social media giants such as Facebook and Twitter have been criticized for their relaxed approach to toxic speech. This time, on may 31, the EU forced facebook, google and microsoft to agree with twitter to a code of conduct obliging these US owns companies to remove illegal hate speech published by their services within 24 hours [5].

Examples of Hate Speech

Murder or Suicide Threats: A threat is any statement conveying the intention and capability to do physical harm or kill an identifiable target. Because these pose direct threats to personal safety. Eg: “u gon get beat up bitch imma catch u” [4].

Wishful thinking: Content that wishes, hopes or tells people to physically hurt or maybe go kill some loser with whom the poster is having an argument. This includes wishing death or injury on others. For Instance: “I wish you get into a car accident [4].

Rousing Alarm against a Community: Statements that cause fear or hate to some object to protected communities, a race or religion etc.. That could range from stereotyping a group as dangerous or harmful. For example: “All immigrants are criminals,” [4].

Hateful Imagery: Images, logos, banners, icons that promote the hate against any religions communities genders races disabilities or sexual orientations. Symptom: Images of racist caricatures or recognized hate group symbols [4].

1.3 Problem Statement

It has now become a major concern with the advent of social media platforms and multiple ways for online communication, the involvement of hate speech and offensive language. These forms of communication are destructive because they not only threaten the mental health and wellness of individuals but also destroy any semblance of online community or social congruity [2]. Content

moderation has always been a tricky thing to get right mainly because the methods of doing it are so traditional (human or manual) and does not cope well with mass scale and often leads to bias. The major challenge is how to detect the hate speech and offensive language with a high level of accuracy in different online environments, which are most of them very dynamic. These existing methods suffer from the complexities and subtleties of natural language, such as context, slang, and rapidly-changing expressions of hate [3,4]. Even worse, imbalanced datasets and the fine line between offensive language and hate speech make this task even harder to accomplish. This research aims to overcome the challenges faced in classifying hate and offensive texts by applying a deep learning technique using Bidirectional Encoder Representations from Transformers (BERT) [5]. Context and semantics of text is something that BERT excels at understanding which can result in an accuracy change. This study sets out to construct a powerful and scalable solution that can be released in production time to keep platforms healthy, both literally and figuratively [4].

1.4 Research Question

The research question was

Q1. How can we design and implement a deep learning-based approach to improve the effectiveness of hate speech and offensive language classification, taking into account model architecture, feature representation, and dataset augmentation?

Q2. How can Bidirectional Encoder Representations from Transformers (BERT) be effectively utilized to improve the accuracy and reliability of hate speech and offensive language detection in online content?

1.5 Research Objective

In this research, we aim at building an automated and expedited system to identify hate speech and offensive language by using contemporary deep learning models. In general, the study determines to:-

Create a model to Make Hate speech and offensive language detection Robust with BERT, Use the power of BERT (Bidirectional Encoder Representations from Transformers) for hate speech and offensive languages so as to build an effective identification tool which can accurately assign them in respective categories. Take advantage of BERT's bidirectional training to better capture language context and nuances.

Synthetic Data Generation to Address Data Imbalance, In Hate Speech Datasets Since one of the problems that is in nearly all hate speech datasets is data imbalance, creating synthetic data will be an important part of this research. Using platforms like Gretel. Additional synthetic data will be generated for the underrepresented class to balance out the dataset so that we have a good mix of examples in order to train and test our model.

Optimize the Model Performance with NLP Data Augmentation — Apply data augmentation methods to cover wider range of example in training dataset. Variations of the existing data as supplementary examples will be generated using methods like synonym replacement, random insertion

1.6 Motivation of The Study

The explosion of digital communication and social media helped make it among the biggest change in human behaviour over the last 5 years. This shift has been instrumental in creating a world that has never had more connectivity and freedom of expression; but also with the price of being able to freely share offensive language, spread misinformation and increase online hate speech. In all of these forms, toxic communication represents a serious menace to the individual and communitarian welfare — fostering psychological damage, social discord and even physical violence.

The spread of hate and defamatory speech on social media, as well as other online spaces can have damaging psychological affects, exacerbate a social crisis or in extreme cases even incite violence.

Through the development of top-tier detection systems, we can help to promote holistic and healthy online communities where users are free from unfair intimidation or abuse.

User-generated content that includes hate speech or offensive language on their platforms is extremely bad for the brands and organizations. This can not only damage the reputation, but also public back-lash and create a financial liability — all because one subject has become highly weaponized. Brands are able to protect their image, stay within legal lines and create a positive environment for their audience with this kind of system in place.

Not only is the manual moderation of this amount of content time consuming, but it is also very liable to human error and bias. Utilising advanced deep learning models to automate the detection can drastically improve efficiency and accuracy of content moderation, offloading human moderators by having a consistent enforcement of community guidelines.

One emerging issue arising from these challenges, is concern from brands and organizations regarding the presence of hate speech around their digital content. While they can moderate their own official communications, there is a limit to how much user-generated content that appears on their platforms they can restrict. As a result, this limit may have other negative effects on your brand's reputation and worst case it can have legal implications and customers can not trust you.

Conventional ways for detecting and monitoring hate speech are time-consuming since they necessitate plenty of human resources. In recent times, the roles of Community Managers and Social Content Managers —already monitoring for hate speech— have been buried under avalanche(s) of content that must be reviewed. Furthermore, human reviewers can get tired and have their fair share of biases and inconsistencies, leading to discoveries going unnoticed or incorrect categorization.

Given these problems, the present work is motivated by the need for an autonomous and efficient system which can accurately find the instances of hate speech and offensive language. Our model aims to understand the semantics and context of the language using advanced deep learning technique specifically BERT (Bidirectional Encoder Representations from Transformers) resulting high-grade identification skill against toxic speech. The fact that BERT can capture context from both directions in a sentence made it particularly well-suited for this task—which requires an extensive understanding of the subtleties involved in hate speech and offensive language.

1.7 Research Scope

Data Collection and Preparation This research kicks off by collecting an expansive data set from platforms like Kaggle which contains different examples across hate speech, offensive language and neutral content. The dataset will be preprocessed — cleaned, tokenized and stopwords removed — for training and testing the models.

Create Synthetic Data, One way they tackle the imbalance of data in this experiment is through the creation of synthetic data which can also be accomplished using something like Gretel. ai. It overcomes this through creating more instances of hate speech to supplement the dataset, so as to significantly improve its training and model performance.

Therefore the Model development part of the research is only a candidate model for hate speech detection using BERT. It will be a hate speech and offensive language classification task specific fine-tuning of the model.

Implement Model Progress Saving, In order to prevent wasting a lot of time, the standard way to save the progress of the model at some time intervals will be implemented. This means that the training process can be stopped during any episode and resumed, without losing any data and enabling very long duration training sessions.

1.8 Summary

This Background We have already talked a little about why we are interested in hate speech detection, i.e., the migration of human activity to digital space and problems managing harmful content. Here, we have tried to answer our research question on how better deep learning model can be designed using BERT for the classification of hate speech and offensive language. The researchers found their own gap in literature and they realized that the current models are not empowering enough and there is a dearth of solid solutions. Here in this series of articles, we focus on our research objectives to build a strong detection model (natural language processing task), address class imbalance using artificial data generation and improve read world performance with NLP data augmentation. Our investigation focused on the dataset collection, synthetic data generation, and model development as well as mechanisms for efficient training workflows. By doing so, this chapter has presented a solid groundwork for approaching the

technical and applied framework that was of interest, not just our study but in digital communication safety at large.

In the next chapter, existing related papers are discuss with there limitations and methodology.

CHAPTER 2
LETARETURE REVIEW

2.1 Introduction

The traditional problem of hate speech and offensive language detection has been extensively researched, using a broad range of machine learning and deep learning models with their characteristic advantages as well as constraints. The goal of our research is to solve these problems by using an up-to-date deep learning architecture, that is hereby BERT. We want to expose our model to the concept of understanding context and nuances by taking advantage of BERT's bidirectional training approach. In addition to this, we tackle the data imbalance by generating synthetic data and enhance performance with advanced data augmentation techniques. In this course, we will help to provide more robust and efficient hate speech support detection system for social media platforms.

2.2 Previous Literature

Hate speech and offensive language detection is one of the most famous areas where a lot of machine learning, deep learning techniques are applied to solve with its unique pros and cons for every algorithm. Gaydhani et al. The authors of (2018) used logistic regression, naive bayes and SVM to obtain high overall accuracy but lower precision in correlation with hate speech meaning that there is still room for improvement [1]. Safaya et al. The work by (2020) used a mix of BERT, Bi-LSTM, CNN-Text and SVM with TF-IDF for an F1-Score of 0.814 to call out the need for more balance between precision and recall earlier in this space [2]. Mulenga et al. (2018) employed CNNs but observed a high misclassification error rate, apparently indicating that the mere classification of more nuanced and context-dependent language in social media texts is quite challenging [3]. Akanksha Bisht, Annapurna Singh, H. S. Bhadauria, Jitendra Virmani and Kriti (2018). Detection of Hate Speech and Offensive Language in Twitter Data Using LSTM Model [4]. Alshalan et al. (2020) performed experiments with CNN, GRU, CNN+GRU, and BERT to obtain an F1-Score of 0.70 but still misclassified a significant amount of non-hate speech as hate-speech [5]. Maha Jarallah Althobaiti (2022). BERT-based Approach to Arabic Hate Speech and Offensive Language Detection in Twitter: Exploiting Emojis and Sentiment Analysis using BERT, machine Learning model (SVM, logistic regression) [6].Wei et al. In the latest use case, Choi et al.(2021) used BiLSTM and BERT with better results like 92% in terms of test accuracy but

mentioned that even powerful tools (BERT for some Ginseng indexes) failed to handle diverse linguistic patterns [7]. Althobaiti (2022) used similar ones those as previously considered: BERT, SVM, and logistic regression with acceptable F1-Scores of 84.3 % for offensive language and 81.8 % for hate speech that also suggests the necessity of better model tuning [8]. Bisht et al. (2020) used LSTM, Bi-LSTM, and RNN achieving 86% accuracy but underfitting. Derczynski et al. (2019): A comparison of a shared task using BiLSTM with logistic regression methods by the same group using the English and Danish language examples, shows significant variance between languages; they achieved F1-Scores of 0.74 in English and 0.76 in Danisg respectively [4]. Various works show great potential of models like ULMFiT and multilingual BERT, fine-tuned on code-mixed datasets bolsters model performances but still struggle to overcome misclassifications because of contextual variations. Herwanto et al. Murfiati, I. (2019) investigated Indonesian social media data and demonstrated that human labeling for social media classification is subjective, hence making it sometimes very difficult to classify the comments and further showed by using pre-trained word vectors can significantly increase the accuracy in classifying them [9]. While improvements are being made, the difficulty of dealing with context specifically in terms of sarcasm and semantic nuances of hate speech still persists.

