Comment ML – YouTube Sentimental & Trend Analyzer

A

Minor Synopsis Report

submitted in partial fulfillment of the requirements for the award of the degree of

Master of Computer Applications

in

Artificial Intelligence

by

Name	**Roll No.**
Suraj Kumar	590016150
Saloni Chaudhary	590017783
Lucky Patel	590016745
Siddharth Kumar	590016229

Under the guidance of

Dr. Prateek Raj Gautam

School of Computer Science, UPES

Bidholi, Via Prem Nagar, Dehradun, Uttarakhand

Month – 2025

CANDIDATE'S DECLARATION

We hereby certify that the project work entitled **"Comment ML – YouTube Sentimental & Trend Analyzer"** in partial fulfillment of the requirements for the award of the Degree of **Master of Computer Applications** with specialization in **Artificial Intelligence**, submitted to the Artificial Intelligence Cluster, School of Computer Science, UPES, Dehradun, is an authentic record of our work carried out during a period from **August, 2025** to **December, 2025** under the supervision of **Dr. Prateek Raj Gautam**.

The matter presented in this project has not been submitted by us for the award of any other degree of this or any other University.

Suraj Kumar - 590016150
Saloni Chaudhary - 590017783
Lucky Patel - 590016745
Siddharth Kumar - 590016229

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

Date: 10-10-2025

Dr. Prateek Raj Gautam
Project Guide

Acknowledgement

We wish to express our deep gratitude to our guide Dr. Prateek Raj Gautam, for all advice, encouragement and constant support he has given us throughout our project work. This work would not have been possible without his support and valuable suggestions. We sincerely thanks to our respected Suvojit Dhara of HoD, Head Department of Computer Science, for his great support in doing our project in Computer Science., for his great support in doing our project in Machine Learning. We are also grateful to Dean SoCS UPES for giving us the necessary facilities to carry out our project work successfully. We also thanks to our Course Coordinator, Suvojit Dhara and our Activity Coordinator Dr. Prateek Raj Gautam for providing timely support and information during the completion of this project. We would like to thank all our friends for their help and constructive criticism during our project work. Finally, we have no words to express our sincere gratitude to our parents who have shown us this world and for every support they have given us.

Suraj Kumar

SAP ID. 590016150

Saloni Chaudhary

SAP ID. 590017783

Lucky Patel

SAP ID. 590016745

Siddharth Kumar

SAP ID. 590016229

Abstract

"Comment ML" – YouTube Sentimental & Trend Analyzer is a project designed to address the complexity of managing and interpreting the massive volume of comments generated on YouTube videos. Influencers and content creators often receive thousands of comments, making manual analysis time-consuming and ineffective. Understanding audience sentiment and detecting trending discussions are essential for improving content strategies and enhancing creator–audience relationships.

The goal of this project is to design and develop an automated system that performs real-time sentiment analysis and trend detection on YouTube comments. The system classifies comments into positive, negative, and neutral categories, while also summarizing key themes, detecting spam/troll comments, and highlighting trending keywords through visualizations such as word clouds and sentiment distribution charts.

From the **creators' perspective**, the system reduces the burden of manually filtering comments and provides actionable insights for improving content quality, engagement, and marketing strategies. From the **users' perspective**, it ensures that their feedback is meaningfully captured, trends are recognized, and spam or harmful comments are filtered, contributing to a healthier online community.

The demonstrator system is implemented as a **Chrome Extension** with a Python backend built using Flask/FastAPI. It leverages Natural Language Processing (NLP) libraries like NLTK and spaCy for preprocessing, while machine learning models are tuned using Optuna and tracked via MLflow. Deployment uses Docker, GitHub Actions, and AWS for scalability. The user-friendly dashboard enables interactive visualization of audience insights, reducing manual effort and enabling data-driven decisions.

Applications of this system include:

- Audience Sentiment Tracking
- Content Strategy Optimization
- Influencer Marketing Insights
- Trend Detection and Keyword Analysis
- Spam & Troll Filtering
- Enhancing Creator–Audience Interaction

"Comment ML" – YouTube Sentimental & Trend Analyzer provides an efficient and practical solution for large-scale comment analysis. By addressing both creator and user perspectives, the system not only saves time for influencers but also ensures that audience feedback is better understood and utilized to foster stronger digital communities.

Contents

List of Figures

Chapter 1

Introduction

In the digital era, social media platforms have become the most powerful medium of communication, interaction, and influence. Among these, YouTube holds a unique position as one of the largest platforms for video sharing, audience engagement, and community building. Millions of users interact with content every day by liking, sharing, and most importantly, commenting. These comments reflect the true voice of the audience — their opinions, preferences, appreciation, criticism, and suggestions. However, with the exponential increase in video consumption, creators and influencers face the challenge of managing thousands of comments that are posted on their videos. Manually analyzing such large volumes of data is not only impractical but also limits the ability to extract valuable insights.

Comment ML – YouTube Sentimental & Trend Analyzer is a project designed to solve this problem by leveraging Natural Language Processing (NLP) and Machine Learning (ML) techniques. The system automatically classifies comments into categories such as positive, negative, and neutral, while also detecting trending keywords, summarizing discussions, and filtering spam or troll comments. By providing visualizations like sentiment distribution graphs and word clouds, it simplifies complex data into actionable insights.

From a creator's perspective, the tool acts as a decision-support system, helping them understand audience reactions, identify what type of content resonates, and refine their strategies accordingly. From the users' perspective, their feedback and opinions are meaningfully captured and represented, ensuring that their voices contribute to shaping the creator's future content. In addition, filtering harmful or spam comments supports a safer and healthier community experience.

This project integrates advanced NLP libraries such as NLTK and spaCy, along with ML frameworks for model building, tuning, and deployment. It is implemented as a Chrome extension backed by a Python-based API, ensuring accessibility and ease of use for creators. Beyond influencers, the system can also benefit brands, marketers, and researchers by providing consumer insights and tracking emerging trends. Thus, this project demonstrates how technology can bridge the gap between creators and their audiences by transforming unstructured comment data into valuable knowledge.

1.1 History

The history of sentiment analysis and trend detection is closely tied to the evolution of social media and user-generated content on the internet. In the early 2000s, senti-

ment analysis began as a research problem in the field of Natural Language Processing (NLP), primarily focused on analyzing product reviews, blogs, and news articles. Initial approaches relied heavily on lexicon-based methods, where predefined word dictionaries were used to classify text as positive, negative, or neutral. Although these methods were simple, they lacked the ability to handle context, sarcasm, or evolving internet language.

As social media platforms such as Facebook, Twitter, and YouTube gained popularity, the demand for large-scale comment and opinion mining grew significantly. Researchers shifted from lexicon-based techniques to machine learning models, which could learn sentiment patterns from labeled datasets. With the rise of deep learning in the 2010s, advanced models like recurrent neural networks (RNNs) and transformers enabled more accurate and context-aware sentiment classification.

