Deep Learning-Based Text Classification for Hate Speech in Online Social Networks

A synopsis submitted in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY
in
Computer Engineering

Under The Supervision of:
Dr Faiyaz Ahmad
Assistant Professor

Submitted By:
Faizan Ahamad 21BCS047
Mantasha Firdous 21BCS049

Department of Computer Engineering
Faculty of Engineering and Technology
Jamia Millia Islamia – New Delhi- 110025

(YEAR-2025)

Abstract

In the digital era, the internet has transformed global communication, enabling individuals to share ideas, information, and awareness effortlessly. However, this freedom of expression is often misused, with online platforms becoming spaces for abusive or racist remarks, frequently made under the veil of anonymity. The increasing prevalence of hate speech highlights the urgent need for robust and effective mechanisms to detect and mitigate its harmful impact. Despite significant advancements in research, hate speech detection remains a challenging task due to the complexity of language and context.

This project proposes a deep learning-based approach to detect abuse and hate speech on online social networks. The methodology involves preprocessing the data, extracting sentence embeddings using the **BERT** model, and classifying the embeddings using a custom **CNN** model. The proposed approach aims to address the limitations of traditional machine learning models and improve detection accuracy. The performance of the model will be evaluated on the Hate Speech and Offensive Language dataset, with the potential to contribute to the development of effective hate speech detection systems.

Introduction

With the rapid adoption of smartphones and affordable internet services, social media platforms have become an integral part of global communication. However, this increased connectivity has also amplified the prevalence of hate speech and offensive content on platforms like X (formerly Twitter), Facebook, and Instagram. Reports indicate that over one million social media accounts were banned in six months, with millions of harmful posts removed, highlighting the urgent need for intervention.

Anonymity on these platforms often emboldens users to share hate speech, targeting individuals or communities. Such content can range from discriminatory remarks to cyberbullying, with severe consequences, including mental health issues and, in extreme cases, suicide. Despite its importance, defining and distinguishing hate speech from offensive speech remains challenging due to overlapping interpretations. This project adopts a three-category classification system: hate speech, offensive speech, and neither.

Manual detection of hate speech is impractical due to the sheer volume of content generated daily, motivating the need for automated solutions. Leveraging natural language processing (NLP) and deep learning, this study aims to develop a robust model capable of classifying text—such as tweets or sentences—into the defined categories.

Problem Statement:

The primary objective is to automate the classification of text into hate speech, offensive speech, or neutral categories. Automated

systems must achieve high precision to minimize false positives, which could lead to unjust account suspensions or content removal. The model must balance accuracy and fairness, recognizing that interpretations of offensive content may vary among individuals.

Motivation:

With increasing access to social media among children and young adults, it is critical to curb exposure to harmful content. Hate speech targeting race, gender, or identity can significantly impact young minds, emphasizing the importance of effective detection mechanisms to foster safer online environments.

Proposed Method

This project employs a structured methodology to classify text into hate speech, offensive speech, or neutral categories. The process involves three key steps: data preprocessing, feature extraction using BERT embeddings, and model training with a custom CNN architecture. The methodology ensures robust handling of textual data and efficient classification.

1.Data Preprocessing

The first step involves cleaning the dataset to remove noise and inconsistencies, which improves the performance of classification models. This process includes:

1. Replacing user mentions with a placeholder.
2. Removing Unicode emojis, URLs, punctuation marks, and stop words.
3. Retaining only meaningful linguistic patterns to enhance feature extraction.

Algorithmically, each tweet undergoes preprocessing to ensure uniformity and focus on relevant semantic information.

2.BERT Embeddings

To extract high-quality features, BERT (Bidirectional Encoder Representations from Transformers) embeddings are utilized. BERT captures both left and right contextual information, providing a robust representation of text. Each sentence is padded with special tokens (“[CLS]” and “[SEP]”), tokenized, and passed through a pre-trained BERT model. The “[CLS]” token from the

final hidden layer serves as the sentence embedding, encapsulating the overall context.

