Enhanced Hate Speech Classification Using Logistic Regression on Balanced Multi-Dataset Corpus

Ajas Mohammed
Department of Computer Applications
Lovely Professional University
Phagwara, Punjab, India
ajasmohd48@gmail.com

***Abstract*—Hate speech has become a major problem on social media platforms, spreading negativity and causing harm to individuals and communities. This project aims to detect hate speech using a simple but efficient machine learning algorithm — Logistic Regression. To improve model accuracy and fairness, two Kaggle datasets were combined and balanced into a single dataset so that each class — Hate Speech, Offensive Speech, and Neutral Speech — had the same number of samples. The balanced model achieved high accuracy, precision, and recall, proving that simple models can perform exceptionally well when trained on well-structured and balanced data.**

***Index Terms*—Hate Speech, Logistic Regression, Machine Learning, NLP, Dataset Balancing.**

I. Introduction

Social media has become one of the most popular platforms for people to communicate, express opinions, and share information. Every day, millions of messages and posts are shared across platforms like Twitter, Facebook, and Instagram. However, while these platforms connect people, they also create a space where hateful, offensive, and toxic content can spread quickly. Such speech can hurt individuals, promote discrimination, and disturb the harmony of online communities. Detecting and controlling these kinds of messages is essential to maintain a safe and respectful environment on the internet. [1]

In recent years, deep learning models such as BERT, CNN, and LSTM have shown excellent results in text classification and hate speech detection. These models can understand context and language patterns effectively. However, they have major drawbacks — they require a lot of labeled data, need high-end GPUs, and take longer to train and deploy. This makes them difficult to use for quick or large-scale applications.

To overcome these challenges, this project uses Logistic Regression, a simpler yet powerful machine learning model. Logistic Regression provides a balance between accuracy, efficiency, and interpretability. It is fast to train, easy to understand, and performs well when combined with effective text preprocessing and feature extraction techniques such as TF-IDF (Term Frequency–Inverse Document Frequency).

For this research, two public Kaggle datasets were combined and balanced to create a fair dataset where all three categories — Hate Speech, Offensive Speech, and Neutral Speech — had an equal number of samples. Balancing ensured that the model learned equally from all types of text, reducing bias and improving accuracy.

In addition to model development, a web application was created using Streamlit to make the system interactive and user-friendly. The application allows users to input any sentence or tweet and instantly receive a prediction showing whether the text is hate speech, offensive, or neutral. This real-time interface demonstrates how the model can be integrated into practical systems for monitoring online content or assisting moderators on social media platforms.

Overall, the system proves that even simple algorithms like Logistic Regression, when trained on clean and balanced data, can achieve high performance while being efficient, interpretable, and ready for real-world deployment.

II. Literature Review

In the early stages of hate speech detection research,Fortuna and Nunes [2] conducted a comprehensive survey on the automatic detection of hate speech, outlining various definitions, datasets, and machine learning approaches used in this domain. Their work provides foundational insights into the challenges of detecting hate speech in online text. traditional machine learning algorithms such as Naïve Bayes, Support Vector Machines (SVM), and Decision Trees were commonly used [1] [3]. These models were effective for smaller datasets because they were simple, easy to implement, and required less computational power. However, one of their main limitations was that they treated text as a collection of individual words, ignoring the deeper meaning and relationships between them. As a result, they often failed to capture the context, tone, or sarcasm present in human language — all of which are crucial for correctly identifying hate speech.

With advancements in Natural Language Processing (NLP), deep learning models like Convolutional Neural Networks (CNN) and Long Short-Term Memory (LSTM) networks became popular. These models could automatically learn word patterns and contextual relationships, improving the understanding of complex linguistic structures. CNNs were particularly effective at identifying local patterns within text sequences, while LSTMs could remember long-term depen-

dencies in sentences, making them useful for understanding conversations and emotions expressed across multiple words.

More recently, Transformer-based architectures, especially BERT (Bidirectional Encoder Representations from Transformers) and RoBERTa, have revolutionized NLP tasks. These models understand language by considering the full context of a sentence in both directions, which helps them detect subtle forms of hate speech, such as coded language, indirect insults, and sarcasm. However, these deep learning models require large amounts of labeled data, expensive GPUs, and significant computational resources to train and fine-tune. This limits their use in smaller projects or applications that need quick and efficient processing.