YouTube, being one of the most influential platforms for content creation, posed unique challenges for sentiment analysis. Comments on videos are often short, informal, multilingual, and filled with slang, emojis, or sarcasm. Traditional models struggled to analyze such data effectively. At the same time, influencers and brands recognized the importance of understanding audience sentiment for shaping their strategies, creating a demand for automated solutions.

Today, the combination of NLP, machine learning, and MLOps practices allows the development of tools like **"Comment ML" – YouTube Sentimental & Trend Analyzer**, which go beyond simple sentiment classification to include trend detection, spam filtering, and visualization. This evolution highlights the growing importance of comment analysis in bridging the gap between creators and audiences in the digital age.

1.2 Requirement Analysis

Requirement analysis is a critical phase of the project, as it defines what the system must achieve to address the problem effectively. In the context of **"Comment ML" – YouTube Sentimental & Trend Analyzer**, the primary objective is to help influencers and creators analyze large volumes of YouTube comments efficiently, extracting actionable insights in the form of sentiment distribution, trending topics, and engagement metrics.

Functional Requirements

- **Sentiment Analysis:** The system must classify YouTube comments into positive, negative, or neutral categories in real time.
- **Trend Tracking:** Monitor changes in audience sentiment over time to understand how new content impacts perception.
- **Comment Summarization:** Generate summaries of the most common themes, suggestions, and criticisms expressed by viewers.
- **Word Cloud Visualization:** Display frequently used keywords and topics in an interactive format to highlight emerging trends.
- **Spam and Troll Detection:** Identify and filter out spam, bot-generated, or harmful comments to improve analysis quality.

- **Export Reports:** Allow influencers to export results and visualizations in formats such as PDF or CSV for further use.
- **Chrome Extension Interface:** Provide a lightweight, user-friendly extension for easy access and visualization of results.

Non-Functional Requirements

- **Scalability:** The system should handle large volumes of comments efficiently without significant latency.
- **Accuracy:** The sentiment classification and trend detection models should achieve reliable performance even with noisy, multilingual, and slang-rich data.
- **Usability:** The interface should be intuitive and interactive, requiring minimal technical expertise from creators.
- **Security and Privacy:** Ensure compliance with data privacy regulations while processing user comments.
- **Maintainability:** The system should support continuous integration and deployment (CI/CD) pipelines for easy updates.
- **Portability:** Deployment must be containerized (e.g., via Docker) to ensure consistent operation across platforms.

Challenges Identified

Based on the problem analysis, the following challenges need to be addressed:

- Availability of clean and labeled datasets for YouTube comments.
- Handling multi-language comments, slang, emojis, and sarcasm.
- Filtering out spam and bot-generated comments.
- Dealing with evolving language usage and data drift (e.g., words with changing sentiment meaning).
- Ensuring low-latency real-time processing.
- Designing a smooth user experience within a Chrome extension.

Thus, the requirement analysis ensures that the system not only delivers sentiment and trend analysis but also addresses the broader needs of both creators and audiences by providing reliable, scalable, and user-friendly insights.

1.3 Main Objective

The main objective of **"Comment ML" – YouTube Sentimental & Trend Analyzer** is to develop an intelligent and automated system that can efficiently analyze large volumes of YouTube comments and provide meaningful insights to both creators and audiences. The project aims to reduce the manual effort required for comment interpretation while ensuring accurate sentiment classification and detection of emerging trends.

More specifically, the objectives include:

- To classify YouTube comments into positive, negative, and neutral categories using Natural Language Processing (NLP) and Machine Learning (ML) techniques.
- To monitor sentiment changes over time and identify how different content influences audience perception.
- To summarize key topics, suggestions, and criticisms expressed by users in the comment section.
- To generate visual insights such as sentiment distribution charts and word clouds for better understanding of audience engagement.
- To detect and filter spam, troll, and bot-generated comments in order to improve the quality of analysis.
- To provide a user-friendly **Chrome Extension** interface for real-time access to sentiment and trend analytics.
- To support influencers, brands, and marketers in making data-driven decisions that improve content strategy, audience interaction, and campaign effectiveness.

In summary, the project's main objective is to bridge the gap between creators and audiences by transforming unstructured YouTube comment data into actionable knowledge, thereby enabling stronger community engagement and smarter decision-making.

1.4 Sub Objectives

To achieve the main objective of **"Comment ML" – YouTube Sentimental & Trend Analyzer**, the following sub-objectives have been identified:

- **Data Collection:** Gather YouTube comments through APIs or scraping methods while ensuring data privacy and compliance.
- **Data Preprocessing:** Clean and prepare raw comment data by handling slang, emojis, special characters, multilingual text, and removing noise such as spam or duplicate entries.
- **Exploratory Data Analysis (EDA):** Understand patterns in comment datasets, detect class imbalances, and identify challenges such as sarcasm or evolving language usage.
- **Model Development:** Build and train machine learning and NLP-based models for sentiment classification and topic extraction.

- **Hyperparameter Tuning & Evaluation:** Optimize models using tools such as Optuna and evaluate performance with standard metrics (accuracy, precision, recall, F1-score).
- **Trend Detection:** Track how audience sentiment and key discussion topics change over time for specific videos or channels.
- **Visualization:** Provide graphical outputs such as sentiment distribution charts, timelines, and word clouds to make analysis more intuitive.
- **Spam/Troll Filtering:** Implement detection techniques to filter irrelevant or harmful comments that distort analysis.
- **System Integration:** Develop a Chrome Extension as the user interface, connected with backend APIs for real-time processing and visualization.
- **Deployment:** Use Docker, GitHub Actions, and AWS services to create a scalable, reliable, and maintainable deployment pipeline.
- **User Accessibility:** Ensure that the system is user-friendly, providing creators with actionable insights and enabling audiences to have their feedback represented fairly.

These sub-objectives collectively ensure that the project not only performs sentiment analysis and trend detection but also delivers a practical, reliable, and user-centered tool for the YouTube ecosystem.

