Addressing Bias and Improving Transparency in Automated Hate Speech Detection Systems

Mohammad Sadri

A thesis submitted to McGill University
in partial fulfillment of the requirements of the degree of

MASTER OF SCIENCE

Department of Electrical and Computer Engineering

McGill University
Montreal

March 2024

Abstract

In the digital era, online hate speech poses a growing challenge for content moderation. This research seeks to illuminate the vital role played by the distribution of target groups in training datasets and its influence on the effectiveness of hate speech detection systems, with the aim of enhancing our understanding of this critical factor. Additionally, the project aims to raise awareness of the limitations of existing hate speech detection systems, highlighting their complexities and the need for further research to achieve more robust solutions.

We selected Davidson et al.'s "Automated Hate Speech Detection and the Problem of Offensive Language" for replication after researching prominent papers in hate speech detection, including "Deep Learning for Hate Speech Detection in Tweets" by Pinkesh Badjatiya and Shashank Gupta, "A Survey on Automatic Detection of Hate Speech in Text" by Paula Fortuna, and "Hate Speech Detection Using a Convolution LSTM Based Deep Neural Network" by Ziqi Zhang. Davidson et al.'s work was chosen due to its widespread recognition and importance in the domain as demonstrated by its number of citations, aligning with standard research practices for validating and building upon existing knowledge.

During our replication efforts based on Davidson's GitHub repository, we encountered various challenges, including outdated and hard-coded elements, compatibility issues with newer Python versions, a lower-quality dataset, and missing information about the distribution of groups targeted by hate speech in the dataset. By revamping the GitHub repository, we have turned it into a flexible and adjustable framework, allowing experimentation with various target group distributions, outcome generation, distribution examination, and testing with different datasets.

To enhance the existing work, we manually labeled specific target groups in the analysis data,

including women, black individuals, and the LGBT community. We introduced a new training dataset from Berkeley, consisting of 39,565 comments annotated by 7,912 annotators, totaling 135,556 rows, to enrich our analysis. Like the original work, our validation process relied on k-fold validation since a dedicated validation dataset was unavailable, involving five groups for training and testing the model iteratively. Additionally, we conducted a comprehensive reassessment of the initial work, incorporating our newly trained classifier for a direct comparison with the original findings.

Our study investigated the impact of distribution on hate speech detection across four scenarios: black-focused, women-focused, LGBT-focused, and balanced. In the first three scenarios, there is a higher number of tweets targeting black individuals, women, and LGBT, respectively. The fourth scenario, balanced, maintains an even distribution of tweets targeting both women and black individuals, as well as LGBT. The precision results for the various scenarios and target groups range from 87% to 96%, the recall results from 55% to 93%, and the F1 scores from 70% to 93%. For the black target group, the women-focused scenario yielded the highest precision (96%), the LGBT-focused scenario the highest recall (72%), and the LGBT-focused scenario the highest F1 score (79%). For the women target group, the women-focused scenario yielded the highest precision (96%) and the balanced scenario the highest recall (84%) and the highest F1 score (89%). For the LGBT target group, the women-focused scenario yielded the highest precision (96%) and the LGBT-focused scenario the highest recall (93%) and the highest F1 score (93%).

In particular, the LGBT target group showcased the best performance in a single scenario, i.e., the LGBT-focused scenario, with the highest recall of 93%, a strong precision of 92%, and an impressive F1 score of 93%.

Our research underscores the intricate impact of distribution on hate speech detection. The significant effects of diversity, dataset specifics, classifier bias, and data quality genuinely influenced our work, emphasizing the importance of aligning training data with natural distributions and addressing biases for a more inclusive AI landscape.

Abrégé

À l'ère numérique, la haine en ligne représente un défi croissant pour la modération de contenu. Cette recherche vise à mettre en lumière le rôle crucial joué par la répartition des groupes cibles dans les ensembles de données d'entraînement et son influence sur l'efficacité des systèmes de détection de discours haineux, dans le but d'améliorer notre compréhension de ce facteur critique. De plus, le projet vise à sensibiliser aux limitations des systèmes existants de détection de discours haineux, soulignant leurs complexités et la nécessité de recherches supplémentaires pour aboutir à des solutions plus robustes.

Nous avons opté pour la réplication de l'étude de Davidson et al. après avoir examiné des articles de référence sur la détection de discours haineux, incluant "Deep Learning for Hate Speech Detection in Tweets" de Pinkesh Badjatiya-Shashank Gupta, "A Survey on Automatic Detection of Hate Speech in Text" de Paula Fortuna, et "Hate Speech Detection Using a Convolution LSTM Based Deep Neural Network" de Ziqi Zhang. Le travail de Davidson et al. a été sélectionné en raison de sa large reconnaissance et de son importance dans le domaine, conformément aux pratiques de recherche standard pour valider et élargir les connaissances existantes.

Au cours de nos efforts de réplication basés sur le référentiel GitHub de Davidson, nous avons rencontré divers défis, notamment des éléments obsolètes et codés en dur, des problèmes de compatibilité avec les versions plus récentes de Python, un ensemble de données de moindre qualité et des informations manquantes sur la répartition des groupes ciblés par les discours de haine dans l'ensemble de données. Pour surmonter ces obstacles, nous avons adapté le code pour le rendre compatible, avons compilé notre propre ensemble de données et avons clarifié la distribution des groupes cibles. Le nouveau travail répliqué permet aux utilisateurs d'exécuter le code avec les nou-

velles versions de Python, réduisant la confusion et la complexité. En remaniant le dépôt GitHub, nous l'avons transformé en un cadre flexible et adaptable, permettant l'expérimentation avec divers scénarios, la génération de résultats, l'examen de la distribution et les tests avec différents ensembles de données.

Nous avons manuellement étiqueté des groupes cibles spécifiques dans les données d'analyse, y compris les femmes, les personnes noires et la communauté LGBT. Nous avons introduit un nouvel ensemble de données de Berkeley, comprenant 39 565 commentaires annotés par 7 912 annotateurs, totalisant 135 556 lignes, pour enrichir notre analyse. Le processus de validation reposait sur une validation en k-fold, puisqu'un ensemble de validation dédié n'était pas disponible, impliquant cinq groupes pour entraîner et tester le modèle de manière itérative. De plus, nous avons réalisé une réévaluation complète du travail initial, incorporant notre classificateur nouvellement entraîné pour une comparaison directe avec les résultats originaux.

Notre étude a examiné l'impact de la distribution sur la détection du discours de haine dans quatre scénarios : centré sur les Noirs, centré sur les femmes, centré sur les LGBT et équilibré. Dans les trois premiers scénarios, il existe un nombre plus élevé de tweets ciblant respectivement les individus noirs, les femmes et les LGBT. Le quatrième scénario, équilibré, maintient une distribution uniforme de tweets ciblant à la fois les femmes et les individus noirs, ainsi que les LGBT. Les résultats de précision pour les différents scénarios et groupes cibles varient de 87% à 96%, les résultats de rappel de 55% à 93%, et les scores F1 de 70% à 93%. Pour le groupe cible noir, le scénario centré sur les femmes a donné la plus haute précision (96%), le scénario centré sur les LGBT le plus haut rappel (72%), et le scénario centré sur les LGBT le plus haut score F1 (79%). Pour le groupe cible des femmes, le scénario centré sur les femmes a donné la plus haute précision (96%) et le scénario équilibré le plus haut rappel (84%) ainsi que le plus haut score F1 (89%). Pour le groupe cible LGBT, le scénario centré sur les femmes a donné la plus haute précision (96%) et le scénario centré sur les LGBT le plus haut rappel (93%) ainsi que le plus haut score F1 (93%).

En particulier, le groupe cible LGBT a démontré la meilleure performance dans un scénario unique, c'est-à-dire le scénario centré sur les LGBT, avec le rappel le plus élevé de 93%, une précision forte de 92%, et un score F1 impressionnant de 93%.

Notre recherche souligne l'impact complexe de la distribution sur la détection de discours haineux. Les effets significatifs de la diversité, des spécificités de l'ensemble de données, du bi-

ais du classificateur et de la qualité des données ont véritablement influencé notre travail, mettant en évidence l'importance d'aligner les données d'entraînement sur les distributions naturelles et de traiter les biais pour un paysage IA plus inclusif.

Acknowledgement

My journey through the master's program has been profoundly enriched and guided by the wisdom and patience of my advisor, Prof. Mussbacher. I am deeply grateful for his invaluable guidance and the pivotal role his expertise has played in the development of my thesis.

I extend my sincere thanks to McGill University, specifically the Department of Electrical and Computer Engineering. The resources and support provided by the department have been instrumental in facilitating my research. The environment and opportunities offered by McGill have been crucial in my academic growth.

My heartfelt appreciation goes out to my friends and family. Your unwavering love, endless encouragement, and belief in my potential have been the bedrock of my perseverance and success. Each of you has played a significant role in this journey, providing support during challenging times and celebrating the milestones with joy.

Special thanks are also due to my peers and faculty members, whose insights and discussions have contributed immensely to my learning and research. Collaborating with such a talented and dedicated group has been both an honor and a privilege.

Lastly, I am grateful to everyone who has been part of this journey, directly or indirectly. Your support, in its many forms, has been a source of strength and inspiration, shaping not only this thesis but also my personal and professional development.

This achievement is not just a reflection of my efforts, but a testament to the collective guidance, encouragement, and support I have received from each one of you. Thank you for being an indispensable part of my master's journey.

Table of Contents

List of Tables

List of Figures

List of Programs

1

Introduction

In our digital age, most online hate speech is a growing concern, posing social challenges and significant obstacles to effective content moderation. As we strive to develop robust hate speech detection systems, it becomes increasingly clear that the distribution of target groups within our training datasets plays a pivotal role in shaping the effectiveness of these systems [29].

Countless studies have explored the application of machine learning and natural language processing to automatically identify hate speech in online content [3], yielding promising results. However, the performance of these models is deeply connected with the composition of our training data.

The distribution of target groups based on ethnicity, race, gender, or other attributes significantly influences how well our hate speech detection models perform [18]. Imbalanced training data, where one group dominates, can lead to biased models that struggle to detect hate speech against underrepresented groups. Contrarily, balanced data can yield models that perform well

across all groups but may fail to capture nuances related to group-specific hate speech.

While past research has offered insights into the challenges of imbalanced data and the potential for bias in hate speech detection models [3], a comprehensive understanding of how various target group distributions impact these models is still lacking [1]. This research seeks to examine the effect of the distribution of groups targeted by hate speech in training datasets on the effectiveness of hate speech detection systems. We aim to enhance our understanding of this aspect and raise awareness of the limitations of existing hate speech detection systems.

1.1 Thesis Methodology and Contribution

We first select a seminal work in the area of hate speech detection (i.e., Davidson et al.'s "Automated Hate Speech Detection and the Problem of Offensive Language") for replication. This work was chosen due to its widespread recognition and importance in the domain as demonstrated by its number of citations. We address challenges encountered during the replication and enhance the replicated hate speech detection system.

This study then employs a systematic approach to examine the influence of target group distribution on the performance of hate speech detection models. Our research methodology can be summarized as follows:

- Data Collection: We gather a diverse dataset comprising a wide range of online content, carefully annotated for hate speech instances and categorized by target group.
- Distribution Scenarios: We construct various distribution scenarios within the training data, experimenting with different proportions of target groups. These scenarios include balanced and imbalanced distributions to explore the landscape comprehensively.
- Model Training and Evaluation: We train and rigorously evaluate hate speech detection models on each distribution scenario. We employ well-established performance metrics to assess model accuracy, precision, recall, F1-score, and bias.

By undertaking this study, we aim to make the following contributions to the field of hate speech detection:

1. The revamped, replicated work is a flexible and adjustable framework, allowing experimentation with various target group distributions, outcome generation, distribution examination, and testing with different datasets.

2. We offer a greater understanding of how the distribution of target groups in training data influences the performance of hate speech detection models, shedding light on the nuances of bias and fairness in this context. Our research contributes to the ongoing discourse on ethical considerations within machine learning and artificial intelligence, particularly in addressing bias and fairness issues.

1.2 Thesis Overview

The thesis is organized as follows:

- Chapter 2: Background — This chapter clarifies the terminology used in our thesis related to hate speech detection systems. It introduces Davidson's et al.'s work and then delves into the concept of hate speech detection systems, along with associated aspects such as preprocessing steps, feature extraction, SVM classifiers, and performance metrics.

- Chapter 3: Understanding Davidson et al.'s Research — In this chapter, we delve into comprehending the code structure of the original research. We proceed to replicate it, adapting it for successful execution in a more up-to-date development environment. We demonstrate successful replication by comparing the outputs of the original and replicated work.

- Chapter 4: Unveiling New Insights — In this chapter, after replicating the results, we proceed to analyze the original work. We begin by examining their group distributions, followed by an investigation into class distributions and potential imbalances to identify any biases. Subsequently, we create a new dataset, elaborating on the reasons for this decision. We analyze the target groups' distributions within the new dataset. Additionally, we present the results for each group and evaluate their performance using 5-fold cross-validation to measure metrics like F1 score, precision, and recall. Finally, we discuss the enhancements made and address potential threats to the validity of our study.

- Chapter 5: Related Work — This chapter positions our thesis in relation to prior research, highlighting its relevance in the field of hate speech detection systems and its alignment with similar approaches employed by other researchers.

- Chapter 6: Conclusions — This chapter provides a concise overview of our thesis's main findings and contributions, along with considerations for potential future research directions.

The author performed all experiments and contributed fully to each chapter.

2

Background

Davidson et al.'s work [3] forms the fundamental basis for this research, providing essential methodologies and insights into automated hate speech detection and the challenges posed by offensive language.

2.1 Introduction to Hate Speech

Hate speech is a complex phenomenon involving language or expression used to insult, belittle, or dehumanize individuals or groups based on race, ethnicity, gender, religion, sexual orientation, or other defining characteristics. Throughout history, hate speech has been used to justify discrimination, violence, and oppression against marginalized groups. It is worth noting that even seemingly small instances of hate speech, such as a tweet targeting a particular group or containing a few words that are considered hateful, can have significant harmful effects.

Hate speech, which involves offensive language or expressions targeting individuals or groups based on various characteristics, has a long history of justifying discrimination and violence against marginalized communities [23]. Recognizing the harmful impact of hate speech, researchers strive to create an inclusive online environment by developing interfaces that effectively detect and address hate speech.

