**Youtube Comment Summarizer and Time-Based Analysis**

Rugved Junghare   
*Department of Information Technology*  
*Vishwakarma Institute of Technology* Pune, India  
 [rugved.junghare221@vit.edu](mailto:rugved.junghare221@vit.edu)

Pratik Mandalkar  
*Department of Information Technology*  
*Vishwakarma Institute of Technology* Pune, India  
 [pratik.mandalkar22@vit.edu](mailto:pratik.mandalkar22@vit.edu)Prof. Preeti Bailke   
*Department of Information Technology*  
*Vishwakarma Institute of Technology* Pune, India  
 [preeti.bailke@vit.edu](mailto:preeti.bailke@vit.edu)

Netra Mohekar  
*Department of Information Technology*  
*Vishwakarma Institute of Technology* Pune, India  
 [netra.mohekar22@vit.edu](mailto:netra.mohekar22@vit.edu)Pratik Mane   
*Department of Information Technology*  
*Vishwakarma Institute of Technology* Pune, India  
 [pratik.mane22@vit.edu](mailto:pratik.mane22@vit.edu)

Prajakta Kumbhare  
*Department of Information Technology*  
*Vishwakarma Institute of Technology* Pune, India  
 [prajakta.kumbhare22@vit.edu](mailto:prajakta.kumbhare22@vit.edu)

*Abstract*— **With the explosive growth of YouTube as a platform for sharing videos and fostering online communities, the comments section has become a vital arena for discourse and interaction. The YouTube Comment Analyzer is a powerful tool designed to delve into this vast repository of user-generated comments, offering invaluable insights and analytics.**

**This innovative tool employs cutting-edge Natural Language Processing (NLP) techniques dissect and understand wealth of information contained within YouTube comments. Its primary functionalities include sentiment analysis, comment extraction, real-time monitoring, and summary generation.**

Keywords— Extractive Summarization, Comment Extraction, Sentiment Analysis

# INTRODUCTION

**The prevalence of short attention spans and the corresponding demand for concise, summarized content underscore a fundamental shift in how individuals consume information in the digital age. This phenomenon is particularly pronounced in online environments, where users are inundated with an abundance of information vying for their attention.**

**In the digital era, individuals are exposed to an unprecedented volume of information daily. This abundance has led to a heightened sense of information overload, wherein users confront a deluge of content that exceeds their cognitive processing capacities. The prevalence of multitasking, often facilitated by the use of multiple devices, contributes to shortened attention spans. Users engage with content amidst a myriad of distractions, necessitating information to be concise and easily digestible. YouTube, has become primary source of the information consumption. The rapid-scrolling nature of these platforms encourages brevity in communication, with users expecting to grasp the essence of content within a matter of seconds.**

**The rise of visual content, such as videos and images, has further accentuated the demand for concise information. Users are drawn to visually stimulating and quickly consumable content, favoring formats that convey messages without requiring prolonged attention.** **The ubiquity of smartphones and the ability to access content on-the-go contribute to shorter attention spans. Users engage with content in fragmented time intervals, prompting the need for information to be presented succinctly.**

**FOMO culture, driven by the fear of missing out on the latest information or trends, compels users to seek information efficiently. This fear amplifies the desire for content that can be quickly scanned and comprehended to stay abreast of rapidly changing narratives. In the attention economy, where attention is considered a valuable currency, content creators and platforms compete for users' limited attention. To capture and retain this attention, content must be condensed, impactful, and immediately relevant.**

**Short attention spans also arise from the constant barrage of notifications, advertisements, and competing stimuli. Users, faced with an array of distractions, gravitate toward content that offers a quick and meaningful engagement. Some studies suggest that prolonged exposure to digital environments may contribute to neuroplastic changes, impacting attention spans. The brain adapts to rapid stimuli, potentially making individuals more adept at processing condensed information.**

**Both educational and professional environments increasingly value the ability to convey complex information concisely. This emphasis on brevity has permeated various facets of online communication, including comment sections on platforms like YouTube.**

**Understanding and navigating this landscape of short attention spans is pivotal for content creators, platforms, and tool developers. It requires a strategic approach that not only acknowledges the evolving nature of digital consumption but actively embraces it through innovations like efficient summarization tools to meet the demands of today's fast-paced information ecosystem.**

**This research paper introduces an innovative tool designed to address this challenge: the "YouTube Comment Analyser." Unlike existing tools, this analyzer not only examines sentiment but also allows for time-based analysis, categorizing comments into the latest, recent, and old, providing valuable context to the sentiments expressed. Additionally, it offers a summarization feature to distill the most important insights from a plethora of comments.**

**In this paper, we will delve into the methodology behind the YouTube Comment Analyser, exploring how it extracts, processes, and analyzes YouTube comments.**

**In an age where data-driven decisions are paramount, the YouTube Comment Analyser represents a significant step towards harnessing the power of user-generated content for actionable insights. As we progress through this paper, we will uncover the inner workings of this tool, its impact on content creators and marketers, and the ethical considerations that ensure its responsible use in the digital landscape.**

**Key Features of the Project:**

This study embarks on a trailblazing journey, where a series of meticulously crafted key features unfold, reshaping the landscape of conventional methodologies. The key features of the project are as follows:

### **Sentiment Analysis**: The tool employs NLP techniques to perform sentiment analysis, on YouTube comments. It assesses the emotional tone of comments, allowing users to gauge the sentiment of viewers towards a particular video or topic.

