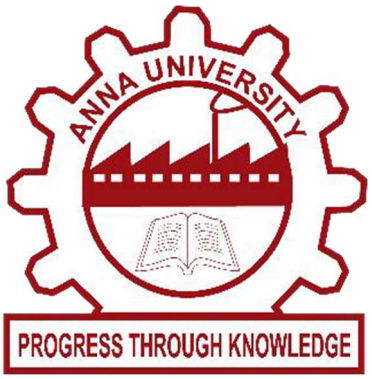
MINI PROJECT REPORT

MACHINE LEARNING TECHNIQUES LABORATORY

(CA5314)

IMPLEMENTING FAIR AND ETHICAL AI IN REAL-WORLD SYSTEMS

[YOUTUBE VIDEO CLASSIFICATION APP]



DEPARTMENT OF INFORMATION SCIENCE AND TECHNOLOGY

COLLEGE OF ENGINEERING, GUINDY

ANNA UNIVERSITY, CHENNAI

SUBMITTED BY:

SAI KIRAN D V (2022179033)

SARINIKA U (2022179034)

MANIGANDAN R (2022179052)

COURSE: MCA SS

SUBMITTED TO:

DR. M. DEIVAMANI

(ASSISTANT PROFESSOR DIST)

ACKNOWLEDGEMENT

I would like to express my heartfelt gratitude to all the individuals who supported and contributed to the successful completion of this mini-project. I extend my sincere thanks to my college faculty and mentors for their guidance, valuable insights, and encouragement throughout the project.

I would also like to thank my friends and classmates for their collaboration and brainstorming sessions, which helped in refining the project's concept and design.

Furthermore, I express my gratitude to the developers and authors of the open-source libraries, tools, and resources used in this project. Their contributions have significantly enriched the functionality and performance of the application.

Lastly, I am deeply indebted to my family for their unwavering love, understanding, and support throughout this endeavor. Their belief in me served as a constant source of motivation and inspiration.

This project would not have been possible without the collective efforts and support of all the individuals mentioned above. Thank you for being a part of this journey and helping me turn my vision into reality.

TABLE OF CONTENTS

ABSTRACT

The YouTube Video Classification App is a mini-project designed to provide users with insights into the sentiment and explicit content of YouTube videos. Built using the Streamlit framework and leveraging various external libraries, the app allows users to input a YouTube video URL, choose the subtitles language, and analyze the video's content.

Key Features:

- **User Interaction:** The app features a user-friendly interface with tabs for Home, Classification, and About. Users can enter a YouTube video URL in the Classification tab and select the desired subtitles language.
- **Sentiment Analysis:** The app employs the Natural Language Toolkit (NLTK) library to perform sentiment analysis on the transcribed content of the provided YouTube video. The sentiment is categorized as positive, neutral, or negative, and the distribution is visualized through pie charts.
- **Word Cloud Generation:** The app generates a word cloud using the WordCloud library, visually representing the most frequently used words in the video transcript. This provides a quick overview of the video's content.
- **Explicit Content Analysis**: The Profanity Check library is utilized to analyze explicit content within the video transcript. An explicit content score is calculated, and the content is classified as either explicit or non-explicit based on a predefined threshold.
- **Data Visualization:** Matplotlib is employed to create interactive visualizations, including pie charts for sentiment distribution, bar charts for sentiment word counts, and word clouds for a graphical representation of the content.
- **YouTube API Integration:** The app integrates with the YouTube API to fetch additional information about the video, such as its title, description, thumbnail URL, and subtitles.

The YouTube Video Classification App is a powerful tool for content creators, educators, and users interested in gaining insights into the emotional tone and content nature of YouTube videos. Its modular design and integration with external libraries make it a versatile and expandable solution for natural language processing and content analysis

INTRODUCTION

In the dynamic landscape of online content, YouTube serves as a colossal platform with a multitude of videos spanning various genres. With the exponential growth of content, the need for effective tools to analyze and classify videos becomes crucial. The YouTube Video Classification App is an innovative solution designed to meet this need, leveraging natural language processing (NLP) techniques to assess sentiments and identify explicit content within video transcripts. As a testament to responsible AI development, the app is committed to ethical considerations, ensuring a positive and inclusive online environment for users.

As a user-friendly web application, the YouTube Video Classification App aims to empower users, including content creators, moderators, and viewers, with insights into the emotional tone and appropriateness of YouTube videos. Leveraging sentiment analysis and explicit content detection, the app provides a systematic approach to content evaluation, contributing to a safer and more regulated online environment.

BACKGROUND

The proliferation of content on YouTube brings forth challenges related to content moderation and ensuring compliance with community guidelines. The YouTube Video Classification App emerges as a response to these challenges, offering a systematic approach to content evaluation by combining sentiment analysis and explicit content detection.

Ethical considerations play a central role in the development of the app. The ethical use of AI, aligned with principles of fairness, transparency, and accountability, is embedded in its core design. As the online space becomes more diverse, addressing biases and promoting inclusivity becomes imperative. The app integrates Ethical AI practices to ensure that content evaluation is conducted without perpetuating societal biases or discriminating against specific groups.

Key Features:

Sentiment Analysis: The application utilizes the Natural Language Toolkit (NLTK) library to perform sentiment analysis on the transcribed text of a YouTube video. Sentiment analysis helps determine whether the overall sentiment of the video is positive, negative, or neutral.

Explicit Content Analysis: The app incorporates the Profanity Check library to assess the likelihood of explicit or inappropriate content within the video transcript. It classifies the content based on a predefined threshold, allowing users to identify videos that may violate community guidelines.