Despite the widespread recognition of hate speech as a significant issue, no globally accepted definition exists [14]. This lack of consensus among scholars and activists can challenge studying and addressing hate speech. Therefore, it is crucial to establish a clear and comprehensive definition of hate speech to advance the understanding and prevention of this phenomenon. While this thesis does not provide a definitive one, it contributes to the ongoing work in the field, drawing upon relevant references to support the exploration of this complex and evolving concept.

The impact of hate speech on individuals and communities can be severe, leading to psychological distress, physical harm, and social exclusion. Hate speech can also perpetuate systemic inequality and discrimination, further marginalizing vulnerable groups. As such, there is a growing consensus among scholars and activists that hate speech must be addressed through legal and social measures, including automated systems for detecting and removing hate speech.

We prioritize Davidson et al.'s "Automated Hate Speech Detection and the Problem of Offensive Language" [3] because it holds enormous significance and impact in the field, being the most cited paper with more than 2000 citations recorded up until the end of 2022 on Google Scholar. The study stands as a widely acknowledged and referenced work in the field. It is widely regarded as a foundational resource for hate speech analysis and detection among researchers, making it essential for our research.

Davidson et al. critically analyze the challenges involved in using automated systems to detect hate speech. They argue that while automated systems can effectively identify patterns of hate speech, they are limited by their inability to understand the broader context of language use and the nuances of human communication. Automated systems rely on predefined algorithms and training data to detect hate speech, which can result in false positives or false negatives [35].

Furthermore, they highlight the ethical considerations of using automated systems for hate speech detection that automated systems can significantly affect free speech and the right to express dissenting opinions.

They note that automated systems can be biased and reflect the values and assumptions of their creators, leading to further marginalization of already marginalized groups. Despite these challenges, they argue that automated systems can be a valuable tool in addressing hate speech, provided that they are developed and deployed responsibly. They also suggest that automated systems should be designed to be transparent and accountable, with human oversight to ensure that decisions made by these systems are fair and just.

In addition to Davidson et al., other scholars have explored the complexities of detecting hate speech [5]. For instance, researchers have developed various methods for detecting hate speech, including supervised and unsupervised machine learning algorithms, natural language processing techniques, and social network analysis. However, each method has limitations, and the challenge of detecting hate speech remains complex.

Furthermore, scholars have explored the ethical implications of using automated systems for detecting hate speech, including privacy, surveillance, and censorship issues [5]. According to some scholars [5], implementing automated systems for hate speech detection might lead to potentially eroding privacy rights. This is because individuals, in an attempt to evade being flagged by such systems, could feel pressured to self-censor their expressions or refrain from sharing particular views and opinions.

Ultimately, detecting hate speech requires a comprehensive and nuanced approach that takes into account both technical and ethical considerations. Despite the challenges and limitations of using automated systems for hate speech detection, there is significant research interest in this area [16].

In the upcoming sections, we will cover various aspects of hate speech detection. We will also dive into Davidson et al.'s work on each topic, exploring classifier implications, dataset use, and other related areas. This will give us a clearer picture of how automated hate speech detection has evolved and the challenges it faces.

2.2 Description and Pre-processing of the Dataset

The issue of hate speech and offensive language has become so ubiquitous that its prevalence has escalated on some of the world's largest social media platforms, such as Facebook, Google, YouTube,

and Twitter [36]. In response to concerns about the harmful effects of this type of language on society, researchers have developed automated tools for detecting hate speech and offensive language in online content.

One standard method for detecting hate speech involves using sentence embeddings that are fed into binary classifiers to predict whether a sentence contains such language. However, the definition of what constitutes hate speech can be subject to bias, making it challenging for machines to detect it accurately.

To aid research in this field, different datasets have been produced to provide a labeled collection of messages that can be utilized for training and assessing machine learning models. Davidson et al. focus on a method for identifying and labeling hate speech and offensive language in tweets. It addresses the growing concern about their usage in digital communication, particularly on social media platforms like Twitter.

To conduct their research, Davidson et al. utilized DWMW17 and FDCL18 datasets [27]. These datasets were selected because they were annotated for various types of online content categorized as hateful or abusive. DWMW17 was labeled for three classes - hate speech, offensive, and neither, while FDCL18 had additional spam category in addition to the standard hateful and abusive categories.

In the SemEval's competition [38], the OffEval dataset was used as a reference to provide context for detecting offensive language in Twitter-based communication. This dataset was annotated to differentiate between comments that were deemed offensive and those that were not, based on their usage of insults, threats, or profanity.

Multiple sources were utilized to obtain the raw data required for training machine learning models to identify instances of hate speech. Some of the sources used included GitHub repositories "hate-speech-and-offensive-language,"[1] which collected tweets by eliminating duplicates through preprocessing techniques such as correcting grammar and spelling errors, expanding contractions, and converting emojis, emoticons, and numbers to text.

Davidson et al. [3] present a dataset that is obtained from a collection of Twitter messages that are labeled based on their likelihood of containing hate speech or offensive language. The dataset mentioned earlied serves as a crucial resource for researchers who aim to create more

[1] https://github.com/t-davidson/hate-speech-and-offensive-language

precise algorithms for detecting hate speech, as well as for those who want to study the usage patterns of such language on social media platforms.

Several sources were utilized to gather the initial raw data for this study. These sources comprised publicly available datasets that were labeled for sexism or racism, as well as other online resources explicitly created for collecting sets of English-language tweets that pertain to the topics of hate speech and offensive language.

After the initial assembly, the raw data underwent several filtering processes aimed at improving its quality. These processes included eliminating duplicates, correcting errors related to language usage or formatting (such as emojis or emoticons), removing unwanted characters and symbols such as numbers or URLs, and expanding contractions where necessary. The resulting preprocessed dataset was then passed through further analysis.

2.3 Feature Extraction and the Use of N-Grams

In the field of *natural language processing* [13], as well as in machine learning, feature extraction plays a significant role in converting raw text data into usable features for analysis. The process includes several steps, such as: pre-processing, tokenization, and n-gram extraction, to further refine these features. Depending on the specific task requirements, various techniques can be utilized, such as statistical measures or deep learning methods like Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN).

Feature extraction is an important factor in detecting hate speech, as this task requires an advanced approach which can capture subtle nuances of language. It is essential to note that feature extraction is not only limited to hate speech detection. Actually, it has numerous applications in different fields, such as image captioning and fake news detection.

In the computer vision domain, feature extraction plays a critical role in generating textual descriptions from visual features through machine learning and natural language processing techniques. As mentioned earlier, deep learning models like AlexNet [17] have enabled significant progress in the capability of feature extraction for computer vision tasks.

Similarly, feature extraction is utilized to detect false news by extracting pertinent features using Natural Language Processing (NLP) tools [22]. These extracted features are subsequently

studied through machine learning algorithms to determine whether the news is authentic or not.

In addition to the typical statistical methods for extracting features, such as n-grams and TF-IDF [11], rule-based encodings can be employed using linguistic annotation tools like CoreNLP [20]. TF-IDF stands for Term Frequency-Inverse Document Frequency. It measures the importance of a word in a document compared to a collection of documents. It helps identify significant words for various NLP tasks.

These tools help measure word-dependency tags and relationships from text data. By combining these techniques, NLP methods can be strengthened to deliver additional features that improve the efficiency of searching scientific data.

It is crucial to subject the extracted features, obtained through intricate statistical models or complex language models often regarded as "black magic," to thorough evaluation before proceeding with their application. "Black magic" is a term that describes highly complex and enigmatic techniques or algorithms that might be difficult for individuals to grasp without specialized knowledge [2].

2.4 SVM Classifier and Parameters in Automated Hate Speech Detection

Support vector machines (SVMs) is a supervised learning model that has associated learning algorithms, which can be utilized for regression analysis and classification in the domain of machine learning [30]. These models were initially developed at ATT Bell Laboratories during the early 1990s by Vladimir Vapnik and colleagues, and are regarded as one of the most powerful prediction methods based on statistical learning frameworks or VC theory proposed by Vapnik and Chervonenkis.

The algorithm constructs a model using a collection of labeled training examples, where each example is associated with one of two categories. This model is then utilized to classify new examples into one of the two categories. It makes a non-probabilistic binary linear classifier. The algorithm maps training examples to points in space to maximize the width of the gap between the two categories. Then, it maps new examples into the same space to predict which category they belong to based on which side of the gap they fall on.

Besides doing linear classification, SVMs can perform non-linear classification efficiently through the use of the kernel trick. This technique implicitly maps inputs into high-dimensional feature spaces, making it possible to perform non-linear classification using a linear classifier in the transformed space. This allows for more complex decision boundaries to be learned and can lead to higher accuracy in classification tasks.

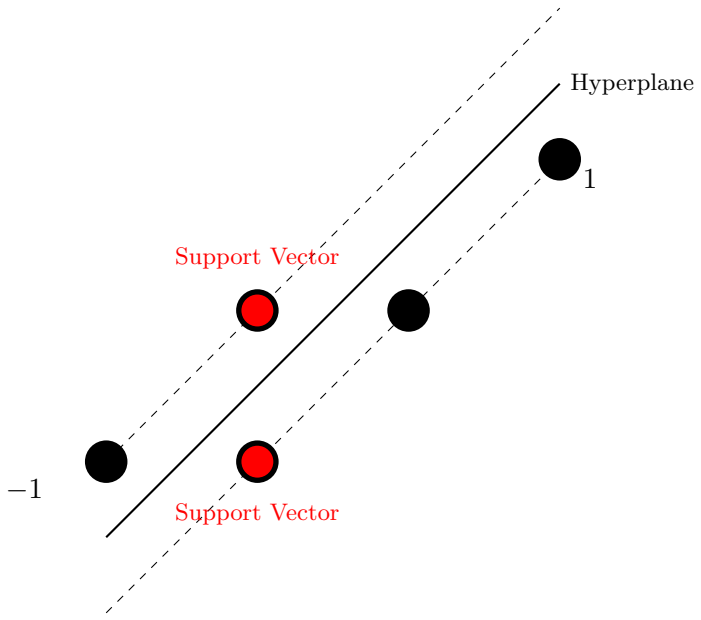

Figure 2.1: Support Vector Machine (SVM) Architecture

The Support Vector Machine (SVM) algorithm is a renowned supervised machine learning technique to tackle classification problems effectively [10]. The main objective of this algorithm is to identify the optimal hyperplane that can categorize data points into different classes as shown in Figure 2.1 . In N-dimensional space, a hyperplane acts as a boundary that separates data points into different classes based on their features. SVM is used to classify data points into different groups. It does this by drawing a line, called a hyperplane, between the groups. It tries to make the line as far away as possible from the data points, which are called support vectors. SVM algorithms can solve different classification problems like image, text, and sentiment analysis. They work well when there are more features than samples. Davidson et al. benefit from SVM in their research as well. SVM finds an optimal line in the data to distinguish different groups and maximizes the margin between them, improving the model's classification of new data points.

The SVM classifier used in the study employed a type of kernel called the "radial basis function (RBF)" kernel [9], which is commonly used in text classification. The RBF kernel is useful for capturing non-linear relationships between features and is well-suited for working with high-dimensional data. The RBF kernel is often used in machine learning, especially in SVM algorithms. It compares two inputs and gives a score based on their distance in a high-dimensional space [9].

Davidson et al. searched for the best settings for the SVM classifier by using a grid search. They tried out different values for the regularization parameter C, which decides how much importance is given to correctly classifying each training example versus maximizing the margin, and the gamma parameter, which controls how much influence a single training example has on the decision boundary. According to the authors, they attained impressive accuracy and F1-score results by utilizing the SVM classifier with the chosen settings.

More detailed information will be provided in Section 2.6. This method of using an SVM classifier with an RBF kernel and fine-tuned hyperparameters is widely used in identifying hate speech and other text classification tasks.

2.5 Understanding the Hate Speech Detection System's Performance Metrics

One of the main ways to measure the performance of a model is through accuracy [30], which calculates the number of correctly predicted instances relative to all cases in a dataset. However, accuracy may not be the most reliable metric when dealing with imbalanced datasets. This issue is commonly seen in hate speech detection models, where non-hate speech instances far outnumber their hate-filled counterparts, resulting in an uneven distribution. To overcome this problem and obtain more accurate results, precision, recall, and F1-score are considered critical alternatives.

Precision is a way of measuring how many instances of hate speech were correctly predicted out of all the predicted instances [37]. We can calculate it by dividing the number of true positives by the total number of predicted positives [30]. Recall, on the other hand, measures the number of true positive predictions out of all the actual occurrences of hate speech in the dataset [37]. It provides an accurate prediction rate for hate speech cases relative to their total count in the dataset. Lastly, F1-score is a weighted average function that balances precision and recall [37].

Precision

$$\text{Precision} = \frac{\text{True Positives}}{\text{True Positives} + \text{False Positives}}$$

Recall

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

F1 Score

$$\text{F1 Score} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

In addition, Davidson et al.'s study uses the receiver operating characteristic (ROC) curve and area under the curve (AUC) as performance evaluation measures. The ROC curve shows the trade-off between the true positive rate and the false positive rate at various classification thresholds. The AUC measures the area under the ROC curve and represents the model's ability to distinguish between the two classes. Davidson et al. use these performance metrics to evaluate the performance of their SVM model in detecting hate speech and compare it with other models in previous studies.

These performance metrics provide a detailed evaluation of the model's precision, recall, accuracy, F1 score, and ability to differentiate between hate speech and non-hate speech instances [30] [37].

2.6 Evaluating Model Performance of the Study

Davidson et al. [3] utilized a crowd-sourced hate speech lexicon to identify tweets containing relevant keywords. Their approach involved training a multi-class classifier for tweet categorization, and the assessment of their SVM model incorporated precision, recall, F1-score, and accuracy metrics.

In the study by Davidson et al., a multi-class classifier was trained to categorize tweets, with logistic regression and SVM models showing promise. The final, best-performing model, however, is logistic regression with L2 regularization. They utilized precision, recall, F1-score, and AUC metrics for a holistic performance evaluation, recognizing the complexities in distinguishing hate speech from offensive language. The study suggests the importance of using a range of metrics for a detailed assessment of a model's ability to identify hate speech. The study's best-performing model achieved an overall precision of 0.91, recall of 0.90, and F1-score of 0.90. The area under

the ROC curve (AUC) was 0.95. However, it's essential to note that nearly 40% of hate speech was misclassified, indicating room for improvement in the model's ability to discern hate speech accurately. The confusion matrix displays the model's classification accuracy for each category: 61% for 'Hate', 91% for 'Offensive', and 95% for 'Neither'. Misclassifications are indicated in non-diagonal cells, such as 31% of 'Hate' instances being incorrectly labeled as 'Offensive' as shown in Figure 2.2.