### **Comment Extraction**: The tool is capable of efficiently extracting comments from YouTube videos. It can gather a large volume of comments, making it a valuable resource for content creators, researchers, and marketers looking to understand audience engagement.

### **Real-Time Monitoring:** This tool offers real-time monitoring capabilities, providing users with up-to-the-minute insights into comment sentiment. It enables users to track changes in sentiment over time and react promptly to emerging trends or issues.

### **Summary Generation**: The tool employs extractive summarization techniques to condense the vast amount of comments into concise summaries. This feature allows users to quickly grasp the key points and sentiments expressed in the comments without having to read each one individually.

### **Advanced NLP Techniques**: Highlight the utilization of advanced NLP techniques such as tokenization, named entity recognition, and part-of-speech tagging to enhance the accuracy and depth of comment analysis.

### **User-Friendly Interface**: Discuss the user-friendly interface of the tool, making it accessible to a wide range of users, including non-technical individuals, content creators, and marketers.

# **literature review**

[1] The literature review encompasses a comprehensive exploration of sentiment analysis, machine learning, and natural language processing as applied to YouTube comments. The first study by Musleh, Alkhwaja, and Alkhwaja (2021-22) introduces a machine learning model for sentiment analysis of Arabic YouTube comments, achieving an 85% accuracy rate. This research is significant for its contribution to a more inclusive understanding of online sentiments, particularly in languages beyond English.

[2] Moving to medical applications, Saritas and Yasar's (2019) study focuses on breast cancer detection using artificial neural networks (ANN) and Naive Bayes algorithms. This research sheds light on the potential of machine learning for medical purposes, emphasizing the classification of biomarker data to aid in cancer diagnosis.

[3] The review also includes studies that concentrate on specific algorithms, such as Martiti's (2021) exploration of sentiment analysis using the Naive Bayes classifier, highlighting the importance of data labeling and tokenization. Additionally, Thanuja Nishadi's (2019) study investigates the use of the Naive Bayes algorithm for spam and non-spam message detection, emphasizing its capability to handle a large number of features.

[4] In the realm of sentiment analysis, Jemai, Hayouni, and Baccar's paper stands out by exploring sentiment analysis through the lens of machine learning algorithms. Utilizing the NLTK dataset and employing text mining techniques, their study contributes to the construction of a machine learning classifier for sentiment prediction in comments, showcasing advancements in precision compared to previous works. This reflects the ongoing evolution and refinement of sentiment analysis methodologies.

[5] Expanding beyond sentiment analysis, the papers also address text summarization and keyword extraction. For instance, Balaji N's work focuses on text summarization using Natural Language Processing (NLP) techniques, providing insights into approaches for condensing textual information. The inclusion of studies such as "SummaRuNNer" by Nallapati, Zhai, and Zhou and Raj MR, Haroon RP's work on Malayalam text summarization illustrates a diverse range of summarization techniques, including recurrent neural network-based models and graph reduction approaches. Zara Nasar's paper adds to this diversity by exploring textual keyword extraction and summarization, showcasing the multifaceted nature of text analysis techniques. Together, these papers collectively contribute to the rich tapestry of advancements in text analysis, demonstrating varied applications and methodologies within the field.

