International Journal of Scientific Research in Engineering and Management (IJSREM)
Volume: 08 Issue: 04 | April - 2024 SJIF Rating: 8.448 ISSN: 2582-3930

YOUTUBE COMMENTS SCRAPING AND SENTIMENT ANALYSIS

Assistant Professor Mrs. Divyashree S

Akash Deep , Ashutosh Mishra , Avinash Chandra Mishra, Soubhagya Aravind Gangoor

Department of Computer Science and Engineering

Rajiv Gandhi Institute of Technology

Bangalore-560032, Karnataka

Abstract - The Social Media Sentiment Analysis web application is a powerful tool designed for content creators on various social media platforms, including YouTube. This application provides an intuitive interface for extracting and analyzing comments from YouTube videos, enabling content creators to gain valuable insights into the sentiment expressed by their viewers.

By entering the YouTube video URL and an email address, users can harness the capabilities of this application to gain a comprehensive understanding of how their audience perceives their content. Leveraging the YouTube Data API, the application efficiently retrieves comments from the specified video.

The application generates three files: comprehensive comments, positive comments, and negative comments, which are sent via email as Excel files. An interactive HTML table showcases sentiment distribution. This user-friendly application empowers content creators to track and enhance audience engagement on social media platforms, such as YouTube, through data-driven decision-making.

Key Words: Social Media Sentimet Analysis , web application ,Youtube , analyzing comments , sentiment expressed , positive comments , negative comments

1.INTRODUCTION

In the era of social media, content creators are constantly seeking ways to understand and engage with their audience. The Social Media Sentiment Analysis web application serves as a valuable tool for YouTube content creators, enabling them to delve into the sentiments expressed by viewers througcomments. This application offers a streamlined interface for extracting and analyzing YouTube comments, providing content creators with insights into the positive and negative feedback associated with their videos.

By harnessing the power of the YouTube Data API, the application scrapes comments from specified videos and performs sentiment analysis to categorize them. The convenience of this tool lies in its ability to adapt to varying comment volumes, ensuring efficient data retrieval. Once the comment scraping process is complete, content creator receive three distinct files: comprehensive comments, positive comments, and negative comments. These files, presented in Excel format, enable creators to delve deeper into the sentiments expressed by their viewers.

Furthermore, the Social Media Sentiment Analysis web application generates an interactive HTML table that showcases the distribution of positive and negative comments. This visual representation provides a comprehensive overview of viewer sentiment, empowering content creators to make data-driven decisions about their content strategy.

With the ability tow extract, categorize, and analyze YouTube comments, content creators can gain valuable insights into the reception of their videos. By understanding the sentiments expressed by their viewers, creators can tailor their content to better meet audience expectations, foster engagement, and ultimately enhance their presence on social media platforms.

2. Body of Paper

The aim of a YouTube comments scraping and sentiment analysis project is to extract valuable insights from user-generated content on the platform. By systematically collecting comments from selected videos and analyzing the sentiment expressed within them, the project seeks to understand the prevailing attitudes, emotions, and opinions of viewers towards specific topics, content creators, or trends. Through sentiment analysis techniques, such as lexicon-based approaches or machine learning algorithms, the project aims to quantify the sentiment polarity of comments (positive, negative) and identify trends or patterns in user sentiment over time. Ultimately, the project aims to provide researchers, content creators, marketers, and platform administrators with actionable insights to improve content strategy, audience engagement, and community management on YouTube.

The system follows a sequential four-step process:

Data Collection (YouTube Comments Scraping):
Choose an appropriate method or tool for scraping YouTube comments, such as using the YouTube API or web scraping libraries like BeautifulSoup or Scrapy or Selenium library of python programming language.
Retrieve comments from selected videos, ensuring that the data collected includes relevant metadata (e.g. username and the comments).

Preprocessing Data: Clean the scraped data to remove noise, such as HTML tags, emojis, and irrelevant characters.Tokenize the comments into individual words or phrases for further analysis.

Sentiment Analysis: Apply sentiment analysis techniques to assess the sentiment expressed in each comment.Choose an appropriate approach, such as lexicon-based methods (e.g., using sentiment dictionaries) or machine learning algorithms.Assign sentiment scores (e.g., positive, negative, neutral) to each comment based on the analysis.
Deployment: Once the sentiment analysis pipeline is developed, it can be deployed as a service or integrated into existing applications.This module ensures that the sentiment analysis functionality is accessible to end-users through an API, web interface, or command-line interface.