1.5 Pert Chart Legend

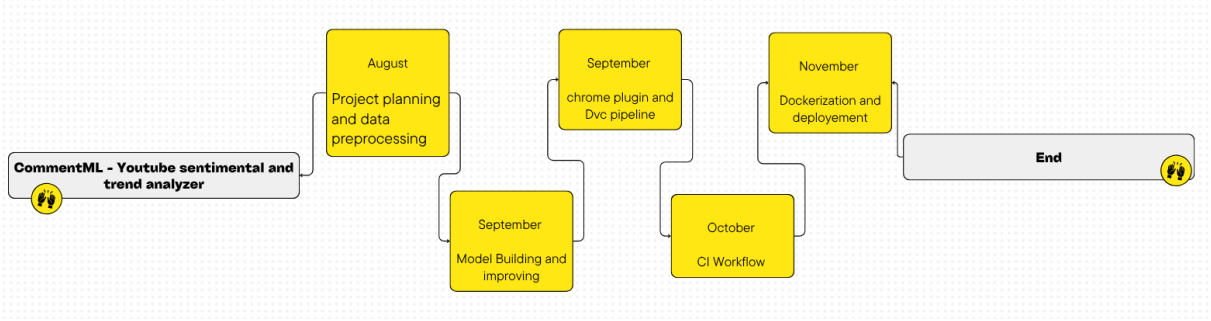

Figure 1.1: Pert Chart Legend

Chapter 2

System Analysis

System analysis is the process of carefully examining the problem domain, identifying the limitations of the existing system, and proposing an improved solution that effectively addresses the identified challenges. It is a crucial step in the software development lifecycle because it lays the foundation for building a system that meets user requirements and performs reliably in real-world conditions.

For **"Comment ML" – YouTube Sentimental & Trend Analyzer**, the focus of the analysis is on the issues faced by content creators, influencers, and audiences on YouTube when dealing with the overwhelming number of comments generated daily. YouTube has emerged as one of the largest social platforms, where billions of users actively engage with video content through likes, shares, and especially comments. These comments often contain valuable insights such as audience appreciation, criticism, suggestions, and discussions of trending topics. However, the sheer volume of data, combined with the unstructured nature of human language, makes it extremely challenging for creators to interpret this information manually.

A detailed system analysis highlights several challenges with the existing platform. YouTube currently offers only very basic tools for comment management, such as sorting by "Top Comments" or "Newest First." These features are insufficient for large creators or marketing teams who need to extract deeper insights. Manually reading through thousands of comments is not only impractical but also highly inefficient. Additionally, spam, bot-generated messages, and troll comments further reduce the quality of feedback and make it difficult to extract useful information. Another limitation is that YouTube does not provide any built-in sentiment analysis, trend detection, or visualization features to help creators understand audience emotions and interests in a structured way.

From a user's perspective, this lack of intelligent comment analysis means that their valuable feedback may go unnoticed, as creators are unable to process every comment effectively. From a creator's perspective, the absence of automated insights leads to missed opportunities in content strategy improvement, audience engagement, and brand collaborations. This gap between audience expression and creator understanding forms the basis of the problem that **"Comment ML"** seeks to solve.

The system analysis further identifies the potential of Natural Language Processing (NLP) and Machine Learning (ML) to address these challenges. Sentiment classification, spam detection, and trend identification are well-studied problems in NLP and can be adapted to the YouTube environment. By integrating these technologies into a Chrome Extension with a simple, intuitive interface, creators can access real-time insights without needing advanced technical knowledge. At the same time, the system ensures that the

opinions of users are properly analyzed, summarized, and reflected in content decisions.

Thus, the system analysis concludes that there is a strong need for an intelligent, automated tool to bridge the gap between creators and their audiences. **"Comment ML" – YouTube Sentimental & Trend Analyzer** directly addresses this gap by transforming unstructured comment data into structured insights, enabling smarter decision-making, stronger engagement, and a healthier digital community.

2.1 Existing System

The existing system for managing and analyzing comments on YouTube is extremely limited. YouTube currently provides only basic options such as sorting comments by **"Top Comments"** or **"Newest First."** While these options help in viewing the most popular or recent comments, they do not provide any analytical insights into the overall audience sentiment or trends within the comment section. For creators with a small number of comments, manual reading may be manageable. However, for influencers, brands, or creators with large followings, the task of going through thousands of comments becomes impractical and highly inefficient.

In the current system, several challenges are observed:

- **Manual Analysis:** Creators must read comments one by one, which is time-consuming and labor-intensive.
- **No Sentiment Analysis:** YouTube does not offer tools to automatically classify comments as positive, negative, or neutral.
- **Lack of Trend Detection:** There is no mechanism to identify recurring themes, trending keywords, or shifts in audience opinion over time.
- **Spam and Troll Comments:** Bots, trolls, and irrelevant comments clutter the comment section, reducing the quality of genuine feedback.
- **No Visualization:** Creators cannot see data in an easily understandable format such as graphs, charts, or word clouds.
- **Language Barriers:** Comments are often written in multiple languages, including slang, emojis, and informal text, which makes manual analysis even harder.

From the user's perspective, the limitations of the existing system mean that their feedback is often overlooked, since creators do not have the capacity to process all comments. From the creator's perspective, the lack of structured analysis prevents them from identifying what resonates with their audience, what improvements are expected, and what kind of content should be prioritized in the future.

Therefore, the existing system can be described as **basic, unstructured, and inadequate** for addressing the needs of modern creators and audiences. This gap creates the strong necessity for an automated, intelligent system like **Comment ML** that can perform large-scale sentiment and trend analysis in an efficient and user-friendly manner.

2.2 Motivations

The motivation for developing **"Comment ML" – YouTube Sentimental & Trend Analyzer** arises from the shortcomings of the existing system and the growing importance of data-driven insights in the digital content ecosystem. YouTube has become one of the most influential platforms worldwide, where millions of creators publish content and billions of users interact daily. Comments play a crucial role in reflecting audience opinions, emotions, and expectations. However, without intelligent tools, the true potential of this valuable data remains untapped.

From the **creators' perspective**, the volume of comments on popular videos can be overwhelming. Manually analyzing thousands of comments is both time-consuming and ineffective, leading to missed opportunities in understanding the audience's reactions. Creators need a solution that can automatically classify sentiments, highlight key themes, and provide visual insights to refine their content strategy, strengthen audience engagement, and respond more effectively to feedback.

From the **users' perspective**, comments are the primary medium to share feedback and interact with creators. However, due to the vast volume and lack of structured analysis, valuable opinions often go unnoticed. A system that can capture, summarize, and present user feedback in an actionable way ensures that their voices contribute meaningfully to content development, thereby improving the overall user experience and fostering stronger creator–audience relationships.

From the **industry perspective**, influencers and marketing agencies rely heavily on audience engagement data to measure campaign effectiveness. Automated sentiment and trend analysis tools provide critical insights for brand collaborations, targeted marketing, and market research.

Finally, from the **academic and technical perspective**, this project is motivated by the opportunity to apply Natural Language Processing (NLP), Machine Learning (ML), and MLOps practices to a real-world scenario. Challenges such as multilingual text, sarcasm, slang, spam detection, and evolving language usage make the problem technically rich and relevant for research and development.