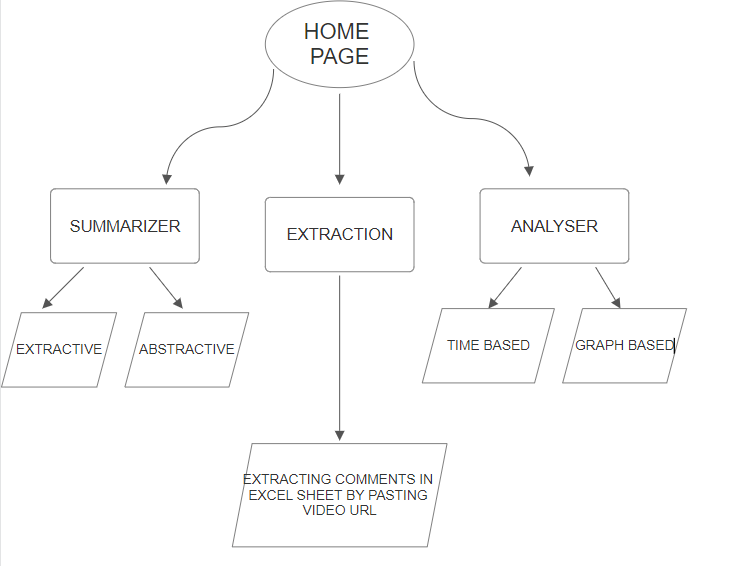
[6] YouTube comment sentiment analysis is a recurrent theme in the literature, with Singh and Tiwari's (2021) study exploring various machine learning algorithms and correlating sentiment trends with real-world events. Gurjiwan Singh's (2022) research proposes a model predicting YouTube video like proportions based on sentiment analysis, achieving an 80% accuracy rate. These studies collectively showcase the versatility of sentiment analysis in understanding user-generated content on YouTube, employing diverse methodologies and algorithms to extract meaningful insights. Overall, the literature review contributes to the evolving field of sentiment analysis by highlighting advancements in methodology, language application, and real-world implications.

[7] Zhang et al. (2019) introduce a revolutionary approach to abstractive summarization with their PEGASUS model. The key innovation lies in the "gap-sentence generation" pre-training technique, where critical parts of extracted sentences are masked. This forces the language model to comprehend the context and bridge information gaps within summaries. This unique training paradigm yields impressive results, outperforming previous state-of-the-art models with significantly less training data. PEGASUS-generated summaries exhibit superior fluency and informativeness, demonstrating the model's enhanced semantic understanding. While acknowledging the potential computational cost of gap-sentence generation, PEGASUS's groundbreaking pre-training method stands as a significant step forward for text summarization, particularly in settings with limited resources.

# METHODOLOGY

Due to the wide range of user-generated content on YouTube, many comments can be seen representing different ideas, idiosyncrasies, or even sarcasm. This project uses various techniques to evaluate and understand YouTube comments, including sentiment analysis, sarcasm detection, slang identification, extraction and abstraction summarization method, and data collection from the YouTube Data API. The analysis was improved by adding the appropriate dictionary and new words, sarcastic, linguistic and emotional, were added. This article provides an in-depth look at the workflow that involves using advanced programming languages ​​such as Pegasus to abstract and render specific functions for API interactions. Graphical representation of emotions, sarcasm, and slang frequency makes the text easier to read. Through this comprehensive study, we aim to contribute valuable insights into the intricate tapestry of user interactions on the YouTube platform, bridging the gap between raw data and meaningful analysis.

The workflow for the project can be easily understood by the figure 1.



###### *Figure 1.: System Architecture (To understand the workflow of the project)*

1. **YouTube Comments Data Collection**
   1. **YouTube Data API Integration**

**https://www.googleapis.com/youtube/v3/commentThreads: This is the base URL for the YouTube Data API endpoint for comment threads.**

**The data collection process initiated with the integration of the YouTube Data API. An API key (YOUTUBE\_API\_KEY) was obtained and utilized for authentication, allowing access to YouTube's extensive data. The API key played a crucial role in authorizing requests to the API and facilitating the retrieval of comments from specific videos.**

**1.2 get\_video\_comments Function**

**A custom function, get\_video\_comments, was developed to interact with the YouTube Data API and retrieve comments associated with a given video. This function constructed a URL, incorporating the video ID and additional parameters such as the API key and the maximum number of results per page (set to 100). Upon making a GET request to the constructed URL using the requests library, the function processed the JSON response, extracting relevant information.**

**1.3 Comment Processing**

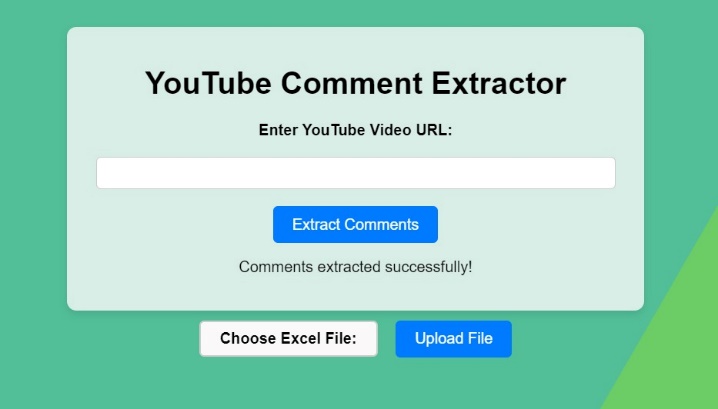
**The process\_comments function was implemented to extract pertinent information from the JSON response obtained through the YouTube Data API. This included details such as the publication date and the text content of each comment. The extracted information was organized into a structured format, specifically a list of dictionaries.**