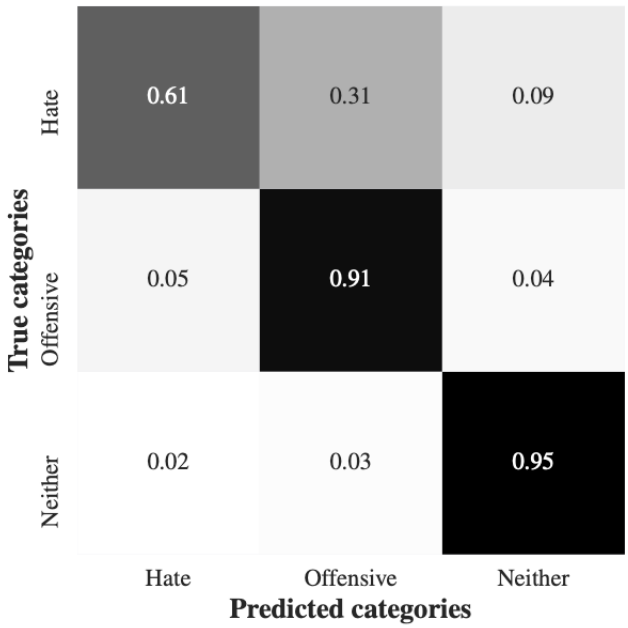

Figure 2.2: Confusion Matrix of the Tweet Classification Model

2.7 Summary

Throughout this chapter, we discussed Thomas Davidson et al.'s research on automated hate speech detection and the issue of offensive language. The study utilized a dataset of tweets that contained instances of both hate speech and non-hate speech, which was preprocessed to remove noise and irrelevant data. The feature extraction process involved generating n-grams that captured the linguistic patterns and context of the tweets.

Overall the study provides multiple machine learning models, including SVM, that were evaluated for detecting hate speech using a dataset of tweets. The final model implemented was logistic regression with L2 regularization, chosen for its interpretability and performance in prior research.

In the upcoming Chapter 3, we will explain the necessary steps to replicate Davidson et al.'s work. Moreover, in Chapter 4, we focus on improving the results by exploring alternative databases, optimizing the code, and seeking unbiased outcomes.

3

Understanding Davidson et al.'s Research: A Detailed Analysis and Reinterpretation of the Original Study

In this chapter, we start by examining the structure of the original code in Section 3.1. This is followed by an exploration of the project's structure and its implementation in Section 3.2, focusing on the components and workflow of the classifier. Moving on to Section 3.3, we enhance the speech classifier and address challenges associated with executing the dataset in older versions of Python. Subsequently, we reproduce and scrutinize the findings of the study within the same section, executing the refined speech classifier in Section 3.4. The chapter concludes with a comparative analysis of the updated results versus those from the original study in Section 3.5.

3.1 Overview

Davidson et al.'s [3] dataset comprises 24,785 English language tweets collected using Twitter's public streaming API [19]. These tweets were manually annotated by human judges to identify instances of hate speech. However, it is essential to note that the dataset was not randomly sampled, and tweets were selected based on certain keywords associated with hate speech [3].

Dataset Size	Training Data Size	Test Data Size
24,802	16,536 entries	8,266

Table 3.1: Size of the Database

The overview in Figure 3.1 is designed to complement Chapter 2, filling in necessary details for this chapter and identifying which elements of the original study were retained or altered for the replication process. We aim to clearly articulate the components from the initial research that are critical for replication but were not discussed in Chapter 2. The sequence presented in the figure follows the practical application order: introduction of the original work, detailing of the methods used by Davidson et al., and preparation for the subsequent replication process. The figure organizes key elements for replication and highlights updates to the original study.

1. **Training**: In Machine Learning applications, the training dataset is critical for instructing the model to detect hate speech on Twitter. This dataset guides 'final_classifier.ipynb' which processes the data and exports a serialized version of the model into pickle files ('final_model.pkl'). These files are then utilized by our speech classifier script ('classifier.py') to carry out evaluations and produce the final results.

2. **Validation**: In the validation phase, the data undergoes k-fold cross-validation, where it's split into 'k' parts. Each subset serves as the test set with the remaining parts as the training set, iteratively. This method evaluates the model's predictive performance and outputs machine learning models (.pkl files). These models are then input into the original classifier ('classifier.py') to obtain evaluation results.

3. **Analysis**: In the analysis phase, we utilize 'labeled_data.csv' in 'classifier.py' for evaluation. This data is crucial for both current analysis and the replication process in Chapter

3, where it is used with an updated classifier 'speech_classifier.py' in Section 3.4, leading to the evaluation detailed in Section 3.5. This dual usage underscores its importance in both understanding the original model and adapting it for new research. Note that the dataset named 'labeled_data.csv' in the figure is identified as the analysis dataset.

Having established the key elements of the Machine Learning application, we now have a clear grasp of both the original study and the modifications for this chapter. As we progress, the chapter will focus on elucidating these changes in greater detail, thereby enhancing our understanding of the replication process.

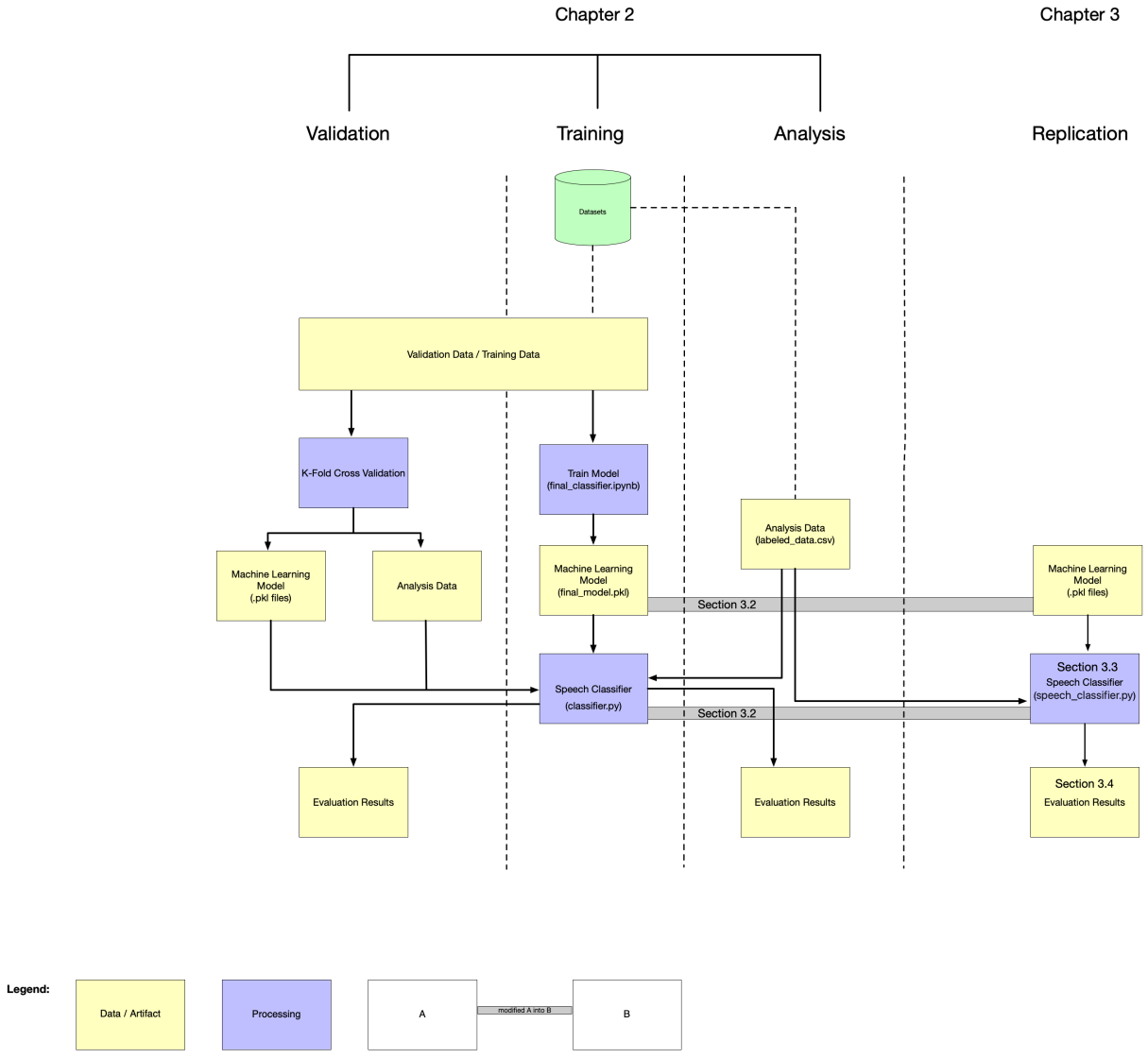

Figure 3.1: Overview

3.2 Project Structure and Execution

This section outlines the structure of our project, focusing primarily on the “classifier” folder, which is central to our work and explains how to execute the original code.

3.2.1 Classifier Folder Overview

In the "classifier" folder, you will find:

1. **classifier.py**: A script that loads the pre-trained classifier and its associated files, transforms new input data into the correct format, and executes the classifier to output results.
2. **final_classifier.ipynb**: A Jupyter notebook used to finalize the classifier, ensuring feature transformation is optimized and results are consistent.
3. **final_idf.pkl**, **final_model.pkl**, **final_pos.pkl**, **final_tfidf.pkl**: These files contain essential data for the classifier's operation, including inverse document frequency values, the final machine learning model, part-of-speech tagging information, and term frequency-inverse document frequency values, respectively.
4. **trump_tweets.csv**: A sample dataset of Donald Trump's tweets, used for testing the classifier.

3.2.2 Additional Project Components

Aside from the classifier, the project structure includes:

- **Data Folder:** Contains *labeled_data.csv* with 24,785 rows of tweets labeled as hate, offensive, or neither.
- **Input Folder:** Holds data regarding the target audience for analyzing the distribution of the original work.
- **Output Folder:** Stores the results generated by the program.

3.2.3 Code Execution Procedure

To run the classifier on the sample data (Donald Trump's tweets), execute the following command in the terminal:

```
python2 classifier.py
```

This command initiates the classifier, which processes the data and outputs the progress and category predictions for each tweet in the terminal.

For applying the classifier to different datasets, modify the inputs on lines 209 and 210 of the **classifier.py** file. This flexibility allows for the classification of diverse datasets, providing insights into various categories of input data.

3.3 Modifying the Speech Classifier

In this section, we give an overview of the modified components, explain the original methods, packages, and components, and discuss encountered challenges.

3.3.1 Overview of Modified Components

To make the pickle (.pkl) files accessible, we converted them into text (.txt) files using a Python program called "CreateNewPkl.py." This process involved copying and pasting the data from the .txt files and saving it as "new_final_*.pkl" files using sklearn version 1.2.2 and the joblib.dump function.

The "classifier.py" script has been updated to incorporate these "new_final_*.pkl" files for further processing. We made necessary adjustments to the code to ensure compatibility with the new datasets.

Before running the program, please execute the following four Python commands to download the required dictionaries or databases necessary for smooth program execution. These commands are crucial for the code's proper functioning.

```
import nltk
nltk.download("stopwords")
nltk.download('averaged_perceptron_tagger')
nltk.download('punkt')
```

Some modifications are required due to incompatible Python versions. Additionally, you need to install various packages using the pip command to ensure compatibility. Here are the steps you

need to take:

1. Open the code files and check for any Python 2.7 specific syntax or functions that may not be compatible with Python 3.x.
2. Modify these parts to make them compatible with the newer version.

- Install the required packages using pip to ensure you have the correct versions:

```
pip install numpy
pip install joblib
pip install scikit-learn
```

- The pip command will automatically install the most recent versions of the packages, which should work with your current Python version (Python 3.x).

To modify the program, follow these steps:

1. Update the code to take a directory (csvinput) as input instead of a single CSV file. The code should loop through all the files in the directory, processing each file, and outputting the results to another directory (results).
2. Incorporate the bug fix given in the GitHub forum (https://github.com/t-davidson/hate-speech-and-offensive-language/issues/14) and replace the lines of code mentioned in the bug fix from:

```
tweet = " ".join(re.split("[^a-zA-Z]*", tweet.lower())).strip()
```

To:

```
tweet = " ".join(re.split("[^a-zA-Z]+", tweet.lower())).strip()
```

The bug fix rectifies the regex pattern issue, resulting in improved program functionality. With these enhancements, the program can now process a directory of files, generate individual results for each file, and save them in the designated "results" directory.

3. Copy the module from the .ipynb file to generate a confusion matrix for labeled_data.csv. The module should produce a file named confusion.pdf.

4. For runs using new pkl training sets, change the output directory to "results_new_training" and the confusion matrix file to "confusion_new_training.pdf".

5. Place all file names and directories at the top of the Python file for easy access and modification.

6. In the function "other_features_(tweet)", consider hardcoding the features according to the training set used. For example, exclude the feature "num_words" for the new dataset "g2g5".

   By implementing these changes, the program will be able to process multiple files in the specified directory, handle different training sets, and generate appropriate output and confusion matrix files.

Changes Made	**Description**	**Enhancements Made**
Replaced two instances of the function call "get_feature_names()" with "vectorizer.get_feature_names_out()".	The updated version of this function now returns a NumPy array instead of a Python list.	The program has been modified to accept a directory (csv-input) as input instead of processing individual files. It now iterates through all the files within the specified directory.
Added the parameter **"solver=liblinear"** to the function "SelectFromModel(LogisticRegression())".	In the previous version, "liblinear" used to be the default solver.	For each file, the program generates results and saves them into separate files within the "results" directory, maintaining the original file names.

Table 3.2: Changes and Enhancements Made

3.3.2 Understanding the Original Methods

It is important to note that the code for this project was developed in 2019 and is accessible through this link: https://github.com/t-davidson/hate-speech-and-offensive-language. It is worth mentioning that the last changes were made four years ago, as Davidson et al. noted. They have made it clear that "the repository is no longer actively maintained, and they won't address compatibility issues with new versions of Python or packages used. They also will

not accept any pull requests. If you intend to use this data or code in your research, it is advisable to review the issues, as several GitHub users have suggested potential changes or improvements to the codebase." Therefore, it is reasonable to expect that the code may require an older version of Python to work as intended.