In summary, the motivation behind **Comment ML** is to empower creators with actionable insights, give users a stronger voice, support brands with data-driven analytics, and demonstrate the practical application of modern AI techniques in solving real-world social media challenges.

2.3 Modules

The proposed system **"Comment ML" – YouTube Sentimental & Trend Analyzer** is organized into multiple interconnected modules, each designed to perform a specific task within the comment analysis workflow. Together, these modules form a complete end-to-end system capable of collecting, cleaning, analyzing, and visualizing large-scale YouTube comment data to extract meaningful insights about user sentiment and content trends.

2.3.1 Data Collection Module

This module is responsible for automatically fetching YouTube comments from videos using the YouTube Data API or efficient web scraping techniques. It gathers raw textual data along with associated metadata such as comment ID, video ID, user ID, likes, replies,

and timestamps. The collected dataset serves as the foundation for all subsequent analysis. It also includes mechanisms to periodically update and synchronize data to track ongoing audience feedback.

2.3.2 Data Preprocessing Module

The preprocessing module ensures that the collected data is clean, structured, and ready for Natural Language Processing (NLP). It involves several steps such as removing stopwords, punctuation, and special symbols, converting text to lowercase, and handling multilingual content using language detection and translation APIs. Emojis and internet slang are standardized or converted into meaningful tokens to retain their emotional value. Additionally, this module removes duplicate entries and filters out incomplete or irrelevant comments, ensuring the dataset is of high quality for model training and analysis.

2.3.3 Sentiment Analysis Module

This is the core analytical component of the system. It classifies each processed comment into one of three sentiment categories: **positive**, **negative**, or **neutral**. The module leverages advanced NLP models and supervised machine learning algorithms such as Logistic Regression, LightGBM, and fine-tuned Transformer-based models (e.g., BERT). It extracts linguistic features such as word embeddings and context-aware semantics to achieve high accuracy in sentiment detection. The sentiment results help identify how viewers emotionally respond to specific videos or topics.

2.3.4 Trend Detection Module

This module focuses on understanding the evolving nature of audience engagement. It identifies trending keywords, frequently discussed topics, and patterns in comment activity over time. By applying topic modeling techniques such as Latent Dirichlet Allocation (LDA) and keyword frequency analysis, it reveals how public sentiment shifts in response to new uploads, viral events, or creator announcements. This helps YouTubers and analysts monitor public perception and content performance dynamically.

2.3.5 Visualization Module

To make data interpretation more intuitive, this module generates interactive and graphical representations of analytical results. Using tools such as Matplotlib, Plotly, or Seaborn, it produces word clouds, sentiment distribution graphs, bar charts, and time-series plots. These visualizations allow users to quickly grasp overall audience mood, trending topics, and engagement patterns without diving into raw data.

2.3.6 Spam and Troll Filtering Module

Since online comment sections often contain spam, bots, or harmful messages, this module implements automated filtering techniques. It uses rule-based and ML-based classification methods to detect and remove irrelevant, promotional, or abusive content. Features such as comment frequency, text repetition, sentiment inconsistency, and keyword density are analyzed to ensure that only genuine audience opinions are used in sentiment and trend analysis.

2.3.7 Report Generation Module

This module compiles the analysis results into structured, user-friendly reports. Users can export data and insights in various formats such as PDF, CSV, or Excel for offline usage. The reports include key metrics, sentiment summaries, visual analytics, and trend observations that can aid content creators, marketing professionals, and researchers in strategic decision-making.

2.3.8 Chrome Extension Module

The Chrome Extension serves as the primary user interface for real-time interaction with the system. Users can access sentiment and trend insights directly from YouTube video pages without switching platforms. The extension displays summarized analytics, trending words, and sentiment graphs in an intuitive dashboard format. It ensures high usability and accessibility for end users, allowing both creators and viewers to instantly understand community reactions and content performance.

Chapter 3

Implementation

The implementation of a machine learning model is a multi-stage process that ensures data quality, consistency, and usability for downstream tasks such as feature extraction and model training. The primary stages of this implementation include data acquisition, data cleaning and preprocessing, and text normalization. Each of these stages is essential for the development of a robust sentiment analysis system and directly influences the performance and generalizability of the resulting model.

3.1 Data Acquisition

Data acquisition is the foundational step in any machine learning project, as the quality, size, and relevance of the dataset heavily impact the learning process. For this study, a dataset of Reddit comments was obtained from a publicly available repository on GitHub: `https://raw.githubusercontent.com/Himanshu-1703/reddit-sentiment-analysis/refs/heads/main/data/reddit.csv`. The dataset consists of "37,249 comments", each labeled with a sentiment category.

The dataset contains two primary columns:

```
clean_comment,category
 family mormon have never tried explain them they still stare puzzled from time time like
some kind strange creature nonetheless they have come admire for the patience calmness
equanimity acceptance and compassion have developed all the things buddhism teaches ,1
buddhism has very much lot compatible with christianity especially considering that sin and
suffering are almost the same thing suffering caused wanting things shouldn want going about
getting things the wrong way christian this would mean wanting things that don coincide with
god will and wanting things that coincide but without the aid jesus buddhism could also seen
proof god all mighty will and omnipotence certainly christians are lucky have one such
christ there side but what about everyone else well many christians believe god grace
salvation and buddhism god way showing grace upon others would also help study the things
jesus said and see how buddha has made similar claims such rich man getting into heaven joke
basically advocating that should rid ourselves material possessions fact distinctly
remembered jesus making someone cry because that someone asked what achieve salvation and
jesus replied with live like buddhist very very roughly translated also point out that
buddha rarely spoke anything about god theory personally because knew well enough leave that
jesus and mohamed who came later just remember conflict difference opinion but education can
fun involving and enlightening easier teach something than prove right like intelligent
design ,1
```

Figure 3.1: Preview of the Dataset

clean_comment: This column contains the textual content of Reddit comments. Although preprocessed to some degree, the text still requires additional cleaning and normalization to make it suitable for machine learning models.

category: This column contains sentiment labels where -1 represents negative sentiment, 0 represents neutral sentiment, and 1 represents positive sentiment.

```python
# --- Core Libraries for Data Manipulation ---
import numpy as np
import pandas as pd
import re
# --- Libraries for Data Visualization ---
import matplotlib.pyplot as plt
import seaborn as sns
from wordcloud import WordCloud
from collections import Counter

# --- Libraries for Natural Language Processing (NLP) ---
from nltk.corpus import stopwords
from nltk.stem import WordNetLemmatizer
from sklearn.feature_extraction.text import CountVectorizer
```

Figure 3.2: Importing libraries

```python
df = pd.read_csv('https://raw.githubusercontent.com/Himanshu-1703/reddit-sentiment-analys
df.head()
```

	clean_comment	category
0	family mormon have never tried explain them t...	1
1	buddhism has very much lot compatible with chr...	1
2	seriously don say thing first all they won get...	-1
3	what you have learned yours and only yours wha...	0
4	for your own benefit you may want read living ...	1

Figure 3.3: loading the dataset into a pandas DataFrame, and previewing the first few rows using `.head()`.