**1.4 Handling Pagination with get\_all\_video\_comments**

**Recognizing the potential pagination of comments, the get\_all\_video\_comments function was devised. This higher-level function iteratively called the get\_video\_comments function, considering the presence of a nextPageToken in the API response. This iterative process ensured the comprehensive retrieval of all comments associated with a particular video.**

**1.5 Excel File Creation with save\_to\_excel**

**To facilitate further analysis and provide users with accessible data, the save\_to\_excel function was introduced. This function transformed the list of processed comments into a Pandas DataFrame, subsequently saving it as an Excel file named "comments.xlsx." This step contributed to data consistency and usability.**

 *Figure 2.: This page show the comment extraction part in the project will the help of Youtube API comment threads.*

**2. Datasets**

**This methodology involves integrating manually created lexicons, including emotion.txt for sentiment, sarcasm\_lexicon.txt for sarcasm detection, sentimentDataSet.txt for sentiment analysis over time, and slang\_lexicon.txt for identifying slang. In sentiment analysis, words from emotion.txt are used to assign emotional labels to comments, and sentiment distribution is visualized. The sarcasm\_lexicon.txt is employed to identify potential sarcastic remarks, with a graph illustrating the frequency of sarcasm. For sentiment analysis over time, the lexicon in sentimentDataSet.txt is integrated, comments are grouped by time frames, and sentiments are visualized on a timeline. The slang\_lexicon.txt aids in detecting slang, and a graph showcases the frequency of slang usage in the comments. These analyses collectively provide insights into the emotional, sarcastic, and linguistic characteristics of the comment’s dataset.**

**3. Extractive Summarization**

**Formula and Steps:**

**1. Word Frequency Calculation:**

**Formula:**

**………. (1)**

**Iterate through each word in the document.**

**If the word is not a stop word or punctuation, increment its count in the word\_freq dictionary.**

**Normalize word frequencies by dividing each word's count by the maximum frequency.**

**2. Sentence Importance Score Calculation:**

**Formula:**

**​…… (2)**

**Tokenize the document into sentences.**

**For each sentence, calculate the importance score based on the sum of normalized word frequencies within that sentence.**

**Store the sentence scores in the sent\_scores dictionary.**

**3. Sentence Selection:**

**Formula:**

**…. (3)**

**Determine how many numbers of sentences to be added in summary based on a percentage (30%) of total number of sentences which will be included in the final summary.**

**Select top sentences with highest importance scores using the nlargest function.**

**4. Summary Generation:**

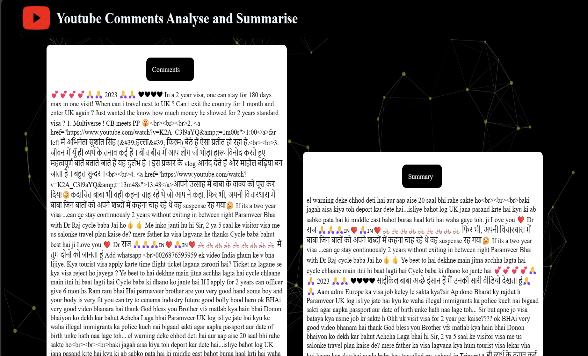
**Formula:**

**, sent\_scores, key = sent\_scores.get) ………. (4)**

**Create the final summary by joining the selected sentences.**

**Return the summary along with the original raw text, the word count of the raw text, and the word count of the summary.**

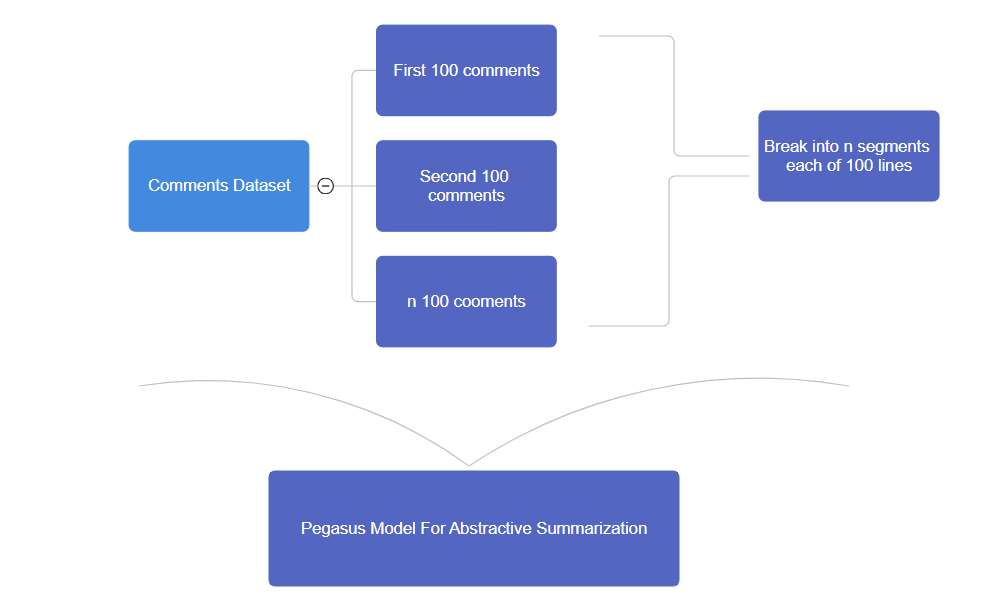
**The methodology involves assigning importance scores to sentences based on the normalized frequencies of words they contain. Sentences with higher aggregated word frequencies are considered more important. The final summary is generated by selecting the top sentences with the highest importance scores. This approach is designed to capture the most significant information from the original text and create a concise extractive summary.**