3.3.3 Understanding Packages and Components

Library name	Usage
numpy	for numerical computations in Python
pandas	data manipulation and analysis library
joblib	particularly useful for saving and loading machine learning models
sklearn.svm	provides implementations of support vector machine algorithms for classification and regression tasks
sklearn.linear_ model	provides implementations of logistic regression models
sklearn.feature _ selection	offers feature selection techniques
nltk (Natural Language Toolkit)	library for natural language processing tasks
string	built-in module in Python
sys.setrecursionlimit(1500)	set maximum depth of the Python interpreter's recursion stack to 1500
pickle	for serializing and deserializing Python objects
vaderSentiment.vaderSentiment	provides a sentiment analysis tool called VADER (Valence Aware Dictionary and sEntiment Reasoner)
textstat	module for computing various text readability metrics

3.3.4 Overcoming Dataset Execution Challenges in Older Python Versions

In summary, during the replication of this research, the most significant challenges encountered were as follows:

- **Compatibility Issues**: The code was developed using Python 2.7, and this posed compatibility problems with newer versions of Python. A separate environment had to be set up to handle the pickle data files, converting them to text files, and repickling them in Python 3.
- **Bug Fixing**: Initially, the code produced different results from the original paper. After identifying a bug in the code through a forum post, implementing the necessary bug fix was crucial to achieving identical results.
- **Unpredictable Behavior**: Understanding the reasons behind the program's occasional opposite behavior from expected results was challenging. The use of the SKLearn package made it difficult to trace the specific operations that led to this unpredictability.
- **Hardcoded Features**: Certain features in "classifier.py" and "final_classifier.ipynb" were hardcoded based on the original dataset's feature selection process. Notably, the "num_words" feature heavily influenced hate speech classification in the original dataset but exhibited less impact in the newer dataset.

3.4 Running the Modified Speech Classifier

After making the necessary modifications and adjustments to the code, we are ready to execute the program and produce the results. This will allow us to compare them with the original findings to assess the accuracy and consistency of the replication process. By conducting this comparison, we aim to validate the code modifications' effectiveness and ensure our replication's reliability.

Loading The Dataset

Starting the process, we load the dataset comprising 24,785 rows and 7 columns. The distribution ratio for each category is as follows:

- Around 28% of the tweets in the dataset are classified as hate speech.
- Around 2.4% of tweets are considered offensive language.
- Approximately 54% of the tweets in the dataset do not belong to any of the specified classes.

	Unnamed: 0	count	hate_speech	offensive_language	neither	class
count	24783.000000	24783.000000	24783.000000	24783.000000	24783.000000	24783.000000
mean	12681.192027	3.243473	0.280515	2.413711	0.549247	1.110277
std	7299.553863	0.883060	0.631851	1.399459	1.113299	0.462089
min	0.000000	3.000000	0.000000	0.000000	0.000000	0.000000
25%	6372.500000	3.000000	0.000000	2.000000	0.000000	1.000000
50%	12703.000000	3.000000	0.000000	3.000000	0.000000	1.000000
75%	18995.500000	3.000000	0.000000	3.000000	0.000000	1.000000
max	25296.000000	9.000000	7.000000	9.000000	9.000000	2.000000

Figure 3.2: Overview of Columns and Rows

Using the table, we can apply the following rules to gain a clearer understanding of the presented data:

- count = number of CrowdFlower (CF) users who coded each tweet (min is 3, sometimes more users coded a tweet when judgments were determined to be unreliable by CF).
- hate-speech = number of CF users who judged the tweet to be hate speech.
- offensive-language = number of CF users who judged the tweet to be offensive.
- neither = number of CF users who judged the tweet to be neither offensive nor non-offensive.
- class = class label for majority of CF users:
  - 0 hate speech
  - 1 offensive language
  - 2 neither

The dataset's histogram reveals a pattern, indicating an imbalanced distribution. Many tweets containing words associated with "hate" based on Hatebase were classified as offensive solely by the CF coders. A more significant proportion of tweets were categorized as neither hate speech nor offensive language compared to those labeled as hate speech.

Data Preprocessing

Several steps are taken to address data irregularities and ensure data quality during preprocessing. These steps aim to standardize the dataset by handling misspellings and errors.

- Firstly, the preprocess function takes a text string as input,
- Then it replaces URLs in the text with the placeholder "URLHERE",
- Then it reduces whitespace to a single instance,
- Next it replaces mentions (like: Twitter usernames) with the placeholder "MENTIONHERE".

By applying these pre-processing steps, we create a cleaner and more standardized dataset. This standardized representation facilitates accurate counting of URLs and mentions, which provides valuable insights for further analysis and modeling.

Following the pre-processing step, the next phase is tokenization, which plays a vital role in standardizing the counts of URLs, mentions, and other textual elements.

Tokenization

After we finish preparing the tweets, the next step is to tokenize them [34]. We can do it using a function called "tokenize". This is how our code works:

- Firstly, tokenize function takes a tweet as input,
- Then it removes punctuation and excess whitespace from the tweet,
- Next it converts the tweet to lowercase,
- Next it has to split the tweet into individual words (based on tokenization rules)
- Stemming is applied to the tweets, reducing words to their base or root form, which captures their meaning and simplifies analysis.

We can see the code of this function here:

Program 3.1: Tokenize Function

```
def tokenize(tweet):
    tweet = " ".join(re.split("[^a-zA-Z]*", tweet.lower())).strip()
    tokens = [stemmer.stem(t) for t in tweet.split()]
    return tokens
```

Program 3.2: Basic Tokenize Function

```
def basic_tokenize(tweet):
    tweet = " ".join(re.split("[^a-zA-Z.,!?]*", tweet.lower())).strip()
    return tweet.split()
```

After tokenization, the next step is constructing the TF-IDF matrix and obtaining relevant scores.

TF-IDF Matrix

TF-IDF, a widely used weighting scheme in natural language processing, stands for Term Frequency-Inverse Document Frequency [31]. The values in the matrix signify the TF-IDF scores for each token in each tweet. This matrix forms the basis for subsequent analysis and modeling.

- For each token in the tokenized tweets, we calculate its Term Frequency (TF), representing its frequency or occurrence within each tweet.
- Next, we use IDF calculation to quantify the rarity of terms across all tweets in the dataset. IDF emphasizes less common terms by taking the logarithm of the total number of tweets divided by the number of tweets containing the term, providing insight into their significance.
- Finally, the TF-IDF score is obtained by multiplying the Term Frequency (TF) with the Inverse Document Frequency (IDF). This score signifies the significance of a term within a specific tweet relative to its occurrence in the entire dataset.

Part Of Speech (POS)

After tokenization, the next step involves using Part Of Speech (POS) tags for the tweets. POS tags [21] help us understand the grammatical structure of words in sentences. We assign grammatical tags to each word in the tokenized tweets and save them as a string.

We will extend this process to include Part of Speech (POS) tagging. A function called "get_pos_tags" will be created to take a list of tweets as input, process them to extract POS tags, and return a list of POS tag strings for each tweet. The function will preprocess each tweet, tokenize it into words, assign POS tags to each word, and combine the tags into a string for each tweet.

Here is the code for the "get_pos_tags" function:

Program 3.3: Function to Retrieve Part-of-Speech Tags

```
def get_pos_tags(tweets):
    tweet_tags = []
    for t in tweets:
        tokens = word_tokenize(preprocess(t))
        tags = nltk.pos_tag(tokens)
        tag_list = [x[1] for x in tags]
        tag_str = " ".join(tag_list)
        tweet_tags.append(tag_str)
    return tweet_tags
```

Extracting Features

Next, our code delves into extracting and analyzing diverse features from Twitter data. It incorporates functions that facilitate tasks such as counting URLs, mentions, and hashtags, as well as calculating sentiment scores, text readability metrics, and other Twitter-specific features.

Let us now elaborate on the two essential functions for this part:

- Count twitter objects:
    1. Replaces URLs, whitespace, mentions, and hashtags in a text string.
    2. Returns the counts of URLs, mentions, and hashtags.
- Other features:

1. Calculates various features such as sentiment scores, syllable count, character count, term count, and readability metrics for a tweet.
2. Includes Twitter-specific features like counts of hashtags and mentions.
3. Returns a list of these calculated features.

Decoding The Code's Functions

Functions in text analysis create metrics like sentiment and readability scores. One metric, the modified Flesch-Kincaid grade, calculates average sentence length by simply using the total word count. This modification enhances the accuracy of the Flesch-Kincaid grade readability score by providing a more precise measure, improving its effectiveness compared to the traditional method.

This function also generates other features such as FKRA (Flesch-Kincaid Reading Age), FRE (Flesch Reading Ease), number of syllables, average syllables per word, number of characters, total number of characters, number of terms, number of words, number of unique words, VADER sentiment scores (harmful, positive, neutral, compound), number of hashtags, number of mentions, number of URLs, and an indicator for retweets.

Building A List Of Generated Feature Variables

For the next step, we compile a list of variable names for the generated features, comprising three categories: the original vocabulary variables, the POS (Part-of-Speech) vocabulary variables, and the names of additional features.

To create this list, we traverse the vocabulary dictionary and allocate each variable name to its corresponding position. The same procedure is applied to the POS vocabulary, ensuring that the variable names are aligned with their respective positions.

We include the names of other generated features in the list, along with the original vocabulary and POS vocabulary variables. By combining these sets of variables, we obtain the final list of feature names. This list serves as a valuable reference, enabling us to associate each feature with its position in the generated feature matrix, facilitating interpretation and analysis. Having these variable names available supports further analyses, including ranking feature importance, model interpretation, and examining the target variable.

Running The Model And Generating Results

In the final step, we execute the model using a GridSearch [28] with 5-fold CV [7]. GridSearch is a systematic technique that explores a predefined set of hyperparameters for the model and identifies the combination that yields the best performance. It allows us to discover the optimal model configuration by evaluating various hyperparameter combinations.

Using GridSearch with 5-fold cross-validation, we fine-tune the model's hyperparameters and obtain an unbiased performance estimate. The best model is identified based on evaluation metrics like accuracy, precision, recall, or F1-score.

3.5 Comparing our Results to the Original Study

Lastly, we present a figure comparing our replication results with the original study. The figure will demonstrate key performance metrics offering a side-by-side analysis of our model versus the original. After overcoming Python compatibility and codebase challenges, our replication yielded results consistent with the original, including the confusion matrix.

Comparing the original (see Figure 2.2) and replicated (see Figure 3.3) results of the tweet classification model, we see minor differences. Both versions maintain high accuracy for "Hate" and "Neither" categories, at 0.61 and 0.95 respectively. The "Offensive" category shows a slight decrease from 0.91 in the original to 0.90 in the replicated results. Misclassifications are very similar between the two, with a negligible decrease in the confusion of "Hate" with "Neither" and "Offensive" with "Neither" in the replicated study. These variations are minimal, suggesting that the replication closely matches the original model's performance.

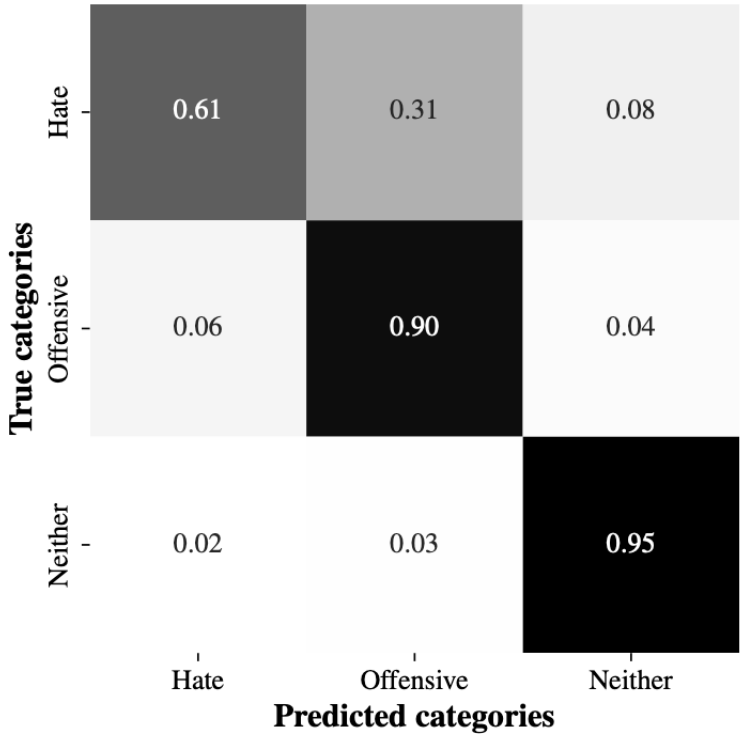

Figure 3.3: Replicated Confusion Matrix

3.6 Summary

In this chapter, we reviewed Davidson et al.'s research, modified and ran the speech classifier, comparing our results to the original study as shown in Chapter 2. Our replication, after addressing Python and codebase issues, mirrored the original model, yielding consistent results and a confusion matrix. The next chapter addresses biases identified, analyzing group distributions and introducing a new dataset for enhanced quality. We will outline our approach, present findings, and analyze results using the new dataset. Our focus will be on understanding the impact of different distributions on our methods and experiments.

4

Unveiling New Insights: Our Approach, Experimental Findings, and Results in Hate Speech Analysis

In this chapter, we start by establishing a new repository in Section 4.1. Then we perform experiments using the original research dataset while considering its codebase limitations in Section 4.2. We analyze group distributions, present results, and address biases in Section 4.3. Due to the lower quality of the original dataset, we introduce a new dataset in Section 4.4, discussing group distributions and performance outcomes in Section 4.5. Lastly, we use confusion matrices to highlight reduced bias and discuss enhancements, limitations, and potential validity concerns in Section 4.6 and in Section 4.7.

4.1 Establishing a New Repository

We set up a GitHub repository and start the modification process. The repository we establish includes the complete project layout. We also detail the necessary procedures for running the speech classifier and executing the k-fold cross-validation.

4.1.1 Creating a GitHub Repository and Initiating Modifications

Upon establishing and configuring the classifier as detailed in Chapter 3, we arranged our project repository on Github [2].

