Hate Speech Detection using Machine Learning

Anurag Wasankar

Moxaben Bhupatbhai Zalawadia(mxz210014)

Astha Thakur

**Problem:**

Social media platforms allow users to express themselves freely, but this can come at a cost. The moderators who are responsible for keeping social media clean are often overworked and underpaid, which can lead to hate speech and other harmful content going unchecked. This can have serious consequences for individuals and communities, including harassment, discrimination, and the spread of harmful ideologies. This project aims to compare and identify the best approach for detecting hate speech on social media. By developing a model that can automatically identify and flag hate speech content, we hope to make social media a safer and more inclusive space for everyone.

**Design:**

The system design of the projects involves data retrieval, data preprocessing, feature engineering, and model training. Data is downloaded from Google Drive and explored for analysis. Text is cleaned by converting to lowercase, removing retweet mentions, URLs, and punctuations, and tokenizing and stemming the words. Feature engineering prepares the text data for the model. The neural network architecture is defined, compiled with categorical cross-entropy loss, and trained using training data with model checkpoints. The model's performance is evaluated with classification metrics on the test set, allowing for potential improvements.

**Implementation:**

The implementation of sentiment analysis involves gathering data from various sources through web scraping or APIs, preprocessing it by removing noise and transforming it into a suitable format. The text is tokenized, normalized, and stop words are removed. Exploratory data analysis is performed to understand the dataset, followed by a train-test split. An appropriate sentiment analysis model is chosen, implemented using a suitable framework, and trained with an appropriate loss function. The model's performance is evaluated using metrics like accuracy, precision, recall, and F1-score, with hyperparameter tuning to optimize the model's performance.

**Challenges:**

The dataset used for this project was based on Twitter, making it noisy and containing modern-day abbreviations and internet language, including emojis and shortened words like "k" for "ok" and "ty" for "thank you." The data also included additional keywords added by the Twitter API, like "RT" for retweets, which complicated the data cleaning process due to the presence of similar abbreviations. Moreover, the dataset was skewed, with one class having significantly fewer samples, leading to models being biased towards the other classes and resulting in high accuracy but poor precision. Due to the resource and time-intensive nature of deep learning models, parameter tuning on cloud resources took longer than expected.

**Screenshots of the result:**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**A screenshot of a computer

Description automatically generated**

**Directions:**

The most effective model we found was the Bidirectional LSTM, a deep learning model. To enhance its performance, training it on a larger and balanced dataset is recommended, as this would expand its predictive capabilities. Additionally, the model's current single-language training could be extended to support multiple languages by utilizing a diverse dataset. To make the model accessible to social websites relying on user text interactions, it can be deployed using a simple Python Flask API. Furthermore, the model's capabilities can be extended by combining it with a deep learning image recognition model, enabling it to extract text from images and broadening its range of applications.