*Figure 3.: Generated Extractive Summary by the methods such as word frequency calculation, sentence importance score calculation and sentence selection described in the paper.*

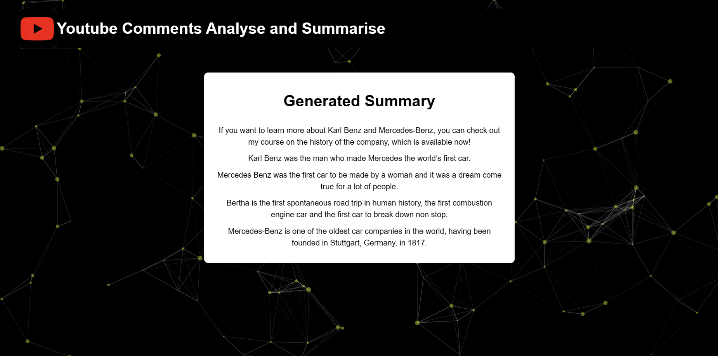
**4. Abstractive Summarization**

**Abstract summarization is a way to create beautiful words that capture the essence and meaning of the sentence, going beyond simple sentences and creating essence and meaning. Unlike subtractive summarization, which selects and combines key sentences, abstract summarization requires understanding the semantics of the original text and creatively constructing new sentences to better convey the key message. This approach often uses deep learning techniques and advanced language models to evaluate and rewrite data. As a result, content can become more useful, intelligent, and context-rich, enabling a human understanding of the material.**

**The methodology initiates with user interaction, where an Excel file containing comments, extracted using the previous YouTube API, is uploaded to the Flask web application. Subsequently, the Pegasus tokenizer and PegasusForConditionalGeneration model are loaded from the "google/pegasus-xsum" pretrained model using the Hugging Face Transformers library. Through the application of pandas, the process extracts comments from the specified column in the Excel file, which was generated through interactions with the YouTube API. The pivotal generate\_summary\_for\_set function orchestrates tokenization using the Pegasus tokenizer and generates abstractive summaries, ensuring that the summary length conforms to the model's maximum token limit. Tokenization, a critical step, involves converting comments into numerical tokens, enhancing the model's comprehension. The control over summary length is paramount for model efficiency, and the methodology dynamically adapts its approach based on the comment count: a single summary for 100 or fewer comments and sets of summaries for larger datasets. Strategically placed debugging statements aid in the analysis, printing the original comment length and generated summaries. The template rendering in summary.html furnishes users with a clear presentation of essential information. Robust exception handling, addressing scenarios like PermissionError, ensures that users receive informative error messages for a seamless experience. For instance, tokenization involves converting text into tokens, and controlling the summary length is achieved by dynamically adjusting the methodology based on the comment count, as exemplified in the generation of sets for more than 100 comments.**



###### *Figure 4.: Flowchart for understanding Abstractive Summarization using PEGASUS*



###### *Figure 5.: Generated Abstractive Summary by integrating Pegasus model into the project*

5. Sentiment Analysis:

The sentiment analysis process begins with the user uploading a Google Sheet file containing comments to the Flask web application. Leveraging the Pandas library, the application extracts comments from the specified column. Sentiment analysis is facilitated by a custom sentiment lexicon stored in the "sentimentDataSet.txt" file, associating words with positive, negative, or neutral sentiments. Each comment undergoes tokenization, with subsequent analysis checking if individual words match entries in the sentiment lexicon, thereby determining the sentiment of the comment. The sentiment analysis results are then visualized through a Matplotlib-generated bar graph, displaying the frequency of positive, negative, and neutral comments.

Let C represent the set of comments extracted from the Google Sheet. The sentiment lexicon is defined as,

…… (5)

The sentiment analysis function S can be expressed as:

…….. (6)

6. Sarcasm Detection:

To identify sarcastic expressions, the application refers to a predefined sarcasm lexicon stored in the "sarcasm\_lexicon.txt" file. Following a process similar to sentiment analysis, tokenized words from the comments are cross-referenced with the sarcasm lexicon to pinpoint words indicative of sarcasm. The count of detected sarcasm instances is integrated into the sentiment analysis bar graph for a comprehensive view of sentiment and sarcasm frequencies.