Visualization: The app provides visualizations in the form of pie charts and bar graphs to represent the sentiment distribution and counts of sentiment-related words in the video transcript. This visual representation enhances the user's understanding of the emotional tone of the video.

Word Cloud Generation: The application generates a word cloud based on the video transcript, offering a visually appealing representation of frequently used words. This feature provides a quick glance at the prominent themes or topics discussed in the video.

Video Details: The app fetches and displays key details about the main video, including its title, description, thumbnail URL, and subtitles. This information allows users to gain context about the content they are analyzing.

Technologies Used:

Streamlit: The user interface is built using Streamlit, a Python library for creating web applications with minimal code.

NLTK (Natural Language Toolkit): NLTK is employed for sentiment analysis, helping to assess the emotional tone of the video transcript.

Matplotlib: Matplotlib is used for creating visualizations, including pie charts and bar graphs.

YouTube API: The application interacts with the YouTube API to retrieve details about the main video, such as title, description, and thumbnail URL.

Profanity Check: Profanity Check is utilized for analyzing explicit content within the video transcript.

Ethical AI as a Solution:

Fair and Inclusive Analysis: The sentiment analysis and explicit content detection modules within the app are carefully designed to minimize biases. The NLP models are trained on diverse datasets to ensure that the analysis remains fair and inclusive, avoiding discrimination based on race, gender, or other sensitive attributes.

Transparency in Content Moderation: The app prioritizes transparency in its content moderation processes. Users are provided with clear insights into how sentiments are assessed and how explicit content is identified. Transparent AI mechanisms contribute to user trust and understanding of the analysis results.

User Empowerment and Consent: The app places a strong emphasis on user empowerment. Users are informed about the analysis process, and their consent is sought before classifying videos. This approach ensures that users have control over the analysis of content that may impact their online experience.

Continuous Monitoring and Improvement: Ethical AI goes beyond initial development; it involves continuous monitoring and improvement. The app is designed to undergo regular updates, incorporating feedback and addressing emerging ethical considerations in AI. This commitment ensures the app evolves with changing norms and standards.

DATA AND APPROACH USED

Data Preparation and Approach used for YouTube Video Classification App:

The data preparation for the YouTube Video Classification App involves a combination of YouTube API interactions, natural language processing (NLP) techniques, and ethical considerations. Below is a detailed overview of the data preparation and the approach used for the mini-project:

1. YouTube API Data Retrieval:

Objective: Obtain essential details about the main video.

Approach:

- Use the YouTube API developer key to build a connection with the YouTube API.
- Extract video details such as title, description, thumbnail URL, and subtitles.
- Utilize the obtained data for contextualizing the video within the application.

2. Transcription Extraction:

Objective: Retrieve the transcript of the main video.

Approach:

- Leverage the **YouTubeTranscriptApi** to fetch the transcript based on the video ID.
- Filter out non-text elements such as background music annotations ('[Music]').
- Concatenate the transcribed text into a cohesive script for further analysis.

3. Language Processing and Tokenization:

Objective: Prepare the transcribed text for sentiment analysis and explicit content detection.

Approach:

- Tokenize the text into sentences or words for further analysis.
- Ensure appropriate preprocessing, removing irrelevant characters or symbols.
- Prepare the tokenized text for input into NLP models.

4. Profanity Check Data Formatting:

Objective: Structure the transcribed text for explicit content analysis using Profanity Check.

Approach:

- Format the transcribed text for input into the Profanity Check library.
- Ensure that the text is correctly processed to identify explicit language or content.

5. Data Cleaning and Validation:

Objective: Ensure the quality and reliability of the data for accurate analysis.

Approach:

- Perform data cleaning to handle missing or inconsistent data.
- Validate the accuracy of the transcript, ensuring that it aligns with the actual content of the video.
- Handle potential errors or discrepancies in the retrieved data.

6. Sentiment Analysis Labels:

Objective: Prepare labels for sentiment analysis results.

Approach:

- Use the **SentimentIntensityAnalyzer** from NLTK to assign sentiment scores to sentences or words.
- Establish a threshold for classifying sentiments (positive, neutral, negative) based on the sentiment scores.

7. Explicit Content Classification Labels:

Objective: Define labels for explicit content analysis results.

Approach:

- Utilize the explicit content scores obtained from Profanity Check.
- Apply a threshold to classify content as explicit or non-explicit.

8. Data Visualization Labels:

Objective: Prepare labels for visualizations such as sentiment pie charts and word clouds.

Approach:

- Aggregate data for sentiment distribution and word frequency analysis.
- Structure data in a format suitable for generating visualizations using Matplotlib and WordCloud libraries.

9. User Input Handling:

Objective: Handle user input for the YouTube video URL and subtitles language.

Approach:

- Implement user input validation to ensure the proper format of the YouTube video URL.
- Provide options for selecting subtitles language and handle user selections appropriately.

10. Ethical AI Considerations:

Objective: Ensure ethical handling of user data and analysis results.

Approach:

- Implement consent mechanisms to inform users about the analysis process and seek their approval.
- Avoid storing sensitive user data unless explicitly required.
- Regularly review and update data handling practices to align with evolving ethical standards.

By following this comprehensive data preparation approach, the YouTube Video Classification App ensures that the subsequent analysis is based on reliable, cleaned, and appropriately formatted data. The approach also integrates ethical considerations, promoting responsible AI practices in the context of content analysis.

ARCHITECTURE OF THE APPLICATION

3.1 ARCHITECTURE OF THE APPLICATION – YOUTUBE VIDEO CLASSIFICATION APP

Application Architecture:

The YouTube Video Classification App follows a structured architecture that incorporates data retrieval from the YouTube API, natural language processing (NLP) for sentiment analysis and explicit content detection, and user interaction through a Streamlit web interface.