Rationale: A well-structured dataset with diverse sentiment examples allows the model to capture subtle patterns in natural language. The inclusion of negative, neutral, and positive comments ensures that the model can distinguish sentiment polarity with higher accuracy. Additionally, using publicly available data ensures reproducibility and provides a benchmark for model performance.

3.2 Data Cleaning and Preprocessing

Raw text data from social media platforms is inherently noisy. It often contains missing values, duplicates, URLs, emojis, newline characters, and inconsistent capitalization, all of which can negatively impact the learning process if not properly handled. To create a dataset suitable for machine learning, the following preprocessing steps were applied.

3.2.1 Handling Missing Values

The first step in preprocessing was to identify and handle missing values. Using `df.info()` and `df.isnull().sum()`, it was observed that 100 entries in the `clean_comment` column were null. Although this represents only 0.27% of the dataset, handling these values is critical to avoid errors during model training.

```
df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 37249 entries, 0 to 37248
Data columns (total 2 columns):
 #   Column         Non-Null Count  Dtype 
---  ------         --------------  ----- 
 0   clean_comment  37149 non-null  object
 1   category       37249 non-null  int64 
dtypes: int64(1), object(1)
memory usage: 582.1+ KB
```

Figure 3.4: Detecting missing values using `.info()`

```
df.isnull().sum()
```

```
clean_comment    100
category           0
dtype: int64
```

Figure 3.5: Detecting missing values using `isnull().sum()`.

Approach and Rationale: Rather than imputing missing comments, which could introduce noise or bias, the null entries were removed using `dropna()`. Removing incomplete data ensures that each training example contains meaningful text, allowing the model to learn effectively without introducing artificial signals that could distort feature extraction.

3.2.2 Removing Duplicate Entries

Duplicate comments can skew the learning process by overrepresenting certain patterns or phrases. Using `.duplicated().sum()`, 350 duplicate rows were detected. These duplicates were removed using `drop_duplicates()`.

```
df.drop_duplicates(inplace=True)
```

Figure 3.6: Identifying and removing duplicate rows to prevent model bias.

Impact on Model Performance: Removing duplicates ensures that the dataset accurately represents the distribution of sentiments. It prevents the model from overfitting to frequently repeated comments and improves its ability to generalize across varied textual data. This is particularly important in NLP tasks, where repeated text can dominate token frequency statistics and bias feature extraction.

3.2.3 Text Normalization

Text normalization ensures uniformity and reduces vocabulary size, which is essential for efficient feature extraction and improved model performance. The following normalization steps were applied:

1. **Lower casing:** Converts all text to lowercase, ensuring that words such as "The" and "the" are treated identically. This reduces redundancy in the vocabulary.
2. **Removing Empty or Whitespace-Only Strings:** Eliminates comments that contain no meaningful content, preventing the introduction of irrelevant data into the model.
3. **Stripping Leading and Trailing Whitespace:** Ensures that tokenization is accurate and that extraneous spaces do not create spurious tokens.
4. **Removing Newline Characters and URLs:** Converts multi-line comments into single-line text and removes web URLs, which do not generally contribute to sentiment information.

```
df['clean_comment'] = df['clean_comment'].str.lower()
df.head()
```

	clean_comment	category
0	family mormon have never tried explain them t...	1
1	buddhism has very much lot compatible with chr...	1
2	seriously don say thing first all they won get...	-1
3	what you have learned yours and only yours wha...	0
4	for your own benefit you may want read living ...	1

Figure 3.7: Converting text to lowercase and removing empty strings.

```
df['clean_comment'] = df['clean_comment'].str.strip()
df['clean_comment'].apply(lambda x: x.endswith(' ') or x.startswith(' ')).sum()
```

Figure 3.8: Stripping leading and trailing whitespace from comments to ensure clean tokenization.

```
comments_with_newline = df[df['clean_comment'].str.contains('\n')]
comments_with_newline.head()
```

	clean_comment	category
448	what missing jpg\nand why this brilliant edit ...	1
781	india has been ruined congress and populist sc...	-1
847	like aap for its stand corruption and making p...	-1
871	reduced trade\ndeficit stronger rupee aren the...	0
1354	amsa press conference australian maritime safe...	1

Figure 3.9: Replacing newline characters with spaces to create single-line comments suitable for analysis.

3.2.4 Advanced Preprocessing Techniques

Beyond basic normalization, several additional preprocessing steps can further enhance the quality of the dataset:

- **Stopword Removal:** Eliminates common words such as "the", "is", and "in" that carry little semantic meaning for sentiment analysis.
- **Lemmatization/Stemming:** Reduces words to their base form (e.g., "running" → "run") to consolidate semantically similar terms and reduce feature dimensionality.
- **Special Character and Emoji Removal:** Eliminates punctuation, emojis, and symbols, which may introduce noise and complicate tokenization.
- **Handling Contractions:** Expands contractions such as "don't" → "do not", improving model understanding of sentiment-bearing words.
- **Part-of-Speech Tagging and Filtering (Optional):** Enables the focus on sentiment-heavy words such as adjectives and verbs, while ignoring function words.

Technical Rationale: Applying advanced preprocessing ensures that the model focuses on meaningful content while reducing irrelevant or noisy data. It also improves the quality of feature extraction methods such as TF-IDF vectorization, word embeddings, or n-gram models, thereby improving the overall sentiment classification performance.

3.3 Tokenization and Feature Extraction Considerations

After cleaning and normalization, the textual data is ready for tokenization and feature extraction. Tokenization converts text into discrete units (words or subwords), which serve as the basic features for machine learning models. Proper preprocessing significantly impacts the quality of extracted features:

- **Reduced Vocabulary Size:** Normalization ensures that words with similar meaning or capitalization are treated identically, reducing the total number of unique tokens.
- **Improved Semantic Consistency:** Lemmatization and stopword removal enhance the quality of tokenized data, allowing embeddings or vector representations to capture sentiment more accurately.
- **Noise Reduction:** Removing URLs, emojis, and special characters prevents irrelevant tokens from skewing feature weights in models such as TF-IDF or word2vec.

3.4 Exploratory Data Analysis (EDA)

Exploratory Data Analysis (EDA) is a crucial step in any data-driven project. It provides a comprehensive understanding of the dataset, uncovers hidden patterns, and guides feature engineering and model selection. In this project, after initial data cleaning and normalization, we performed a detailed EDA to examine various textual and statistical properties of Reddit comments. Our analysis focused on sentiment distribution, comment length, stop words, characters, punctuation, word frequencies, and n-grams.

