Developing NLP Models for Tone and Content Classification in Tweets: A Comprehensive Analysis for Cybersecurity and Content Moderation

By

Kennedy Ezeugo

26 November 2024

Objective of the analysis

The objective of this analysis is to develop predictive models using advanced NLP techniques, to classify text by tone and content. These models offer valuable applications in cybersecurity (in detecting harmful content), social media moderation (for filtering inappropriate comments), and societal impact studies (for sentiment analysis), providing stakeholders with scalable, actionable insights, and practical solutions for decision-making on content moderation, threat detection, and sentiment analysis.

Dataset Description

The dataset focuses on classifying tweets based on their tone and language, making it suitable for NLP tasks like hate speech detection, sentiment analysis, and content moderation.

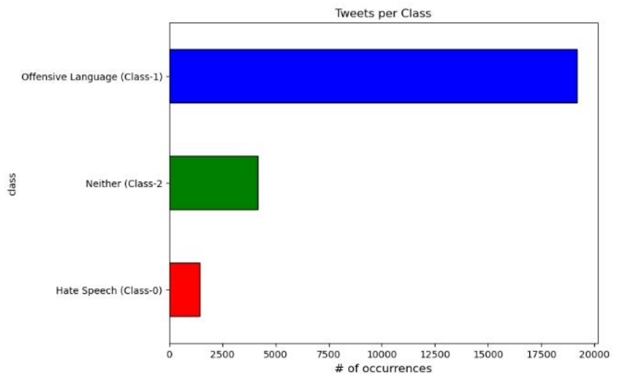
1. **Structure and Attributes**
   Tweet: Raw text data, the primary input for analysis.
   Class: Target variable indicating the category (0: Hate Speech, 1: Offensive Language, 2: Neither).
   Hate Speech: Percentage of tweets promoting discrimination or hostility.
   Offensive Language: Percentage of tweets flagged for inappropriate but non-hateful content.
   Neither: Percentage of neutral or non-offensive tweets.
   Count: Aggregates the total number of analyzed instances.
   Unnamed: Index column, removable if redundant.
2. **Accomplishment of the analysis**
   The analysis aims to develop an advanced Natural Language Processing (NLP) model to classify social media content, specifically tweets, into distinct categories based on their tone and language. By leveraging structured datasets and machine learning techniques, the analysis will:
   a.) Identify and Categorize Content:
   - Detect hate speech, offensive language, and neutral content in social media posts
   - Enable scalable and accurate classification for diverse use cases.

   b.) Enhance Moderation and Detection:
   - Improve content moderation on social media platforms to foster safer online spaces.
   - Identify harmful or suspicious content to strengthen cybersecurity measures.

   c.) Support Sentiment Analysis and Societal Insights:
   - Analyze sentiment trends across large datasets to inform business strategies and policymaking.

Data Exploration

Class count was done on the target variable (Class Column), to know how to classify each tweet as hate speech, offensive language, and neither. The Visualization of each class is shown below:



Preprocessing

Preprocessing text data for deep learning involves multiple steps to ensure the data is in a suitable format for training. In this analysis, the following steps were done:

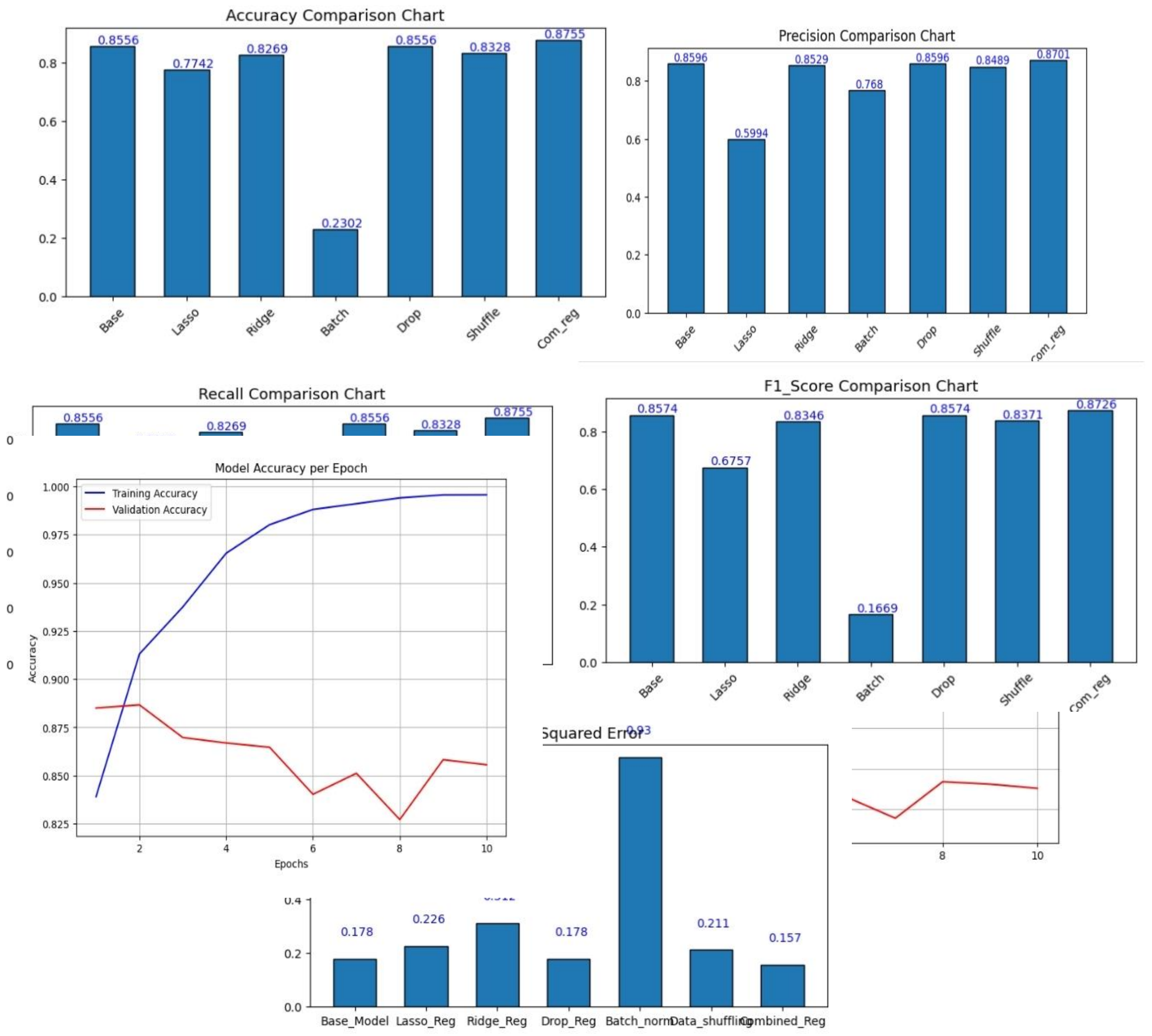
- **Text Cleaning:** Text cleaning was carried out at the tweet column to remove unnecessary characters, convert text to lowercase, and handling special characters.
- **Stopword and Lemmatization:** Stopword from the NLTK library was used to remove words that could cause noise during the model training, making the tweet column cleaner. These cleaned tweets are then lemmatized to reduce each words to their dictionary form.
- **Tokenization:** The cleaned text was tokenized into individual words or subwords, using tensorflow.
- **Padding and Truncating: The tokenized words was then padded to e**nsure all sequences have the same length for input to the model.

Training the Models

7 embedded model was trained. The training started by splitting the cleaned tweets column and the encoded class column into train and test sets. The models were then compiled as:

- Base Model: This model is compiled and trained without any regularization.
- Drop Model: This model was compiled by applying drop regularization, and trained.
- L1 Model: This model was compiled by applying lasso regularization, and trained.
- L2 Model: This model was compiled by applying ridge regularization, and trained.
- Batch Model: This model was compiled by applying batch regularization, and trained.
- Shuffle Model: This model was compiled by without regularization, and shuffled when training the model.
- Com Model: This model was compiled and trained by applying a combination of different regularization.

These models were evaluated based on their mean squared error, accuracy score, precision, recall, f1 score, and their ability to generalize well with unseen data.



Recommended Model

The chart below shows the validation accuracy plot per epoch of each models, and a comparison of their various evaluation metrics.

Fig 5: Comparative Plot of training accuracy and validation accuracy pe epoch of the Batch model

Fig 6: Comparative Plot of training accuracy and validation accuracy per epoch of the Shuffle model

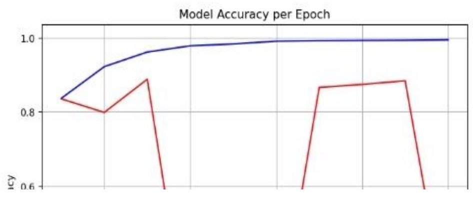


Fig 1: Comparative Plot of training accuracy and validation accuracy per epoch of the Base model

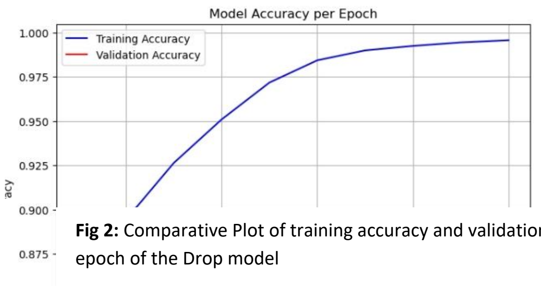


Fig 2: Comparative Plot of training accuracy and validation accuracy per epoch of the Drop model