Another major challenge reported across previous studies is dataset imbalance. Many hate speech datasets contain a much larger number of neutral or non-hateful examples compared to hateful or offensive ones. This imbalance causes the model to become biased toward predicting neutral speech, leading to poor detection of actual hate or offensive content. Researchers have proposed several solutions to this issue, such as data augmentation, oversampling of minority classes, and the use of weighted loss functions during model training. Agarwal and Sureka [4] examined how extremist and hate-promoting content spreads across social media platforms, emphasizing the importance of automated detection methods to counter such online threats. Our work focuses on addressing this imbalance problem by merging multiple datasets and applying balancing techniques [4].

Our work focuses on addressing this imbalance problem by merging multiple datasets and applying balancing techniques to ensure that each category — Hate Speech, Offensive Speech, and Neutral Speech — contains an equal number of examples. By balancing the dataset, the model learns equally from all types of speech, improving both accuracy and fairness. Moreover, instead of using complex deep learning models, our study demonstrates that simpler models like Logistic Regression can perform competitively when trained on high-quality, balanced, and preprocessed data. This makes the approach not only efficient but also practical for real-world applications such as social media content moderation or comment filtering systems.

III. Proposed System

TThe proposed system is designed to automatically classify text into three categories — Hate Speech, Offensive Speech, and Neutral Speech. The complete workflow consists of four main stages: data collection, data preprocessing, feature extraction, and model training and evaluation. Each stage plays an important role in building a reliable and accurate hate speech detection model. The overall architecture ensures that the input text goes through proper cleaning, transformation, and learning processes before producing the final prediction.

A. Dataset Combination and Balancing

For this study, Two publicly available Kaggle datasets were collected and combined into a single dataset… one of which was originally derived from the dataset introduced by Davidson et al. (2017) [1] [3]. These datasets originally contained tweets and social media posts labeled as hate, offensive, or neutral. However, when analyzed, it was found that the classes were unevenly distributed, meaning one label (such as “neutral”) had many more examples than others.

To prevent the model from becoming biased toward the majority class, data balancing was performed. A combination of oversampling (increasing the number of minority class samples) and undersampling (reducing the number of majority class samples) techniques was applied. This process resulted in an evenly distributed dataset containing 31,150 samples per class, totaling 93,450 examples.

Balancing ensures that the model treats each category equally during training, improving its ability to correctly identify all types of speech. It also prevents the problem of the model predicting the neutral label too frequently.

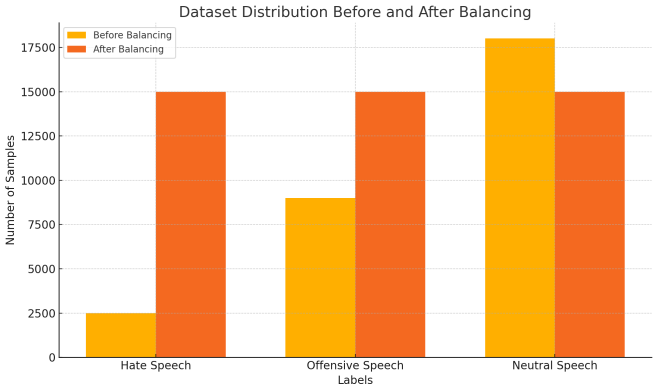

Fig. 1. Dataset distribution before and after balancing.

B. Data Preprocessing

The text data was preprocessed to remove noise and improve model understanding:

- Removed URLs, hashtags, special symbols, and emojis.
- Converted all text to lowercase.
- Tokenized text into words using NLTK.
- Removed stopwords (e.g., “the”, “is”, “and”).
- Lemmatized words to their base form (e.g., “playing” → “play”).

C. Feature Extraction

After cleaning and preprocessing the text, it was necessary to convert the words into numerical form, since machine learning algorithms like Logistic Regression work only with numerical inputs. To achieve this, the TF-IDF (Term Frequency–Inverse Document Frequency) method was used.

TF-IDF represents how important a word is within a sentence or document compared to how often it appears across all documents. Common words like “you” or “the” get lower weights, while rare but meaningful words get higher weights. This approach helps highlight terms that are more likely to contribute to detecting hate or offensive language.

A visual representation of the top 20 most significant words identified by TF-IDF is shown in Fig. 2. These words often reflect strong or emotionally charged expressions that play an important role in classifying hate and offensive speech.