3.2 C4 Model:

Level 1: System Context Diagram

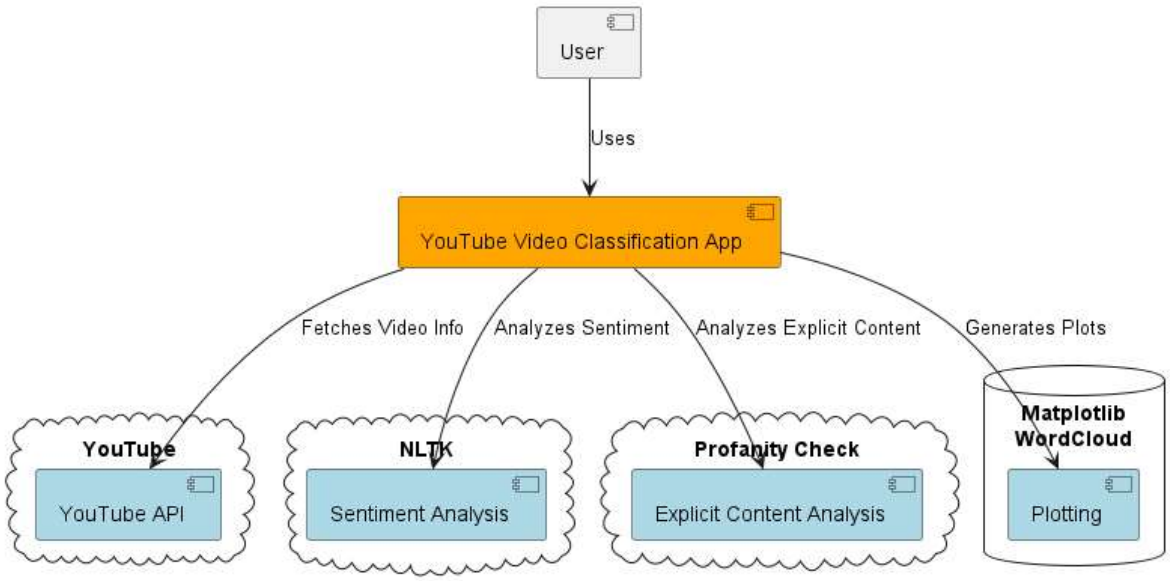


Figure 1 Context Diagram

1. App:

- The central component, representing the YouTube Video Classification App.
- Orchestrates the entire process of video analysis.

2. User:

- Represents the end-user who interacts with the app.

3. YouTube API:

- Cloud icon representing the YouTube API, responsible for fetching essential details about the main video.

4. NLTK (Natural Language Toolkit):

- Cloud icon for NLTK, specifically for sentiment analysis.
- Analyzes the sentiments conveyed in the video transcript.

5. Profanity Check:

- Cloud icon for the Profanity Check library, specialized in explicit content analysis.
- Assesses the likelihood of explicit language in the video transcript.

6. Matplotlib and WordCloud:

- Database icon for Matplotlib and WordCloud, responsible for generating plots and visualizations.
- Creates pie charts, bar graphs, and word clouds based on the analysis results.

Interactions:

1. The App interacts with the YouTube API to fetch information about the main video.
2. The App utilizes NLTK for sentiment analysis and Profanity Check for explicit content analysis.
3. The results of the analysis are then passed to the Plotting component (Matplotlib and WordCloud) to generate visualizations.
4. The User interacts with the App, triggering the entire process.

Level 2: Container Diagram

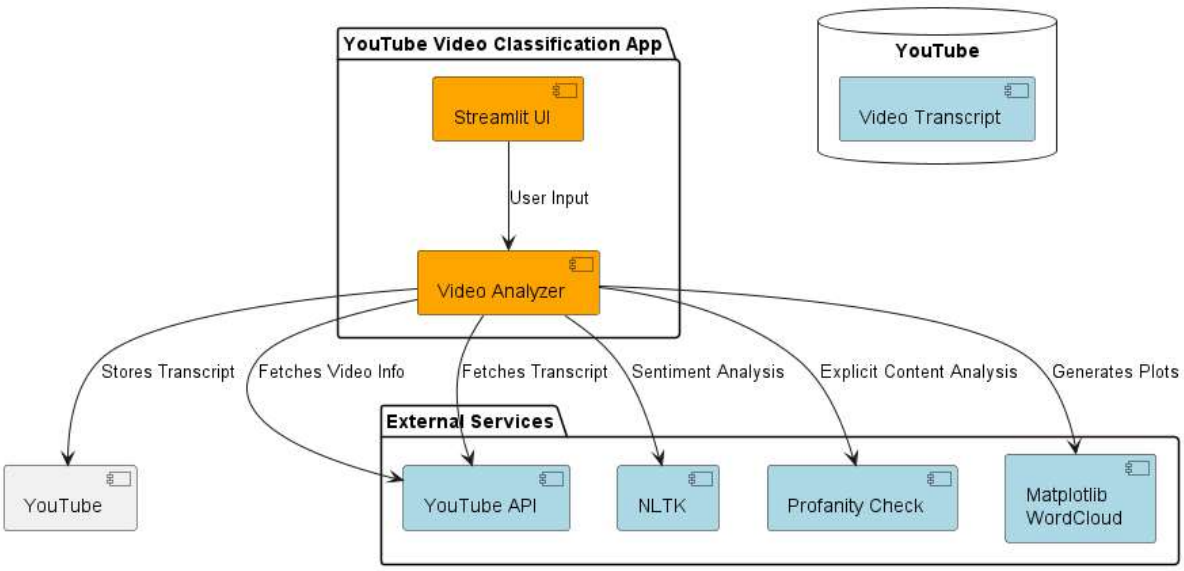


Figure 2 Container Diagram

1. YouTube Video Classification App:

Contains two main components:

- **Streamlit UI (UI):** Represents the user interface where users interact with the application.
- **Video Analyzer (Analyzer):** The core of the application responsible for analyzing YouTube videos.