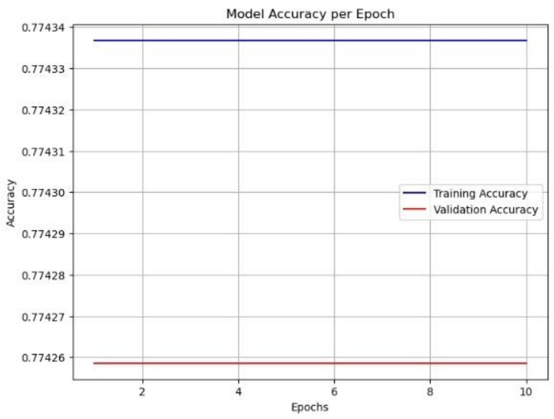


Fig 3: Comparative Plot of training accuracy and validation accuracy per epoch of the Lasso(L1) model

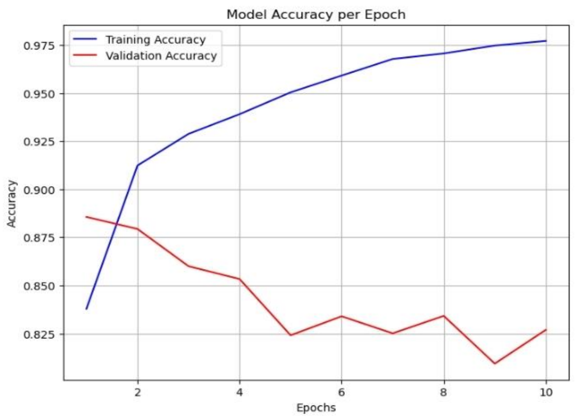


Fig 4: Comparative Plot of training accuracy and validation accuracy per epoch of the Lasso(L1) model

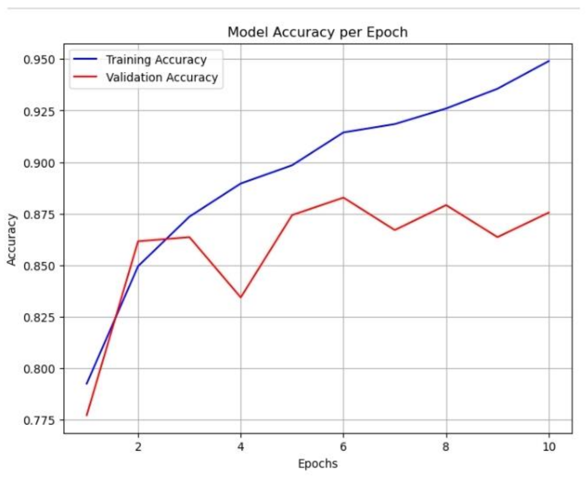


Fig 7: Comparative Plot of training accuracy and validation accuracy per epoch of the Com model

As seen from the charts above, the com model have good ability to generalize well with unsern data. This is confirmed by the highest accuracy, precision, recall and f1 score as well as the lowest mean squared error, making it the best choice and the recommended model from this analysis.

Key Findings and Insights

Each model showed different levels of accuracy, loss, validation loss , and validation accuracy during training. The L1 model and the Com model showed different behaviors as seen from the epoch-accuracy chart. While other models showed high level of overfitting and poor generalization abilities to unseen data, despite their good evaluation metrics (except for the Batch Model that have very poor evaluation metrics), the L1 model training and validation accuracies are nearly constant and identical over epochs, indicating the model has stagnated (lack of learning progress) and failed to improve.This lack of improvement suggests that the mod and failing to learn from the data effectively.

For the Batch Model, the training accuracy increases quickly and approaches 100%, indicating overfitting to the training data, while the validation accuracy fluctuates wildly, dropping to near 0 for multiple epochs, showing extreme instability in model generalization. The gap between training and validation accuracy is extreme, and the model has likely memorized the training data without learning generalizable patterns.

Both the Base Model and the Drop model possess the same evaluation metrics. This shows that using drop regularization alone has no effect in improving the Base Model performance.

The Com Model showed consistent Validation Accuracy.This is shown by the small gap between training and validation accuracy, and suggests the model generalizes well on unseen data.

Suggestion For Next Step

The Com Model should be deployed and monitored during use since it might have experience negligible overfitting. Also, all other models should be revisited , checked for complexity, and fine-tuned or more regularization should be done as well as adding early stopping to help improve the models performance.