3.4.1 Sentiment Category Distribution

Understanding the distribution of the target variable is essential for modeling decisions. The 'category' column represents sentiment polarity with three classes: -1 (negative), 0 (neutral), and 1 (positive). By plotting the distribution of these categories, we observed a moderate class imbalance, with positive comments forming the largest portion of the dataset.

This insight indicates the necessity of considering class imbalance techniques such as weighted loss functions or oversampling during model training to ensure that neutral and negative comments are properly represented.

3.4.2 Comment Length Analysis

Comment length is a valuable feature in sentiment analysis as longer comments may contain more sentiment-bearing words, while very short comments may be ambiguous. We created a 'word_count' feature to quantify the number of words in each comment.

Visualizing the distribution of comment lengths revealed a right-skewed pattern, with most comments being short, but a small proportion extending to very long comments.

A boxplot further confirmed the presence of numerous outliers—long comments that deviate significantly from the majority.

Segmenting by sentiment revealed that neutral comments tend to be shorter, while positive and negative comments are slightly longer, indicating that users often elaborate when expressing strong sentiment.

Finally, a bar plot of median word counts per category confirmed that neutral comments have the lowest median length, while negative and positive comments are slightly longer, indicating more expressive content.

```
sns.countplot(data=df,x="category")
```

```
<Axes: xlabel='category', ylabel='count'>
```

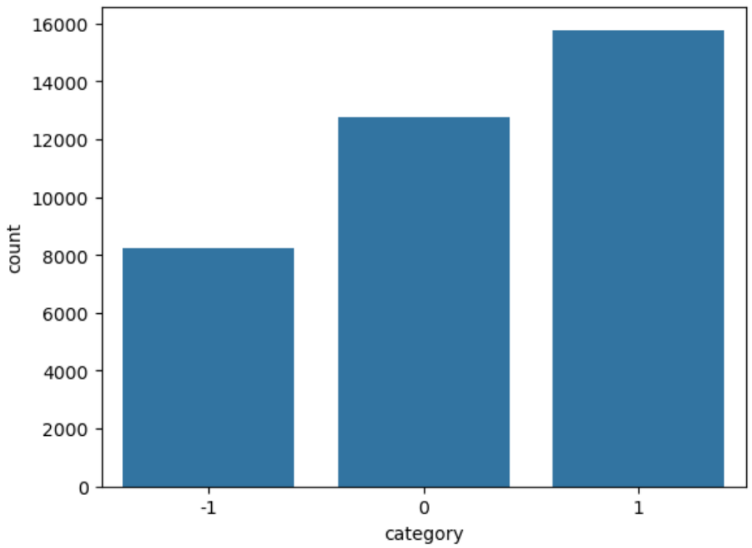

Figure 3.10: Distribution of sentiment categories, highlighting a higher frequency of positive comments.

```
df['word_count'].describe()
```

```
count    36793.000000
mean        29.667464
std         56.790738
min          1.000000
25%          6.000000
50%         13.000000
75%         30.000000
max       1307.000000
Name: word_count, dtype: float64
```

Figure 3.11: Creation and statistical summary of the ‘word_count‘ feature.

3.4.3 Stop Words, Characters, and Punctuation

Stop words, punctuation, and character counts offer additional insights into text structure. Stop words are common words such as “the”, “and”, or “that”, which often do not carry sentiment but contribute to sentence structure.

We created a ‘num_stop_words‘ feature and analyzed its distribution.

The distribution by category mirrored the word count patterns, confirming that longer comments tend to contain more stop words.

3.4.4 Word Frequencies and N-gram Analysis

N-gram analysis revealed commonly occurring bigrams and trigrams, offering insight into frequently co-occurring word patterns.

These EDA insights provide a comprehensive understanding of the dataset, revealing

```
plt.figure(figsize=(10, 6))
sns.histplot(df['num_stop_words'], kde=True)
plt.title('Distribution of Stop Word Count in Comments')
plt.xlabel('Number of Stop Words')
plt.ylabel('Frequency')
plt.show()
```

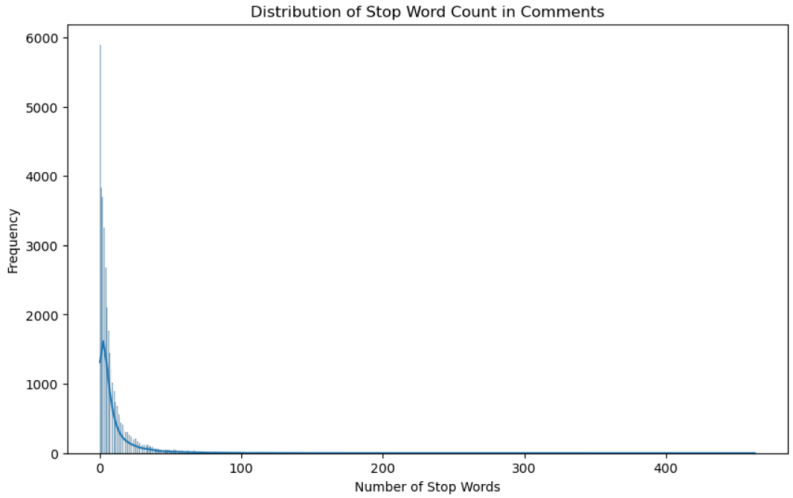

Figure 3.12: Distribution plot of ‘word_count‘ showing right-skewness in comment length.

```
sns.boxplot(df['word_count'])
```

```
<Axes: ylabel='word_count'>
```

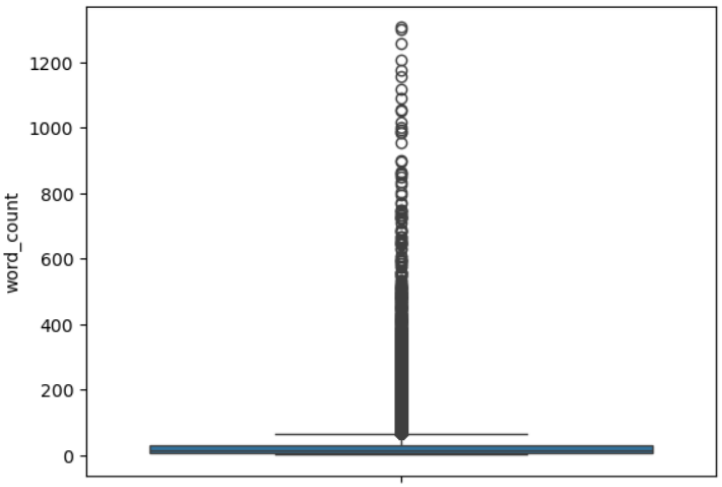

Figure 3.13: Boxplot of ‘word_count‘ highlighting outliers in comment length.

