# Hate Speech and Offensive Language Detection using LSTM

Raiyanul Islam Siam
24141261
B.Sc in Computer Science, BRAC University, Badda
raiyanul.islam.siam@g.bracu.ac.bd

## Abstract

This paper describes a project that uses a deep learning approach to detect hate speech and offensive language in tweets with a model based on **LSTM** (Long Short-Term Memory). The researchers used the hate_speech_offensive dataset, which they expanded through **data augmentation** using parallelized back-translation. The model was evaluated using metrics like Accuracy, F1-score, Precision, and Recall. It showed that the model achieved results around 90% validation accuracy and a balanced F1-score. Explainable methods and a confusion matrix were also used to interpret the results and ensure the model's reliability.

**Keywords:** Hate Speech, Offensive Language, LSTM, Data Augmentation, Explainable AI, Text Classification

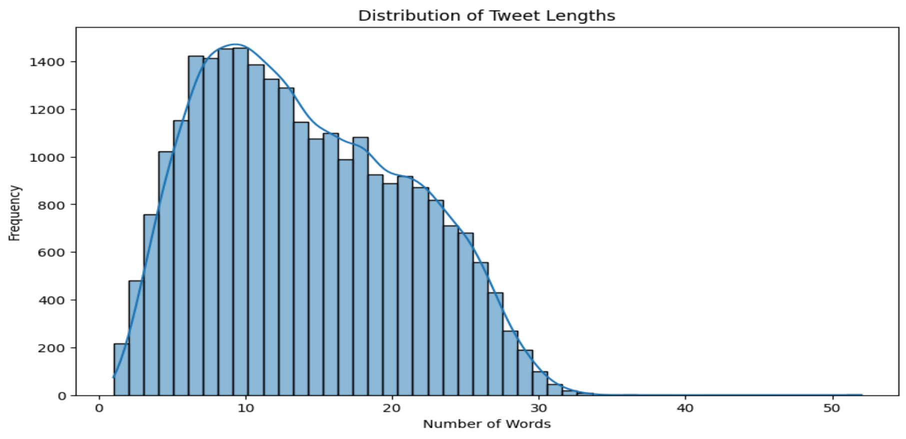
## 1 Introduction

Social media has become a powerful communication tool, but it's also a place for harmful conversations. **Hate speech** attacks and harms humans depending on their race, religion, gender, or identity. It has serious societal consequences (e.g "All Muslims are ter*orists."). **Offensive language**, on the other hand, is general abuse or profanity that creates a toxic environment (e.g "You're an idi*t, shut up."). Detecting and moderating this content is crucial for creating safer online communities. Traditional systems have a hard time with the complexity of natural language. This is why deep learning models, particularly recurrent models like LSTM, are useful because they can capture the sequence and context of language. This project aimed to implement and improve an LSTM-based text classifier for identifying hate speech and offensive language in tweets

## 2 Methods

### 2.1 Dataset

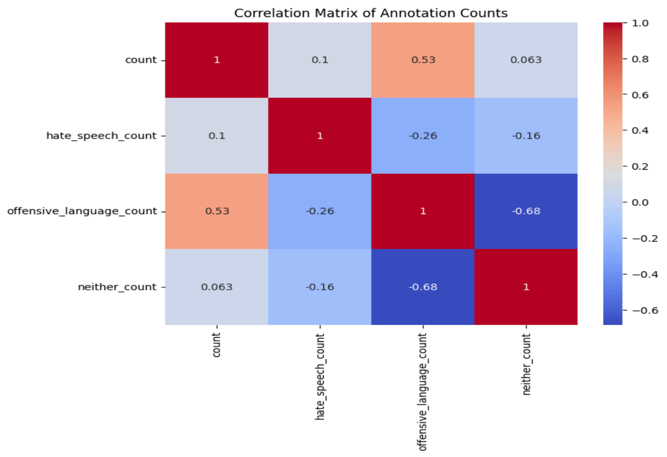
The project used the **hate_speech_offensive** dataset, which consists of English tweets. Each tweet was labeled by multiple reviewers as either hate speech, offensive language, or neither. The dataset was **imbalanced**, with fewer examples of hate speech compared to offensive or neutral tweets



## 2.2 Preprocessing

The dataset was prepared for training by:

- Removing usernames, URLs, HTML, and special characters.
- Applying tokenization and removing common words (stopwords).
- Converting words into numerical indices for the model to understand.