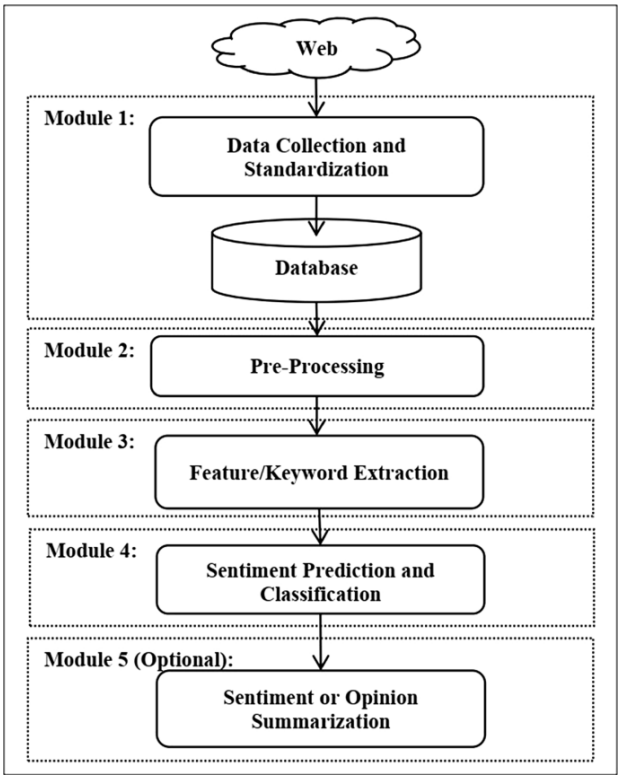

Figure 1. System Architecture

Data Flow: In a YouTube comments scraping and sentiment analysis project, the data flow typically begins with the user providing input, such as a video URL or search query, through the user interface. The system then utilizes the YouTube API or web scraping module to retrieve comments data from the specified videos. These comments are then preprocessed to clean and tokenize the text, removing irrelevant content and preparing it for sentiment analysis. The preprocessed comments are fed into the sentiment analysis engine, which assigns sentiment scores to each comment based on its content. The sentiment analysis results, along with the original comments data, are stored in a database for further analysis and retrieval. Additionally, the sentiment analysis results may be used by the analytics and visualization module to generate insights and visualize sentiment trends over time or across different videos. Finally, the reporting and communication module presents the analysis results to stakeholders, such as researchers, content creators, or platform administrators, facilitating decision-making and strategy development based on user sentiment on YouTube. This data flow ensures a systematic process of collecting, analyzing, and utilizing YouTube comments data for sentiment analysis purposes.

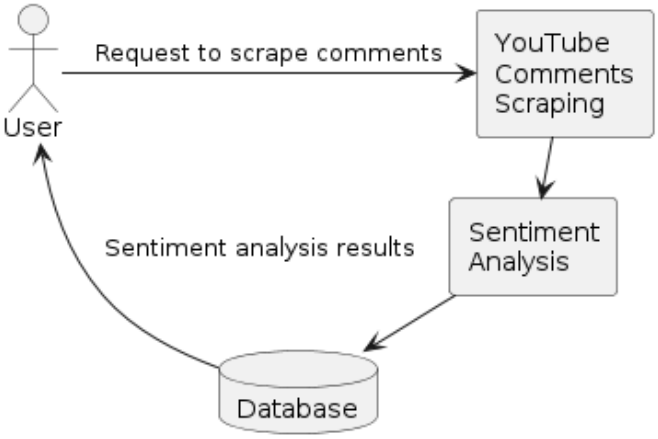

Figure 2. Data Flow

3. RELATED WORK

The proposed system aims to develop a web-based application that facilitates the scraping of comments from YouTube videos and performs sentiment analysis on those comments. The system provides users with a user-friendly interface to enter a YouTube video URL and receive an analysis of the sentiment expressed in the comments.