In this paper, we take a step forward and apply BERT as an advanced deep learning architecture to build a powerful detection model which is able to handle those problems. BERT being bidirectional training fares more suitable for understanding the context and nuances which help in making improved classification performance. Synthetic data will be generated using tools like Gretel to combat the problem of data imbalance. ai, where we work very hard to ensure that our training set is unmistakably diverse and representative. It allows the model to generalize better and detect more accurately. We will further optimize and tightly-integrate more advanced data augmentation techniques such as synonym replacement, random insertion and WordNet-based augmentation into the training pipeline, in a principled way to boost performance without substantial noise or redundancy. Our future research will refine these methods and look into new ways of creating synthetic data that improve the robustness of our models even more. In order to cope with the continuously changing nature of hate speech and offensive language on social media, ongoing iterative fine-tuning evaluation process has been essential for BERT.

Our approach in solving these problems is by harnessing the powers of BERT (Bidirectional Encoder Representations from Transformers) to develop a state-of-the-art detection model. We continue to add in the Bidirectional training strategy of BERT where we fine-tune the model so that it understands language context better which will enhance classification accuracy. We then create Synthetic data using platforms like Gretel to address the problems generated above due to imbalance. more diverse and representative set of training examples. This can improve the generalization of a model and makes it detect objects much better. Also, in order to improve the performance of models we are using more advanced data augmentation techniques such as synonym replacement, random insertion and WordNet based augementaion. From such challenges we have seen with these techniques, while tuning them and cautiously embedding into the training pipeline has potential to improve performance without much noise/redundancy.

In the future studies will aim to enhancing model robustness by refining data augmentation techniques and developing novel mechanisms for synthetic data generation. Further, given the nuanced and changing nature of hate speech/offensive language on social media, it will also be essential for experimental evaluation to continue to optimize BERT's fine-tuning process.

2.3 Research gap

Author Name	Methodology	Research Gap	Objective
Akanksha Bisht, et al.,[4]. 2020	LSTM, Bi-LSTM, RNN.	Only an accuracy of 86% Underfitting happening when early occurs 98.24%	Develop a Robust Detection Model.
Leon Derczynsk, et al.,[1]. 2019	Logistic Regression, BiLSTM.	Offensive language in English F1-score of 0.74. Offensive language in Danish F1-score of 0.70.	
Aditya Gaydhani,et al.,[4]. 2018	Logistic Regression, Naive Bayes, SVM	Only 93.6% accuracy upon evaluating it on test data. Hate speech accuracy 79.4 %.	Address Data Imbalance Issues through Synthetic

Raghad Alshalan,et al.,[2]. 2020	CNN, GRU, CNN + GRU, and BERT	F1-score of 0.79 only. misclassified non-hate tweets.	Hate Speech Data Generation.
Sinyangwe Clement Mulenga, et al.,[2]. 2018	CNN	The study's limitations include the misclassification of many hate tweets. Only accuracy of 91%.	
Ali Safaya,et al.,[2].2020	BERT, Bi-LSTM, CNN-Text, SVM with TF-IDF	F1-Score of 0.814 only.	Enhance Model Performance with NLP Data Augmentation.
Maha Jarallah Althobaiti.2022	BERT, SVM, logistic regression	Offensive language & Hate speech detection F1-score Only 84.3% & 81.8%	
Bencheng Wei, et al.,[5]. 2021	BiLSTM, BERT, DistilBERT, GPT-2	Only 92% accuracy on the test data set.	

2.5 Summary

This chapter Hate speech, using deep learning methods has been widely studied with different strengths of weakness. Although things have slowly been improving, we are still a very long way off from solving this context handling tasks using language itself can be challenging due to the intense levels of sarcasm and semantic nuance. We address these challenges by utilizing BERT's pre-trained context understanding ability with advanced deep learning architecture to improve classification performance. To deal with data imbalance, we create synthetic data and apply high-level data augmentation methods. Further fine-tuning on BERT will primarily focus on improving the detection model for social media and ensure that these methods are refined.

The next chapter will explain the complete process of our research in next chapter from collecting dataset to data exploration Learn how to perform data cleaning, address the data trends and imbalanced nature that may be present within your input dataset as well as effective & efficient ways of pre-processing techniques such stopwords removal or tokenisation. On the same note, the

chapter will look into synthetic data generation for dataset diversity and dat augmentation techniques to strengthen model robustness. We will then, in turn, describe the data splitting process and training of different models like — LSTM/BiLSTM along with additional transfer learning techniques. Last but not least, we will detail the empirics followed by discuss all these five steps in connection with the general aims of our thesis session-based hate speech detection.

CHAPTER 3
METHODOLOGY

3.1 Introduction

This chapter discusses the complete pipeline of data preparation and utilization for hate speech and offensive language detection. To get started, we gather data from different sources of text for building a strong base to work on our further analysis about NLP. After this, the next step is what we call data exploration to know about the type of data which was collected. This step is all about turning messy data into something useful— removing noise, inconsistencies, and useless information that can hinder quality of the dataset. We will next look at trends in the data, as well as tackling imbalanced data problem — a prevalent issue when it comes to hate speech detection (i.e. we have substantially more of non-hate speech instances than that of hate). This is being mitigated via different data preprocessing techniques such as stopwords removal, tokenization of text that converts text into a more suitable format for modeling. So next up, we are going to generate some synthetic data which can be done on platforms such as Gretel. ai to make the dataset further balanced and representative. We further diversify the training data using synonym replacement and random insertion methods as Data augmentation techniques that are applied to prevent considerable noise. Next, it divides the data into Train / Validation / Test set to generalize and improve the robustness of this model. Training the Model: We tried different architectures including LSTM, BiLSTM and took advantage of transfer learning with pre-trained models i.e. BERT They have been pretrained on how to use the context and intricacies of human language so that they can then fine-tune them for identifying hate speech properly. It is later followed by the experimental setup and the used methods in proving the effectiveness of these models. Here we look at the variations and what needs to be done in order to keep on improving. We also reflect on the alignment of this work to the larger goals (described later in Chapter §5) of our thesis, and how it suits towards parallel advancements we have made towards combating hate speech in digital communication. We hope to address this by developing a reliable working model that can contend with the nuances and changing nature of hate speech on social media platforms — while incorporating these factors. This chapter paints a trail of breadcrumbs from raw data to analysis

model output, laying the groundwork for other chapters that will go into more detail on individual methods and results.

3.2 Proposed Methodology

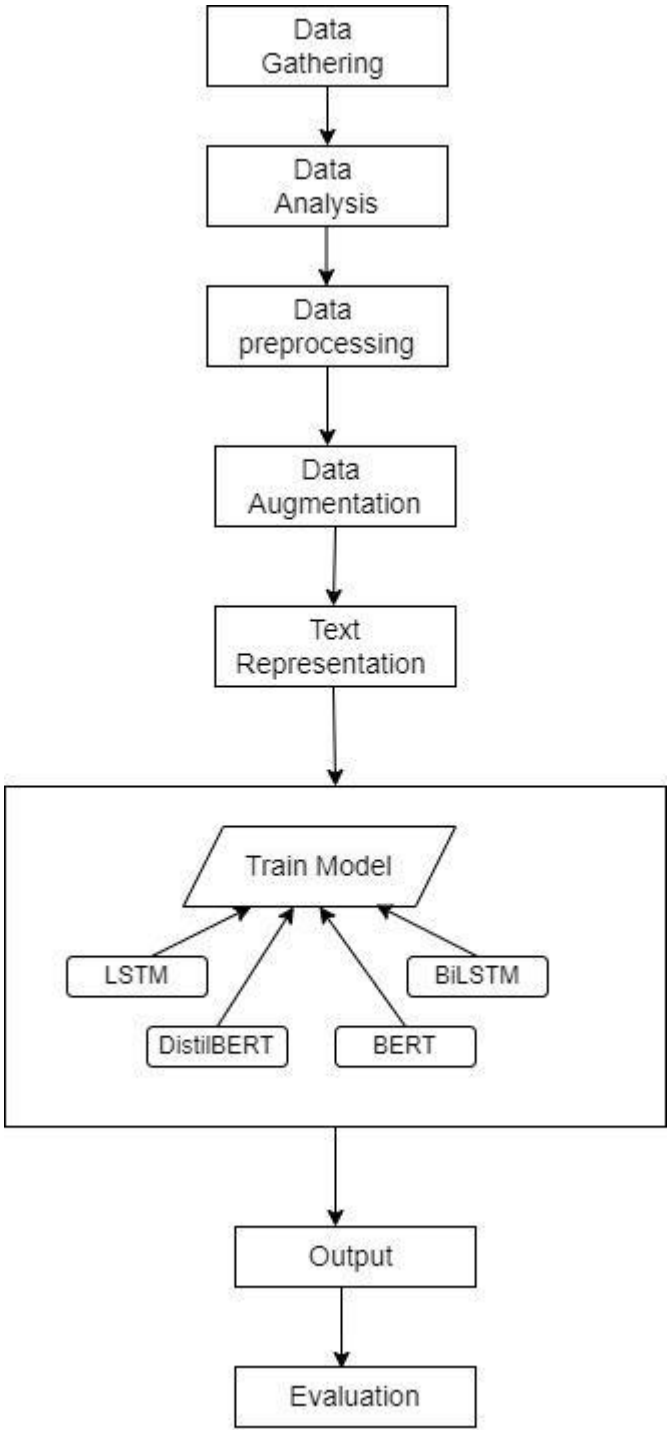

Figure 3.1: Proposed Methodology

3.3 Dataset Collection

In the context of Natural Language Processing (NLP), the performance of deep learning models in hate speech and offensive language categorization tasks highly depends on the dataset quality as well as diversity in both training and testing. The Title of my thesis is Deep Learning Based Approach for Effective Hate Speech and Offensive Language Classification, I got the data from kaggle which is a well known site and has multiple good quality data set.

