INFOSYS SPRINGBOARD INTERNSHIP

+

HATE SPEECH DETECTION ON TELEGRAM PLATFORM

BY: A. BHARGAVI

BUSINESS PROBLEM

Hate speech on social media platforms like Telegram presents a significant challenge, threatening user safety and the platform's reputation. Telegram users send millions of messages daily, making it impossible for human moderators to review all content manually. The sheer volume and speed of content generation necessitate a more efficient solution for detecting and mitigating hate speech.

SOLUTION

The project aims to develop an AI model for text classification system designed to automatically detect whether the speech is hateful or not and categorize hateful content within user-generated text, such as social media posts, comments, and reviews.The model is trained on a labeled dataset and utilizes multiple natural language processing (NLP) methods to detect and categorize hate speech.This system leverages a deep learning model that combines the capabilities of BERT (Bidirectional Encoder Representations from Transformers) and LSTM (Long Short-Term Memory) networks, ensuring accurate and context-aware identification of hateful speech. By implementing this system, we aim to enhance the ability to moderate content and maintain a positive community environment.

DATASET DESCRIPTION

The Gab Hate Corpus (GHC), consisting of 22,037 posts from the social network service gab.ai, each annotated by a minimum of three trained annotators. Annotators were trained to label posts according to a coding typology derived from a synthesis of hate speech definitions across legal, computational, psychological, and sociological research.It consists of CV (a Call for Violence), HD (an Assault on Human Dignity), VO (Vulgarity/Offensive Language directed at an individual): HD and CV, or NH (Not Hateful) which is labelled as 0. If none apply, the document is to be considered NH (Not Hateful) which is labelled as 0.

Dataset URL: https://osf.io/edua3/

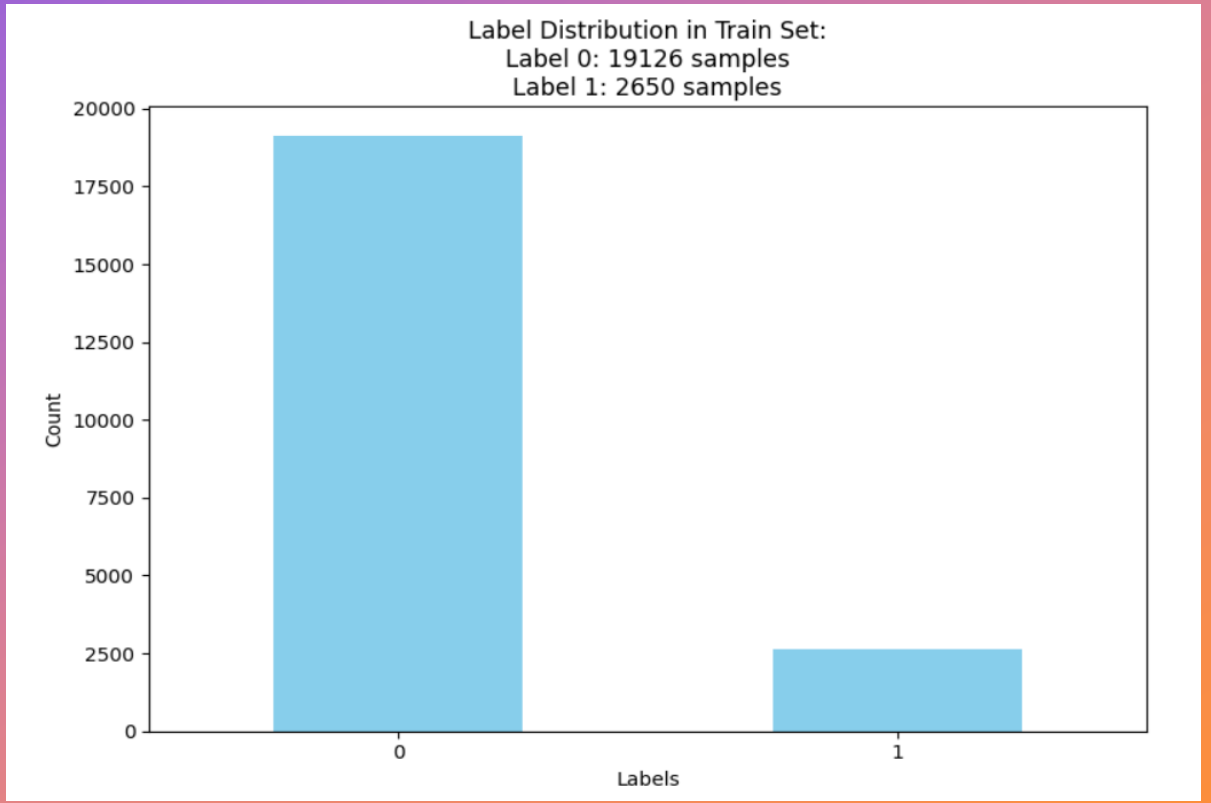
What evil lurks at the rotten core of you pension fund? https://pension.solari.com/2018/01/22/pension-fund-stats/	0	0	0
No way. That bitch is STUPID as a rock. No I won't. Also what sort of CUCK willingly allows ANY WOMEN to give him orders? BOY.	1	0	1
Obama and his entire administration should be in Prison!!	1	0	0