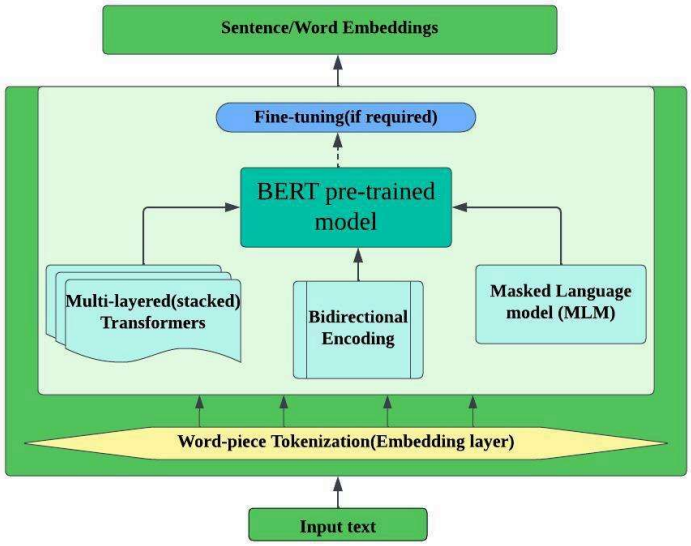

Figure 1:BERT model architecture[7]

3.Dataset Splitting

After preprocessing, the dataset is split into training and testing sets using an 80:20 ratio, ensuring a balance between sufficient training data and generalizability. The stratify parameter is used to maintain the class distribution in both subsets, addressing potential class imbalance.

4.Model Training

The processed embeddings are fed into a custom CNN model designed for text classification. The performance of this model is compared against baseline machine learning models to evaluate its effectiveness. The model showing the best results is selected for further analysis.

Tools and Framework

For this project proposal, following technology stack for implementing and evaluating the hate speech classification model will be used:

- Development Platform: Google Colab.
- Proposed Libraries and Frameworks:
  - **Pandas**: To load datasets and manipulate data frames.
  - **NumPy**: For handling multidimensional arrays and storing embeddings.
  - **Keras**: To design and implement custom deep learning models, including CNNs.
  - **TensorFlow**: For building and executing deep learning computations.
  - **Transformers**: To access pre-trained BERT models for generating contextual embeddings.
  - **WordCloud**: For visualizing the dataset through word clouds.
  - **Matplotlib**: For plotting and visualizing data insights.
  - **NLTK**: For natural language preprocessing tasks.
  - **Re**: For text string manipulations, such as removing stop words and hashtags.
  - **Scikit-learn**: For dataset splitting and implementing baseline machine learning models.

References

[1] Ghosal, Sayani, et al. "Inculcating Context for Emoji Powered Bengali Hate Speech Detection using Extended Fuzzy SVM and Text Embedding Models." ACM Transactions on Asian and Low-Resource Language Information Processing (2023).
[2] Chhabra, Anusha, and Dinesh Kumar Vishwakarma. "Fuzzy and Machine Learning Classifiers for Hate Content Detection: A Comparative Analysis." 2022 4th International Conference on Artificial Intelligence and Speech Technology (AIST). IEEE, 2022.
[3] K J, Bhanushree, et al. "Automatic Hate Speech Detection using Ensemble Method and Natural Language Processing Techniques." 2023 7th National Conference on Machines and Intelligent Techniques *(*NMITCON*)*. IEEE, 2023.
[4] Jemima, P. Preethy, et al. "Hate Speech Detection using Machine Learning." 2022 7th International Conference on Communication and Electronics Systems (ICCES). IEEE, 2022.
[5] Melton, Joshua, Arunkumar Bagavathi, and Siddharth Krishnan. "DeL-haTE: a deep learning tunable ensemble for hate speech detection." 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA). IEEE, 2020.
[6] Chopra, Abhishek, et al. "A Framework for Online Hate Speech Detection on Codemixed Hindi-English Text and Hindi Text in Devanagari." ACM Transactions on Asian and Low-Resource Language Information Processing 22.5 (2023)
[7]https://www.geeksforgeeks.org/how-to-generate-word-embedding-using-bert/