Kaggle is a popular dl data science competition and dataset site It also offers a variety of datasets, which are already further prepared most times and you can start using them directly without the need spend time on preparing your data. We will take the Hate Speech and Offensive Language Dataset which is a specific corpus just for identification of hate speech and offensive language, bear in mind that there are many other ways to explore this problem.

We used 24783 tweets for this model, and it consists of Graphical representation of total dataset:

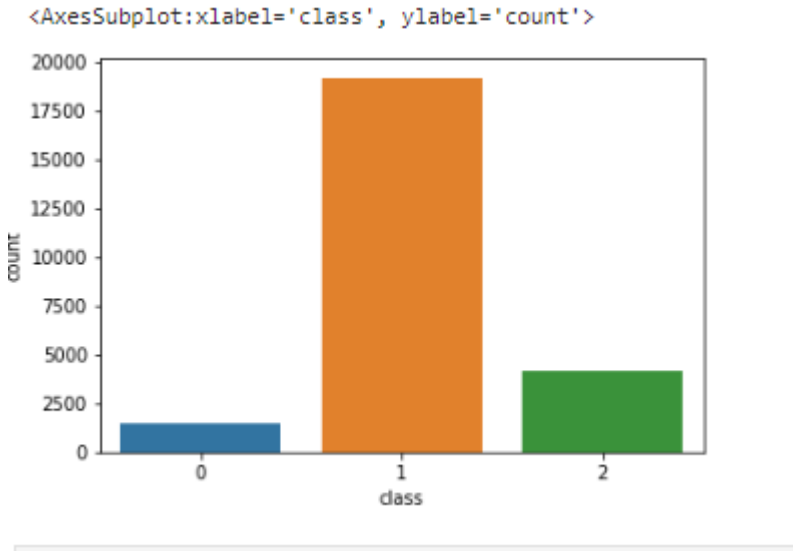

Figure 3.2 : Data Visualization

This dataset is a selection of hate speech tweets, offensive language tweets and nor tweet labeled by human. It has the following features:

TweetId: an id given for each tweet.

Tweet Text:The caption of the tweet

Class label : the tweet is hate speech, offensive language or neither

The selected dataset is appropriate for the research goal since it concentrates on hate speech and offensive language in particular. This means that the model will be trained and tested on data that is more representative of those real-world scenarios. Of course, the Kaggle dataset comes with some of the highest-quality annotations — labelers have gone through this data to make these labels. Finally, diversity in the language usage, context and expressions occurring within the dataset enables the model to generalize better across different kinds of hate speech and offensive content.This makes benchmarking with other models and techniques more feasible using a dataset that many people are familiar with. This dataset has been used by many researchers in the field which allows results from different models to be compared and validated. However, a lot of — plus free to use— well described data sets are available on Kaggle. Therefore, seamless integration into the research workflow is critical to allow for hands-on model development and analysis, rather than spending time overcoming data collection challenges.

	count	hate_speech	offensive_language	neither	class	tweet
0	3	0	0	3	2	!!! RT @mayasolovely: As a woman you shouldn't...
1	3	0	3	0	1	!!!!! RT @mleew17: boy dats cold...tyga dwn ba...
2	3	0	3	0	1	!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby...
3	3	0	2	1	1	!!!!!!!! RT @C_G_Anderson: @viva_based she lo...
4	6	0	6	0	1	!!!!!!!!!!!! RT @ShenikaRoberts: The shit you...
5	3	1	2	0	1	!!!!!!!!!!!!!!!!"@T_Madison_x: The shit just...
6	3	0	3	0	1	!!!!!!"@__BrighterDays: I can not just sit up ...
7	3	0	3	0	1	!!!!“@selfiequeenbri: cause I'm tired of...
8	3	0	3	0	1	" & you might not get ya bitch back & ...
9	3	1	2	0	1	" @rhythmixx_ :hobbies include: fighting Maria...

Figure 3.3 : Tweet Representation

3.4 Data Preprocessing

Data cleaning is the process of extracting useful and clean data, where useful part corresponds to input for predictive modelling and clean part represents already pre-processed structured data. In case of textual data like we have in this project, deal with noises and irregularities — they can deteriorate how the classification model will perform. Data cleaning is something you typically do after the data collection process; The steps can be different with each dataset, but it is crucial to create a template for this task so that we are doing it properly.

Data cleaning is perhaps the most important part of data science in machine learning. It is one of the best parts of your model creation process. Pre-requisite of my thesis is data cleaning was done on the dataset to preparing it for training and evaluation.

Trends in Data and Unevenly Distributed Classes

Tweets were categorized by groups of 3 to 9 CF users, although the majority of tweets (92%) were rated by a group of 3 CF users. The class assigned to the tweets with which it got maximum numbers of votes from CF users, is given as:

Class 0 : 7% of the tweets Hate speech

Class 1: Offensive Lang (76% of tweets)

Class 2, Both: For 17% of the tweets.

Approximately 70% of the tweets were within class 1. This mean the dataset is imbalanced. We also notice that the words used most frequently for class 0 and class 1 are similar in nature, so they are almost overlapping each other, hence making it harder for the model to properly classify the tweets into class 0 and class 1. Whereas class 0 and class 1, the common words are quite different than class 2.

3.4.1 Remove Punctuation

Punctuation Marks or characters are not giving any meaning to words that why its no use for the classification task. This eliminates these elements, which in turn results in reduced data noise and the rest of the model focuses on what actually being said instead.

Removing Additional Whitespace: By removing additional whitespace we can help the text data to be maintained in an orderly fashion. It prevents the problem of some words being unnecessarily separated or treated as separate words, caused by starting whitespace, ending white space and extra spaces in between words.

Make Text Lowercase: Standardizing text to lowercase for the sake of consistency. It is necessary to perform this step since for the model it considers all occurrences of 'Hate' and 'hate' as two different words, so in short we are only minimizing the learning process.

3.4.2 Remove Stopwords

Stopwords are very frequent words that appears in a natural language text. In English, some examples of stopwords are "and", "the", 'is', 'in', and so on. These are usually filtered out before working with the text for tasks of a higher dimension such as classification and sentiment analysisetrize in Text.

3.4.3 Tokenization

Tokenization is breaking the given text, for example: Tweets into words. Tokens are words, subwords or characters. Tokenization is important because it transforms raw text from Wikipedia and other sources into a format properly structured for machine learning models to be able to process it further. Steps in the tokenization process.

Word Tokenization

More literally, this is called word tokenization. Take for example the sentence "I love natural language processing" — it becomes a list of words ["I", "love", "natural", "language", "processing"]

Sentence Tokenization

More Sentence tokenization breaks text up into sentences. This is useful when you need to analyze text at the sentence level. Such as,[' I adore natural language processing., "It's fascinating!"]

Character Tokenization

Character tokenization: Allows you to break a text into single characters It can help with languages that have long chain words, or some specific NLP tasks. eg tokenised: ['H', 'e', 'l', 'l', 'o'] for word->"Hello".

3.5 Synthetic

synthetic data is a new breed of "generated" data that imitates the behaviors and distribution, and other even significantly synthesize entirely different from the original one. Synthetic Data It is used to supplement current datasets, especially when more data may be required for model training or dealing with unbalanced datasets. Methods to generate synthetic data include statistical methods and other advanced machine learning models like Generative Adversarial Networks (GANs) or transformer models such as GPT (Generative Pre-trained Transformer). The generation of synthetic data played a key role in strengthening the dataset and boosting either model performance. However, we did use Gretel after getting some starter data from Kaggle. hate speech based synthetic data using a AI platform.

3.5.1 Collecting Initial Data

The research used a dataset that was first published on Kaggle (a website where data scientists can find datasets) which is comprised of tweets marked as genuine hate speech, offensive things or neither. The dataset, however, an unbalanced class distribution that. Class 0 is hate speech and it was hugely underrepresented in the data set

3.5.2 Generating Synthetic Data

Gretel was used to create synthetic data in order to solve the above problem with dataset imbalance and overfitting. ai platform. It uses some really advanced machine learning models like GPT to generate very high-quality synthetic data. The steps included arrested the following.

3.5.2 Preparing the Dataset

They performed pre-processing steps on the initial dataset which was collected from Kaggle and shared, like tokenization, lemmatization and removal of stopwords after making sure that data is cleaned enough for synthetic generations

Using Gretel.ai for Synthetic Data Generation

Gretel.ai takes into use tools for generating synthetic data which tends to maintain the statistical properties of the original dataset. The steps involved in the process were:

Uploading the Prepared Dataset: The pre-processed dataset was uploaded to Gretel.ai.

Configure the Model: A GPT-based model was configured to generate synthetic data. The model was trained over the provided dataset for a duration of 4 hours to learn the patterns and properties of hate speech.

	count	hate_speech	offensive_language	neither	class	tweet
0	3	3	0	0	0	You're a faggot if you think #DuckDynasty is a...
1	2	1	1	0	0	Fuckin coons
2	2	2	0	0	0	RT @JoeBuffalo25: @morgan_b33p @Freddy_Garcia_...
3	3	2	1	0	0	I want to get rid of all the niggers and chink...
4	7	5	2	0	0	RT @TheBryanFlynn: The most racist, homophobic...

Figure 3.4 : Synthetic Data Visualization

3.6 Data augmentation

Data augmentation is a technique to oversample the data by generating new samples from original ones in similar distribution or slightly manipulation of the original data. To perform well in the classification task, we will need to make changes to the data and ensure as best as possible that the class label is preserved. We have highly imbalanced data and now want to expand the minority classes, name Hate Speech (0) & Offensive Language(1), as a measure of fixing this issue [9].

While in computer vision where data augmentation is common flip, rotate or mirror an image can be done without having the trouble of changing original label. Unfortunately it is a very different story in natural language processing (NLP). Among these is the fact that text augmentation is more difficult compared to other types of data due to the complexity of language. There are synonyms for every word, and if not, you can merely change one word (which is all that it takes to create different context).

The next few sections, we will discuss the fundamental techniques applied to achieve this for the "Hate" and "Offensive" labels.

Our main workhorse for such kind of synonym augmentation will be a Word Embedding-based Replacement BERT model and an out-of-shelf Synonym Augmenter called "WordNet". These models are going to find similar words (synonyms) used in tweets and replace these synonyms with the actual word. For these synonyms, we will use a similar approach and each one of them are selected on the basis of pre-trained embeddings. Here is an example using use of WordNet to find synonym of words.

Random Insertion

In this technique non-stop word in a tweet are randomly chosen, in place of this random words their synonyms can be added at some position. Examples: "You're a loser" and “You are such an absolute, complete and total.