The GitHub repository contains the following project structure. Within the "speech_classifier" folder, six primary files can be found:

1. **generate_group_csv.py**: This program reads speech data from the "data" folder (provided by Berkeley) and allows users to select the number of target groups for analysis. It creates different scenarios for testing by altering the number of each targeted group based on the "data_name" variable. For our discussion, we use the "balanced" data name as an example.

2. **count_groups.py**: This simple script provides the actual number of groups resulting from the previous program. Additionally, it calculates the percentage of each targeted group produced, considering potential overlaps between different groups.

3. **generate_trained_model.ipynb**: This Jupyter Notebook script generates the trained model necessary for use in the speech classifier program. It produces five .pkl files which are essential components for the subsequent steps.

4. **speech_classifier.py**: This program functions as the actual speech classifier. It analyzes all speech files in the input folder and determines the number of hate speech instances in the input files. Additionally, it is configured to analyze a pre-labeled data file named "labeled_data.csv" for evaluation purposes. Trained model files, as mentioned earlier, are required components for this program.

5. **generate_cv_data.py**: This program is designed for preparing data for k-fold cross-validation, while ensuring the desired target group distribution in the splits. It splits the files into k+1

[2] https://github.com/MoSadri/Thesis_2023

pieces, with one piece designated as the analysis dataset and the remaining k pieces as training datasets. This process repeats k times, ensuring each piece of data has an opportunity to be the analysis dataset. Additionally, this program generates the necessary .pkl files for each trained dataset.

6. **run_cross_validation.py**: This program executes the k-fold cross-validation process. It iterates through all analysis and training sets for each fold and generates quality scores, including accuracy, precision, recall, and F1 score.

4.1.2 Steps to Run the Speech Classifier

1. Run *generate_group_csv.py* to create training data with desired target group numbers.
2. Execute *generate_trained_model.ipynb* in a Jupyter Notebook.
3. Run *speech_classifier.py* to analyze speech files.

4.1.3 K-fold Cross-validation

- Two additional files, *generate_cv_data.py* and *run_cross_validation.py*, are used for k-fold cross-validation.
- By running *generate_group_csv.py* and then *generate_cv_data.py*, you prepare the data.
- Set k (currently 5) in *run_cross_validation.py* to perform k-fold cross-validation. Run it with these commands:

    1. `python generate_group_csv.py`
    2. `python generate_cv_data.py`
    3. `python run_cross_validation.py`

4.2 Analyzing the Original Work: Group Distribution

4.2.1 Dissecting the Original Dataset

For model training and testing, 80% of the annotated tweets were randomly chosen for training the hate speech detection model, while the remaining 20% were used for testing the model's performance. The dataset is available in both CSV and pickled pandas dataframe formats in Python 2.7, offering different options for data exploration and analysis. The five columns in each data file include:

Index	Count	Hate Speech	Offensive Language	Neither	Class	Tweet
0 - 24,784	0 - 6	0	1	2	0 - 2	Text

Table 4.1: Data Labels According to Original Columns on the Tweets

Count: These labels correspond to the original columns in the tweets, indicating the presence or absence of hate speech based on human annotation.

Hate Speech: The number of CrowdFlower [3] users who labeled the tweet as hate speech.

Offensive language: The number of CrowdFlower users who labeled the tweet as offensive language.

Neither: The number of CrowdFlower users who labeled the tweet as neither offensive nor non-offensive.

Class: The class label for the majority of CrowdFlower users. The class labels include 0 for hate speech, 1 for offensive language, and 2 for neither.

In the initial phase of the experiment, we focused on tweets categorized as group 0 (hate speech). Out of the 24,785 tweets, a Python script was utilized to exclusively extract the tweets labeled as hate speech, resulting in a total of 1432 such tweets.

Next, we generated Table 4.2 with four columns. The first column displays the index of the tweets. The second column represents the class, with all values set to 0, denoting that the tweets belong to the hate speech category. The third column contains the tweet content, and the fourth column indicates the targeted group. To clarify the labeling of hate speech tweets, we manually assign labels to identify the specific group(s) targeted in each tweet. We then calculate the number of times a group is targeted, taking into account that one tweet is targeting one or more groups.

This process allows us to analyze the distribution for each target group.

Index	Class	Tweet	Group
0 - 1431	0	Content of the tweet	Targeted group(s)

Table 4.2: Hate Speech Tweets and Their Targeted Groups

Understanding the number of targeted groups in each tweet is crucial as it affects the evaluation metrics for hate speech detection and reveals the extent of bias in the results. The targeted groups in the dataset are as follows:

- Women
- Black People
- LGBT Community
- Mexicans
- Disabled People
- White/Caucasian
- Muslims
- Political
- Latinos
- Jews
- Asians
- Immigrants

4.2.2 Examining Target Group Counts

The bar chart in Figure 4.1 visually represents the target groups along with their respective counts, providing a clear illustration of the varying levels of targeting intensity. It prominently indicates

that the LGBT group is the most frequently targeted, with the black group closely following in frequency, and the women and white/Caucasian groups not far behind.

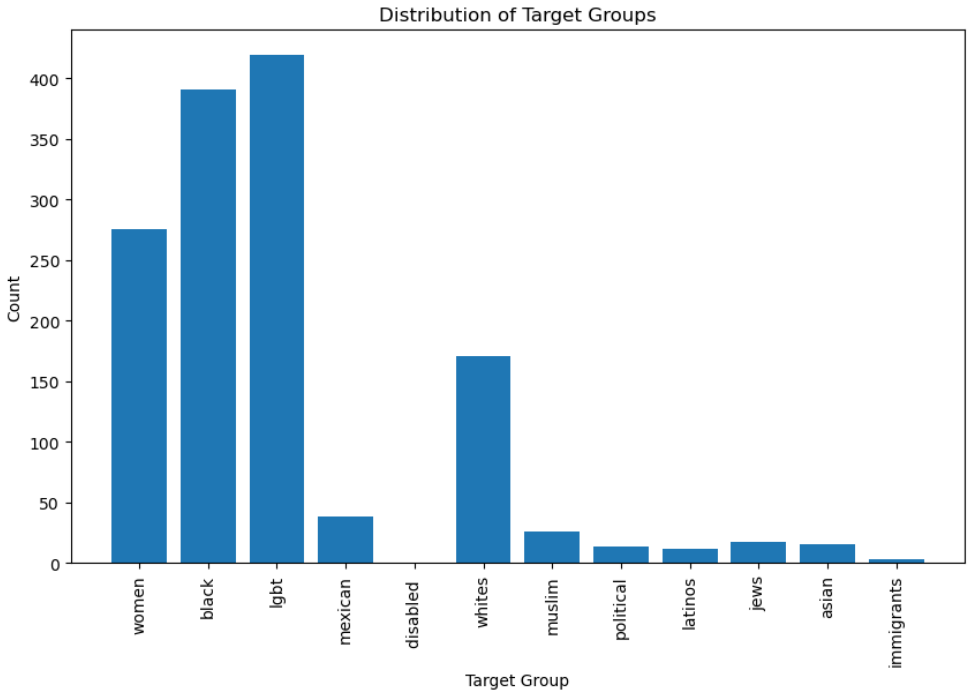

Figure 4.1: Distribution of Groups - Bar Chart

4.2.3 Proportional Representation of Target Groups

The pie chart in Figure 4.2 gives another perspective on the distribution of target groups. It visually represents the proportions of each group within the entire dataset. The LGBT group stands out as the largest target, making up 28.3% of the distribution. The black group follows closely, constituting 27.3% of the total. Women form 18.1% of the target groups, while the white/Caucasian group accounts for 12.1% of the pie chart. Additionally, we find the remaining target groups - including Mexican, Latino, Muslim, immigrant, political, Jews, and Asian - collectively make up the rest of the chart. These groups demonstrate different levels of targeting, which collectively contribute to the overall distribution pattern among the target groups.

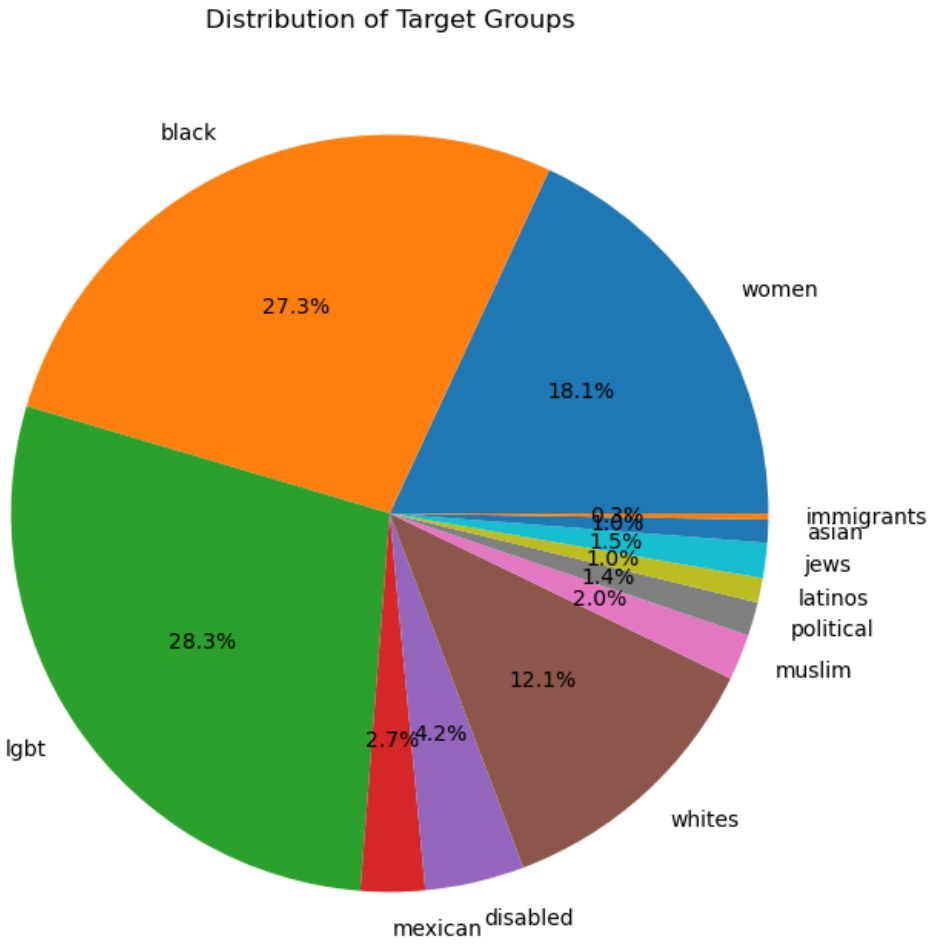

Figure 4.2: Distribution of Groups - Pie Chart

4.3 Examining Class Distribution and Imbalance: Investigating Bias

Next, we delved into the class distribution of the original dataset, revealing significant imbalances among the target groups. For evaluating our research, we chose metrics like accuracy, recall, precision, and F1 scores, consistent with the evaluation methods of the original study.

Target Group	Accuracy	Precision	Recall	F1 Score
Black	65%	83%	73%	75%
LGBT	90%	91%	82%	86%
Women	22%	90%	41%	47%

Table 4.3: Performance Metrics by Target Group

The table shows a disparity in accuracy among target groups, with lower accuracy for Black (65%) and Women (22%) compared to LGBT (90%), indicating potential bias or imbalance in the dataset or model.

4.3.1 Group Distribution: Pointing Out Variations and Bias

The current dataset in Table 4.4 shows a bias by often labeling tweets from the LGBT community as hate speech, and tweets from women as offensive, hinting at a lack of diverse perspectives in its creation. Similarly, the high number of negative labels on tweets from Black individuals suggests cultural biases in the data. This imbalance highlights the critical need for a new dataset that is created with a broad and inclusive approach.

Group	**Hate Speech %**	**Offensive %**	**Neither %**
LGBT	93.32	4.06	2.63
Women	7.97	88.41	3.62
White/Caucasian	70.76	8.77	20.47
Black	59.34	30.43	10.23
Disabled	61.29	19.35	19.35
Mexican	55.56	36.11	8.33
Muslim	89.66	3.45	6.90
Political	76.32	13.16	10.53
Latino	79.31	8.62	12.07
Jews	76.67	10.00	13.33
Asian	75.00	12.50	12.50
Immigrant	70.59	17.65	11.76

Table 4.4: Percentage Distribution of Tweets by Category and Group

4.4 Creating A New Dataset

4.4.1 Addressing Lower-Quality Original Data

The original dataset by Davidson et al. [3] faced significant quality issues and was limited in size, leading us to develop a new, more comprehensive dataset for our model's training. This upgraded dataset enhances the original by improving the sources of data, the timing of data collection, and offering a more varied class distribution. These critical factors set our dataset apart from the original, directly impacting the model's training process and its overall effectiveness.

4.4.2 Collection and Annotation

To ensure unbiased research on hate speech, we have invested significant time and effort in researching and analyzing trustworthy sources to construct a dataset with minimal bias. Our primary objective is to uphold the integrity of the research while avoiding data of low quality or potential harm that could skew the results. For this challenge, we have adopted a systematic approach and chosen the following databases for in-depth analysis as shown in Table 4.5. When choosing a dataset for our research, several factors need consideration as they can greatly influence the outcomes.

Dataset	**Link**
UC Berkeley Dataset	[https://huggingface.co/datasets/ucberkeley-dlab/measuring-hate-speech]
100k English (27593 hate, 30747 offensive, 41660 none)	[https://hatespeechdata.com/English-header]
Dynamically Generated Hate Speech Dataset	[https://github.com/bvidgen/Dynamically-Generated-Hate-Speech-Dataset]
The 'Call me sexist but' Dataset (CMSB)	[https://search.gesis.org/research˙data/SDN-10.7802-2251]
HateCheck: Functional Tests for Hate Speech Detection Models	[https://github.com/paul-rottger/hatecheck-data]
HateXplain: A Benchmark Dataset for Explainable Hate Speech Detection	[https://github.com/hate-alert/HateXplain]
hatEval, SemEval-2019 Task 5 (English)	[https://aclanthology.org/S19-2007/]
HASOC (2019) Hate Speech and Offensive Content Identification	[https://www.ims.uni-stuttgart.de/forschung/ressourcen/korpora/stance-hof/]
Hate speech dataset from a white supremacist forum	[https://hasocfire.github.io/hasoc/2019/dataset.html]
Labelled Hate Speech Detection Dataset from Reddit	[https://github.com/aymeam/Datasets-for-Hate-Speech-Detection]
Large-Scale Hate Speech Dataset (English and Turkish)	[https://github.com/avaapm/hatespeech]

Table 4.5: Overview of Hate Speech Datasets and Links

The selection of the UC Berkeley dataset was based on meticulous criteria essential for maintaining the integrity of hate speech research:

- **Representativeness:** The dataset provides comprehensive coverage of various social groups, ensuring a representative sample for generalizable machine learning outcomes.
- **Diversity:** It includes a broad spectrum of linguistic expressions and cultural contexts, which is crucial for reducing bias in the analysis.
- **Annotation Quality:** High-quality annotations from trained annotators and verified for consistency contribute to the dataset's reliability.
- **Timeliness:** The recent compilation of the dataset captures the current state of online discourse, reflecting the evolving nature of language use in social media.
- **Ethical Sourcing:** The dataset is publicly available and ethically sourced, ensuring transparency and adherence to research ethics.