2. External Services:

A package containing external services crucial for video analysis.

- **YouTube API (YouTubeAPI):** Connects to YouTube to fetch video information and transcripts.
- **NLTK (Natural Language Toolkit):** Performs sentiment analysis on the video transcript.
- **Profanity Check (ProfanityCheck):** Analyzes the video transcript for explicit content.
- **Matplotlib and WordCloud (Plotting):** Responsible for generating visual plots based on the analysis results.

3. YouTube Database:

- Represents the storage for video transcripts obtained from YouTube.

Interactions:

The user interacts with the application through the Streamlit UI, providing input.

The Video Analyzer coordinates the analysis process:

- Utilizes the YouTube API to fetch essential video information and transcripts.
- Performs sentiment analysis using NLTK.
- Analyzes explicit content using Profanity Check.
- Generates visual plots using Matplotlib and WordCloud.
- Stores the transcript in the YouTube Database.

Level 3: Component Diagram

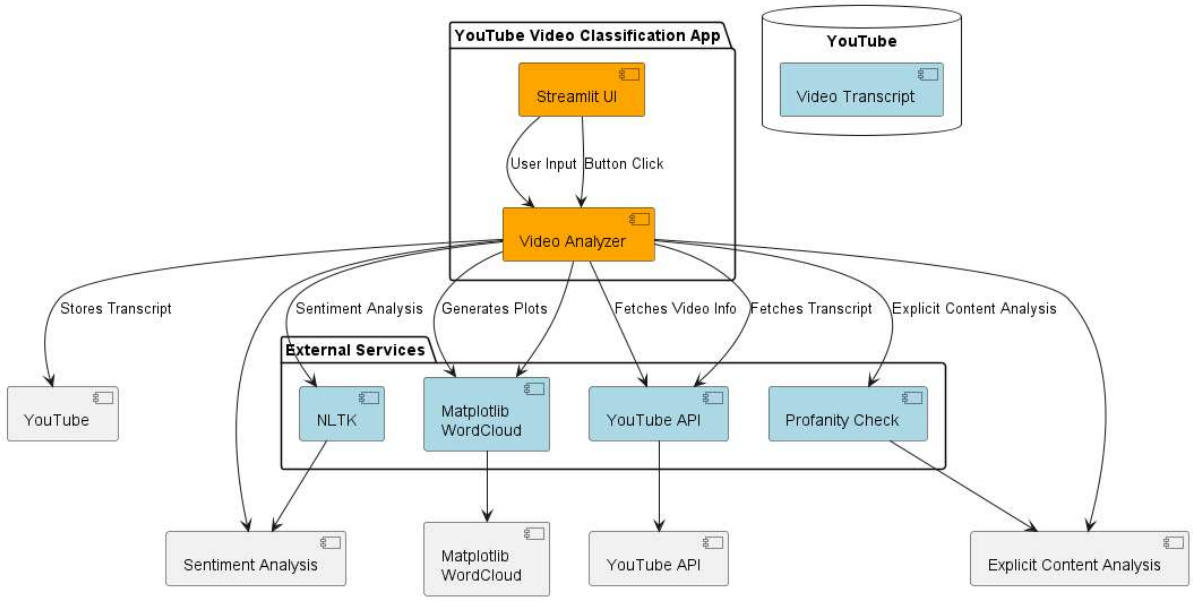


Figure 3 Component Diagram

1. YouTube Video Classification App:

Contains two primary components:

- **Streamlit UI (UI):** Represents the user interface where users interact with the application.
- **Video Analyzer (Analyzer):** The core of the application responsible for coordinating video analysis.

2. External Services:

A package comprising external services essential for video analysis.

- **YouTube API (YouTubeAPI):** Connects to YouTube to fetch video information and transcripts.
- **NLTK (Natural Language Toolkit):** Conducts sentiment analysis on the video transcript.
- **Profanity Check (ProfanityCheck):** Analyzes the video transcript for explicit content.
- **Matplotlib and WordCloud (Plotting):** Generates visual plots based on the analysis results.

3. YouTube Database:

- Represents the storage for video transcripts obtained from YouTube.

Interactions:

User Interaction:

- Users interact with the application through the Streamlit UI, providing input.
- The "Button Click" event triggers the analysis process.

Video Analysis Process (Within Analyzer):

The Video Analyzer coordinates the analysis process:

- Utilizes the YouTube API to fetch essential video information and transcripts.
- Performs sentiment analysis using NLTK.
- Analyzes explicit content using Profanity Check.
- Generates visual plots using Matplotlib and WordCloud.
- Stores the transcript in the YouTube Database.

External Services Interactions:

Components within the Analyzer package interact with their respective external services:

- YouTubeAPI fetches video information.
- NLTK performs sentiment analysis.
- ProfanityCheck analyzes explicit content.
- Plotting generates visual plots.