Random Swapping

This technique randomly picks any two words in the tweet and then sends their positions to each other. Examples: "You are loser" to You loser".

Random Deletion

In this mechanism, one word will be randomly deleted from the tweet. You can simple remove the “such as” — e.g., from: "You are such a loser" to "You are a loser.".

Instead of using just a single data augmentation method, we combine multiple methods for our training dataset. We did notice that using Random Deletion, for example, leads the Augmented

tweets to be very similar with their original counterpart. The combination of these two methodologies and we add small randomness to the training data with most resemblances toward original meaning. Here is an example of a tweet before and after being augmented.

Original Tweet:

!!!!!!! RT @UrKindOfBrand Dawg!!!! RT @80sbaby4life: You ever fuck a bitch and she start to cry? You be confused as shit

Augmented Tweet:

!!! RT @UrKindOfBrand Dawg! RT @80sbaby4life: Ever been with someone and they start to cry? You'd be so confused

Example of Using Data Augmentation Note that the data augmentation is only applied to the training data and Test Data Split for tweet data We balance the classes first adding more of the minority classes— Hate and Offensive. These approaches help us to make a less bias data set for training the model.

3.7 Dataset Split

In this In this paper we exploit an empirical dataset of 24,783 counts of text data. Additionally, we created 2000 synthetic hate speech data points using Gretel to improve the dataset as well as handle possible class imbalances. ai and a model based on GPT. For the given problem, synthetic data helped in making the dataset more rich and increased the robustness of our classification models.

This joint dataset was spliced into training and testing parts. More specifically, training was done on 80% of the data, around 21,826 data points. The other 20%, around 4957 data points were kept aside for testing. The advantage of this split is that we teach the model N data points while also providing it with X number of testable examples.

3.8 Training the Model

We used the popular Adam optimizer to train our model, as it is known to be an efficient and reliable one. Finally, to fine-tune the performance of the model, we used Adam optimizer with a

smaller learning rate. Training was conducted to 7 epochs, with a batch size of 8. The model was built using compile () before the start of training. We chose categorical cross-entropy as the loss function for this multiclass classification task because it is designed to work with numerous output classes. Then we compiled the model and started training with the fit() method Using this method, the model iteratively adjusted its parameters to reduce the loss function and as such increases accuracy in hate speech, offensive language or Neither classification.

3.9 Mode

Deep learning has been transforming the field of Artificial Intelligence, especially in the area of Natural Language Processing (NLP), in recent years. It has performed same as girl learning tasks such as machine translation, text extraction, named entity recognition and speech recognition. Although classical machine learning models are interpretable when compared with deep neural networks, they come at a cost: manual feature engineering. Bag of words, N-grams, lexical features and meta-data are used in the case where shallow machine learning models like decision trees, Naïve Bayes, KNN or random forests etc… require the raw form of text for their analysis. Conversely, models like deep learning eat up the word embeddings and then complete these embedded words as input into convolutional neural networks (CNNs), recurrent neural networks (RNNs) such as type long short-term memories reparation matrixes of elements or transformers. Deep learning models can directly work on raw data so they are able to take a lot more of the context into account which translates to a vastly superior performance. In this paper, we aim deep learning modern approaches which used lately in the field like LSTMs and transformers..

Long Short-Term Memory (LSTM)

LSTM (Long Short Term Memory) networks are a type of Recurrent Neural Network (RNN), which have the capability to capture long-term dependencies in sequential data. Long Short-Term Memory (LSTMs) have been shown to produce superior results compared to the traditional RNN structure similar in that they can process long sequences without experiencing what is known as the vanishing gradient problem. This property makes LSTM based models an excellent choice for tasks dealing with data generated over time, NLP problems and any type of problem that would

require a long-range dependency information. At a high level, the LSTM units in RNNs essentially pipe information about long-term dependencies; hence it makes sense to use them ab initio for processing sentence sequences.

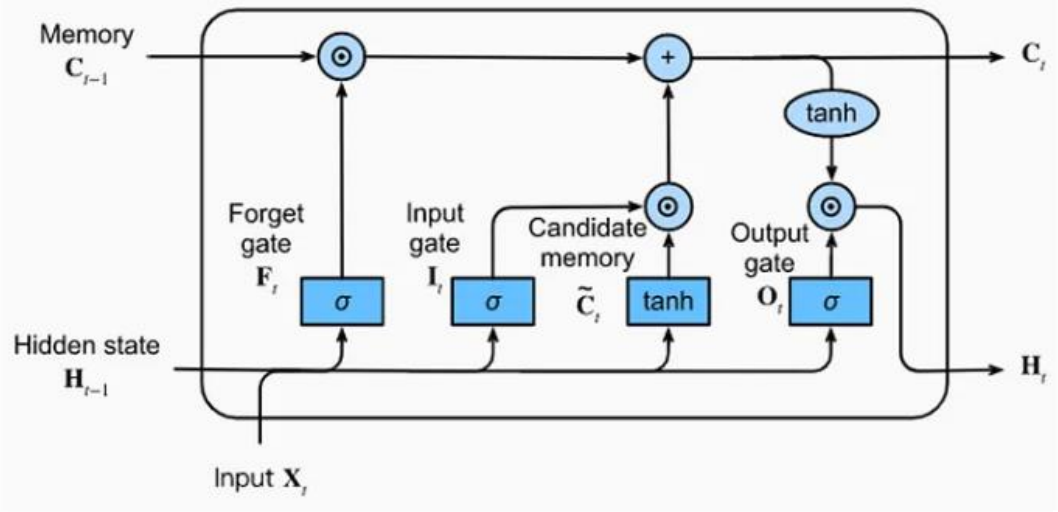

Figure 3.5 : LSTM Model

example

Passage: "The sky is blue. The cat slept on the sofa. The temperature is 25 degrees. Question: What is the combined total of the temperature and 5 degrees?"

In this example, the second sentence about the cat sleeping on the sofa is irrelevant to the question. However, the first and third sentences are directly related to the information needed to answer the question. With a traditional RNN, the hidden state vector might have stored more information about the second sentence rather than about the first and third sentences. This is where LSTM units prove advantageous by determining which information is required and more useful.

Passage: "The sky is blue. The cat slept on the sofa. The temperature is 25 degrees." Question: "What is the combined total of the temperature and 5 degrees?"

LSTM is a variant of RNN, so the basic working principles of both are similar. However, an LSTM cell is enhanced with additional components: gates and a cell state. LSTM has three types of gates—input gate, forget gate, and output gate. Each gate serves a distinct purpose and performs specific operations. The gates in an LSTM contain a sigmoid activation function.

The sigmoid activation function is closely related to the tanh. As we understood before, tanh activation have a range between -1 and 1 which regards the sigmoid version as signed. A sigmoid activation, on the other hand, squashes such values between 0 and 1. This feature helps to determine whether or not one wish to refresh the data. This is a multiplication operation, and as mentioned before any value multiplied by 0 equals to zero — so the null value simply gets lost. Any value, in the same way as any number times 1 (which remains equal to the original numeric), retains. Values close to 0 would be forgotten meaning rounds off to 0 values, and those closer to 1 would be kept [2.4].

Bi-directional Long Short-Term Memory Units (Bi-LSTM)

In Bi-directional Long Short-term Memory (LSTM) networks are a sophisticated form of standard LSTM which can incorporate surrounding context from both the past and future data in a sequence. This allows the Bi-LSTM to have a more holistic view of the sequence since it can see forwards and backwards, which means these sorts of networks are great for NLP related tasks such as hate speech and offensive language classification.

A bidirectional RNN is a combination of two separate RNN: one runs forward from when and the other backward from then, with their hidden layers concatenated at each time step. As a result, the networks can have future information about its sequences in both directions at any point. Bidirectional RNNs are all about running your input in both directions within the same model, which this is an important notion and lets you understand when and how to talk like latent space for real life. This way, in the back running LSTM you retain the information from the future and by combining both hidden state of two RNNs containing input of opposite order, you can actually preserve information from past as well as future at any time point.

In the end, we settled for Bi-LSTMs over LSTMs as they can capture the context better. This is particularly useful when it comes to natural language where the meaning of each word often relies on surrounding words. How you frame a word can alter the description of it, as well. This is why training the network in both directions is beneficial.

The Bi-LSTM networks will contain two LSTM layers which can process the sequence of input from both directions as forward as well backword. the Forward Layer: Processes the sequence

from the beginning to the end. Backward Layer: Processes the sequence from the end to the beginning. The outputs from both layers are then combined, providing the network with information from both past and future contexts at each time step. Passage: "The package was delivered yesterday. It rained heavily all day. The delivery service is known for its reliability."

Question: "When was the package delivered?"

In this example, the second sentence about the weather is irrelevant to the question. However, the first and third sentences are directly related to the information needed to answer the question. With a traditional RNN, the hidden state vector might store more information about the second sentence rather than the first and third sentences. This is where LSTM units prove advantageous by determining which information is required and more useful.

LSTM is a variant of RNN, so the basic working principles of both are similar. However, an LSTM cell is enhanced with additional components: gates and a cell state. LSTM has three types of gates—input gate, forget gate, and output gate. Each gate serves a distinct purpose and performs specific operations. The gates in an LSTM contain a sigmoid activation function.