DATA VISUALIZATION

The Gab Hate Corpus (GHC) dataset, consisting of 22,037 posts to detect whether it is hate speech or not.

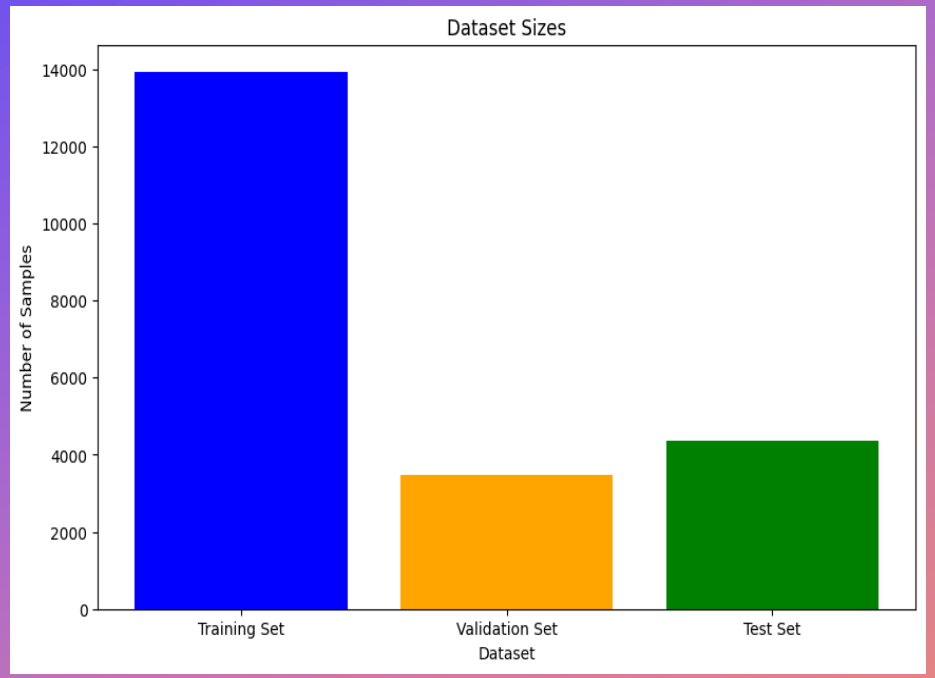
Label 0: 19126

Label 1: 2650

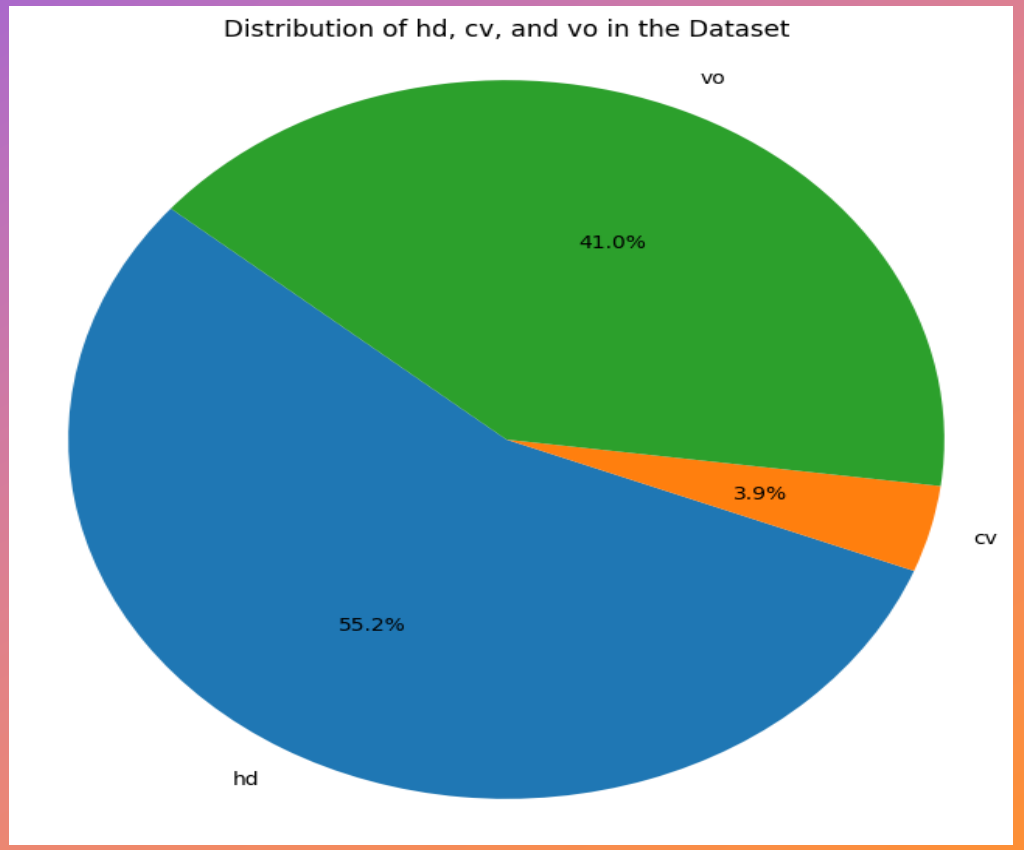


DATA VISUALIZATION

Distribution of Train, Test and Validation sets



Distribution of Metadata



DATA PREPROCESSING

The raw text data undergoes several preprocessing steps to clean and prepare it for analysis or modeling:

Lowercasing: All letters are converted to lowercase to ensure uniformity and consistency in the text data.

Remove URLs: Elimination of URLs or hyperlinks present in the text data, which are typically irrelevant for analysis or modeling purposes.

Stemming: Reduction of words to their root form by removing suffixes, allowing for variations of a word to be treated as the same token.

Lemmatization: Conversion of words to their base or dictionary form (lemma) to normalize variations and ensure consistent representation across different forms of the same word.

Emoji Conversion: Conversion of emojis to text data to focus solely on textual content.

DATA PREPROCESSING

Remove Extra Spaces: Trimming excess whitespace between words to maintain uniform spacing and improve readability

Abbreviation Treatment: Handling of abbreviations to expand them into their full forms for clarity and consistency.

Remove Stop Words: Elimination of common words (e.g., "the", "and", "is") from the text data that do not carry significant meaning or contribute to the analysis.

Tokenization: Splitting the text into individual tokens or words, which are then processed through the aforementioned steps.

