Cyberbullying Detection using Hybrid RNN and LSTM

Introduction

Cyberbullying is a pervasive issue in today's digital era, particularly on social media platforms. Detecting harmful and offensive content online is crucial for creating a safer internet environment. This project aims to develop a model to detect cyberbullying based on comments extracted from social media platforms such as YouTube and Reddit.

Objective

The objective of this project is to build a dataset by scraping comments from YouTube and Reddit, clean and preprocess the data, and label the content as 'cyberbullying' or 'non-cyberbullying' which is then followed by tokenization.

Data Collection

We collected data by web scraping comments from YouTube and Reddit. We used the YouTube Data API to extract comments, and Python's BeautifulSoup and PRAW libraries for scraping Reddit. The collected data was stored as CSV files for further processing.

Technologies Used for Web Scraping

1. **YouTube Data API**: Used to fetch comments from YouTube videos.
2. **PRAW (Python Reddit API Wrapper):** Used to extract comments from Reddit threads.
3. **CSV:** The scraped data was saved as CSV files using Python's built-in CSV module for further analysis.

YouTube Scraping - Code Snippet

```
import googleapiclient.discovery
import pandas as pd
api_service_name = "youtube"
api_version = "v3"
api_key = "AIzaSyCJqXW3sC6RYsUhOFvyARThnMx6_eJyNfI"
youtube = googleapiclient.discovery.build(api_service_name, api_version,
developerKey=api_key)
def get_all_video_comments(video_id):
    comments_list = []
    request = youtube.commentThreads().list(
```

```
        part="snippet",
        videoId=video_id,
        maxResults=100
    )
    while request:
        response = request.execute()
        for item in response["items"]:
            comment = item["snippet"]["topLevelComment"]["snippet"]
            text = comment["textDisplay"]
            comments_list.append(text)
        request = youtube.commentThreads().list_next(request, response)
    return comments_list

video_id = "RZp07vLSak8"
comments = get_all_video_comments(video_id)
df = pd.DataFrame(comments, columns=["Comments"])
df.to_csv("youtube_comments-2.csv", index=False)
print("All comments have been saved to youtube_comments.csv")
```

Reddit Scraping - Code Snippet

```
import asyncio
import pandas as pd
async def fetch_multiple_subreddits(subreddits, limit):
    comments_data = []
    for subreddit_name in subreddits:
        try:
            subreddit = await reddit.subreddit(subreddit_name)
            async for comment in subreddit.comments(limit=limit):
                comment_text = comment.body.lower()
                if comment.author:
```

```
                comments_data.append([comment.created_utc, comment.author.name,
comment.body])
        except Exception as e:
            print(f"Error fetching comments from subreddit '{subreddit_name}': {e}")
            continue
    comments_df = pd.DataFrame(comments_data, columns=['Date', 'User', 'Comment'])
    return comments_df


async def run_async():
    comments_df = await fetch_multiple_subreddits(subreddits=subreddits_list, limit=20000)
    await reddit.close()
    return comments_df


subreddits_list = [
    'roastme', 'unpopularopinion', 'cringe', 'insults',
    'choosingbeggars', 'relationship_advice', 'rant', 'confessions',
    'TrueOffMyChest', 'bullying', 'depression'
]


filtered_comments = asyncio.run(run_async())
```

Text Cleaning

After collecting the data, we performed several text cleaning operations to remove unwanted characters, URLs, emojis, and punctuation. We also removed extra whitespaces to prepare the text for further analysis.

Text Cleaning - Code Snippet

```
import pandas as pd
import re
import string
```

```
df = pd.read_csv('youtube_comments-2.csv')
def remove_html_tags(text):
    clean = re.compile('<.*?>')
    return re.sub(clean, '', text)
def remove_links(text):
    return re.sub(r'http\S+|www\S+|https\S+', '', text, flags=re.MULTILINE)
def remove_emojis(text):
    emoji_pattern = re.compile(
        "["
        "\U0001F600-\U0001F64F"
       "\U0001F300-\U0001F5FF"
      "\U0001F680-\U0001F6FF"
        "\U0001F700-\U0001F77F"
        "\U0001F780-\U0001F7FF"
        "\U0001F800-\U0001F8FF"
        "\U0001F900-\U0001F9FF"
        "\U0001FA00-\U0001FA6F"
        "\U00002702-\U000027B0"
        "\U000024C2-\U0001F251"
        "]+",
        flags=re.UNICODE)
    return emoji_pattern.sub(r'', text)
def remove_punctuation(text):
    return text.translate(str.maketrans('', '', string.punctuation))
def clean_text(text):
    text = remove_html_tags(text)
    text = remove_links(text)
```

```
    text = remove_emojis(text)
    text = remove_punctuation(text)
    text = re.sub(r'\s+', ' ', text)
    return text.strip()
df['cleaned_comments'] = df['Comments'].apply(clean_text)
df.to_csv('cleaned_data.csv', index=False)
from google.colab import files
files.download('cleaned_data.csv')
print("Text cleaning complete! Cleaned data saved to 'cleaned_data.csv'.")
```

Data Labeling

After cleaning the data, we manually labeled the comments as 'cyberbullying' or 'non-cyberbullying'. This labeling process was crucial in providing labeled training data for the machine learning model.

Flow of the process