patterns in sentiment, comment length, stop words, character distribution, and common word/phrase usage. Such analysis is invaluable for guiding feature engineering, preprocessing decisions, and modeling strategies.

can u foramte thsi

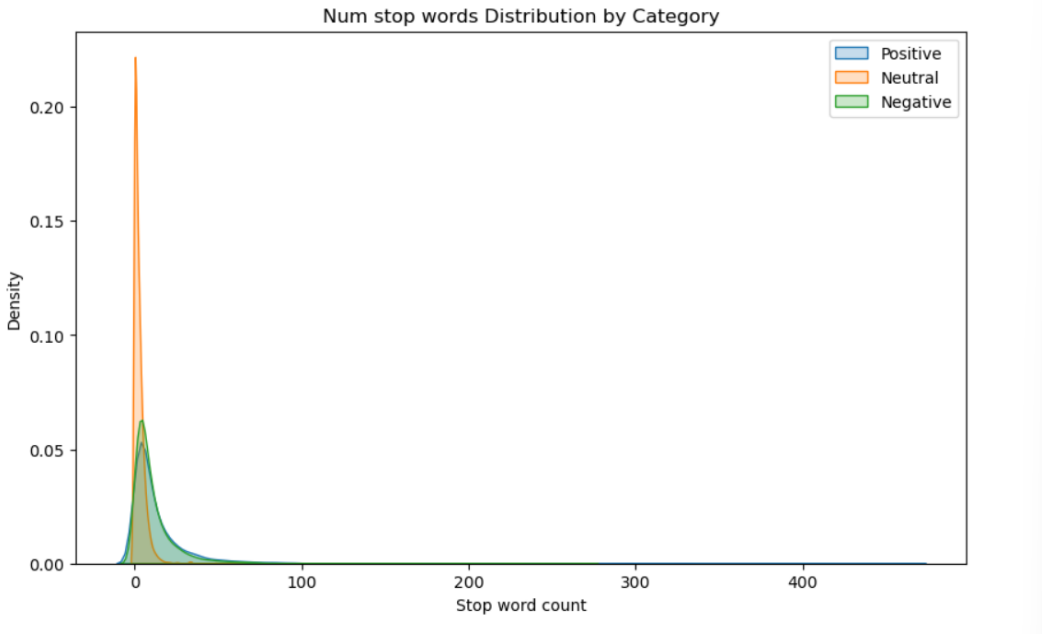

Figure 3.14: Density plot of ‘word_count‘ by sentiment category. Neutral comments are typically shorter.

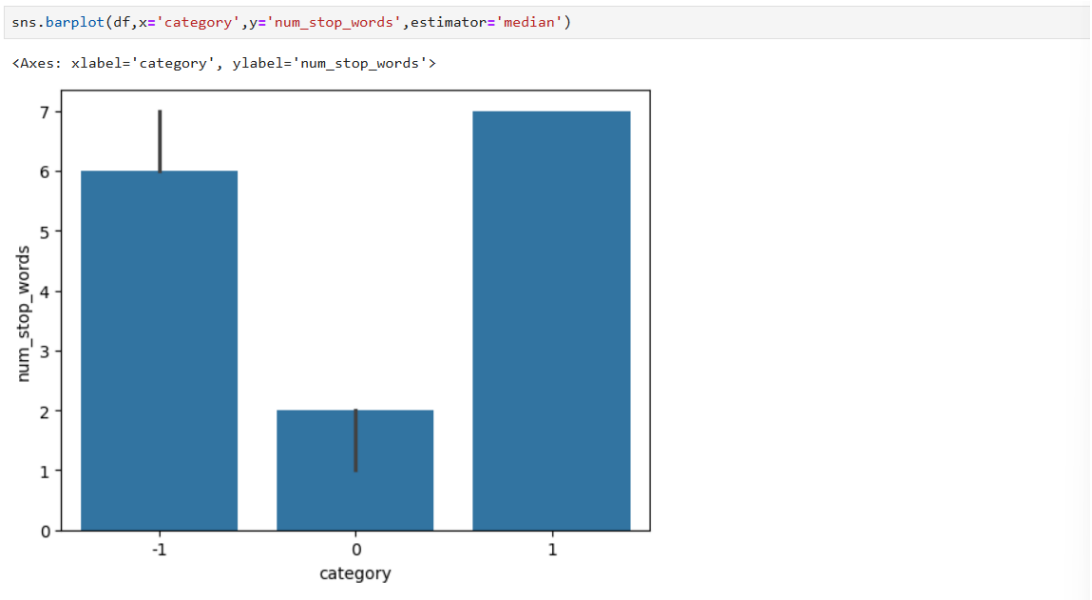

Figure 3.15: Boxplot of ‘word_count‘ by category showing differences in median and spread.

```
df['num_chars'] = df['clean_comment'].apply(len)

df.head()
```

	clean_comment	category	word_count	num_stop_words	num_chars
0	family mormon have never tried explain them th...	1	39	13	259
1	buddhism has very much lot compatible with chr...	1	196	59	1268
2	seriously don say thing first all they won get...	-1	86	40	459
3	what you have learned yours and only yours wha...	0	29	15	167
4	for your own benefit you may want read living ...	1	112	45	690

Figure 3.16: Median ‘word_count‘ for each sentiment category, providing a simple summary of comment length differences.

```
all_text = ' '.join(df['clean_comment'])
char_frequency = Counter(all_text)
char_frequency_df = pd.DataFrame(char_frequency.items(),
                                 columns=['character', 'frequency']).sort_values(by='frequency', ascending=False)
char_frequency_df['character'].values
```

```
array([' ', 'e', 't', ..., '段', '她', '谁'], dtype=object)
```

Figure 3.17: Creation of ‘num_stop_words‘ feature.