Level 4: Code Diagram

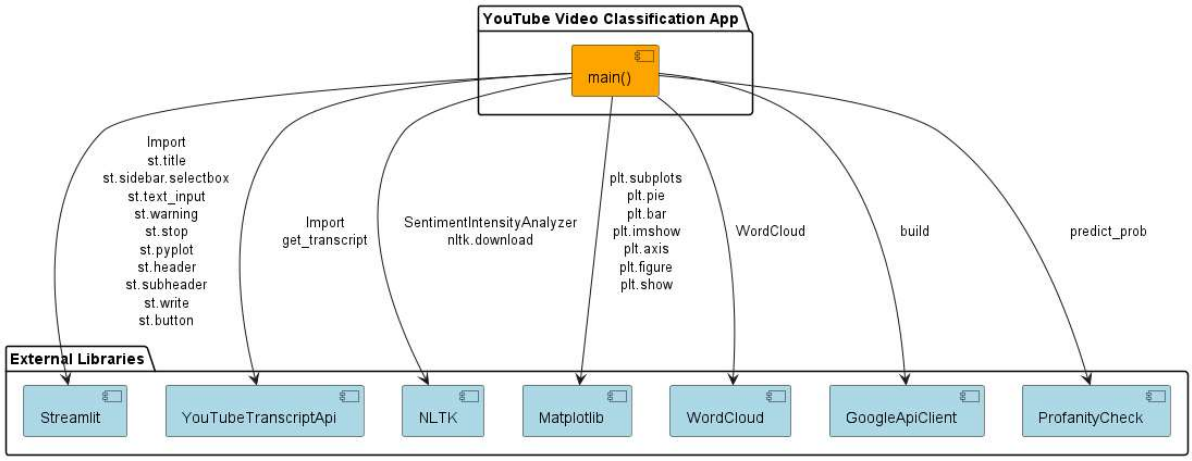


Figure 4 Code Diagram

1. YouTube Video Classification App:

Contains a single component representing the main application logic:

- **Main():** The main function or module responsible for orchestrating the entire application.

2. External Libraries:

A package comprising external libraries and modules used by the application.

Streamlit: A web application framework used for creating the user interface.

- Methods imported for handling titles, input, warnings, and plots.

YouTubeTranscriptApi: An API for fetching YouTube video transcripts.

- The get_transcript method is utilized.

NLTK (Natural Language Toolkit): A library for natural language processing.

- The SentimentIntensityAnalyzer is used for sentiment analysis.
- nltk.download is used for downloading required resources.

Matplotlib: A plotting library for creating visualizations.

- Methods like plt.subplots, plt.pie, plt.bar, plt.imshow, plt.axis, plt.figure, and plt.show are employed.

WordCloud: A library for generating word clouds.

- The WordCloud class is imported.

GoogleApiClient: Part of the Google API client library for interacting with Google services.

- The build method is used to set up the YouTube API client.

ProfanityCheck: A library for predicting the probability of profanity in text.

- The predict_prob method is imported.

Interactions:

- The **Main()** module orchestrates the application logic and interacts with various external libraries to achieve different functionalities.
- Streamlit is used for creating the application's user interface and handling user inputs.
- External libraries like YouTubeTranscriptApi, NLTK, Matplotlib, WordCloud, GoogleApiClient, and ProfanityCheck are imported and utilized to fetch data, perform sentiment analysis, generate visualizations, and analyze explicit content.