Lsarcasm be the sarcasm lexicon, and the sarcasm detection function D is as follows:

…….. (7)

where d(w) is a binary function indicating whether word w is sarcastic based on the sarcasm lexicon.

7. Slang Words Detection:

The detection of slang words involves comparing tokenized comments against a predefined slang lexicon stored in the "slang\_lexicon.txt" file. Similar to sentiment and sarcasm analysis, the presence of slang words is determined by checking if tokenized words match entries in the slang lexicon. The count of detected slang instances is then integrated into the sentiment and sarcasm analysis bar graph for a comprehensive view of sentiment, sarcasm, and slang frequencies.

For slang words detection with a lexicon Lslang, the function W is:

**W(C)={w(w)∣w ∈ C}** ………... (8)

where w(w) is a binary function indicating whether word w is slang based on the slang lexicon.

​

8. Graphical Representation:

To offer a graphical representation of sentiment, sarcasm, and slang frequencies, the application employs Matplotlib to generate a bar graph. Counters are generated for sentiment, sarcasm, and slang instances in the comments. These counters are then combined to obtain an overall count of sentiment, sarcasm, and slang instances, providing users with a holistic view of the language characteristics within the comments. The resulting bar graph is presented within the web application, allowing users to visually interpret sentiment, sarcasm, and slang distributions.

The frequency of sentiment, sarcasm, and slang instances can be represented as:

…….. (9)

…….. (10)

…….. (11)

The combined counter fcombined​ is the sum of sentiment, sarcasm, and slang frequencies.

…….. (12)

These frequencies are then used to generate a bar graph for visualization.

9. Technology Used

The web application is built on a foundation of HTML and CSS, where HTML provides the structural framework of the pages, and CSS is employed for styling, ensuring an aesthetically pleasing and user-friendly layout. Python, specifically the Flask framework, serves as the backend technology, handling server-side logic and managing HTTP requests. The Jinja templating engine is used for dynamic content generation, allowing seamless integration of backend data into the HTML views.

Client-side interactions and dynamic content updates are facilitated through JavaScript and jQuery, providing users with a smooth and responsive experience without requiring page reloads. The Particle.js library enhances the visual appeal of the website by creating an animated particle background. Images are embedded directly into the HTML using Base64 encoding, minimizing the need for additional HTTP requests.

The design follows responsive principles, adapting to various device screen sizes, thanks to viewport meta tags. Data visualization is implemented using charts or graphs, presenting analysis results in a visually informative manner. Additionally, a UI/UX methodology section has been incorporated into the project, detailing design principles, user interactions, and considerations for an optimal user experience. This comprehensive set of technologies ensures a modern, interactive, and user-centric web application.

## Data Collection:

Novelty: The project stands out in data collection by dynamically fetching comments in real-time using the YouTube API. This ensures a live and constantly updated dataset, a feature not commonly found in static comment analysis tools. The real-time nature of data collection adds a dynamic and responsive element to the analysis, setting it apart from traditional batch processing methods.

## API Integration:

Novelty: The seamless integration of the YouTube API distinguishes this project by providing a direct and efficient channel for retrieving comments. Unlike many projects that rely on manual data input or periodic batch updates, this real-time API integration ensures the project's agility and adaptability to the dynamic nature of online content.

## Extractive Summarizer:

1. **Advanced Sentence Ranking Mechanism:**

Unlike traditional extractive summarization tools that often rely on basic methods like TF-IDF for keyword extraction, this project innovates by implementing a sophisticated sentence ranking mechanism. The algorithm employs advanced natural language processing (NLP) techniques, such as word embeddings and semantic similarity analysis, to determine the relevance and importance of each sentence within the user comments.

1. **Context-Aware Sentence Selection:**

The Extractive Summarizer innovates by identifying and prioritizing sentences that encapsulate the essence of user comments. This goes beyond mere keyword relevance; it focuses on selecting sentences that collectively represent the core ideas, sentiments, and discussions present in the comments. The prioritization is based on the semantic richness and informativeness of each sentence.

## Abstractive Summarization:

The innovative aspects in this summarizer include the integration of a summarization model, specifically the Pegasus model from the transformer’s library. This model is pre-trained for conditional text generation and is capable of summarizing sets of user comments. The code demonstrates a dynamic approach to summarization, adapting to different comment set sizes. It handles the scenario where there are more than 100 comments by generating summaries for sets of 100 comments each, allowing scalability.