TOKENIZATION AND EMBEDDING TECHNIQUES

For Machine Learning Models

White Space Tokenization: White space tokenization efficiently splits text into tokens using spaces, tabs, and newlines, making it straightforward and effective for many applications. It is a simple approach and works well.

Embedding Techniques:

One-Hot Encoding: One-hot encoding converts categorical data into binary vectors, where each category is represented by a unique vector with one element set to 1 and the rest to 0, facilitating their use in machine learning algorithms.

Label Encoding: Label encoding assigns a unique integer to each category in the data, transforming categorical variables into numerical values, making them suitable for algorithms that require numerical input.

TF-IDF Embeddings: TF-IDF is a statistical measure used in text mining to evaluate the importance of a word in a document relative to a collection of documents. It combines term frequency and inverse document frequency to highlight important words while reducing the weight of common terms.

Finalized Embedding Technique: TF-IDF (Term Frequency-Inverse Document Frequency) Embedding

TOKENIZATION AND EMBEDDING TECHNIQUES

For Deep Learning Models

BERT Tokenizer: The BERT (Bidirectional Encoder Representations from Transformers) tokenizer processes text into tokens by splitting words into subwords and converting them into numerical IDs, ensuring that even rare and unknown words are handled effectively for BERT model input. It also preserves the context and meaning of words by leveraging a large vocabulary of subword units.

BERT Embeddings: BERT (Bidirectional Encoder Representations from Transformers) embeddings are dense vector representations of text generated by the BERT model, capturing contextual information and relationships between words to enhance the performance of downstream NLP tasks. These embeddings are pre-trained on large corpora and can be fine-tuned for specific tasks, providing robust and versatile representations.

MODELING

Machine Learning Models

1. Logistic Regression
2. Random Forest
3. Multinomial Naïve Bayes
4. Gradient Boosting

Finalized ML Model: Gradient Boosting

- It builds an ensemble of weak learners (typically decision trees) in a sequential manner, where each new tree corrects the errors of the previous ones, leading to high predictive accuracy.
- It can handle various types of data, including numerical and categorical features, without the need for extensive preprocessing.
- By using techniques like shrinkage, subsampling, and regularization, gradient boosting can be tuned to avoid overfitting while maintaining high performance.
- It provides insights into feature importance, helping to understand the underlying data and model behavior.