- **Academic Precedent:** Its prior use in scholarly works provides a benchmark for comparison and lends credibility to our research methodology.

These factors establish the UC Berkeley dataset as a robust choice, positioned to provide valuable insights into the dynamics of hate speech detection and classification. Through this selection, our study aims to contribute meaningfully to the field, building upon a foundation of methodological excellence.

4.4.3 Key Findings From Kennedy et al. and Sachdeva et al.

The dataset used in our research as shown in Table 4.6 is a publicly available version derived from the dataset described in Kennedy et al. [12] and Sachdeva et al [26]. It contains a total of 39,565 comments that have been annotated by 7,912 annotators, resulting in 135,556 rows of data. The primary focus of this dataset is the "hate speech score," which serves as the main outcome variable. Additionally, it includes 10 ordinal labels associated with sentiment, disrespect, insult, humiliation, inferior status, violence, dehumanization, genocide, attack/defense, and hate speech benchmark.

Total Comments	39,565
Annotators	7,912
Rows	135,556
Outcome Variable	Hate Speech Score
Ordinal Labels	10
Target Identity Groups	8
Target Identity Subgroups	42
Annotator Demographics	6 Categories
Subgroup Classifications	40

Table 4.6: Dataset Details

In terms of target identity groups, the dataset covers 8 categories, including race/ethnicity, religion, national origin/citizenship, gender, sexual orientation, age, disability, and political ideology. Moreover, it consists of 42 target identity subgroups, providing detailed insights into various segments of these identity groups.

An important aspect of this dataset is the incorporation of annotator demographics, which are categorized into 6 groups. This information allows for a better understanding of any potential

biases introduced during the annotation process. Additionally, the dataset includes 40 subgroup classifications that further enhance the granularity of the data.

To access the dataset, use the following code on Jupyter NoteBook:

Program 4.1: Loading and Describing the UC Berkeley Hate Speech Dataset

```
import datasets
dataset = datasets.load_dataset('ucberkeley-dlab/measuring-hate-speech', 'binary')
df = dataset['train'].to_pandas()
df.describe()
```

4.5 Examining Group Representation in the Updated Dataset

To analyze the distribution of groups in our dataset of 135,556 rows, we developed the script **count_groups.py**. This program is specifically written to efficiently count and categorize each group's representation within the dataset.

- Open the CSV file and initialize variables for column indices and total row count.
- Create a dictionary called `target_groups` to store the count of occurrences for each targeted group.
- Initialize the counts of all targeted groups to 0 in the `target_groups` dictionary.
- Iterate through each row of the CSV file.
- For each row, check the columns corresponding to the targeted groups. If a group is marked as 'TRUE', increment the count for that group in the `target_groups` dictionary.
- Update the total row count after each row is processed.
- Open an output text file called "groups_counts.txt".
- Write the total row count to the file, indicating how many rows were processed in the CSV.
- For each targeted group, calculate the percentage of its occurrence and write both the count and percentage to the output file.

After running the program, we obtained the distribution for each targeted group, which is showcased below in Table 4.7.

Targeted Group	**Percentage (%)**	**Count**
Women	20.5	27,889
Black People	16.89	22,899
LGBT Community	49.32	65,957
Latinos/Mexicans	6.26	8,497
Disabled People	0.83	1,126
Whites/Caucasians	8.77	9,797
Muslims	9.22	12,509
Jews	5.10	6,924
Asians	5.18	7,025
Immigrants	7.03	9,525

Table 4.7: Hate Speech Distribution in New Dataset by Targeted Groups

Note that the percentages for certain groups may not total 100% because an individual entry can target multiple groups simultaneously. Here are the key observations and insights from the visualization:

1. **LGBT Community**: The LGBT community is the most heavily targeted group, accounting for 49.32% of hate speech instances.

2. **Women**: Women are the second-most targeted group, with 20.5% of the hate speech instances.

3. **Black People**: The percentage of hate speech targeting Black people is significant at 16.89%.

4. **Muslims and Immigrants**: Hate speech directed towards Muslims and immigrants comprises 9.22% and 7.026%, respectively.

5. **Whites/Caucasians**: While at a lower percentage of 7.22%, hate speech targeting Whites or Caucasians is still present.

6. **Latinos/Mexicans, Jews, Asians, and Disabled People**: These groups encounter relatively lower instances of hate speech, with percentages varying between 5.10% and 6.26%.

4.6 Impact of Distribution on Hate Speech Detection Across Scenarios

We selected three primary groups (black, women, LGBT) for analyzing hate speech impact across four scenarios: 1) Tweets predominantly about the black group, 2) Tweets focusing on women, 3) Tweets focusing on LGBT, and 4) A balanced mix targeting black, women, and LGBT groups. These scenarios will help us understand how group distribution affects hate speech detection.

4.6.1 Code Structure Outline

This code is available in the generate_group_csv.py file, as mentioned in the GitHub repository outlined in Chapter 3.

Our main goal is to use this code to create a personalized CSV dataset for training a hate speech detection model. The code gives one the flexibility to choose how different target groups are distributed in a dataset. You can select options like 'balanced', 'women', or 'black' to control the number of instances from each group in the dataset, and you can change these settings as needed.

Additionally, you can adjust the 'hate_score_threshold' to decide what qualifies as hate speech. If a text's hate score is higher than this threshold, it is labeled as '0' for hate speech; otherwise, it is labeled as '2' for non-hate speech.

Below is a step-by-step explanation of the script's workflow:

1. **Reading the Dataset:** Initially, the script loads the 'berkeley_speech_dataset.csv' containing textual data and attributes, including labels for target groups.

2. **Data Packing:** Depending on the chosen distribution, such as 'black,' the script selects the random sample of 'n' rows pertaining to the specified group.

3. **Class Label Assignment:** It labels each row as '0' (indicating hate speech) or '2' (indicating non-hate speech), based on the hate score threshold. The script ensures a balanced representation of these labels for each target group.

4. **Column Selection:** Only essential columns, such as the text, target group labels, and the 'class' label, are retained for clarity and relevance.

5. **Shuffling the Dataset:** The script shuffles the rows at this stage to guarantee a randomized distribution, which is crucial for unbiased training of machine learning models.

6. **Saving the New Dataset:** Finally, the processed dataset is saved as a new CSV file, with its name derived from the selected 'data_name' like 'balanced_dataset.csv'.

4.7 Assessing Classifier Effectiveness: Four Target Group Scenarios

We use the "berkeley_speech_dataset.csv" to create four different scenarios to observe the effectiveness of this speech classifier program. Different scenarios can be created by setting different numbers when running the program "generate_group_csv.py". Here are our configurations shown in Table 4.8.

Scenario	**Configuration**
Configuration for scenario 1	9000 black, 500 women, 200 trans, 150 gay, 150 lesbian
Configuration for scenario 2	9000 women, 500 black, 200 trans, 150 gay, 150 lesbian
Configuration for scenario 3	15000 LGBT, 3300 black, 3300 women
Configuration for scenario 4	3300 black, 3300 women, 2800 trans, 100 gay, 500 lesbian

Table 4.8: Distribution of Target Groups Across Different Scenarios

4.7.1 Scenario 1: Targeting the Black Community

In the first scenario, the emphasis is placed on tweets predominantly featuring black individuals, allowing for a focused and detailed examination.

4.7.2 Scenario 2: Targeting the Women Community

In the second scenario, the emphasis is placed on tweets predominantly featuring the women group, allowing for a focused and detailed examination.

4.7.3 Scenario 3: Targeting LGBT

In the third scenario, the emphasis is placed on tweets predominantly featuring the LGBT group, allowing for a focused and detailed examination. However, the dataset is more balanced compared

to the datasets in the first and second scenario but not as balanced as the dataset in the fourth scenario. The size of the dataset is also larger compared to the other three scenarios to also examine the impact of dataset size on the results.

4.7.4 Scenario 4: Balanced Distribution of Target Groups

In the fourth scenario, our focus shifts to tweets evenly distributed across the black, women, and LGBT groups for a comprehensive analysis.

4.7.5 Original Data Classified with New Classifier

First, we will compare the results obtained from the new classifier with those generated by Davidson et al.'s original classifier using the labeled_data.csv. We will assess the impact of group distribution on hate speech by comparing all four scenarios against the original dataset.

Scenario	**Target Group**	**Accuracy**	**Precision (Hate)**	**Recall (Hate)**	**F1 Score (Hate)**
Black	Black	67%	91%	65%	76%
Black	Women	74%	94%	71%	81%
Black	LGBT	84%	95%	86%	90%
Women	Black	62%	96%	55%	70%
Women	Women	75%	96%	71%	81%
Women	LGBT	70%	96%	68%	76%
LGBT	Black	70%	87%	72%	79%
LGBT	Women	77%	87%	83%	85%
LGBT	LGBT	87%	92%	93%	93%
Balanced	Black	68%	94%	64%	76%
Balanced	Women	83%	94%	84%	89%
Balanced	LGBT	85%	95%	88%	91%

Table 4.9: Performance Metrics by Scenario and Target Group

Comparing the original and updated classifiers (Table 4.3 and Table 4.9, respectively), we see that accuracy for the Black group is steady and accuracy for the Women group has increased, while LGBT accuracy remains high in balanced scenarios. Precision has significantly improved for all groups, now mostly above 90%. The recall for women has notably increased from 41% to 71%, and the F1 score for women has jumped from 47% to 81%, showing enhanced balance and performance for these groups.

Table 4.9 details the calculations derived from the confusion matrix in Figure 4.3. While we

present the confusion matrix of the 'Black' scenario as an example, the same calculation process applies to the other groups and scenarios, ensuring a comprehensive evaluation across various demographics. This approach underscores the comparative analysis of our classifier with Davidson et al. [3]'s original dataset, highlighting the impact of group distribution on hate speech detection systems. The confusion matrix in Figure 4.3 shows the performance of a hate speech detection model for the Black target group under the Black scenario. It shows that 65% of the hate speech was correctly identified (true positives), while 75% of non-hate speech was also correctly recognized (true negatives), with some instances misclassified (35% false negatives and 25% false positives).

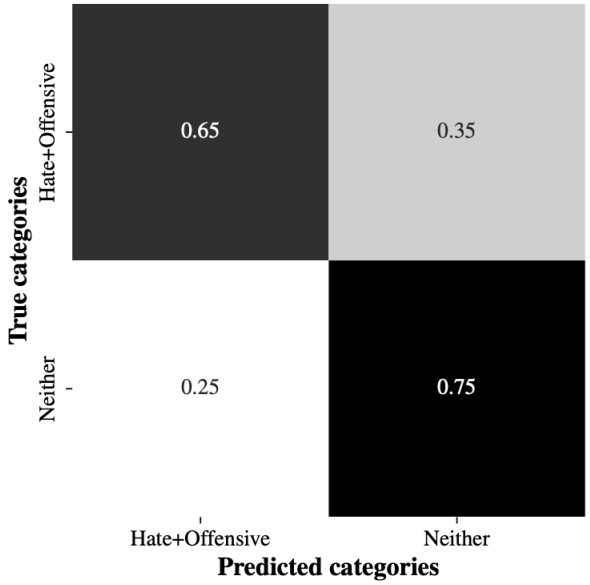

Figure 4.3: Black Target Group: Confusion Matrix

Second, drawing from Table 4.9, we can draw the following conclusions by comparing the results of the three target groups in the four scenarios:

Black Scenario:

- **Accuracy**: The system performs moderately well for detecting hate speech against Black individuals (67%) but is better at detecting hate speech against LGBT (84%). The model's moderate performance may stem from an imbalanced representation of Black individuals in

the training data, limiting the system's learning capacity for this group (Data Imbalance).

- **Precision vs. Recall**: Precision is notably high (91%-95%), but recall is relatively lower (65%-86%), indicating a cautious approach that prioritizes avoiding false positives over capturing all hate speech instances. This may suggests challenges in detecting subtle or varied expressions of hate speech against Black individuals (Linguistic Variability).

- **F1 Score**: Consistent with precision and recall, F1 scores are robust, especially for LGBT (90%). The annotation process might have introduced biases, affecting the model's ability to accurately identify hate speech against Black individuals, as reflected in the F1 score (Bias in Annotation).

Women Scenario:

- **Accuracy:** There is a noticeable dip when detecting hate speech against Black individuals (62%), suggesting possible biases in the model when the data distribution changes. Significantly, the decline in numbers is even more pronounced for the LGBT group compared to the Black scenario (84% to 70%). The notable performance drop for the LGBT group could be attributed to the model's limited grasp of the cultural and contextual subtleties inherent in hate speech directed at these individuals (Cultural and Contextual Factors).

- **Precision vs. Recall:** Precision remains high (96%), but recall drops significantly for Black targets (55%), indicating a higher rate of missed hate speech instances. For the LGBT group, recall also significantly drops from 86% to 68%, pointing to a similar issue of under-detection. The high precision coupled with significantly lower recall suggests that the model's sensitivity settings may be too conservative, leading to the omission of numerous instances of hate speech across groups (Model Sensitivity).

- **F1 Score:** The F1 score is lowest for Black targets (70%), underscoring the critical need for a training dataset that is more representative of the diverse groups targeted by hate speech and encompasses a wider array of linguistic expressions.