RESULTS

IMPLEMETATION OF THE APPLICATION

4.1 HOME PAGE

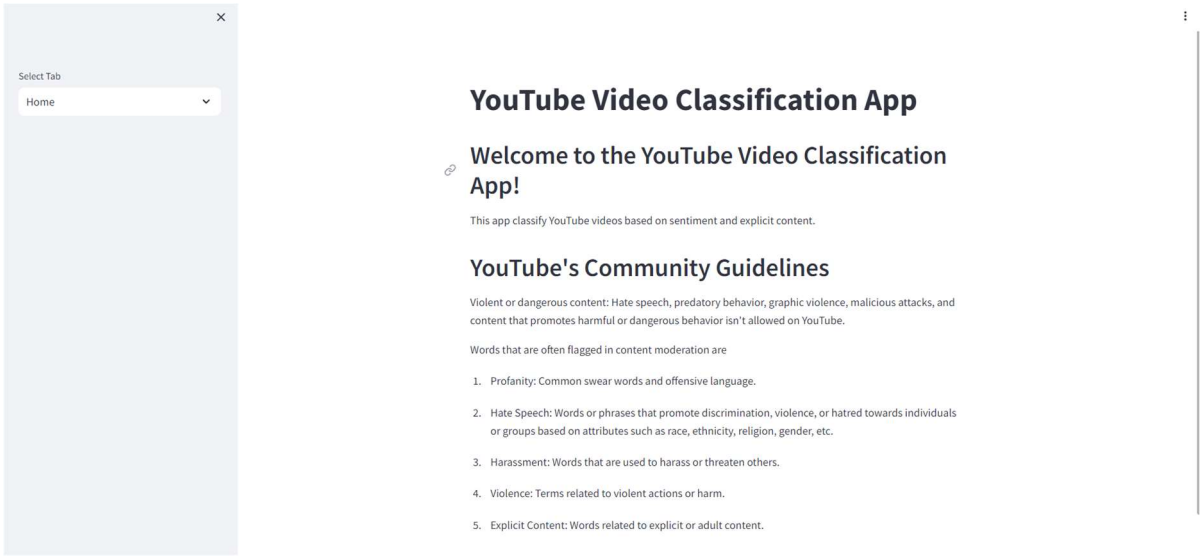


Figure 5 Home Page

4.2 ABOUT PAGE

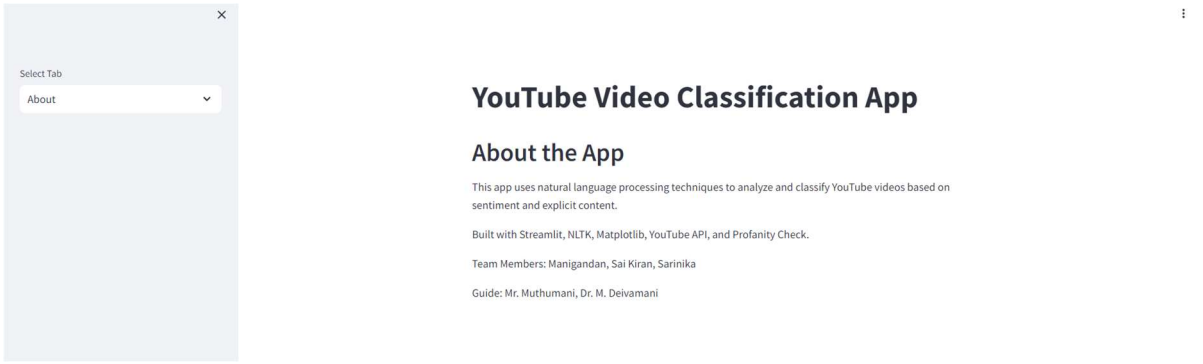


Figure 6 About Page

4.3 CLASSIFICATION PAGE

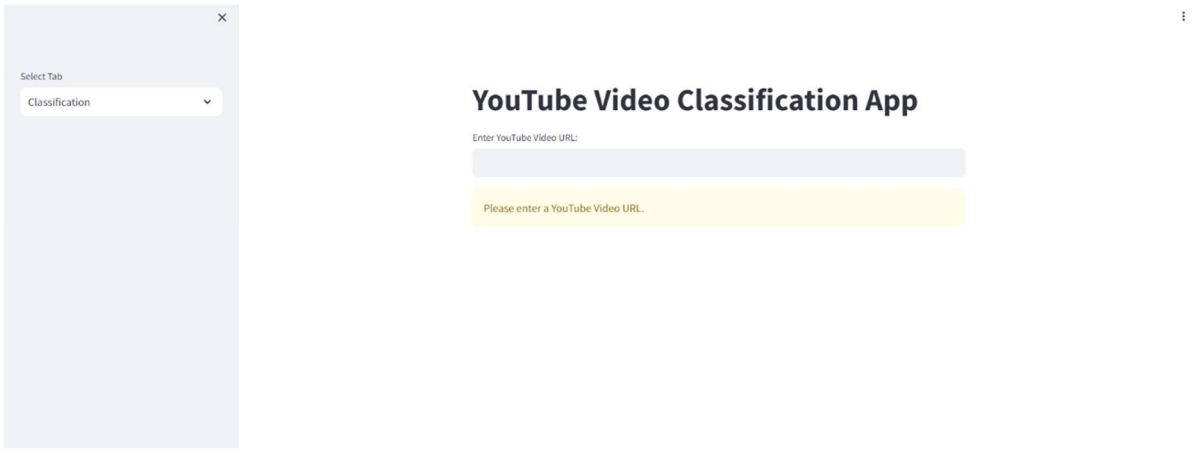


Figure 7 Classification Page

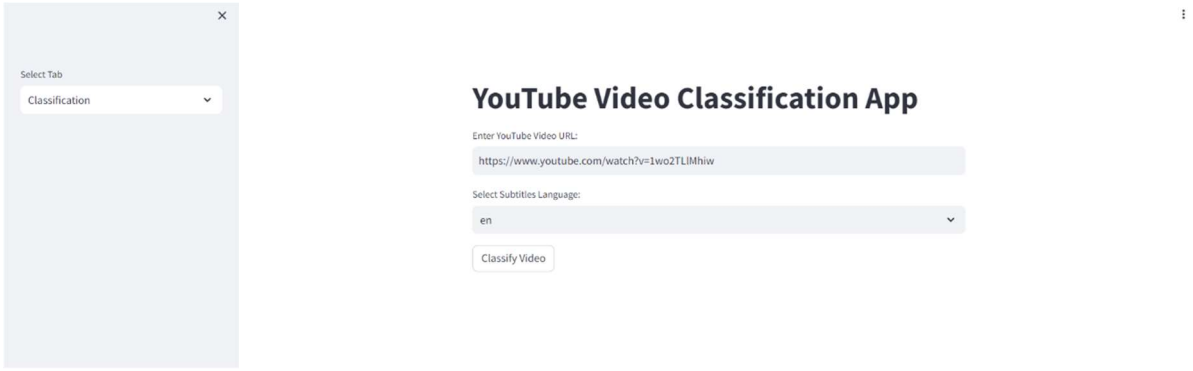


Figure 8 User Input

4.4 CLASSIFICATION RESULTS:

Select Tab

Classification

About

Details for Main Video - 1wo2TLlMhiw

Video ID: 1wo2TLlMhiw

Title: Conflict in Israel and Palestine through 2015: Crash Course World History #223

Description: *This video was published in 2015 and therefore does not reflect recent historical events, nor context about the ongoing conflict between Israel and Palestine.*

In which John Green teaches you about the conflict in Israel and Palestine. This conflict is often cast as a long-term beef going back thousands of years, rooted in a clash between religions. Well, that's not quite true. What is true is that the conflict is immensely complicated, and just about everyone in the world has an opinion about it. John is going to try to get the facts across in under 13 minutes.

Note: We mispronounced the Arabic word "nakba" at 6:06. Sorry about that!

Citation 1: Arthur James Balfour, Balfour Declaration (letter to Baron Rothschild, leader of British Jewish community). 1917.