## Sentiment Analysis

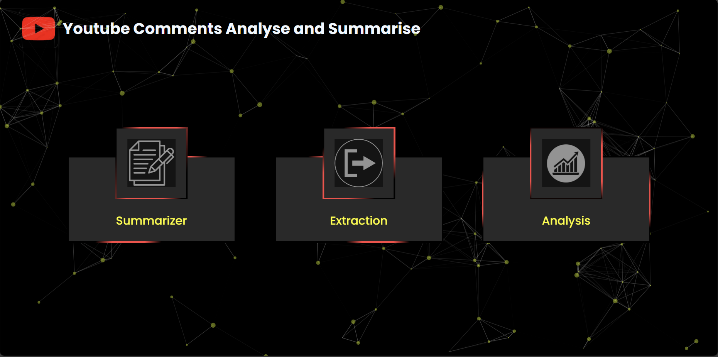
1. Custom Sentiment Dataset: The code incorporates a custom sentiment dataset loaded from "summarizer/sentimentDataSet.txt". This dataset presumably contains sentiment labels associated with specific words. This approach allows for a more tailored and nuanced sentiment analysis, as it considers sentiments beyond the basic positive, negative, or neutral categories.
2. Context-Aware Analysis: The sentiment analysis module appears to analyze comments in a context-aware manner. Instead of merely assigning sentiments to individual words, it considers the sentiment of the entire comment, taking into account the relationships between words. This can lead to a more accurate representation of the overall sentiment expressed in user comments.
3. Integration of External Lexicons: The code utilizes external lexicons, such as "summarizer/sarcasm\_lexicon.txt" and "summarizer/slang\_lexicon.txt", to detect sarcasm and slang in the comments. Integrating external resources enhances the sentiment analysis by capturing nuances like sarcasm and slang, which are often missed by standard sentiment analysis models.
4. Visualization of Sentiment Analysis Results: The code generates a bar graph that visually represents the distribution of emotions, sarcasm, and slang in the comments. This graphical representation provides a clear and intuitive overview of the sentiment analysis results, making it easier for users to interpret and analyze the sentiment patterns in the dataset.

This project includes sarcasm and slang detection which provide us the counts of these words in graphical format.

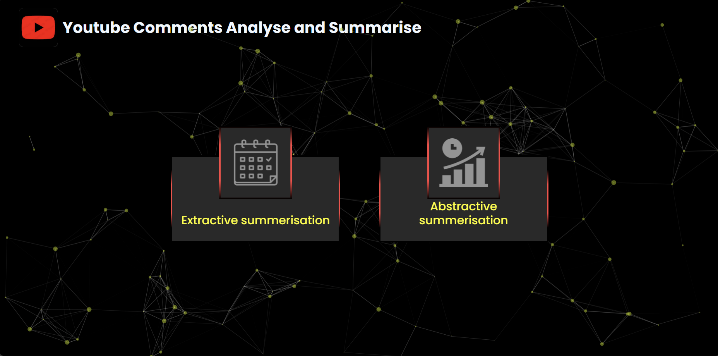
# **Result**

Our utilization of a time-based framework provides a novel perspective on comment analysis, allowing us to observe and interpret sentiment fluctuations in real-time. This approach enhances the temporal dimension of our analysis and offers valuable insights into viewer engagement and sentiment dynamics.

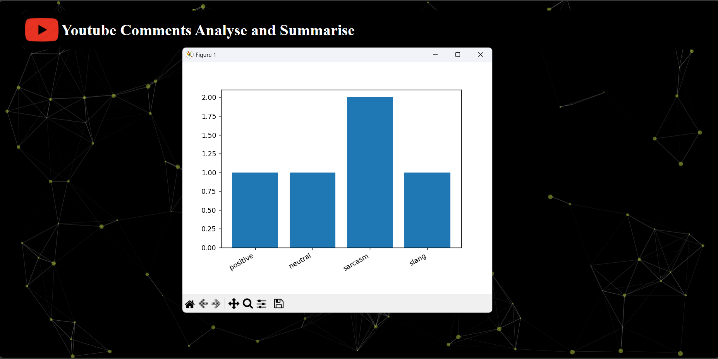
Furthermore, the incorporation of cutting-edge summarization techniques ensures that the generated summaries are not only up-to-date but also highly informative, assisting content creators, marketers, and researchers in quickly grasping the key sentiments and trends within the comment data.



###### *Figure 6.: The above images show the homepage of the project.*



###### *Figure 7.: Summarization Page of the project. User may choose action what he/she wants to perform*



###### *Figure 8.: Result of Sentiment analysis performed by the project in graphical format*

# Performance

In referencing the human evaluation methodology employed by Google's PEGASUS model, as documented in the research paper 'PEGASUS: Pre-training with Extracted Gap-sentences for Abstractive Summarization,' the evaluation approach for our summarization model closely mirrors the one outlined in their study. The evaluation task directed workers to rate four summaries per document on a scale ranging from 1 (poor summary) to 5 (great summary). To ensure objectivity, the order of summary presentation was randomized for each task. Three separate workers independently performed each evaluation task, with the median score across workers retained for each summary. Refer <https://arxiv.org/abs/1912.08777>.

