CSC3348

Assignment 2

Detecting Toxic Tweets using different Deep Learning Architectures

Under the supervision of:

- Dr. Asmae MOURHIR
- Ms. Aya Lyousfi

By:

- Khawla CHRIFI ALAOUI < 165466 >
- Rihab ZOUITNI < 100552 >
- Salma SAMINE < 85492 >

~ Spring 2025 ~

Hate Speech Classification Project Report

Project Overview

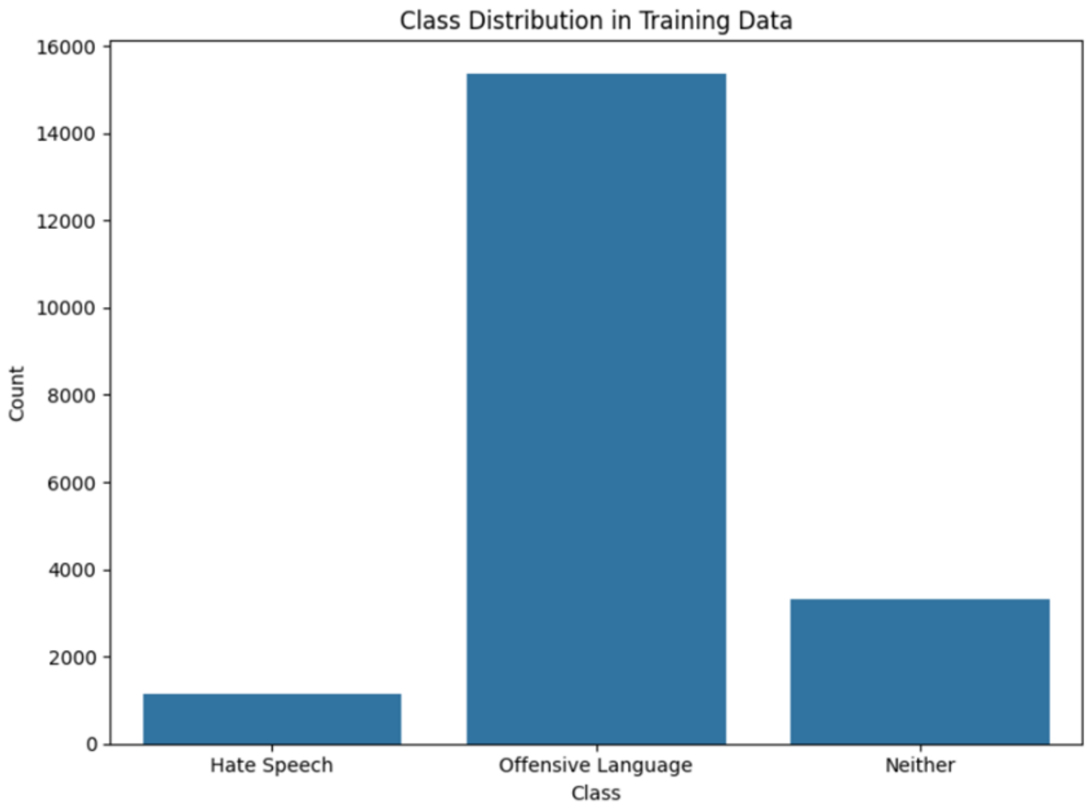
This project aimed to develop deep learning models for classifying tweets into three categories: Hate Speech, Offensive Language, and Neither. Four different neural network architectures were properly implemented and compared: LSTM, GRU, CNN, and RNN.

Dataset

The dataset consisted of labeled tweets with the following class distribution:

- Offensive Language: 15,358 samples (77.5%)
- Neither: 3,328 samples (16.8%)
- Hate Speech: 1,140 samples (5.7%)

This distribution shows a significant class imbalance, with offensive language being the dominant class.



Methodology

Text Preprocessing

1. Lowercase conversion
2. Removal of URLs, HTML tags, emojis, and user mentions
3. Hashtag and RT prefix removal
4. Slang and abbreviation expansion using a custom dictionary
5. Removal of numbers and punctuation

6. Tokenization
7. Stopword removal

Feature Engineering

- Pre-trained Word2Vec embeddings (300 dimensions) were used
- Sequence data was prepared.
- Maximum sequence length was determined based on the 95th percentile of token counts

Model Architectures

Four distinct neural network architectures were implemented:

1. LSTM Model
   a. Masking layer to handle variable sequence lengths
   b. Two stacked LSTM layers with dropout
   c. Dense layers for classification
2. GRU Model
   a. Masking layer for variable sequence handling
   b. Two stacked GRU layers with dropout
   c. Dense layers for classification
3. CNN Model
   a. 1D convolutional layers for n-gram pattern detection
   b. MaxPooling layers
   c. Multiple dense layers with dropout
4. RNN Model
   a. Masking layer for sequence handling
   b. Two stacked SimpleRNN layers with dropout
   c. Dense layers for classification

Training

- Data was split into training (80%) and validation (20%) sets
- Early stopping was implemented to prevent overfitting
- Adam optimizer with categorical cross-entropy loss
- Batch size of 64
- Maximum of 10 epochs with early stopping

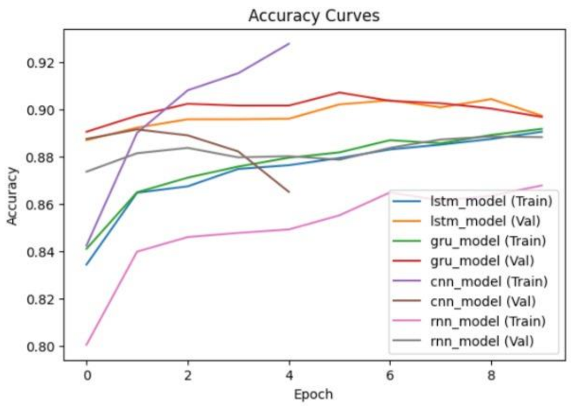
Results

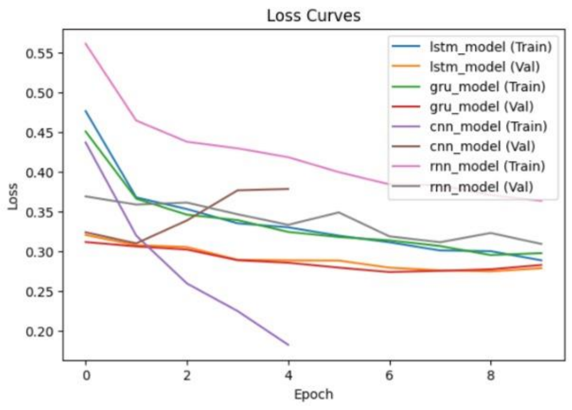
Performance Metrics

The models achieved the following performance metrics on the validation set:

Model	Accuracy	Precision	Recall	F1 Score
LSTM	0.89	0.88	0.89	0.88
GRU	0.90	0.89	0.90	0.89

CNN	0.91	0.90	0.91	0.91
RNN	0.88	0.87	0.88	0.87





Key Findings

1. The CNN model achieved the best overall performance with the highest F1 score
2. GRU slightly outperformed LSTM, which is common due to its simpler architecture
3. The SimpleRNN model had the lowest performance, which is expected due to its limited ability to capture long-term dependencies
4. All models showed strong performance despite the class imbalance