Crash Course is on Patreon! You can support us directly by signing up at http://www.patreon.com/crashcourse

Want to find Crash Course elsewhere on the internet? Facebook - http://www.facebook.com/YouTubeCrashCourse Twitter - http://www.twitter.com/TheCrashCourse Instagram - https://www.instagram.com/thecrashcourse/ CC Kids: http://www.youtube.com/crashcoursekids

Thought Café's series on the subject: https://www.youtube.com/channel/UCGAL9TcH76MBKR5hywFZ4CA

Thumbnail URL: https://i.ytimg.com/vi/1wo2TLlMhiw/default.jpg

Select Tab

Classification

Total Number of Negative Words - 1wo2TLlMhiw

850

Explicit Content Analysis - 1wo2TLlMhiw

Explicit Content Score: 0.0038500876692325386

Explicit Content Class: non-explicit

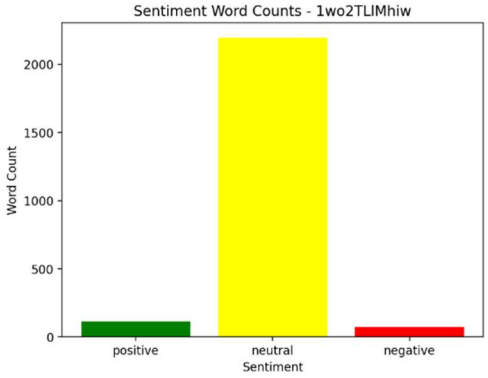
Word Cloud - 1wo2TLlMhiw

Select Tab

Classification

Sentiment Word Counts:


```
{
  "positive" : 115
  "neutral" : 2195
  "negative" : 72
}
```

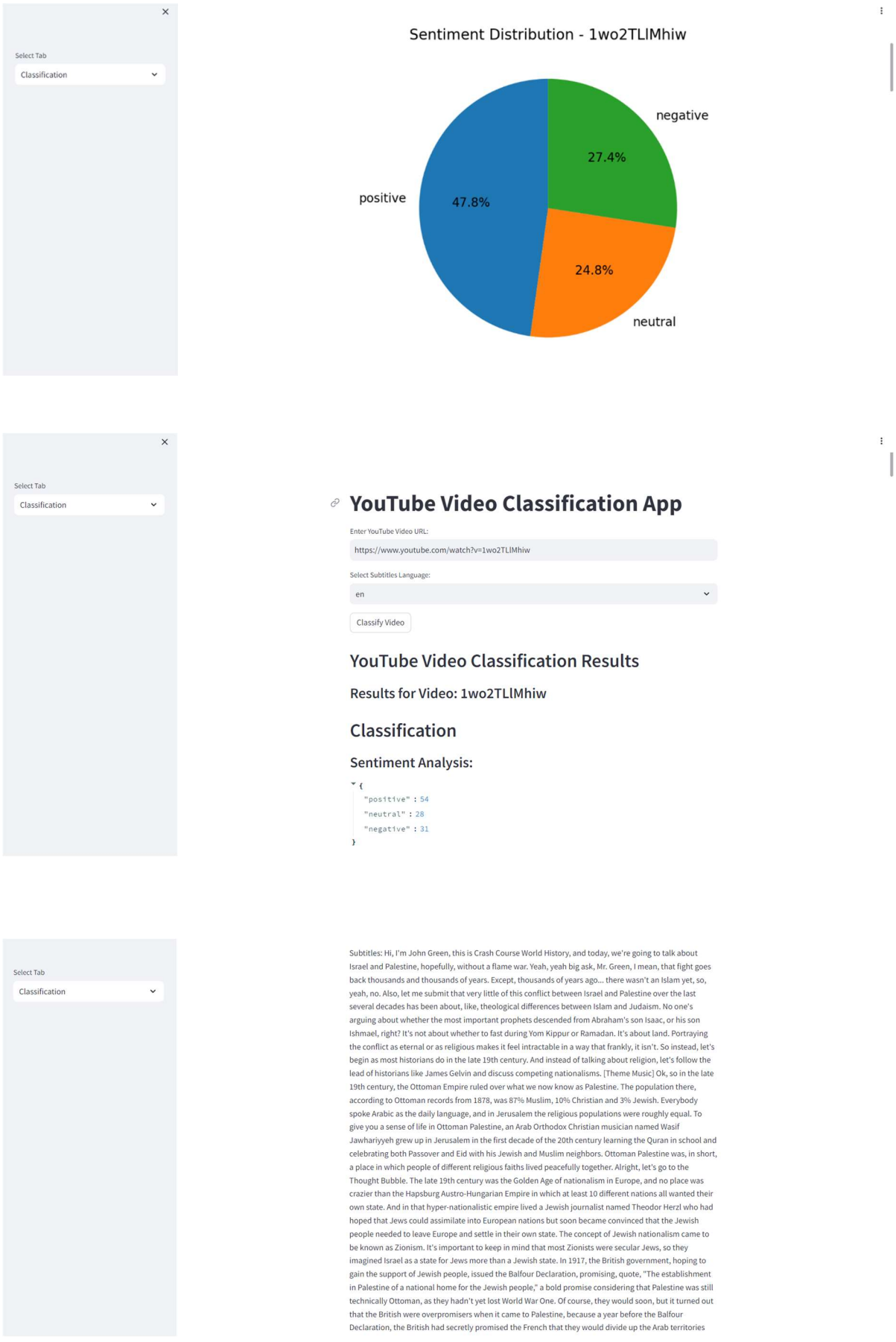


Figure 9-14 Classification Results

DISCUSSION AND CHALLENGES

1. Data Privacy and Ethics:

Analyzing video content and user interactions in the YouTube Video Classification App raises ethical considerations, particularly regarding data privacy. Careful handling of sensitive information and explicit content is essential. Ensuring user consent, transparent communication about data usage, and implementing privacy safeguards are critical aspects to address in the development process.