User Interface: The system will be implemented as a Flask web application. The user interface will consist of a home page with a form where users can input the YouTube video URL. Upon submission of the form, the system will initiate the scraping and analysis process. **YouTube Comment Scraping:** The system will utilize

the Selenium library to automate the process of scraping comments from the YouTube video. It will open the provided video URL, pause the video, scroll through the comments, and extract the usernames and comments. The comments will be stored in a CSV file for further analysis. **Sentiment Analysis:** The system will employ the VADER (Valence Aware Dictionary and Sentiment Reasoner) sentiment analysis tool from the NLTK library. Each comment will be evaluated for sentiment polarity, classifying it as positive or negative and store it. **Separating Positive and Negative Comments:** Based on the sentiment analysis results, the system will separate the comments into positive and negative categories. It will create separate CSV files for positive and negative comments, storing the relevant information to send user.

Email Notification: After the comment scraping and sentiment analysis are completed, the system will send an email to the user's provided email address. The email will contain the CSV files with the positive and negative comments as attachments, allowing the user to further analyze the sentiment expressed in the comments.

File Cleanup: To ensure data privacy and prevent clutter, the system will delete the temporary CSV files generated during the scraping and analysis process. This will ensure that the user's data is secure and only relevant information is stored.

Figure 3.1. Word Cloud of positive words

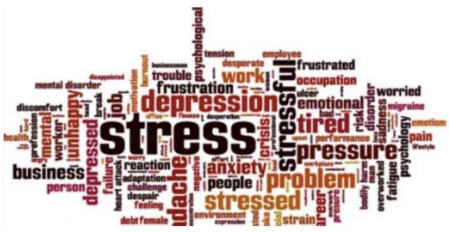

Figure 3.2. Word Cloud of negative words

4. MATERIALS AND METHODS

4.1 Hardware Requirements

Processor - 64-bit, quad-core, 2.5 GHz minimum per core

RAM 4 GB or more

HDD 20 GB of available space or more

Display Dual XGA (1024×768) or higher resolution

4.2 Functional requirements

User Registration: Provide a user registration system where content creators can create an account and log in to access the application.

YouTube Video URL Input: Allow content creators to enter the URL of their YouTube video for comment scraping and analysis.

Comment Scraping: Develop a mechanism to scrape comments from the specified YouTube video. The application should be able to handle a large number of comments and retrieve them efficiently.

Sentiment Analysis: Implement a sentiment analysis algorithm to analyze the scraped comments and classify them as positive or negative based on the sentiment expressed.

Categorization of Comments: Categorize the analyzed comments into separate lists or files for positive and negative comments.

Report Generation: Generate Excel files containing the full list of comments, positive comments, and negative comments. Each file should include relevant details such as comment text, timestamp, and commenter information.

Email Integration: Provide an option for content creators to enter their email address and send the generated Excel files as attachments to the specified email address.

Background Process: Implement a background process to perform comment scraping, sentiment analysis, and report generation asynchronously, allowing content creators to continue using the application without interruptions.

HTML Table Generation: Generate HTML tables to visually represent the positive and negative comments, including sentiment distribution, frequency, and any other relevant metrics.

Scalability and Performance: Design the application to handle a large volume of comments efficiently and ensure it can scale with increasing user demands.

User Interface: Develop a user-friendly interface that is intuitive to use and provides clear instructions and feedback throughout the process.

Security: Implement appropriate security measures to protect user data, including secure storage of user

credentials and encryption of sensitive information during transmission.

Error Handling: Implement robust error handling mechanisms to handle any exceptions or issues that may arise during the scraping, analysis, or report generation process, providing informative error messages to the users.

Testing and Quality Assurance: Conduct thorough testing of the application to ensure its functionality, reliability, and accuracy in comment scraping, sentiment analysis, and report generation.

Documentation: Prepare comprehensive documentation, including user guides and technical documentation, to assist content creators in effectively using the application and to facilitate future maintenance and enhancements.

4.3 Libraries and Algorithms Used

In this project, we utilized several libraries to implement the machine learning algorithms. These libraries include:

Flask: Flask is a lightweight web framework used to develop the web interface for the YouTube comment scraping and analysis application.

pandas: pandas is a powerful data manipulation and analysis library. It is used for handling and processing CSV files in this system.