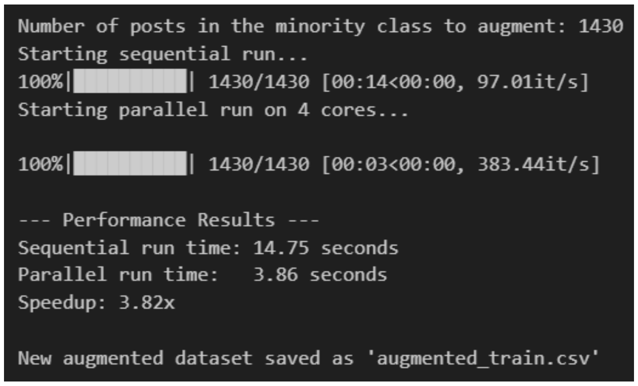
| | count | hate_speech_count | offensive_language_count | neither_count |
|---|---|---|---|---|
| count | 1 | 0.1 | 0.53 | 0.063 |
| hate_speech_count | 0.1 | 1 | -0.26 | -0.16 |
| offensive_language_count | 0.53 | -0.26 | 1 | -0.68 |
| neither_count | 0.063 | -0.16 | -0.68 | 1 |

## 2.3 Parallel Data Augmentation

A key part of the project was using **back-translation** for data augmentation using ***googletrans***. This process translates tweets into another language and then back to English to generate new, but similar, sentences. Because this is computationally expensive, the researchers used Python's ***multiprocessing*** module to distribute the tasks across multiple CPU cores. This parallel approach reduced the processing time from several hours to under one hour. The parallel processing was essential to make data augmentation practical at a larger scale

- **On Colab:** Used T4 GPU + 2 vCPUs for preprocessing.
- **On Local Machine:** Intel i7 11th Gen CPU with 8 cores / 16 threads and RTX 3070 Ti.

This reduced augmentation time from several hours to under one hour, enabling faster experimentation. The parallel approach was crucial to make data augmentation practical at scale.



```
Number of posts in the minority class to augment: 1430
Starting sequential run...
100%|██████████| 1430/1430 [00:14<00:00, 97.01it/s]
Starting parallel run on 4 cores...

100%|██████████| 1430/1430 [00:03<00:00, 383.44it/s]

--- Performance Results ---
Sequential run time: 14.75 seconds
Parallel run time:   3.86 seconds
Speedup: 3.82x

New augmented dataset saved as 'augmented_train.csv'
```

### 2.4 Model Architecture

- The model was a classifier based on **LSTM**.
- It included an **Embedding Layer** to convert tokens into dense vectors
- A **LSTM Layer** to capture the relationships between words
- A **Dense Layer** to learn higher-level representations
- An **Output Layer** to classify tweets into one of three categories: Hate Speech, Offensive, or Neither

### 2.5 Why LSTM?

LSTM models have a memory cell that helps them capture long-term dependencies in text, such as understanding negation. This is especially important for hate speech detection, where a word's meaning often depends on its context

### 2.6 Training Setup

- Optimizer: Adam
- Loss: Categorical Crossentropy
- Batch size: 64, Epochs: 10
- Metrics: Accuracy, Precision, Recall, F1-score
- Validation: Stratified train-test split to preserve class ratios