MODELING

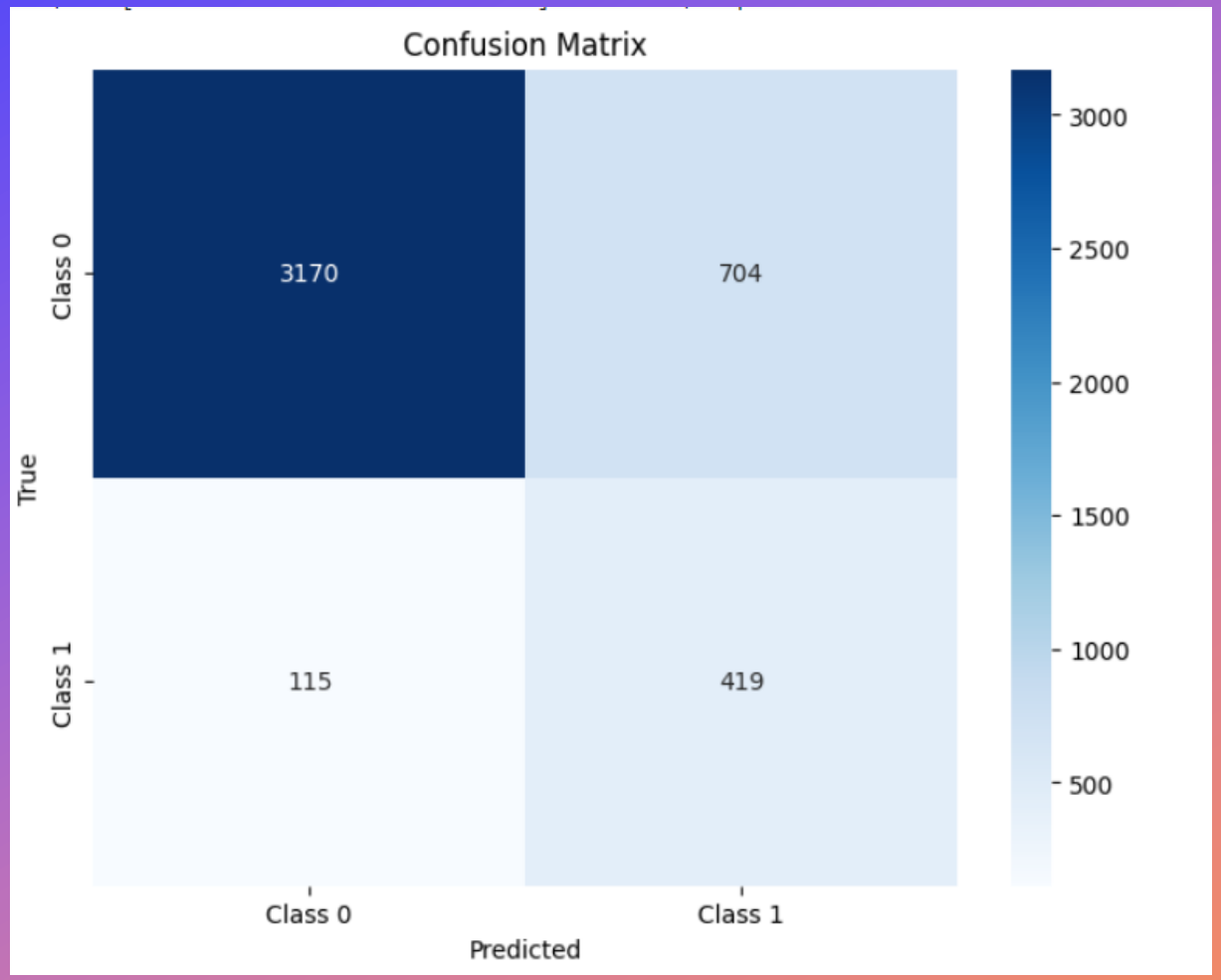
Deep Learning Models

1. CNN
2. Bidirectional LSTM
3. Bidirectional LSTM with BERT Embeddings

Finalized DL Model: Bidirectional LSTM with BERT Embeddings

- BERT embeddings capture deep contextual information from both directions (left-to-right and right-to-left), while the BiLSTM processes the input sequence in both forward and backward directions, providing a comprehensive understanding of the context.
- The model is particularly effective at capturing complex dependencies and relationships in text, benefiting tasks that require understanding long-range dependencies and intricate patterns.
- BERT embeddings are pre-trained on vast amounts of text data, enabling the model to leverage this extensive knowledge and transfer learning to improve performance on downstream tasks with limited labeled data.
- The combination leverages BERT's strength in encoding language representations and the BiLSTM's ability to model temporal dependencies, resulting in superior performance for sequence-based tasks like text classification, named entity recognition, and language translation.

Confusion Matrix of Bidirectional LSTM with Bert Model



Recall and ROC AUC Score of the Bidirectional LSTM with BERT Model

Recall: 0.7340823970037453

ROC AUC Score: 0.8813711016881971

EVALUATION METRICS

Recall: Recall emphasizes the model's ability to identify all relevant positive instances, which is crucial in applications where missing positive cases is more critical than incorrectly labeling negatives. It is particularly valuable in situations with imbalanced datasets where the number of positive instances is significantly lower than the number of negative instances, ensuring that the model doesn't overlook the minority class.

For ML Model: 54%

For DL Model: 73%

ROC AUC Score: ROC AUC score measures the ability of a binary classification model to distinguish between classes. A higher ROC AUC score indicates better performance of the model.

For ML Model: 85%

For DL Model: 88%

THANK YOU