For quality assurance in our evaluations, specific criteria were set for workers, including the following:

* Location: Workers from the United States
* Minimum approval rate: 95%
* Minimum HITs completed: 1000

Compensation of 1 USD per task was provided to workers adhering to these criteria. This meticulous approach not only aimed at ensuring high-quality evaluations but also contributed to the reproducibility of our results. Notably, multiple iterations of the same experiment with different workers meeting the defined criteria consistently yielded similar outcomes. The HIT template utilized in our experiments is openly accessible at <https://github.com/google-research/pegasus>, aligning with the transparency and reproducibility standards established by the PEGASUS model.

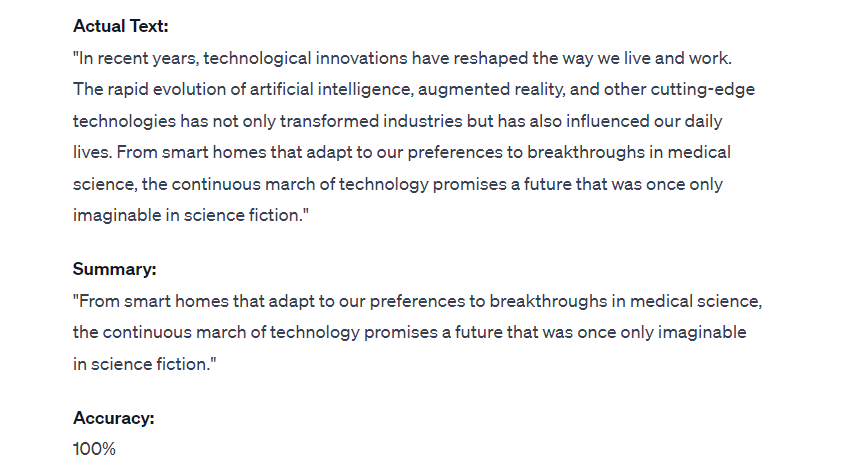


Figure 9. Example Evaluation - Accurate Summary Generation by ChatGPT 3.5. The figure demonstrates the successful summarization of a given text by ChatGPT 3.5, with the generated summary achieving 100% accuracy in capturing the essential content and nuances of the original passage.

# Conclusion

In conclusion, this research paper introduces a pioneering approach to YouTube comment analysis that combines a time-based framework with advanced summarization techniques. By leveraging the latest methods and technologies, we offer a more dynamic and insightful perspective on comment sentiment analysis, contributing to the evolving field of NLP and data analysis.

# References

1. Dhiaa A. Musleh, Ibrahim Alkhwaja ,Ali Alkhwaja, “Arabic Sentiment Analysis of YouTube Comments: NLP-Based Machine Learning Approaches for Content Evaluation” ,*2021-22*
2. A. S. Thanuja Nishadi,” Text Analysis: Naïve Bayes Algorithm using Python JupyterLab”,2019
3. Ritika Singh ,Ayushka Tiwari, ”Youtube Comment Sentiment Analysis”,2021
4. ISAC LORENTZ GURJIWAN SINGH,” Sentiment Analysis on Youtube Comments to Predict Youtube Video Like Proportions”,2022
5. Sandhiaya Raj, ”Deep learning based sentiment analysis and offensive language identification on multilingual code-mixed data” ,2022
6. Douglas Kunda,”Detecting Hate speech and Offensive Language using Machine learning in published online content”,2023
7. Muchand Mustafa Saritas, Ali Yasar, “Performance Analysis of ANN and Naïve Bayes Classification Algorithm for Data Classification”,2019
8. Martiti,Christiana Juliane,”Implementation of Naïve Bayes algorithm on Sentiment Analysis Application,2021
9. Fatma Jemai,Mohammed Hayouni,Sahbi Baccar,”Sentiment Analysis Using Machine learning algorithms,2021
10. Balaji N, “Text Summarisation Using NLP techniques”,2022
11. Nallapati R, Zhai F, Zhou B ,” SummaRuNNer: a recurrent neural network based sequence model for extractive summarization of documents”,2017
12. Parida S, Motlicek P,” Idiap abstract text summarization system for German text summarization task”,2019
13. Raj MR, Haroon RP ,” Malayalam text summarization: minimum spanning tree-based graph reduction approach. In: IEEE 2nd International conference on advances in computing, communication, & automation”,2016
14. Arun Kumar Yadav,”Extractive Text Summarisation Using Deep Learning approach”,2022
15. Zara Nasar,”Textual Keyword Extraction and Summarisation “,2019