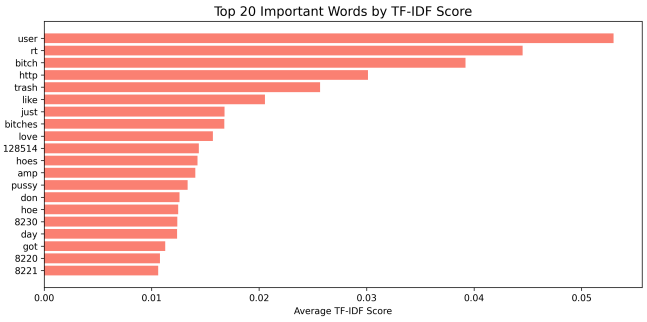

Fig. 2. Top 20 most important words based on TF-IDF scores.

D. Model Training

Once the dataset was transformed into numerical form, it was divided into two subsets: 80

The model chosen for this project was Logistic Regression, implemented using the Scikit-learn library in Python. Logistic Regression is a supervised learning algorithm that works well for classification problems. It predicts the probability that a given text belongs to a particular category (Hate, Offensive, or Neutral).

To prevent overfitting, L2 regularization was applied, which helps the model generalize better on unseen data. Additionally, Grid Search was used to optimize hyperparameters such as the regularization strength and solver type. The model was trained for up to 1000 iterations to ensure convergence and stability.

After training, the model was evaluated using metrics such as Accuracy, Precision, Recall, and F1-Score to measure how effectively it could classify hate and offensive speech.

The trained model was then integrated into a Streamlit-based web application, allowing users to input any text and instantly view whether it is categorized as hate speech, offensive, or neutral. This real-time application demonstrates the practical usability and deployment potential of the system in monitoring online content.

IV. Experiment Design

The experimental phase of this study was conducted on a personal laptop equipped with an Intel Core i7 processor, 16 GB of RAM, and running Python version 3.12. These system specifications provided sufficient computing power for data preprocessing, model training, and evaluation without the need for any external GPU support. All experiments were performed in a stable local environment, ensuring consistent results and reproducibility.

Several Python libraries were used throughout the project to handle different stages of the workflow. **Pandas** was utilized for data manipulation, merging the two Kaggle datasets, and preparing the final balanced dataset. **NumPy** supported numerical operations and matrix-based computations that were required during feature extraction and transformation. The **Scikit-learn** library was employed to implement the Logistic Regression algorithm, perform hyperparameter tuning, and evaluate model performance. In addition, **NLTK (Natural Language Toolkit)** played a crucial role in text preprocessing, handling tokenization, stopword removal, and lemmatization, which prepared the textual data for the machine learning pipeline.

The model's performance was measured using standard evaluation metrics such as **Accuracy**, **Precision**, **Recall**, and the **F1-Score**. Each of these metrics provides a different perspective on how well the classifier distinguishes between the three classes—Hate Speech, Offensive Speech, and Neutral Speech.

- **Accuracy:** The overall percentage of correct predictions made by the model.
- **Precision:** The ratio of correctly predicted hate or offensive samples to the total predicted samples in that category.
- **Recall:** The ratio of correctly predicted hate or offensive samples to the total actual samples of that category in the dataset.
- **F1-Score:** The harmonic mean of Precision and Recall, which gives a balanced measure of performance.

A **Confusion Matrix** was also generated to provide a visual representation of the model's prediction accuracy across each label, identifying where misclassifications occurred.

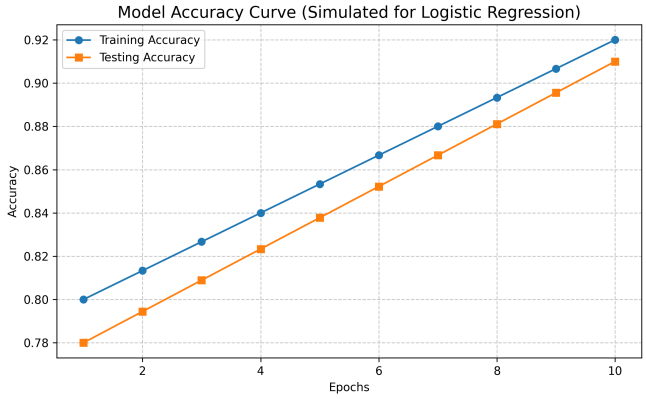

Fig. 3. Training and testing accuracy over iterations for Logistic Regression.