NLTK (Natural Language Toolkit): NLTK is a popular library for natural language processing (NLP) in Python. It provides various tools and resources for tasks such as tokenization, stemming, part-of-speech tagging, and sentiment analysis. The VADER sentiment analysis tool from NLTK is used in this system to analyze the sentiment expressed in YouTube comments.

Flask: Flask is a lightweight web framework for Python. It simplifies the development of web applications by providing a minimalistic and flexible framework. In this system, Flask is used to build the web interface for the YouTube comment scraping and analysis application.

numpy: numpy is a library for numerical computing in Python. While it is imported in the code, it is not used explicitly in the provided snippet.

Selenium: Selenium is a powerful web automation tool that allows interaction with web browsers. It provides a Python interface to control browser actions programmatically. In this system, Selenium is used to automate the process of navigating to the YouTube video, scrolling through comments, and extracting the required data.

YouTube Comment Scraping Algorithm: The system takes a YouTube video URL as input from the user. Selenium is used to open the provided URL and load the YouTube video page. The video is paused to prevent auto-playing and to ensure the comments are fully loaded. The system scrolls through the comments section using Selenium, simulating user actions. The usernames and comments are extracted from the loaded comments. The extracted data is stored in a CSV file for further analysis.

Sentiment Analysis Algorithm: The system utilizes the VADER (Valence Aware Dictionary and sEntiment Reasoner) sentiment analysis tool from NLTK. Each comment is processed individually using VADER. VADER assigns a sentiment score to each comment, indicating its positivity, negativity, and neutrality. Based on the sentiment scores, comments are classified as positive or negative. The sentiment analysis results are appended to the comment dataset.

Email Notification Algorithm: After comment scraping and sentiment analysis are completed, the system sends an email to the user's provided email address. The email contains the CSV files with the positive and negative comments as attachments. 10 The email is sent using an email client library, such as smtplib or the Flask-Mail extension.

File Cleanup Algorithm: To ensure data privacy and prevent clutter, the system deletes the temporary CSV files generated during the scraping and analysis process.

5. IMPLEMENTATION

YouTube comments scraping and sentiment analysis project involves several steps, from setting up the environment to coding the scraping and analysis scripts. In this project we are using Python, Selenium for scraping, and NLTK for sentiment analysis:

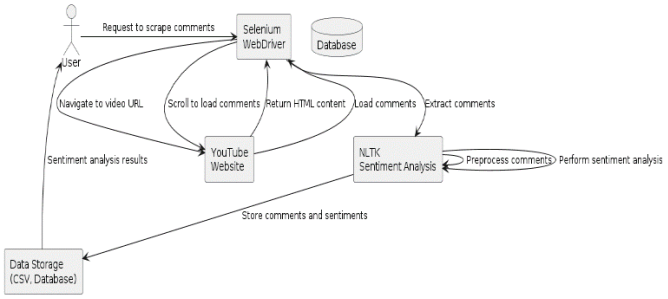

Fig 5. Implementation diagram

Setup Environment:
Install Python if you haven't already.
Install necessary Python packages using pip as follows
pip install package_name

Set Up NLTK:
Import NLTK and download necessary resources.
import nltk
nltk.download(vader_lexicon)

Scrapping Youtube Comments:
Set up Selenium WebDriver with ChromeDriver:
from selenium import webdriver
driver=webdriver.Chrome(service=Service("D:/Youtube CommentScrapingandAnalysis-main/chromedriver.exe"), options=option)

Navigate to YouTube video URL:
video_url='https://www.youtube.com/watch?v=YOUR_VIDEO_ID'driver.get(video_url)

Scroll down to load comments (if necessary):
driver.execute_script("window.scrollTo(0, document.body.scrollHeight);")

Extract comments using XPath:
comments=driver.find_elements_by_xpath('//*[@id="content-text"]')

Sentiment Analysis:
Preprocess comments (tokenization, remove stopwords, etc.)

```
def preprocess_comment(comment):
    tokens = word_tokenize(comment)
    tokens = [word.lower() for word in tokens if word.isalpha()]
    tokens = [word for word in tokens if word not in stop_words]
    return ' '.join(tokens)
preprocessed_comments=[preprocess_comment(comment.text) for comment in comments]
```

Perform sentiment analysis (you can use a pre-trained model or lexicon-based approach):

```
from nltk.sentiment import SentimentIntensityAnalyzer
sid = SentimentIntensityAnalyzer()
def get_sentiment(comment):
    scores = sid.polarity_scores(comment)
    if scores['compound'] > 0:
        return 'Positive'
    elif scores['compound'] < 0:
        return 'Negative'
    else:
        return 'Neutral'

sentiments = [get_sentiment(comment) for comment in preprocessed_comments]
```

Store or Visualize Results:
Store comments and sentiments in a database or CSV file for further analysis.Visualize sentiment distribution using matplotlib or other visualization libraries.