LGBT Scenario:

- **Accuracy:** In the LGBT scenario, the model's accuracy for detecting hate speech against Black individuals, the Women group, and LGBT is 70%, 77%, and 87%, respectively. This is the highest across all scenarios, which indicates that training data size plays a crucial role for the general performance of the classifier.

- **Precision vs. Recall:** When identifying hate speech within the LGBT scenario, precision is generally lower but recall is generally higher, often substantially. This suggests a better balance between precision and recall compared to the other scenarios. The balanced precision and recall across groups suggest an optimized model sensitivity that effectively identifies and confirms hate speech instances. Furthermore, it shows an improved model capability in handling the linguistic variability of hate speech.

- **F1 Score:** The resulting F1 scores for the Black, Women, and LGBT groups are 79%, 85%, and 93%, respectively. This is the highest score for the Black and LGBT groups, and almost the highest for the Women group, again highlighting the importance of training data size. The F1 score for the LGBT group is impressive at 93%, reflecting a highly effective balance of precision and recall. The highest accuracy and F1 scores, especially for LGBT (87% and 91%), indicate a more representative training dataset that addresses previous imbalances.

Balanced Scenario:

- **Accuracy vs. Precision vs. Recall:** Overall, the Balanced scenario performs second best, better than the Black and Women scenarios but not as good as the LGBT scenario.

- **F1 Score:** The highest accuracy and F1 scores for the Women group (83% and 89%, respectively) indicate a more representative training dataset for that group that addresses previous imbalances.

4.8 Outcomes, Advancements, and Limitations of the Approach

Our findings reveal that a balanced and diverse dataset is fundamental in the creation of AI systems that are both inclusive and proficient across various demographic groups. The LGBT Scenario's superior performance in nearly all evaluated metrics, the Balanced Scenario's superior performance

for key metrics of the Women target group, and the Balanced Scenario's improved performance over the Black Scenario and Women Scenario accentuates the significance of diversity and equity in training data but also highlights the impact of training data size. Specifically, the Balanced Scenario's heightened accuracy for all groups compared against the Black and Women Scenarios illuminates the enhanced inclusivity and precision afforded by representative datasets. Similarly, the LGBT Scenario is more balanced than the Black and Women Scenarios, which is reflected in the results of the LGBT Scenario. In contrast, the LGBT Scenario's less effective performance within the LGBT Scenario in identifying hate speech directed at Black individuals with an accuracy score 17% less compared to the LGBT group exposes underlying biases, signaling a need for corrective measures in data representation. Such disparities underscore the influence of data composition on the potential for AI bias, especially highlighted by the F1 scores within each scenario, which presented crucial biases that necessitate urgent attention. The difference between the lowest F1 score and highest F1 score within a scenario is at least 11% and as high as 15%. Reflecting on the results for the LGBT scenario specifically, it is clear that when the training data is well-represented, the AI system's precision and recall are notably high. This reinforces the importance of inclusivity in the dataset for robust AI detection capabilities, particularly for groups often targeted by hate speech. The 93% F1 score for the LGBT group is a testament to the model's ability to identify hate speech with high accuracy when provided with diverse and comprehensive data.

Additionally, using strict methods like k-fold validation and creating an 'updated.csv' file with clear labels for different groups was very important. These steps show how crucial it is to carefully test and check AI models with different kinds of data to greatly improve an AI's ability to detect hate speech accurately.

In conclusion, the effectiveness of an AI model deeply relies on the quality and diversity of its training data. Our research not only demonstrates the crucial impact of having a wide range of data reflecting real-world demographics but also highlights the significant influence of group distribution on the model's performance. For AI systems to be fair and efficient, they must be developed with diverse data and undergo thorough testing and validation. This approach ensures that the AI is robust and reliable for varied practical applications.

4.9 Threats to Validity

Our research highlights the nuanced impact of data distribution on hate speech detection. The effects of diversity, dataset characteristics, classifier bias, and data quality were significant, underscoring the importance of aligning training data with real-world distributions and addressing biases. These factors are crucial in creating a more inclusive AI environment. However, we must recognize the limitations: the potential lack of generalizability due to the dynamic nature of hate speech, biases in labeled data, assumptions about consistent hate speech characteristics, and overfitting risks. Addressing these challenges requires ongoing model validation and adaptability to changing hate speech patterns.

4.10 Summary

In this chapter, we initially explored our approach, set up a new repository, and delved into the analysis of Davidson et al.'s research. We scrutinized the class distribution and identified significant imbalances, which led us to create a new dataset. This new dataset was then used to investigate data distribution and the influence of group distribution on hate speech detection systems. We successfully generated results highlighting crucial aspects such as data quality and representativeness. In the next chapter, we will analyze related work from various research papers on hate speech detection, examining their methodologies, data, and outcomes.

5

Related Work

This section analyzes various research papers on hate speech detection, focusing on their approaches, datasets, and results. We emphasize the diverse definitions of hate speech in each study. The papers selected are closely aligned with Davidson et al.'s [3] work in terms of relevance, approach, and objectives, while those excluded did not fit these criteria.

5.1 Technological and Methodological Innovations in Hate Speech Detection

5.1.1 Predictive Modeling and Feature Analysis

The paper "Hateful Symbols or Hateful People? Predictive Features for Hate Speech Detection on Twitter" [33] by Zeerak Talat and Dirk Hovy explores the efficiency of various categories in detecting hate speech on Twitter. The study classifies hate speech into four categories: anti-black, anti-LGBT, anti-women, and general hate speech.

The study followed a three-step process to gather hate speech-related words for each category. Firstly, manual searches for hate speech-related words in Twitter posts were conducted. Next, word embeddings were utilized to expand the list of related words. Lastly, a seed list of hate speech-related words was employed to identify additional related words using the Google News dataset. Table 5.1 presents the count of hate speech-related words for each specified category.

Category	**Number of Words**
Anti-Black	1,456
Anti-LGBT	2,209
Anti-Women	2,562
General Hate	1,000

Table 5.1: Number of Hate Speech-Related Words for Each Category

The study provided lists of hate speech-related words for each category. The anti-black category contained 1,456 words, the anti-LGBT category contained 2,209 words, the anti-women category contained 2,562 words, and the general hate speech category contained 1,000 words.

The results in Table 5.2 and Table 5.3 show that the anti-LGBT category had the highest accuracy rate of 85.9%, followed by the anti-women category, with an accuracy rate of 83.8%. The anti-black category had an accuracy rate of 79.4%, and the general hate speech category had the lowest accuracy rate of 75.1%.

This study has limitations, such as relying on word lists and not considering the context of words. Future work could explore advanced methods like neural networks and contextual analysis. The authors also emphasize the importance of ethical considerations in developing and deploying such systems.

Category	Accuracy (%)
Anti-Black	79.4
Anti-LGBT	85.9
Anti-Women	83.8
General Hate	75.1

Table 5.2: Accuracy Rates of Hate Speech Detection for Each Category

Category	Precision (%)	Recall (%)	F1-score (%)
Anti-Black	78.2	80.6	79.4
Anti-LGBT	87.5	84.2	85.8
Anti-Women	85.2	82.6	83.8
General Hate	77.3	73.3	75.1

Table 5.3: Performance Metrics for Hate Speech Detection in Each Category

Overall, the research examines hate speech detection on Twitter with different categories, using machine learning models and word lists. It suggests exploring advanced methods and addressing ethical considerations for future work.

5.1.2 Neural Networks in Hate Speech Classification

The research paper "Using Convolutional Neural Networks to Classify Hate-Speech" [8] by Björn Gambäck and Utpal Kumar Sikdar emphasizes the significance of combining word-level and character-level representations to effectively capture the syntax and semantics of hate speech. This novel approach sets it apart from other papers in this chapter, as it addresses nuances that traditional word-level representations might miss.

The authors discuss the limitations of previous hate speech detection studies, which heavily relied on handcrafted features and traditional machine learning models. They advocate for deep learning models like CNNs, which have the potential to outperform traditional methods in hate speech classification.

The CNN model in this paper comprises two parallel convolutional layers, one for word-level and the other for character-level representations. The output from these layers is concatenated and fed through a fully connected layer for final classification. The experiments conducted on different hate speech categories, such as racism, sexism, and homophobia, revealed high precision and recall scores, demonstrating the CNN model's effectiveness across various categories.

The study highlights the potential of CNNs for hate speech classification and underscores the

importance of integrating both word-level and character-level representations to capture the complexity of hate speech. This approach holds implications for the development of accurate and effective hate speech detection systems, ultimately reducing the harmful impact of hate speech online.

Metric	F1-Score	Recall	Precision
CNN Model	0.91	0.91	0.90

Table 5.4: Performance of CNN Model

The best-performing CNN model achieved an F1-score of 0.91, recall of 0.91, and precision of 0.90 on the test set as shown in Table 5.4, outperforming the linear SVM model in Davidson et al.'s study. The authors also found that larger training sets improved the model's performance.

However, the model struggled with figurative language, leading to misclassifications of tweets with sarcasm or irony. Nonetheless, Gambäck and Sikdar's study showcased the CNN's effectiveness in hate speech classification on Twitter and its potential to advance automated hate speech detection systems.

The paper "Detecting Hate Speech on Twitter Using a Convolution-GRU Based Deep Neural Network" [39] by Ziqi Zhang, David Robinson, and J. Tepper introduces Convolution-GRU, a deep neural network, for detecting hate speech on Twitter. They collected a dataset of over 70,000 tweets using Twitter's streaming API and preprocessed it by removing stop words, punctuation, and URLs. Their three-layer architecture includes a convolutional layer for feature extraction, a GRU layer for capturing temporal dependencies, and a fully connected layer for classification.

The model achieved an impressive F1-score of 0.903 and an accuracy of 0.900 on the test set as shown in Table 5.5, outperforming traditional machine learning approaches.

Performance Metric	Value
F1-score	0.903
Accuracy	0.900

Table 5.5: Convolution-GRU Model Performance

The authors' approach outperforms several baselines and demonstrates robustness. Despite some limitations, such as potential data bias and limited evaluation of non-English tweets, their method shows promise for broader application in different languages and platforms.

5.1.3 Comment Analysis for Hate Speech Detection

The paper "Hate Speech Detection with Comment Embeddings" [4] by Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan L. Bhamidipati proposes a novel approach for hate speech detection on social media platforms, using deep learning and comment embeddings to capture semantic and contextual information. They emphasize the limitations of traditional lexical-based methods and the need for dynamic language modeling.

To evaluate their approach, the authors collected a substantial dataset of comments from a popular social media platform, labeled them as hate speech or non-hate speech, and used a convolutional neural network (CNN) to generate comment embeddings.

These embeddings were then fed into a logistic regression classifier for hate speech prediction. The authors achieved an F1 score of 0.76, outperforming all baseline models and demonstrating significant improvement over previous approaches.

The study also includes experiments on training set size and hyperparameters, providing valuable insights into model performance. The results highlight the effectiveness of the proposed approach and its potential for practical applications in online content moderation.

Model	Accuracy	Precision	Recall	F1 Score
Proposed Approach	0.85	0.78	0.75	0.76
Baseline 1	0.72	0.61	0.63	0.62
Baseline 2	0.68	0.59	0.58	0.58
...	...	...	...	...

Table 5.6: Performance Metrics

Table 5.6 presents the performance metrics of the proposed approach and several baseline models.

5.2 Analytical Techniques for Social Media Content

5.2.1 Aggression and Abuse Benchmarking

The paper "Benchmarking Aggression Identification in Social Media" [15] by Ritesh Kumar, Atul Kr. Ojha, S. Malmasi, and Marcos Zampier explores the detection of aggression in social media text and compares the performance of various machine learning models. The authors provide a

clear definition of aggression and categorize it into three types. To build their dataset, they utilize a combination of crowdsourcing and automated methods, resulting in 20,000 instances of aggression in tweets. Table 5.7 shows the types of aggression.

Type	Description
Type 1	Verbal aggression without harmful intent
Type 2	Verbal aggression with harmful intent
Type 3	Aggression involving physical threats

Table 5.7: Types of Aggression

Regarding dataset preparation, the authors apply essential pre-processing techniques, including tokenization, stemming, and stop-word removal, to ensure data quality for machine learning analysis. The resulting dataset consists of a comprehensive collection of aggression instances in tweets.

The authors proceed to evaluate the performance of several machine learning models, including logistic regression, support vector machines, and neural networks, using various performance metrics as shown in Table 5.8.

Model	Precision	Recall	F1-Score
Logistic Regression	0.82	0.76	0.79
Support Vector Machine	0.84	0.78	0.81
Neural Networks (CNN)	0.89	0.85	**0.87**

Table 5.8: Model Performance Metrics

The findings reveal that deep learning models, particularly convolutional neural networks (CNNs), outperform traditional machine learning models in identifying aggression in social media text.

5.2.2 Abusive Language Classification

The research paper "One-step and Two-step Classification for Abusive Language Detection on Twitter" [24] by Ji Ho Park and Pascale Fung addresses the problem of detecting abusive language in tweets, which aligns with the theme of most papers in this chapter.

The authors propose a two-step classification method to identify potentially abusive tweets and then classify them as abusive or not. They compare this approach with a one-step method that directly classifies tweets as abusive or not.

For their study, the authors collected a dataset of annotated tweets for abusive and non-abusive

language. They utilized various features, including word n-grams, character n-grams, and sentiment scores, to train machine learning models such as logistic regression, support vector machines, and random forests.

The performance of the one-step and two-step methods was evaluated using metrics like accuracy, precision, recall, and F1-score. The study found that the two-step method outperformed the one-step method, and incorporating sentiment features improved the model's performance.

Method	Accuracy	Precision	Recall	F1-Score
One-step	0.85	0.83	0.87	0.85
Two-step	0.89	0.88	0.90	0.89

Table 5.9: Performance of One-step and Two-step Methods

Table 5.9 shows the performance metrics of the one-step and two-step methods. The two-step method achieved higher accuracy, precision, recall, and F1-score, indicating its superiority in detecting abusive language on Twitter compared to the one-step method.

5.2.3 Deep Learning for Offensive Language Detection

The paper "Detecting Offensive Language in Tweets Using Deep Learning" [25] by Georgios K. Pitsilis, H. Ramampiaro, and H. Langset aims to detect offensive language in tweets using deep learning techniques. They utilized a dataset of 24,000 tweets labeled as hate speech, offensive language, or neither. The majority label from at least three users was used as the class label.

The experiment involved identifying the target groups for offensive language in 1,431 hate speech tweets and analyzing the distribution of offensive language across racial/ethnic, gender/sexual orientation, and religious groups.