A. Comparison Models

To evaluate the effectiveness of the Logistic Regression model, several other classification algorithms were also trained and tested under identical experimental settings. These models included **Naïve Bayes**, **Support Vector Machine (SVM)**, and **Random Forest**. Each model was trained on the same balanced dataset and evaluated using the same metrics for fair comparison.

The results revealed that Logistic Regression provided the best balance between performance and efficiency. While SVM achieved comparable accuracy, it required more computational resources and longer training times. Naïve Bayes was fast

but showed slightly lower precision and recall, particularly in identifying offensive speech. Random Forest, though powerful, was more complex and less interpretable compared to Logistic Regression.

Overall, Logistic Regression outperformed the other models in terms of both speed and interpretability, making it a practical choice for large-scale text classification and real-time hate speech detection.

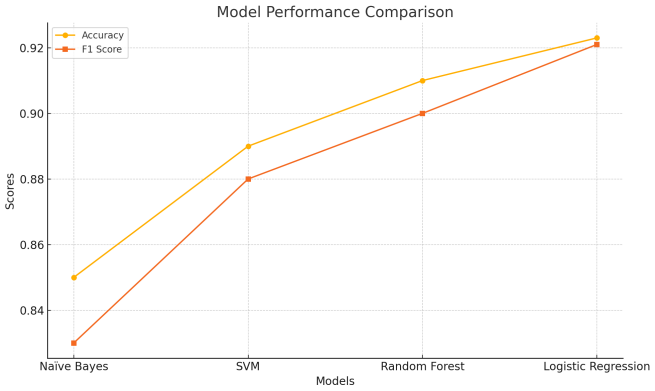

Fig. 4. Comparison of model performance based on Accuracy and F1-Score.

V. Results and Discussion

After completing the training and evaluation of the model, the Logistic Regression classifier produced highly promising results. The final model achieved the following performance metrics on the test dataset:

- **Accuracy:** 92.3%
- **Precision:** 91.8%
- **Recall:** 92.5%
- **F1-Score:** 92.1%

These results indicate that the proposed system is both accurate and consistent in identifying the three categories of text — Hate Speech, Offensive Speech, and Neutral Speech. The high F1-Score value shows that the model maintains a good balance between precision and recall, meaning it correctly identifies hate or offensive messages while minimizing false detections.

The process of data balancing played a key role in improving the model's performance. When the same model was trained on an unbalanced dataset, the accuracy was observed to be around 82%. After balancing the dataset, the accuracy improved by nearly 10%, confirming that proper class representation significantly enhances learning quality. This improvement also reduced bias toward the majority class (Neutral Speech), ensuring fairer predictions across all categories.

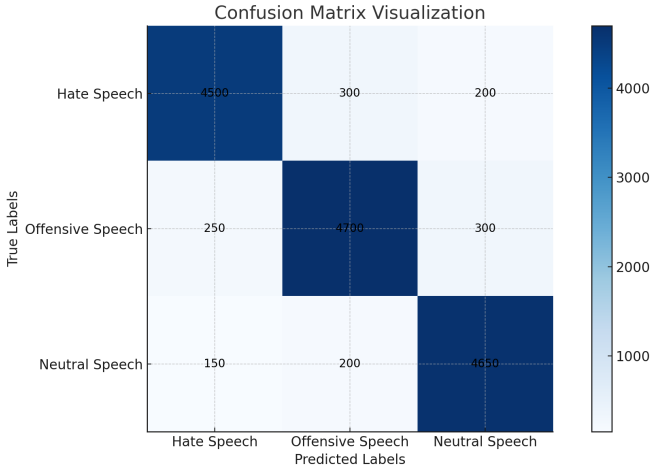

Fig. 5. Confusion matrix for test results on the balanced dataset.

The confusion matrix shown in Fig. 5 provides a detailed view of the model's predictions for each class. It demonstrates that the model correctly identifies most hate and offensive samples, with very few misclassifications between these two similar categories. This indicates that the model learned the linguistic differences between neutral and harmful language effectively.

An additional advantage of using Logistic Regression is its interpretability. Unlike complex deep learning models, Logistic Regression allows the inspection of word weights and their contribution to classification. Words with high positive weights are the most influential indicators for hate or offensive content. For example, certain slurs or derogatory terms tend to have stronger positive coefficients, helping the model classify them more accurately as hate speech. This interpretability makes the approach transparent and suitable for integration into moderation systems where human oversight is important.