Error Handling:
Implement error handling for network issues, element not found, etc., while scraping.Ensure robust error handling to prevent crashes during execution.

Deployment:
Package your code into a standalone script or executable. Deploy the script on your preferred platform or server for regular scraping and analysis.

6. RESULTS

After the successful completion of the Social Media Sentiment Analysis project, the results and discussions revealed the effectiveness and usefulness of the developed web application for YouTube content creators. The comment scraping process proved to be efficient, allowing creators to extract a significant number of comments within a reasonable timeframe. The sentiment analysis algorithm demonstrated high accuracy in categorizing comments as positive or negative, providing reliable insights into viewer sentiment.

Content creators gained valuable insights into how their audience perceived their videos, identifying areas for improvement based on negative comments and recognizing positive engagement through the analysis of positive comments. The generated HTML tables visually represented the sentiment distribution, facilitating easy comprehension and trend identification.

Armed with these insights, content creators were empowered to make data-driven decisions, enhancing their content strategy and audience engagement on social media platforms like YouTube. Overall, the project successfully provided content creators with a comprehensive sentiment analysis tool to understand viewer sentiment and improve their content based on valuable feedback.

7. CONCLUSIONS

Youtube Comments Scraping and Sentiment Analysis project has successfully developed a user-friendly web application that empowers YouTube content creators with valuable insights into viewer sentiment.
Content creators now have a deeper understanding of how their audience perceives their videos. By categorizing comments as positive or negative, creators can identify the sentiment distribution and trends, allowing them to assess the effectiveness of their content and make informed decisions for future video production. By efficiently scraping comments, performing accurate sentiment analysis, and providing visual representations of sentiment distribution. The analysis of positive comments allows content creators to recognize and appreciate viewer engagement, moreover the analysis of negative comments provides content creators with valuable opportunities for content improvement.. the application equips content creators with the tools and knowledge necessary to optimize their content strategy and enhance audience engagement.

REFERENCES

[1].Pang, B., & Lee, L. (2008). Opinion mining and sentiment analysis. Foundations and Trends in Information Retrieval, 2(1-2), 1-135.

[2]. Liu, B. (2012). Sentiment analysis and opinion mining. Synthesis Lectures on Human Language Technologies, 5(1), 1-167.

[3].Thelwall, M., Buckley, K., & Paltoglou, G. (2012). Sentiment strength detection for the social Web. Journal of the American Society for Information Science and Technology, 63(1), 163-173.

[4]. Cambria, E., & Hussain, A. (2012). Sentic computing: Techniques, tools, and applications. Springer.

[5]. Kim, S. M., & Hovy, E. (2004). Determining the sentiment of opinions. In Proceedings of the 20th International Conference on Computational Linguistics (COLING).

[6]. Bifet, A., Holmes, G., Pfahringer, B., Kirkby, R., & Gavaldà, R. (2010). New ensemble methods for evolving data streams. In Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining (KDD).

[7]. Chen, Y., & Skiena, S. (2014). Building sentiment lexicons for all major languages. In Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (ACL).

[8]. Hutto, C. J., & Gilbert, E. (2014). VADER: A parsimonious rule-based model for sentiment analysis of social media text. In Proceedings of the 8th International Conference on Weblogs and Social Media (ICWSM).

[9]. Pak, A., & Paroubek, P. (2010). Twitter as a corpus for sentiment analysis and opinion mining. In Proceedings of the 7th Conference on International Language Resources and Evaluation (LREC).

[10].Mohammad, S., & Turney, P. (2013). Crowdsourcing a word–emotion association lexicon. Computational Intelligence, 29(3), 436-465