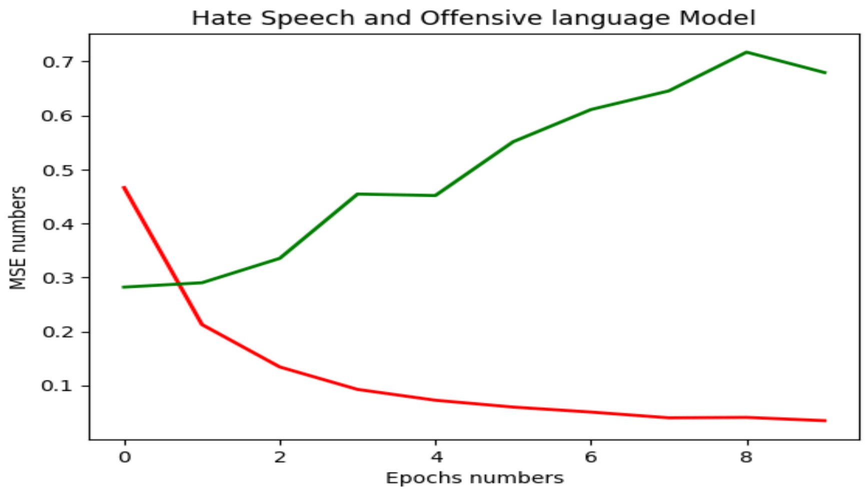
### 2.7 Evaluation & Validation

- Metrics: Accuracy, Precision, Recall, F1-score.
- Stratified train-test split for class balance.
- Visualizations: Training/validation curves, confusion matrix.
- Validation with Explainable AI: attention visualization to highlight key words influencing predictions.

## 3 Results and Findings

| Epoch | Train Acc | Train F1 | Val Acc | Val F1 |
|---|---|---|---|---|
| 1 | 82.8% | 0.80 | 89.8% | 0.89 |
| 5 | 97.5% | 0.97 | 88.9% | 0.89 |
| 10 | 98.7% | 0.98 | 88.1% | 0.88 |

- The model achieved **~90% validation accuracy** and **~0.89 F1-score**.



- **The confusion matrix** showed strong classification. But occasional confusion between *Hate Speech* and *Offensive*. (Fig showed above)
- Augmentation improved diversity and reduced overfitting compared to the baseline model.

## 4 Discussion

The results confirm that LSTM is an effective model for detecting hate speech. The use of **parallel processing** was essential, as it made data augmentation more efficient and prevented a bottleneck in the preprocessing stage. The model was able to distinguish between offensive insults and group-based hate. However, due to the dataset imbalance, the minority classes (like hate speech) were harder to classify.

**Explainable Methods** (XAI) such as attention heatmaps were used to validate the model's behavior. These visualizations showed that the model correctly focused on discriminatory words in hate speech, which increases trust in its decisions.

## 5 Conclusion and Future Work

This project successfully created a strong pipeline for detecting hate speech and offensive language using an **LSTM** model with augmented data. The model's validation accuracy was about 90% with balanced F1-scores. The use of parallel preprocessing was a major factor in making the data augmentation efficient and scalable.

Future work could include:

- Using more advanced models like **Transformer-based models** (BERT, ROBERTa).
- Expanding the dataset to include multiple languages.
- Using more advanced balancing techniques for imbalanced data.
- Integrating
  **Explainable AI** dashboards for real-time interpretation.
- Deploying the model as a web application for live moderation.

## 6 References

1. Davidson, T., Warmsley, D., Macy, M., & Weber, I. (2017). Automated hate speech detection and the problem of offensive language. *Proceedings of ICWSM (International AAAI Conference on Web and Social Media)*, 512–515.
2. Zhang, Z., Robinson, D., & Tepper, J. (2018). Detecting hate speech on Twitter using a convolution–GRU based deep neural network. *Lecture Notes in Computer Science*, Springer.
3. Badjatiya, P., Gupta, S., Gupta, M., & Varma, V. (2017). Deep learning for hate speech detection in tweets. *Proceedings of the 26th International Conference on World Wide Web Companion*, 759–760.
4. Fortuna, P., & Nunes, S. (2018). A survey on automatic detection of hate speech in text. *ACM Computing Surveys (CSUR)*, 51(4), 1–30.
5. Hochreiter, S., & Schmidhuber, J. (1997). Long short-term memory. *Neural Computation*, 9(8), 1735–1780.
6. Vaswani, A., Shazeer, N., Parmar, N., et al. (2017). Attention is all you need. *Advances in Neural Information Processing Systems (NeurIPS)*, 5998–6008.
7. Ribeiro, M. T., Singh, S., & Guestrin, C. (2016). “Why should I trust you?” Explaining the predictions of any classifier. *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, 1135–1144. (LIME – Explainable AI)