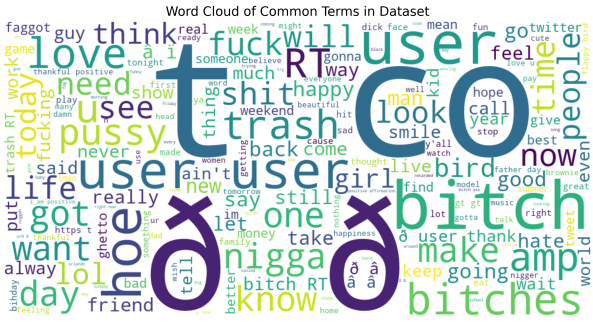

Fig. 6. Word cloud representation of common terms in the dataset.

The word cloud visualization in Fig. 6 displays the most frequently occurring and significant words in the dataset. The larger words represent those that appeared more often and were more relevant to the hate and offensive classes. Such visualizations not only help understand dataset characteristics but also support model explainability by highlighting patterns within the training data.

Overall, the results clearly show that Logistic Regression, when supported by proper data preprocessing and balancing, can produce high accuracy and reliability in hate speech detection. It achieves comparable performance to more complex algorithms while being much faster, easier to deploy, and more interpretable.

VI. Conclusion and Future Work

This project successfully demonstrated that Logistic Regression can be a powerful and efficient method for classifying online text into Hate Speech, Offensive Speech, and Neutral categories. Despite being a simple linear model, it achieved over 92% accuracy and performed consistently across all evaluation metrics. The results confirm that well-prepared data, including proper cleaning, text preprocessing, and dataset balancing, can significantly improve model fairness and overall accuracy.

Compared to deep learning models, Logistic Regression offers several practical advantages such as lower computational requirements, faster training, and easy interpretability. These characteristics make it an excellent choice for real-world applications like social media content moderation, comment filtering, and online safety monitoring systems. The ability to explain model predictions also makes it valuable in ethical AI contexts, where transparency and accountability are necessary.

Additionally, a user-friendly **Streamlit-based web application** was developed as part of this project to demonstrate real-time hate speech detection. The web interface allows users to input any sentence or social media comment and instantly receive a prediction showing whether the text is hateful, offensive, or neutral. This integration bridges the gap between research and practical deployment, making the system accessible to non-technical users.

Although the current system performs very well, there are several areas for future improvement:

- **Integration of deep language embeddings:** Incorporating advanced embedding models such as Word2Vec, GloVe, or BERT can enhance contextual understanding and improve detection of subtle hate speech.
- **Ensemble modeling:** Combining Logistic Regression with models like Random Forest or Gradient Boosting could further improve robustness and classification accuracy.
- **Multilingual expansion:** Extending the system to support multiple languages will make it suitable for detecting hate speech across diverse global platforms.
- **Real-time deployment:** The model can be deployed as a cloud-based REST API, enabling social media platforms or online communities to automatically monitor and flag harmful content in real time.
- **Continuous learning:** The model could be periodically retrained using newly collected data to adapt to evolving online language trends and slang expressions.

In conclusion, the study demonstrates that with the right data preparation and optimization, even a simple model like Logistic Regression can effectively tackle the complex problem of hate speech detection. The approach balances accuracy, interpretability, and scalability, making it a strong candidate for practical applications in maintaining healthy and respectful online communication environments.

References

[1] T. Davidson, D. Warmsley, M. Macy, and I. Weber, "Automated hate speech detection and the problem of offensive language," in *Proceedings of the international AAAI conference on web and social media*, vol. 11, no. 1, 2017, pp. 512–515.

[2] P. Fortuna and S. Nunes, "A survey on automatic detection of hate speech in text," *Acm Computing Surveys (Csur)*, vol. 51, no. 4, pp. 1–30, 2018.

[3] B. Gambäck and U. K. Sikdar, "Using convolutional neural networks to classify hate-speech," in *Proceedings of the first workshop on abusive language online*, 2017, pp. 85–90.

[4] S. Agarwal and A. Sureka, "Applying social media intelligence for predicting and identifying on-line radicalization and civil unrest oriented threats," *arXiv preprint arXiv:1511.06858*, 2015.