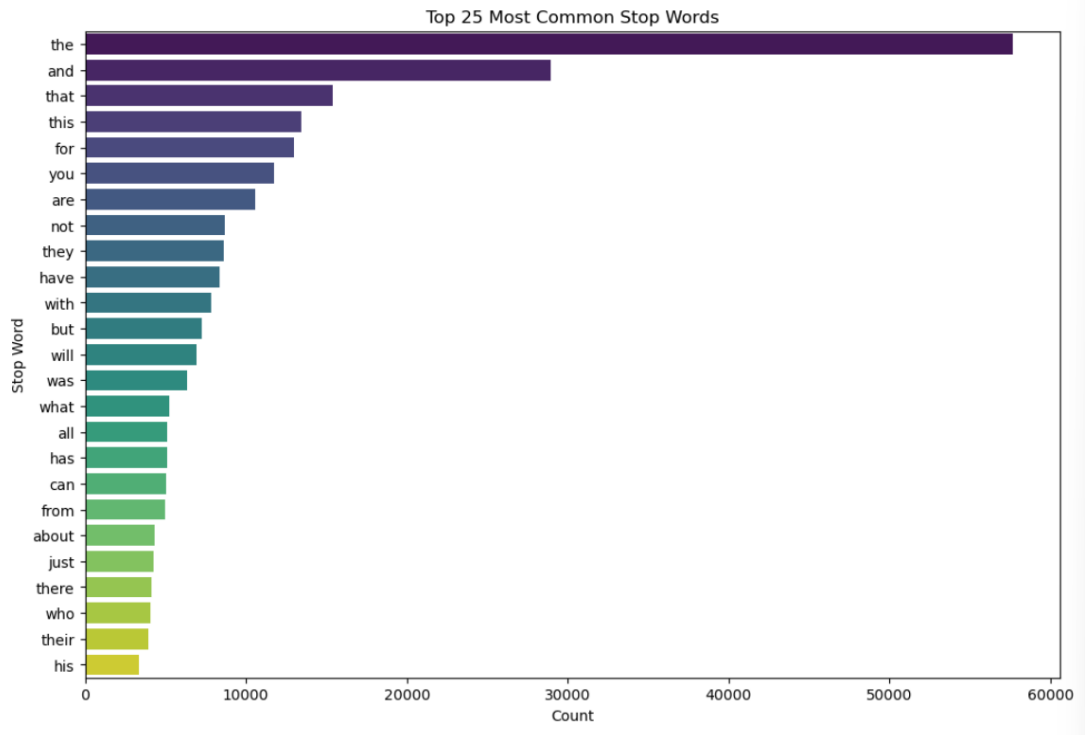

Figure 3.18: Top 25 Most Common words

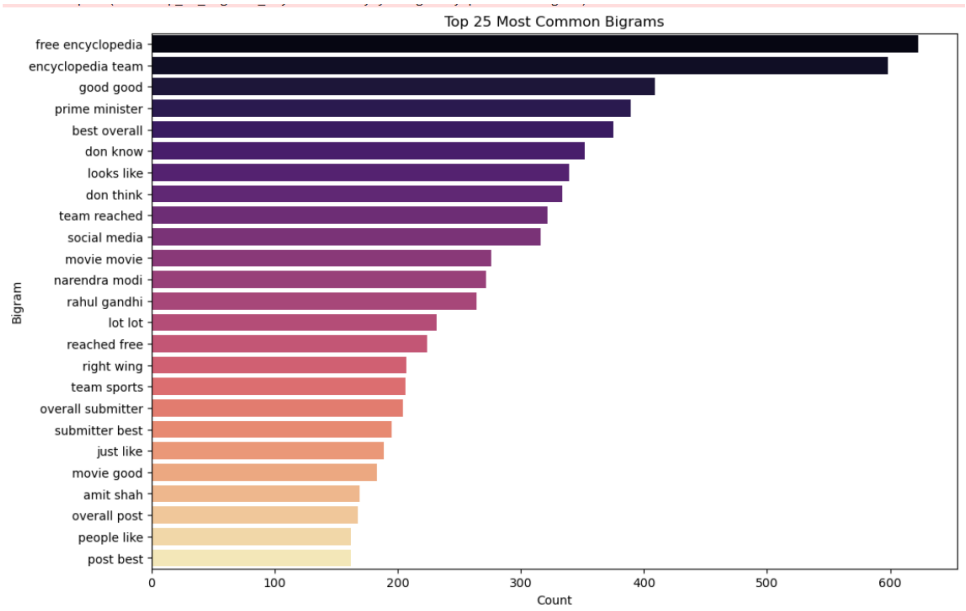

Figure 3.19: Top 25 most common bigrams in the dataset.

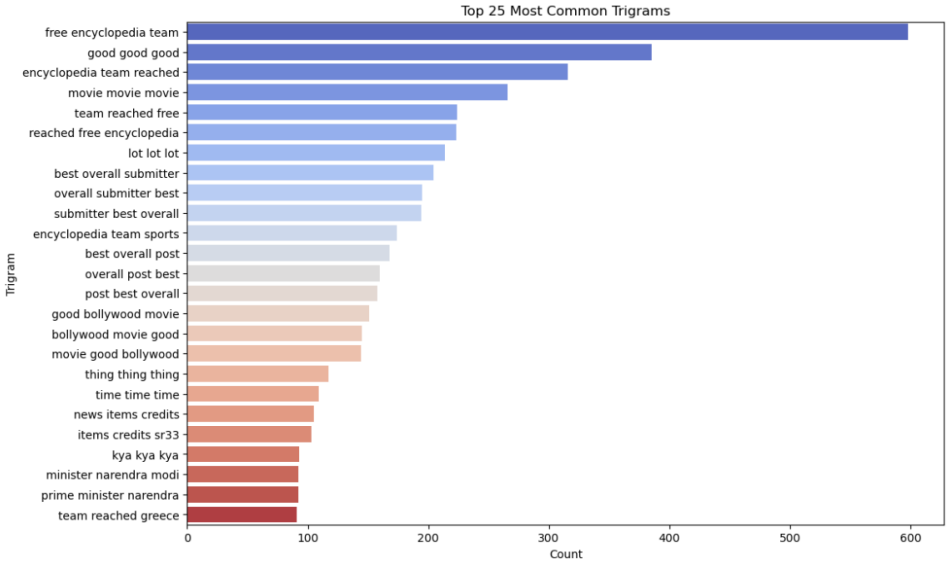

Figure 3.20: Top 25 most common trigrams in the dataset.

Chapter 4

Conclusion

...

Bibliography

[1] Sui, X. (2022). *Measurement and Sentiment Analysis of YouTube Video Comments.*

[2] Aiswarya, A. S., & Rajeev, H. (2024). *YouTube Comment Sentimental Analysis.*

[3] Möller, A., et al. (2024). *A Supervised Machine Learning Approach to Identifying Relevant YouTube Comments.*

[4] Hoiles, W., Krishnamurthy, V., & Pattanayak, K. (2019). *Rationally Inattentive Inverse Reinforcement Learning Explains YouTube Commenting Behavior.*

[5] Shajari, S., Agarwal, N., & Alassad, M. (2023). *Commenter Behavior Characterization on YouTube Channels.*

[6] Chakravarthi, B. R., Muralidaran, V., Priyadharshini, R., & McCrae, J. P. (2020). *Corpus Creation for Sentiment Analysis in Code-Mixed Tamil-English Text.*

[7] Bhattacharya, S., Singh, S., Kumar, R., et al. (2020). *Developing a Multilingual Annotated Corpus of Misogyny and Aggression.*