2. API Key Management:

The reliance on the YouTube API necessitates secure API key management. Discussions should cover secure key storage, prevention of unauthorized access, and key rotation strategies. Managing API keys securely is crucial to prevent exposure and ensure the integrity of the application's interactions with external services.

3. Accuracy of Sentiment Analysis:

Sentiment analysis, a key aspect of the app, faces challenges due to the intricacies of human language. Limitations of the NLTK library and potential inaccuracies in sentiment classification should be discussed. Improving accuracy through advanced NLP models and addressing nuances in language expressions are ongoing challenges.

4. Explicit Content Detection:

While Profanity Check is employed for explicit content analysis, it may not cover all forms of inappropriate content. Discussions should revolve around the need for a comprehensive content moderation strategy, including the integration of multiple tools and adapting to evolving language trends and emerging explicit content.

5. User Interface and Experience:

Streamlit is chosen for the UI, prompting discussions on user experience. Balancing simplicity with informative displays and handling errors or unexpected inputs gracefully are essential considerations. Ensuring a user-friendly interface that effectively communicates analysis results is a continuous challenge.

6. Multilingual Support:

The app supports multiple languages for subtitles, requiring discussions on language coverage and challenges in sentiment analysis for non-English languages. Expanding language support, addressing variations in language expressions, and considering cultural context are ongoing challenges.

7. Deployment and Scalability:

Deployment options and scalability considerations are critical topics. Discussions may include considerations of local deployment versus cloud services and optimizing resource usage for scalability. Ensuring availability and performance under varying loads is a key challenge.

8. Continuous Integration and Deployment (CI/CD):

Discussions on automating testing and deployment through CI/CD pipelines are essential for a smooth development lifecycle. Setting up efficient CI/CD pipelines, managing version control, and handling potential conflicts are challenges in maintaining a consistent and reliable release process.

9. Model Explainability:

The interpretability of sentiment analysis and explicit content detection models should be discussed. Addressing the need for model explainability to provide insights into decisions and balancing accuracy with understandability are ongoing challenges.

10. Feedback Mechanisms:

Incorporating user feedback into the development process is crucial. Discussions on collecting user feedback, creating a feedback loop for continuous improvement, and implementing mechanisms for users to report inaccuracies or provide suggestions are important for refining and adapting the application over time. Handling feedback in a manner consistent with ethical AI practices ensures responsible development and user satisfaction.

CONCLUSION AND FUTURE WORKS

5.1 CONCLUSION

The YouTube Video Classification App represents a promising initiative in leveraging natural language processing and content analysis for classifying YouTube videos based on sentiment and explicit content. The app, built with Streamlit, NLTK, Matplotlib, YouTube API, and Profanity Check, demonstrates the potential to provide valuable insights to users regarding the content of videos.

The development process involved addressing various challenges, including data privacy and ethical considerations, API key management, accuracy of sentiment analysis, explicit content detection, user interface design, multilingual support, deployment strategies, continuous integration, model explainability, and feedback mechanisms. These challenges were navigated through thoughtful discussions and solutions, emphasizing ethical AI practices and user-centric design.

5.2 FUTURE WORKS

1. Enhanced Sentiment Analysis Models:
Explore and integrate advanced natural language processing models to improve the accuracy of sentiment analysis, considering nuances, sarcasm, and cultural context.

2. Advanced Explicit Content Detection:
Investigate and implement more sophisticated techniques for explicit content detection to enhance the app's ability to identify and moderate inappropriate content effectively.

3. Multimodal Analysis:
Extend analysis beyond textual content to include audio and visual cues, enabling a more comprehensive understanding of video content.

4. Dynamic Language Support:

Dynamically adapt language support for subtitles based on user preferences and video content, improving the app's effectiveness across diverse linguistic contexts.

5. Integration of Machine Learning Models:

Explore the integration of machine learning models to continuously learn from user feedback and adapt to evolving language trends, thereby enhancing the accuracy of sentiment analysis.

6. Real-Time Analysis:

Implement real-time analysis capabilities to provide users with instantaneous insights into the sentiment and content of live streaming videos.

7. User-Centric Improvements:

Gather user feedback systematically and implement iterative improvements to the user interface and overall user experience based on user preferences and needs.

8. Security Measures:

Strengthen security measures, especially in API key management, by considering advanced key protection techniques and encryption methods.

9. Comprehensive Testing and Quality Assurance:

Implement a comprehensive testing and quality assurance strategy to ensure the robustness and reliability of the application under various scenarios.

10. Community Collaboration:

Foster collaboration with the user community to co-create features, address emerging challenges, and gather diverse perspectives for continuous enhancement.

By pursuing these future works, the YouTube Video Classification App can evolve into a more sophisticated and user-friendly tool, catering to the dynamic landscape of online content and user expectations. Continuous innovation and responsiveness to user needs will be key to the sustained success and impact of the project.

REFERENCES:

1. https://github.com/BetterForAll/HonestyMeter
2. https://github.com/armin-pousti/COMP400
3. https://docs.streamlit.io/
4. https://vppstereo.github.io/
5. https://www.youtube.com/intl/ALL_in/howyoutubeworks/policies/overview/
6. YouTube Community guidelines: https://support.google.com/youtube/answer/9288567?hl=en-GB
7. Stack Overflow: https://stackoverflow.com/
8. Chat Generative Pre-Trained Transformer

APPENDIX

GitHub Link:

https://github.com/RincisM/Responsive_AI/tree/main/Ethical_AI