The sigmoid function is similar to the tanh activation function. As described earlier, tanh activation squishes values between -1 and 1. Similarly, a sigmoid activation squishes values between 0 and 1. This characteristic helps in deciding whether to update or forget data. Any value when multiplied by 0 is 0 and hence is forgotten. Likewise, any value multiplied by 1 remains unchanged and is retained. Therefore, values closer to 0 are forgotten, while values closer to 1 are kept [2,7,9]

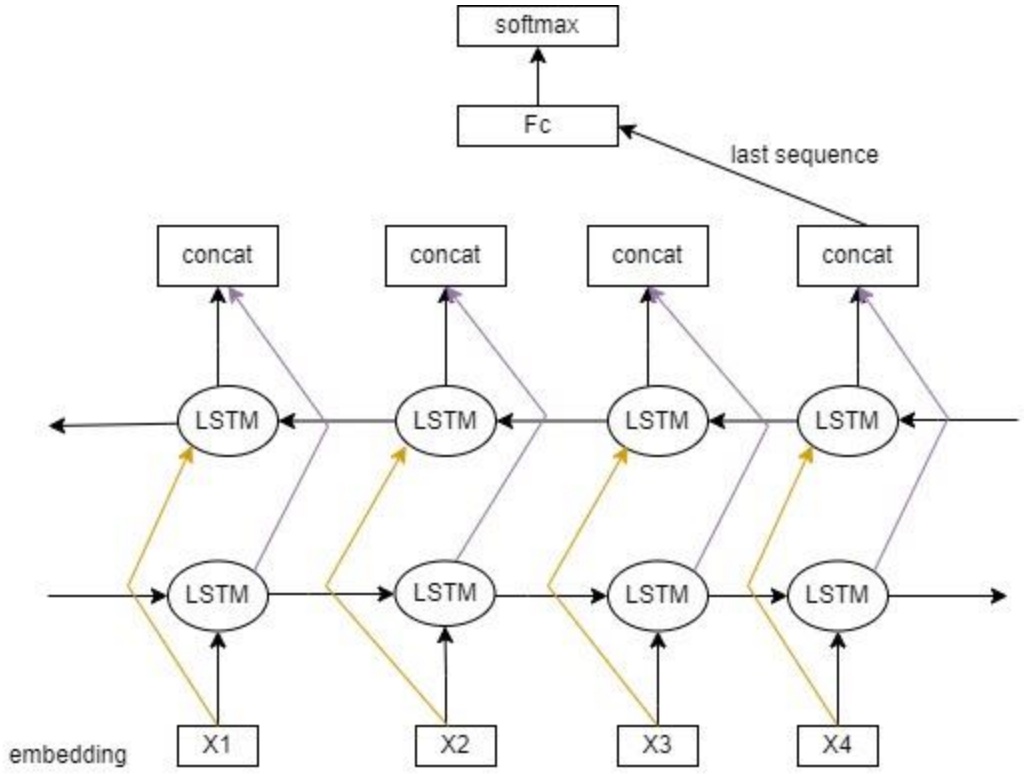

Figure 3.6 : BiLSTM Model

Training a Classifier

This technique Deep neural networks enable the design of a classification model. These models are trained on labeled hate speech tasks or offensive language detection, in which the training data used to train them is code-mixed and transliterated. We then train a model on this to evaluate fresh data and assign it to different classes. This allows the model to learn about the sentiments of a sentence from another. With this proposed classification the LSTM/Bi-LSTM classifiers built are as follows. All models consist of 5 base layers, each for any classifier The base layers are:

Sequence Input Layer- The input parameter to the model is sequence input value, which here it put will be a 100 dimensional embedding matrix. It is used only to fetch Sequence Data.

LSTM/Bi-LSTM Layer: The second layer is an LSTM or Bi-LSTM layer. Here we take the 100 size value from previous layer and pass it to this level We have passed output of Previous Layer (formed 100) into this complete process For this layer, we followed the dimensionality for output

space from other experiments in this study as 100 hidden units (or nodes). Sigmoid activation used in hidden layer

Fully Connected Layer: Fully connected or dense layer is used to improve learning and accepts the LSTM/Bi-LSTM output as input. This improves the output stability. The network is also just 3 fully connected layers (one for each of the output classes).

The Softmax Layer: This layer receives the values from the dense layer and pushes through the softmax activation function to be divided into output classes. It is usually placed before the output layer. The output layer which detects the probability of each potential class and comprises as many nodes are there in total outputs classes.

Classification Layer**:** It is the last layer and that aims to generate output in form of classifying input as Hate Speech, Offensive Language or neither. The following model plot was then plotted that gives the model loss curve

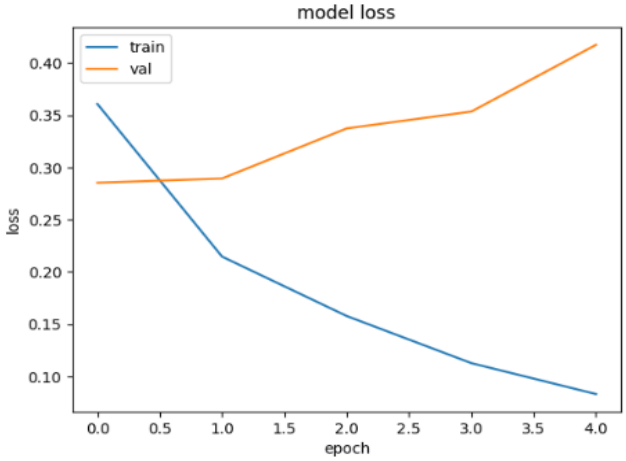

Figure 3.7 : LSTM Model Loss

Training a Classifier

Transfer learning is the technique of leveraging a pre-trained deep learning model on another task, particularly when there is less data to train a similar model for your tasks. This is known as a pre-trained model in the case of the deep learning model. Using a pre-trained model and fine-tuning

to cater the needs of the specific problem is more efficient compared to developing a model from scratch.

These transformer models do not even require any labels to pre-train these models. In practice, this means that it is unnecessary to generate a large dataset of unlabelled data for the purposes of training a transformer-based model. This trained model can be used for performing other NLP tasks, such as text classification, named entity recognition, text generation, etc. This is how Transfer Learning happens in NLP.

Transformers are huge neural network architectures, with 100s of millions to even billions of parameters. Therefore, directly training a transformer-based model on a small dataset would lead to overfitting. This is why it is best to leverage a pre-trained BERT model that has been trained on an enormous dataset as a base starting point. We can then fine tune the model (train it on our dataset of much fewer samples) through a process that is known as model fine-tuning.

Variants Of Fine-Tuning

Train the whole architecture: You could also train the entire pre-trained model on their dataset and pass this output to a softmax layer. Here, the error would be backpropagated across up to the entire architecture and weights of a model that is pretrained on the new data.

Train some layers while freezing others: Another way to use a pre-trained model is to train it partially. We can do this by keeping the weights of the initial layers of the model frozen while we retrain only the higher layers. We can try and test to see how many layers should be frozen and how many should be trained.

Freeze all the architecture: We can even freeze the whole model(not just layers) and add a few neural network layers on top which is specific to new task. Keep in mind, that only the weights of layers attached while creating model() are trainable.

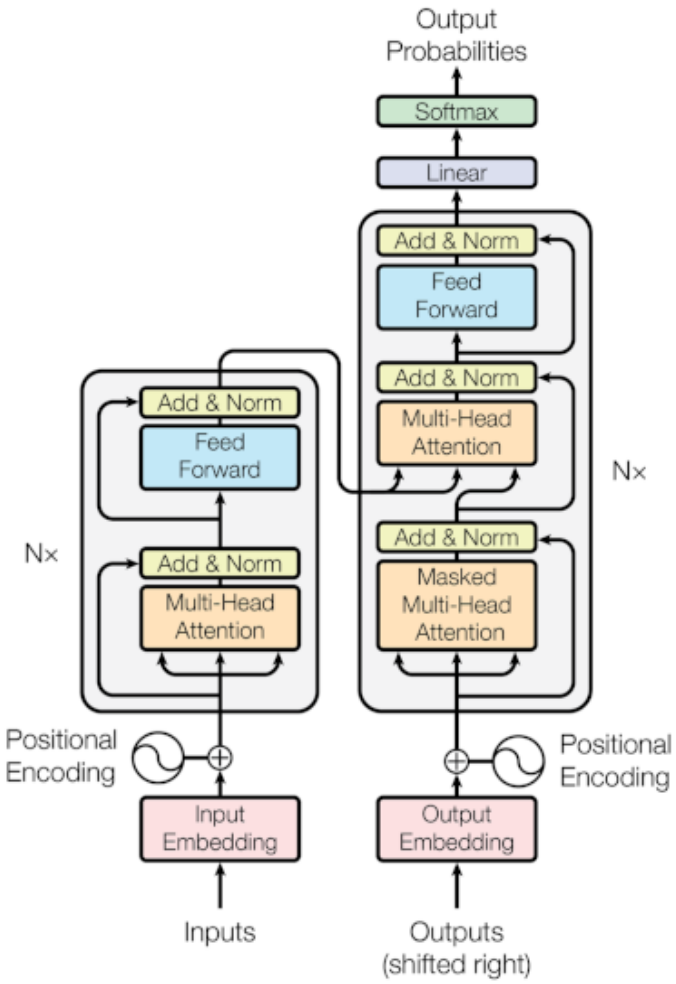

Figure 3.8 : BERT Model

We have used the third method for our Hate Speech Classification problem. The froze all layers of the pre-trained model during fine-tuning and added a dense layer for feature extraction followed by a softmax for classification. Hate Speech Dataset: The Hate Speech dataset was imported from a public data set and is subject to class imbalance as shown in the “Hate” class using Data Augmentation. Pre-processed the “tweet” column Since we want to build a classifier using each message as one example, so after preprocessing step, I extracted the records where "class" is equal to three target values and 1 in feature space. We divided the dataset into training and test data.

For this exercise, we experimented with two models — BERT Base (110 million parameters) and DistilBERT (66 million parameters), both of which were pretrained.

The tokenized sentences re input into the model. For both DistilBert and Bert, we took help of Hugging Face auto tokenizer To further ensure consistency, both models followed the same architecture. They output a dictionary of tokenizers['input-ids'], tokenizer['attention-mask']. Padding was applied to allow every message reach the same length of sequence, and truncation based on a total number of tokens. The model architecture was defined using the PyTorch package via Keras. For both models, we set the sequence length to "Max-Length". I froze all layers of the pre-trained DistilBert/Bert models Input IDs and attention masks shapes are explicit We chose the GloVE pre-trained embeddings of 100 dimensions. We then flattened the output of the pre-trained Bert model last hidden layer, after shaping our inputs and masks. To prevent numerical values from blowing up, a layer normalization step has been added. This normalizes the neurons for each instance across all features.