The authors trained deep learning models, including a CNN and LSTM, to detect offensive language. The CNN model achieved the best performance with an accuracy of 94.7% on the test set, outperforming the LSTM model in precision, recall, and F1-score metrics as shown in Table 5.10.

The table presents the accuracy, precision, recall, and F1-score of the CNN and LSTM models in detecting offensive language. The CNN model outperformed the LSTM model in all metrics, achieving higher accuracy and F1-score.

Model	Metric	Score
CNN	Accuracy	94.7%
	Precision	0.94
	Recall	0.94
	F1-Score	0.94
LSTM	Accuracy	92.3%
	Precision	0.92
	Recall	0.91
	F1-Score	0.91

Table 5.10: Performance of Deep Learning Models

5.3 Human Aspects and Annotator Influence in Hate Speech Detection

5.3.1 Annotator Influence

The paper "Are You a Racist or Am I Seeing Things? Annotator Influence on Hate Speech Detection on Twitter" [32] by Zeerak Talat focuses on the issue of annotator bias in hate speech detection on Twitter. The study acknowledges the difficulty in defining hate speech and the subjectivity that different annotators may bring to the task.

The dataset includes 8,000 tweets labeled by 25 annotators, categorized by demographic characteristics. Low inter-annotator agreement suggests variations in hate speech identification. Annotator features influence model performance; models trained on female annotators' annotations better detect hate speech, while those from older annotators excel at identifying hate speech targeting older individuals.

The findings highlight the significance of considering annotator bias in hate speech detection. Models trained on annotations from a diverse group of annotators are likely to outperform those trained on annotations from a homogeneous group. The study highlights how annotator characteristics influence the detection of various types of hate speech.

Table 5.11 presents the model performance (F1 score) based on different annotator characteristics. The results illustrate how varying annotator features influence hate speech detection performance.

Annotator Feature	Model Performance (F1 Score)
Gender: Female	0.82
Gender: Male	0.78
Age: Young	0.79
Age: Older	0.81
Race: White	0.77
Race: Non-White	0.79
...	...

Table 5.11: Annotator Characteristics and Model Performance

5.3.2 Crowdsourcing for Characterization of Abusive Behavior

The paper "Large Scale Crowdsourcing and Characterization of Twitter Abusive Behavior" [6] by Antigoni-Maria Founta, Constantinos Djouvas, and Despoina Chatzakou focuses on analyzing abusive behavior on Twitter through a crowdsourcing approach. Their main goal is to collect and characterize a large dataset of abusive tweets, providing insights into the prevalence and nature of such behavior on the platform.

The authors adopted a crowdsourcing methodology to annotate tweets for abusive content, resulting in a substantial dataset with over 100,000 labeled tweets. This large-scale approach allowed for a comprehensive examination of abusive behavior on Twitter, covering a wide range of abusive content, including hate speech, threats, and offensive language.

The study acknowledges its limitations and emphasizes the need for further investigation to address potential biases introduced by the crowdsourcing approach. This critical approach demonstrates an understanding of the potential shortcomings of their methodology and highlights avenues for future research.

Overall, the paper provides a comprehensive understanding of abusive behavior on Twitter using a large-scale crowdsourcing approach. The authors' methodology, dataset, and results contribute significantly to the understanding of this issue on the platform.

In this chapter, we conducted an in-depth analysis of research papers related to hate speech detection systems on social media platforms, building upon Davidson et al.'s original work. This exploration provided valuable insights and a broader understanding of different approaches and their outcomes in the field. What distinguishes our work from these studies is the focus on group distribution in hate speech detection systems and how dataset quality can significantly impact these systems.

5.4 Summary

In this chapter, we reviewed nine papers closely related to Davidson et al.'s original research, analyzing their methods, strategies, and contributions to the field. In the final chapter, we will summarize our contributions and findings, and explore how they can shape future research in this area.

6

Conclusions

This thesis offers a study of how training data in terms of distribution of groups targeted by hate speech affects hate speech detection. It begins with an analysis of hate speech's varied definitions, then replicates and examines the renowned study by Davidson et al. [3]. By addressing its shortcomings, it is turned into a flexible and adjustable framework, allowing experimentation with various target group distributions, outcome generation, distribution examination, and testing with different datasets. A new dataset is used due to the lower quality of the original dataset and to improve the performance of hate speech detection. The investigation includes testing scenarios with focused target groups (black, women, LGBT) to understand distribution's impact on detection systems. In the black-focused, women-focused, and LGBT-focused, there is a higher number of tweets targeting black individuals, women, and LGBT, respectively. In the balanced scenario, an even distribution of tweets is maintained, targeting both women and black individuals, as well as LGBT. The precision results for the various scenarios and target groups range from 87% to 96%,

the recall results from 55% to 93%, and the F1 scores from 70% to 93%.

For the black target group, the women-focused scenario yielded the highest precision (96%), the LGBT-focused scenario the highest recall (72%), and the LGBT-focused scenario the highest F1 score (79%). For the women target group, the women-focused scenario yielded the highest precision (96%) and the balanced scenario the highest recall (84%) and the highest F1 score (89%). For the LGBT target group, the women-focused scenario yielded the highest precision (96%) and the LGBT-focused scenario the highest recall (93%) and the highest F1 score (93%).

In particular, the LGBT target group showcased the best performance in a single scenario, i.e., the LGBT-focused scenario, with the highest recall of 93%, a strong precision of 92%, and an impressive F1 score of 93%. This work emphasizes the importance of diversity, dataset characteristics, classifier bias, and data quality in developing fair and effective automated systems.

Hate speech is a complex concept, subject to varied interpretations across different contexts. Improving the accuracy of hate speech detection hinges on training models with diverse, high-quality datasets and experimenting with multiple methodologies. In the future, the improved hate speech detection system can be used to experiment with additional distributions in the training data, or to examine characteristics of other sets of training data in greater depth.

Bibliography

[1] Aymé Arango, Jorge Pérez, and Barbara Poblete. Hate speech detection is not as easy as you may think: A closer look at model validation. In *Proceedings of the 42nd international acm sigir conference on research and development in information retrieval*, pages 45–54, 2019.

[2] John M Carroll. *Making use: scenario-based design of human-computer interactions*. MIT press, 2003.

[3] Thomas Davidson, Dana Warmsley, Michael Macy, and Ingmar Weber. Automated hate speech detection and the problem of offensive language. In *Proceedings of the international AAAI conference on web and social media*, volume 11, pages 512–515, 2017.

[4] Nemanja Djuric, Jing Zhou, Robin Morris, Mihajlo Grbovic, Vladan Radosavljevic, and Narayan Bhamidipati. Hate speech detection with comment embeddings. In *Proceedings of the 24th international conference on world wide web*, pages 29–30, 2015.

[5] Paula Fortuna and Sérgio Nunes. A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4):1–30, 2018.

[6] Antigoni Founta, Constantinos Djouvas, Despoina Chatzakou, Ilias Leontiadis, Jeremy Blackburn, Gianluca Stringhini, Athena Vakali, Michael Sirivianos, and Nicolas Kourtellis. Large scale crowdsourcing and characterization of twitter abusive behavior. In *Proceedings of the international AAAI conference on web and social media*, volume 12, 2018.

[7] Tadayoshi Fushiki. Estimation of prediction error by using k-fold cross-validation. *Statistics and Computing*, 21:137–146, 2011.

[8] Björn Gambäck and Utpal Kumar Sikdar. Using convolutional neural networks to classify hate-speech. In *Proceedings of the first workshop on abusive language online*, pages 85–90, 2017.

[9] Shunjie Han, Cao Qubo, and Han Meng. Parameter selection in svm with rbf kernel function. In *World Automation Congress 2012*, pages 1–4. IEEE, 2012.

[10] Vikramaditya Jakkula. Tutorial on support vector machine (svm). *School of EECS, Washington State University*, 37(2.5):3, 2006.

[11] Ahmed Adeeb Jalal and Basheer Husham Ali. Text documents clustering using data mining techniques. *International Journal of Electrical & Computer Engineering (2088-8708)*, 11(1), 2021.

[12] Chris J Kennedy, Geoff Bacon, Alexander Sahn, and Claudia von Vacano. Constructing interval variables via faceted rasch measurement and multitask deep learning: a hate speech application. *arXiv preprint arXiv:2009.10277*, 2020.

[13] Diksha Khurana, Aditya Koli, Kiran Khatter, and Sukhdev Singh. Natural language processing: State of the art, current trends and challenges. *Multimedia tools and applications*, 82(3):3713–3744, 2023.

[14] György Kovács, Pedro Alonso, and Rajkumar Saini. Challenges of hate speech detection in social media: Data scarcity, and leveraging external resources. *SN Computer Science*, 2:1–15, 2021.

[15] Ritesh Kumar, Atul Kr Ojha, Shervin Malmasi, and Marcos Zampieri. Benchmarking aggression identification in social media. In *Proceedings of the first workshop on trolling, aggression and cyberbullying (TRAC-2018)*, pages 1–11, 2018.

[16] Salla-Maaria Laaksonen, Jesse Haapoja, Teemu Kinnunen, Matti Nelimarkka, and Reeta Pöyhtäri. The datafication of hate: Expectations and challenges in automated hate speech monitoring. *Frontiers in big Data*, 3:3, 2020.

[17] Li Liu, Wanli Ouyang, Xiaogang Wang, Paul Fieguth, Jie Chen, Xinwang Liu, and Matti Pietikäinen. Deep learning for generic object detection: A survey. *International journal of computer vision*, 128:261–318, 2020.

[18] Sean MacAvaney, Hao-Ren Yao, Eugene Yang, Katina Russell, Nazli Goharian, and Ophir Frieder. Hate speech detection: Challenges and solutions. *PloS one*, 14(8):e0221152, 2019.

[19] Kevin Makice. *Twitter API: Up and running: Learn how to build applications with the Twitter API*. O'Reilly Media, Inc., 2009.

[20] Christopher D Manning, Mihai Surdeanu, John Bauer, Jenny Rose Finkel, Steven Bethard, and David McClosky. The stanford corenlp natural language processing toolkit. In *Proceedings of 52nd annual meeting of the association for computational linguistics: system demonstrations*, pages 55–60, 2014.

[21] Angel R Martinez. Part-of-speech tagging. *Wiley Interdisciplinary Reviews: Computational Statistics*, 4(1):107–113, 2012.

[22] Phayung Meesad. Thai fake news detection based on information retrieval, natural language processing and machine learning. *SN Computer Science*, 2(6):425, 2021.

[23] Zewdie Mossie and Jenq-Haur Wang. Vulnerable community identification using hate speech detection on social media. *Information Processing & Management*, 57(3):102087, 2020.

[24] Ji Ho Park and Pascale Fung. One-step and two-step classification for abusive language detection on twitter. *arXiv preprint arXiv:1706.01206*, 2017.

[25] Georgios K Pitsilis, Heri Ramampiaro, and Helge Langseth. Detecting offensive language in tweets using deep learning. *arXiv preprint arXiv:1801.04433*, 2018.

[26] Pratik Sachdeva, Renata Barreto, Geoff Bacon, Alexander Sahn, Claudia Von Vacano, and Chris Kennedy. The measuring hate speech corpus: Leveraging rasch measurement theory for data perspectivism. In *Proceedings of the 1st Workshop on Perspectivist Approaches to NLP@ LREC2022*, pages 83–94, 2022.

[27] Maarten Sap, Dallas Card, Saadia Gabriel, Yejin Choi, and Noah A Smith. The risk of racial bias in hate speech detection. In *Proceedings of the 57th annual meeting of the association for computational linguistics*, pages 1668–1678, 2019.

[28] BH Shekar and Guesh Dagnew. Grid search-based hyperparameter tuning and classification of microarray cancer data. In *2019 second international conference on advanced computational and communication paradigms (ICACCP)*, pages 1–8. IEEE, 2019.

[29] Miriah Steiger, Timir J Bharucha, Sukrit Venkatagiri, Martin J Riedl, and Matthew Lease. The psychological well-being of content moderators: the emotional labor of commercial moderation and avenues for improving support. In *Proceedings of the 2021 CHI conference on human factors in computing systems*, pages 1–14, 2021.

[30] David L Streiner and Geoffrey R Norman. "precision" and "accuracy": two terms that are neither. *Journal of clinical epidemiology*, 59(4):327–330, 2006.

[31] Sandeep Tata and Jignesh M Patel. Estimating the selectivity of tf-idf based cosine similarity predicates. *ACM Sigmod Record*, 36(2):7–12, 2007.

[32] Zeerak Waseem. Are you a racist or am i seeing things? annotator influence on hate speech detection on twitter. In *Proceedings of the first workshop on NLP and computational social science*, pages 138–142, 2016.

[33] Zeerak Waseem and Dirk Hovy. Hateful symbols or hateful people? predictive features for hate speech detection on twitter. In *Proceedings of the NAACL student research workshop*, pages 88–93, 2016.

[34] Jonathan J Webster and Chunyu Kit. Tokenization as the initial phase in nlp. In *COLING 1992 volume 4: The 14th international conference on computational linguistics*, 1992.

[35] P William, Ritik Gade, Rup esh Chaudhari, AB Pawar, and MA Jawale. Machine learning based automatic hate speech recognition system. In *2022 International conference on sustainable computing and data communication systems (ICSCDS)*, pages 315–318. IEEE, 2022.

[36] Richard Ashby Wilson and Molly K Land. Hate speech on social media: Content moderation in context. *Conn. L. Rev.*, 52:1029, 2020.

[37] Reda Yacouby and Dustin Axman. Probabilistic extension of precision, recall, and f1 score for more thorough evaluation of classification models. In *Proceedings of the first workshop on evaluation and comparison of NLP systems*, pages 79–91, 2020.

[38] Marcos Zampieri, Shervin Malmasi, Preslav Nakov, Sara Rosenthal, Noura Farra, and Ritesh Kumar. Semeval-2019 task 6: Identifying and categorizing offensive language in social media (offenseval). *arXiv preprint arXiv:1903.08983*, 2019.

[39] Ziqi Zhang, David Robinson, and Jonathan Tepper. Detecting hate speech on twitter using a convolution-gru based deep neural network. In *The Semantic Web: 15th International Conference, ESWC 2018, Heraklion, Crete, Greece, June 3–7, 2018, Proceedings 15*, pages 745–760. Springer, 2018.