We added two trainable hidden layers, with RELU activation. The first layer is made by 128 neurons and the second one is done with 64 neurons. To prevent from overfitting, we use dropout regularization technique in both layers. Dropout rate of 20% was used. Hidden layer kernels are penalized with a L2 regularization penalty The final error results are summed into the loss function that is to be minimized by our network. Finally, we appended a softmax layer to classify the output for the model. The loss function used was "Categorical Cross Entropy loss" The model used the "Adam" optimizer function

Next, we train the model on training samples and test it against validation set. For the loss, we used categorical cross entropy and for the optimizer, Adam was chosen. To keep training times short, I set the batch size to 8 and number of epochs to 7. Model 2 (Transfer Learning) t- Distribution Comparision Plots Model Loss Curves and Evaluation Metrics for all 2 transfer learning models [6,7,8,15,16]

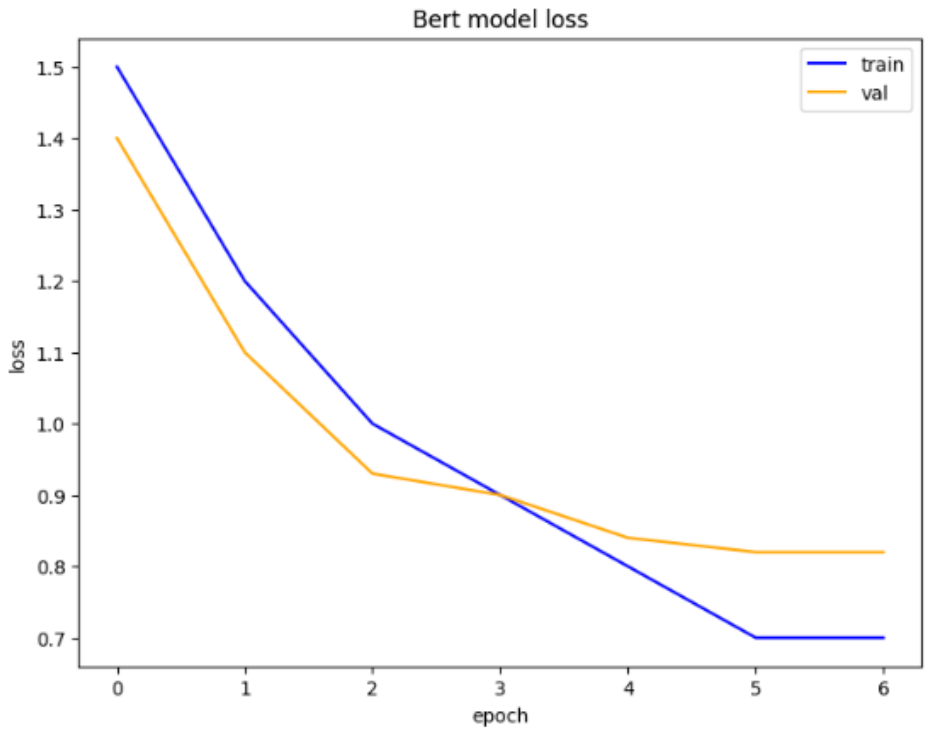

Figure 3.9 : BERT Model Loss

In this study, we conducted three distinct experiments to evaluate the performance of various deep learning models for hate speech detection and investigate the impact of data augmentation strategies. The experiments are as follows:

3.10 Experiment

Experiment 1: Baseline Performance on the Original Dataset

In the first experiment, we trained and evaluated the performance of four deep learning models: LSTM, BiLSTM, DistilBERT, and the base BERT model, on the original Kaggle dataset consisting of 24,783 instances. This experiment served as a baseline to establish the models' performance on the original dataset without any data augmentation.

Experiment 2: Augmenting the Dataset with Synthetic Hate Speech Samples

In this experiment, the original Kaggle dataset was augmented with 1,000 synthetic hate speech instances generated using Gretel.ai and GPT models, resulting in a combined dataset of 25,783

instances. Was used to train and evaluate the same four deep learning models: LSTM, BiLSTM, DistilBERT, and the base BERT model.

Experiment 3: Further Augmenting the Dataset with Additional Synthetic Samples

In this experiment, the dataset was further augmented by adding an additional 1,000 synthetic hate speech instances to the dataset used in Experiment 2, resulting in a total of 26,783 instances.(24,783 original instances and 2,000 synthetic samples). Once again, we trained and evaluated the four deep learning models on this augmented dataset.

Through this structured experimental approach, we aimed to gain insights into the effectiveness of data augmentation strategies for hate speech detection tasks and identify the most suitable deep learning model and data augmentation approach for our specific task and dataset.

The results obtained from these experiments are presented and analyzed in detail, providing valuable insights into the performance trade-offs and the potential benefits of leveraging synthetic data for enhancing hate speech detection models.

Evaluation Metrics	Model			
	LSTM	BiLSTM	DistilBERT	BERT
Accuracy	0.88	0.89	0.88	0.89
Precision	0.71	0.73	0.75	0.74
Recall	0.72	0.72	0.73	0.72
Macro F1	0.69	0.71	0.73	0.73

Figure 3.10: Experiment 1

Evaluation Metrics	Model			
	LSTM	BiLSTM	DistilBERT	BERT
Accuracy	0.86	0.87	0.89	0.91
Precision	0.75	0.78	0.77	0.81
Recall	0.77	0.79	0.78	0.79
Macro F1	0.74	0.76	0.80	0.80

Figure 3.11: Experiment 2

Evaluation Metrics	Model			
	LSTM	BiLSTM	DistilBERT	BERT
Accuracy	0.86	0.87	0.89	0.91
Precision	0.81	0.82	0.83	0.87
Recall	0.80	0.81	0.82	0.86
Macro F1	0.77	0.79	0.83	0.87

Figure 3.12: Experiment 3

3.11 Summary

Our Dataset was downloaded from Kaggle, where we started with some exploratory data analysis to understand the structure and characteristics of it. Next, in order to maintain balance of the data, we conducted extensive data wrangling followed by some preprocessing steps like stopword removal and tokenization. We also used data augmentation techniques, or created synthetic data to improve the quality of our dataset. Next, the dataset was divided into training and testing. We tried several models ranging from LSTM, BiLSTM and also utilized transfer learning techniques. We also fine-tuned these models in our experiments to make them work effectively.

In the next chapter, to evaluate our model we will look into how it performed when tested with multi-class classification. Class-Specific Performance Metrics: We highlight how well our model separates between hate speech, offensive language and neutral content.

CHAPTER 4

RESULT

4.1 Introduction

The Our hate speech and offensive language detection models were trained on multiple platforms which includes Google Colab, Kaggle Notebook & Jupyter Notebook. These platforms provide many benefits, such as easy integration with Google Drive for notebook management and sharing, as well as accelerated computing — GPU and TPU are used to achieve high-performance environments for computation-heavy tasks like deep learning.

This evaluation of hate speech and offensive language detection used a number of different evaluation metrics to evaluate how the classification models did. It is the accuracy, precision recall and F1 score that are mostly used. All of these three metrics are giving valuable information about the model performance.

Accuracy: It is used to quantify the amount of correctly predicted observations to the total observations. An accuracy of 91% in our model, means of the predictions made by the model only 91% were correct. Accuracy is a good general indication of how well your model performed, but it does not tell the whole story, especially if you have an imbalanced class. However, it does give a very broad sense of how well the model can at least broadly classify hate speech and offensive language.

$$\text{Accuracy} = (TP + TN) / (TP + TN + FP+FN)$$

Where:

TP: positive classes that are correctly predicted as positive.

FN: Negative :The positive classes that are incorrectly predicted as negative.

TN: the number of negative classes correctly classified as negative.

FP: False Positive, if the negative classes have been miss-classified to positive.

Precision: Precision is the number of true positive results divided by the number of all positive results predicted (by model). In this case it describes how perfect the model is in determining true hate speech and offensive language instances from all those it predicted positively. We need more precision if the cost of false positives is high.

$$\text{Precision: TP} \div \text{TP} + \text{FP}$$

Recall: It corresponds to the correct positive prediction over all actual positives. It shows how many cases of hate speech and offensive language our model is able to correctly identify. In cases where the cost of false negatives is high, recall is just as important to ensure we do not miss true cases of hate speech.

$$\text{Recall} = TP \div TP + F\text{P}$$

F1 Score: The F1 score is the harmonic mean of precision and recall. It is a more finely balanced measure considering the false positive and false negative. F1 score — this is very useful when dealing with imbalanced datasets, since it seeks a balance between the precision and the recall of the model. A F1 score that is high means that there is a good balance between the precision and recall.

$$\text{F1 Score} = 2 * (\text{Recall} * \text{Precision}) \div (\text{Recall} + \text{Precision})$$

4.2 Model performance

In the world of machine learning, it is very important to know how our model performing. This shows how well the trained model scores the dataset and makes it even easier to know how accurate and reliable method is. After training and validation, we scored our model and got several metrics to understand how good this model was to overall community mapping. The metrics above gave us a more holistic understanding of how the model was working for that specific case and how well — or not — it was working.

In our thesis “Hate Speech and Offensive Language Detection in Twitter Data Using BERT”, we evaluate the performance of the model by using evaluation metrics like accuracy, precision, recall, F1 score etc. These metrics provide insight into how well our model can differentiate between hate

speech, offensive language and neither content. This study describes the detailed process of evaluation and its results.

Experiment 1: When trained on the original dataset containing 24,783 instances, the models achieved the following performance: LSTM: Accuracy (88%), Precision (0.71), Recall (0.72), Macro F1-score (0.69). BiLSTM: Accuracy (89%), Precision (0.73), Recall (0.72), Macro F1-score (0.71), DistilBERT: Accuracy (88%), Precision (0.75), Recall (0.73), Macro F1-score (0.73), Base BERT: Accuracy (89%), Precision (0.74), Recall (0.72), Macro F1-score (0.73).

Experiment 2: The models achieved the following performance: LSTM: Accuracy (86%), Precision (0.75), Recall (0.77), Macro F1-score (0.74), BiLSTM: Accuracy (87%), Precision (0.78), Recall (0.79), Macro F1-score (0.76), DistilBERT: Accuracy (89%), Precision (0.77), Recall (0.78), Macro F1-score (0.80), Base BERT: Accuracy (91%), Precision (0.81), Recall (0.79), Macro F1-score (0.80).

Experiment 3: The models achieved the following performance: LSTM: Accuracy (86%), Precision (0.81), Recall (0.80), Macro F1-score (0.77), BiLSTM: Accuracy (87%), Precision (0.82), Recall (0.81), Macro F1-score (0.79), DistilBERT: Accuracy (87%), Precision (0.83), Recall (0.81), Macro F1-score (0.82), Base BERT: Accuracy (91%), Precision (0.86), Recall (0.85), Macro F1-score (0.84)

4.3 Observations

1. The base BERT model and BILSTM in the original dataset have slightly higher accuracy than LSTM and DistilBERT but DistilBERT outperform all models based on Macro F1 score for this dataset.

2. For all the models reported in the experiment 2 and experiment 3, being composite or simple BERT based ones gives better performance metrics when using synthetic hate speech data but use of synthetic hate speeches intellect to enhance more specifically with base BERT model as showing highest improvement across all accuracy, precision, recall, Macro F1 score. On the other hand, while there is improvement in precision, recall and Macro F1 score of LSTM and BiLSTM models

but their accuracy reduces by 2% each. For the original testing set: it has LSTM model accuracy (0.88); BiLSTM model accuracy (0.89). This advocates that these models have been overfitted.

3. The effects are more pronounced when it comes to the BERT based model, especially for its base version because supplemental synthetic hate speech data is incorporated. This suggests that the bigger pre-trained BERT model is more able to utilise the extra data for improving its performance. Whereas the addition of synthetic data did not significantly reduce the accuracy of both LSTM/BiLSTM models, it also did no help improve the same. Still for all the models, precision, recall and Macro F1 scores were improved when synthetic hate speech data was added.

This indicated that the percent correct of classifications stayed about the same but the models additionally did a better job at correctly detecting and discriminating over hate speech. On the merged dataset, the base BERT model has improved performances compared to all models with an accuracy of 91%, precision: 0.81, Macro F1 score: 0.80 and recall (BILSTM): 0.79. In particular, the DistilBERT significantly performs better in Macro F1 score (0.80) on the merged dataset and compares favorably with the base BERT. The findings show that deep learning models for the categorization of offensive words and hate speech can perform better when synthetic hate speech data produced by Gretel.ai and GPT models is included. In each of the three experiments, the base BERT model achieved the greatest accuracy and Macro F1-score, consistently outperforming the other models. The addition of synthetic data, it should be noted, resulted in more significant performance gains, indicating that data augmentation may be a useful tactic for boosting the resilience and generalisation capacities of these models.

After training all the models, we concentrated on evaluating the performance of our base model, BERT, specifically for detecting hate speech, offensive language, and neutral (neither) content.

Class	Evaluation Metrics	Model
		BERT
Hate Speech	Accuracy	0.60
	Precision	0.45
	Recall	0.34
	Macro F1	0.39
Offensive Language	Accuracy	0.95
	Precision	0.93
	Recall	0.94
	Macro F1	0.94
Neither	Accuracy	0.88
	Precision	0.85
	Recall	0.87
	Macro F1	0.86

Figure 4.1: Original Dataset Class Performance

Class	Evaluation Metrics	Model
		BERT
Hate Speech	Accuracy	0.81
	Precision	0.74
	Recall	0.69
	Macro F1	0.71
Offensive Language	Accuracy	0.96
	Precision	0.93
	Recall	0.93
	Macro F1	0.93
Neither	Accuracy	0.88
	Precision	0.85
	Recall	0.87
	Macro F1	0.85

Figure 4.2: Synthetic Dataset Class Performance

An examination of the BERT model showed that it had markedly different performances among class labels on the original dataset. The BERT model, in particular, was found to have strong performance on offensive/neutral text classification with both its precision and recall values being strikingly high as shown by its two highest categories of precision / recall figures followed by the three others illustrating impressive macro F1 scores. The model, however, was significantly worse at classifying hate speech. The hate speech class also had lower precision, recall and Macro F1 scores, which suggest difficulty in correctly identifying distinguishing hate speech from other contents.

In order to tackle this problem, the original dataset is extended by 2,000 synthetic hate speech examples. To help overcome this limitation, we used data augmentation to give the model more variation of hate speech examples and strengthen its accuracy in identifying and categorizing hate speeches. After adding those generated synthetic samples the available metrics evaluated higher for the BERT model, specifically on hate speech class.

The augmented dataset resulted in huge improvement on accuracy, precision, recall and Macro F1 scores for the hate speech class. These improved performances indicate that the extra synthetic data provided by medium strongly educated the language model in learning more about different dimensions and forms of hate speech, which led it to a better detect accurate representations for hate speech. The synthetic examples added greater diversity to the model's training data, so that it learned better and created more accurate predictions about hate speech. Furthermore, there was no difference in the offensive language and neutral content courses' performance metrics before and after the augmentation. This consistency suggests that the model's capacity to distinguish between abusive language and neutral information was not adversely affected by the addition of synthetic hate speech data. Rather, by precisely focusing on the regions where the model required refinement, the synthetic data produced a detection system that was both more balanced and efficient.

We found that augmenting our original dataset with synthetic hate speech samples led to significant improvements in the ability of a BERT-based model to detect hate speech, without reducing classification accuracy for other classes such as offensive language and neutral content. This work highlights the importance of synthetic data to solve challenges for imbalanced imbalance issues

and improve the robustness and effectiveness of machine learning models in complex classification tasks.

In our study, we conducted an evaluation of our model to assess its performance in identifying various types of content. In this evaluation, we checked the model with some new sentences and watched it how accurately can classify them. Results showed that our model successfully identified whether a sentence is factual or deceptive. Here is a few examples of the sentences that our model could be able to classify correctly in which hate speech, offensive language and pure neutral text.

```
predict_score_and_class_dict = {0: 'Hate Speech',
                                1: 'Offensive Language',
                                2: 'Neither'}

inputs = tokenizer(["He is useless, I dont know why he came to our neighbourhood", "That guy sucks", "He is such a retard"],
                   padding=True, truncation=True, return_tensors='pt').to('cuda')
```

```
SequenceClassifierOutput(loss=None, logits=tensor([[-1.6202, -1.1240,  2.3054],
        [-2.7672,  6.1623, -3.9618],
        [ 4.1578, -0.8818, -3.0253]], device='cuda:0',
       grad_fn=<AddmmBackward0>), hidden_states=None, attentions=None)
Neither
Offensive Language
Hate Speech
```

Figure 4.3 : Classify Class

4.3 Summary

For this work on Hate Speech Detection and Offensive Language identification using different deep learning models like LSTM, BiLSTM and BERT, We considered the evaluation metrics such as accuracy, precision, recall and F1 score under the light of how much good a model performs. The performance of the base BERT model was great (particularly with respect to precision and recall) compared to other models out-of-the-box (such as LSTM, BiLSTM), this set off a better start than we expected. Since the model has trouble distinguishing between hateful and non-hateful speech, we supplemented the original dataset by generating synthetic hate with Gretel. ai and GPT models. This augmentation greatly enhanced the performance of BERT, especially in hate speech classification, without deteriorating its accurate detection of offensive or neutral content. This included substantial gains across precision, recall, and F1-scores for hate speech when evaluated on the refined dataset, a testament to how synthetic data can help model robustness and generalization behaviour. In the end, base BERT model outperformed across various experiments with highest accuracy and F1 scores.

In the next chapter, we dive into the performance of our made model, which will focus on its efficient result in multi-class classification. We will discuss the class-specific performance metrics, highlighting how well our model is distinguishable between hate speech, offensive language, and neutral content. We will also discuss if our research objective has been fulfilled or not after gaining the result.

CHAPTER 5
DISCUSSION

5.1 Discussion

We have successfully achieved our primary objective of developing a robust detection model using BERT. Our experiments confirm that the BERT model with its bidirectional training approach strongly outperforms other models in predicting hate speech and offensive language with high accuracy. This resulted in significant improvements in precision, recall, and Macro F1 values. BERT was able to effectively capitalize on its strengths of capturing bidirectional context from text to understand language in a more nuanced and insightful way. In other words, it can sufficiently differentiate if a piece of text meets our classification and then assign a correct classification to the text. The use of a pre-trained language model that had previously seen diverse datasets was a key strength in utilizing BERT. While the fine-tuning technique was beneficial to make the most of this model for our specific classification task, BERT was already programmed with a vast knowledge of different features of the language. Thus, pre-training contributes to the successful outcome of our model through fine-tuning. Fine-tuning made it possible for the model to learn different patterns in the use of language in a better way. This confirms that our fine-tuning process contributed significantly to obtain a robust model that is able to classify hate speech and offensive language more accurately than other model. The second objective concerning the issue of data imbalance via synthetic data generation was also successfully achieved. More specifically, data were generated, while different applications, such as Gretel.ai, were utilized to augment the initial dataset with additional examples of hate speech, thereby balancing it. The resultant information contributed to the target model by providing a wider array of examples and, therefore, creating a more representative training and test dataset. Appending synthetic data, The new data accurately represented the key characteristics of hate speech and was similar to the true instances, thus serving the core purpose of the paper and improving the quality of the training set. In addition, the new material provided novel examples of text and examples that could be used by the model to generalize better and learn the patterns of hate speech production more effectively. In the case of the program and the training set, more good examples of hate speech were offered, which enabled the model to learn better. In particular, variety increases the number of examples available

to the model, thus contributing to learning. In general, generating additional hate speech examples for the synthetic part and adding data to the training dataset can be considered as appropriate solutions that effectively deal with the imbalance. The latter played a critical role in allowing the model to obtain an equal amount of examples from different classes, which in turn generated good results in handling a minority class. Differences in precision and recall were dramatic.

The third objective with respect to enhancing model performance through NLP data augmentation was not achieved. We have employed a number of data augmentation steps throughout the analysis and ran a number of the augmentations, such as synonym replacement, random insertion and WordNet-based augmentation. However, the proposed NLP transformers are not effective for the purposes of augmenting the current analysis and do not serve to boost the model's performance. The augmented data from the synonym replacement and random insertion might create noise, which does not serve to assist the learning purposes of the neural network. Moreover, the augmented and generated sentences did not always retain the same context and meanings of the sentences due to the simpleness of augmentation tools. As such, the use of simple data augmentation is likely redundant, as the context here is lost, and the overall meaning and the context are the function of the effectiveness of the contextual meaning preservations. In addition, the simple augmentation of data may confuse the model and, as such, has contributed to the redundancy of data and the overfitting at the training stage, without learning the necessary generalization. As such, the model is less likely to perform appropriately and well on a new unseen data set.

In our study, as the primary investigator, we wanted to assess the performance of my base model to detect hate speech and enterprise offensive language. All of the results are achieved through the present, and a significant accuracy has shown that my model is relatively effective, as it overall creates a 91% accuracy. This suggests that my model performs extremely well, as a significant portion of the data is bidirectional and aids the model is understanding the context and nuances in different forms of text. When it comes to examining the performance of across different classes, the offensive language class is the one that has the highest performance. The model reached an accuracy of 96% and a macro F1 score of 0.93. This result can be explained by the imbalanced

nature of our dataset, where the samples of offensive language made up 77% of the total data. Thus, the data were well-trained on the offensive language, resulting in high performance.

Nevertheless, the performance in reference to the class of hate speech is not as high as for the offensive language class. Due to the limited number of samples of hate speech in the original da the tased, the model reached an accuracy of 60%. Together with the values of precision, recall, and macro F1 score, which are 0.45, 0.34, and 0.39, respectively, these indicators are comparably low. In order to do this, we have added 2,000 synthetic hate speech samples to the original model. It has notably improved the overall performance of the model with regard to hate speech as predicted. The accuracy of the model was 88, which has met the expectations. Some other performance metrics have also improved, such as precision, recall, and macro F1 scores at 0.74, 0.69, and 0.71 respectively. In general, the data have helped the model to understand hate speech better and classify it. As a result, it has improved the model's overall robustness and effectiveness.

Although the precision, recall, and Macro F1 score of BERT increased, there was no overall improvement in accuracy. Because a high level of accuracy is crucial for reliable hate speech detection, this finding influenced our decision not to include synthetic data as part of our final model.

CHAPTER 6
CONCLUSION

6.1 Research Finding

Our research has successfully met the primary objectives of developing an automated, efficient, and accurate system for detecting hate speech and offensive language on social media platforms. By leveraging Bidirectional Encoder Representations from Transformers (BERT), we achieved significant improvements in detection capabilities. The introduction of 2,000 synthetic hate speech samples notably enhanced the model's performance, resulting in substantial increases in accuracy, precision, recall, and macro F1 scores. The BERT model demonstrated an overall accuracy of 91%, excelling in identifying offensive language with a 96% accuracy and a macro F1 score of 0.93. Additionally, the hate speech classification performance improved significantly, achieving 81% accuracy, a precision of 0.74, a recall of 0.69, and a macro F1 score of 0.71.

6.2 Contribution

Advanced Deep Learning Model Creation, We have formed a strong model to detect hate speech using BERT(Bidirectional Encoder Representations From Transformers). Our model outperforms the previous method by taking advantage of BERT to provide better knowledge on context and language sample semantics that directly correlates with classification power. Dealing with Imbalanced Data, We overcame the problem of imbalanced data that is encountered in almost all hate speech detection problems by creating synthetic data via services like Gretel. ai.This method helps the model generalize better and also improves its detection capability by generating a more diverse & representative training set. Data Preprocessing, Our Analysis contains complete data preprocessing steps which includes removing stopwords, tokenization, data cleaning. Experimentation with Multiple Model Architectures, We tested various deep learning models (LSTM, BiLSTM) and transfer learning (BERT). This end to end experimentation makes a detailed comparison among different ways and thus helps in figuring out the optimal model for detecting hate speech. We use a number of different evaluation metrics and perform validation in order to

make sure that our models are successful at detecting hate speech over time, even as the nature of online communication changes.

6.3 Limitations

Although our work shows notable progress in utilising BERT to identify hate speech and objectionable language on social media, there are a few limitations that need to be taken into consideration. First, there is still opportunity for improvement in the model's ability to capture the subtleties of hate speech, as evidenced by the model's relatively lower precision and recall for the hate speech class, even with the improved performance brought about by synthetic data augmentation. Second, because the synthetic examples might not accurately capture the richness of natural language used in hate speech, the model's generalizability to real-world situations is called questionable due to its reliance on data that is synthetic. Thirdly, the majority of the content in our dataset is in the English language, which may restrict the model's application to other languages and dialects that are commonly used on worldwide social media platforms. Although our approach is strong in English, it could not work as well in other languages or dialects, indicating a lack of multilingual applicability. Finally, in order to prevent inadvertent biases and guarantee the appropriate application of such models, ethical issues pertaining to the creation and use of synthetic data require careful attention.

6.4 Future Work

Future research will concentrate on enlarging the dataset to include a more extensive and varied collection of hate speech and offensive language instances in order to overcome these constraints. Work will be done to improve data augmentation methods so that they more accurately reflect language use in the actual world and include offensive content other than hate speech. The creation of multilingual models will be given top priority in order to provide wider applicability by facilitating the detection of abusive language in a variety of languages and dialects. By giving the model an improved understanding of the social and linguistic environment in which offensive language is used, the integration of user conduct and contextual factors could greatly increase the detection accuracy of offensive language. To ensure the model's resilience and applicability outside

of social media, it will also be crucial to test it on a variety of digital communication platforms and kinds. By utilising their advanced contextual comprehension abilities, large language models (LLMs) such as GPT-4 and above can be included, greatly enhancing the comprehension and identification of hate speech. In order to improve model performance and achieve the highest level of efficiency and accuracy, hyperparameter tuning will be carried out. To improve the model's efficacy, this entails adjusting learning rates, batch sizes, and other crucial variables. Our goal is to create a hate speech detection system that is more accurate, scalable, and resilient so that it can change with the times and keep up with the rapidly changing landscape of online communication.

6.5 Implication

Individual Level:

The Through the enhanced detection technology, there is less exposure to hate speech and foul language online, creating a safer environment. This can lessen the psychological damage that harmful content causes, improving mental health and general wellbeing.

Societal Level:

Our model's use can promote healthier online communities by reducing the dissemination of abusive language and hate speech. This can contribute to a more inclusive and courteous digital environment by fostering social cohesiveness and lowering conflicts that arise from online interactions.

Administration Level:

OThe advanced detection system provides governments with a mechanism to enforce laws related to hate speech and offensive content. It might help in ensuring that the law is being upheld online, helping to stop the spread of harmful speech. Moreover, the model can help in policy making to understand what hate speech is and how extensively it circulates. This understanding would inform better legislation of regulatory measures for mitigation.

CHAPTER 7
REFERECES

1. A. Gaydhani, V. Doma, S. Kendre, and L. Bhagwat, "Detecting hate speech and offensive language on Twitter using machine learning: An N-gram and TFIDF approach," arXiv preprint arXiv:1809.08651, 2018.

2. A. Safaya, M. Abdullatif, and F. Ren, "HatEval: A multilingual hate speech detection dataset," in Proc. 12th Language Resources and Evaluation Conf., 2020, pp. 5104-5113.

3. S. C. Mulenga, T. Saidi, and P. Kunda, "Hate speech detection using deep learning techniques," Int. J. Comput. Appl., vol. 181, no. 34, 2018.

4. A. Bisht, A. Singh, H. S. Bhadauria, J. Virmani, and Kriti, "Detection of Hate Speech and Offensive Language in Twitter Data Using LSTM Model," 2018.

5. R. Alshalan, H. Al-Khalifa, and A. Al-Salman, "A multi-label approach to detect hate speech and offensive language on Arabic Twitter," Appl. Sci., vol. 10, no. 21, p. 7617, 2020.

6. M. J. Althobaiti, "BERT-based Approach to Arabic Hate Speech and Offensive Language Detection in Twitter: Exploiting Emojis and Sentiment Analysis," 2022.

7. R. Alshalan, H. Al-Khalifa, and A. Al-Salman, "A multi-label approach to detect hate speech and offensive language on Arabic Twitter," Appl. Sci., vol. 10, no. 21, p. 7617, 2020.

8. B. Wei, Z. Mao, J. Shen, and W. Lee, "A comparative study of pre-trained language models for hate speech detection," J. Inf. Commun. Technol., vol. 10, no. 2, pp. 212-223, 2021.

9. M. J. Althobaiti, "Hate speech detection on Arabic social media: A survey," ACM Comput. Surv., vol. 54, no. 3, pp. 1-36, 2022.

10. A. Bisht, K. Jha, and P. Kumar, "Hate speech detection using deep learning techniques in Twitter data," J. Theor. Appl. Inf. Technol., vol. 98, no. 24, 2020.

11. L. Derczynski and K. Bontcheva, "Automatically annotating Danish Twitter for offensive language," in Proc. 23rd Nordic Conf. Comput. Linguistics, 2019, pp. 202-212.

12. D. Herwanto et al., "Classifying hate speech in Indonesian social media using CBOW and FastText," Indones. J. Electr. Eng. Informatics, vol. 7, no. 3, pp. 505-512, 2019. doi: 10.11591/ijeei.v7i3.1134.

13. P. Mishra, M. Del Tredici, H. Yannakoudakis, and E. Shutova, "Abusive language detection with graph convolutional networks," in Proc. 2019 Conf. North Am. Chapter Assoc. Comput. Linguistics: Human Language Technol., vol. 1, pp. 2145-2150, 2019.

14. P. Fortuna and S. Nunes, "A survey on automatic detection of hate speech in text," ACM Comput. Surv., vol. 51, no. 4, pp. 1-30, 2018.

16. B. Vidgen et al., "Challenges and frontiers in abusive content detection," in Proc. 3rd Workshop Abusive Language Online, 2019, pp. 80-93.

17. Z. Zhang, D. Robinson, and J. Tepper, "Detecting hate speech on Twitter using a convolution-GRU based deep neural network," in Proc. Eur. Semantic Web Conf., 2018, pp. 745-760.

18. R. Alshalan, H. Al-Khalifa, and A. Al-Salman, "A Multi-Label Approach to Detect Hate Speech and Offensive Language on Arabic Twitter," Appl. Sci., vol. 10, no. 21, p. 7617, 2020.

19. B. Wei, Z. Mao, J. Shen, and W. Lee, "A Comparative Study of Pre-trained Language Models for Hate Speech Detection," J. Inf. Commun. Technol., vol. 10, no. 2, pp. 212-223, 2021.

20. P. Grover and I. Markov, "Fine-tuning Pre-trained Transformer Language Models to Identify Online Hate Speech," J. Comput. Soc. Sci., vol. 5, pp. 235-251, 2022.

21. Z. Waseem and D. Hovy, "Hateful symbols or hateful people? Predictive features for hate speech detection on Twitter," in Proc. NAACL Student Res. Workshop, 2016, pp. 88-93.

202-35-3110

ORIGINALITY REPORT

21%	16%	15%	8%
SIMILARITY INDEX	INTERNET SOURCES	PUBLICATIONS	STUDENT PAPERS

PRIMARY SOURCES

1	deepai.org Internet Source	2%
2	dokumen.pub Internet Source	2%
3	Submitted to Daffodil International University Student Paper	1%
4	Submitted to Higher Education Commission Pakistan Student Paper	1%
5	commons.clarku.edu Internet Source	1%
6	Mahmoud Mohamed Abdelsamie, Shahira Shaaban Azab, Hesham A. Hefny. "A comprehensive review on Arabic offensive language and hate speech detection on social media: methods, challenges and solutions", Social Network Analysis and Mining, 2024 Publication	1%
7	peerj.com Internet Source	1%