TRANSCRIPT SUMMARIZATION ON VISUAL DATA

A MINOR PROJECT REPORT

Submitted by

VASUPRATHA M- RA2111003020439

NANDHINI V- RA2111003020488

PRAMIKHA K-RA2111003020633

Under the guidance of

Dr.B. Deepa

(Assistant Professor, Department of Computer Science and Engineering)

in partial fulfilment for the award of the

degree of

BACHELOR OF TECHNOLOGY

in

COMPUTER SCIENCE AND ENGINEERING

of

FACULTY OF ENGINEERING AND TECHNOLOGY

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY
RAMAPURAM, CHENNAI -600089

NOV 2024

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

(Deemed to be University U/S 3 Of UGC Act,1956)

BONAFIDE CERTIFICATE

Certified that this project report titled **"TRANSCRIPT SUMMARIZATION ON VISUAL DATA"** is the bonafide work **VASUPRATHA M[REG NO: RA2111003020439], NANDHINI V [REG NO: RA2111003020488],PRAMIKHA K [REG NO: RA2111003020633]** who carried out the project work under my supervision. Certified further, that to the best of my knowledge the work reported herein does not form any other project report or dissertation on the basis of which a degree or award was conferred on an occasion on this or any other candidate.

SIGNATURE	SIGNATURE
Dr. B. Deepa	Dr. K. RAJA, M.E., Ph.D.,
Assistant professor	Professor and Head
Department of Computer Science and Engineering, SRM Institute of Science and Technology,	Department of Computer Science and Engineering, SRM Institute of Science and Technology,
Ramapuram, Chennai.	Ramapuram, Chennai.

Submitted for the project viva-voce held on____________at SRM Institute of Science and Technology, Ramapuram, Chennai -600089.

INTERNAL EXAMINER 1 **INTERNAL EXAMINER**

SRM INSTITUTE OF SCIENCE AND TECHNOLOGY

RAMAPURAM, CHENNAI - 89

DECLARATION

We hereby declare that the entire work contained in this minor project report titled "**TRANSCRIPT SUMMARIZATION ON VISUAL DATA**" has been carried out by **VASUPRATHA.M**[REGNO:RA2111003020439],**NANDHINI.V**[REGNO:RA1811003020 488], **PRAMIKHA K** [RA2111003020633] at SRM Institute of Science and Technology, Ramapuram Campus, Chennai- 600089, under the guidance of **Dr. B. Deepa, Assistant Professor**, Department of Computer Science and Engineering.

Place: Chennai

Date:

VASUPRATHA M

NANDHINI V

PRAMIKHA K

ACKNOWLEDGEMENT

We place on record our deep sense of gratitude to our lionized Chairman **Dr.R.SHIVAKUMAR** for providing us with the requisite infrastructure throughout the course.

We take the opportunity to extend our hearty and sincere thanks to our **Dean, Dr.M. SAKTHI GANESH., Ph.D.,** for maneuvering us into accomplishing the project.

We take the privilege to extend our hearty and sincere gratitude to the Professor and Head of the Department, **Dr.K.RAJA,M.E.,PhD.,** for his suggestions, support and encouragement towards the completion of the project with perfection.

We express our hearty and sincere thanks to our guide **Mrs.Dr.B.Deepa , B. Tech, Assistant Professor,** Computer Science and Engineering Department for his encouragement, consecutive criticism and constant guidance throughout this project work.

Our thanks to the teaching and non-teaching staff of the Computer Science and Engineering Department of SRM Institute of Science and Technology, Ramapuram Campus, for providing necessary resources for our project.

ABSTRACT

With the prolific growth of online video content, extracting valuable information from lengthy videos has become a significant challenge. This paper presents a comprehensive YouTube video summarization tool that seamlessly integrates various natural language processing (NLP) techniques, translation functionalities, and text-to-speech capabilities to enhance the accessibility and usability of video content. The tool begins by retrieving the transcript of a YouTube video using the YouTubeTranscriptApi library. Leveraging the power of spacy for NLP, the system employs an extractive summarization approach, selecting the most salient sentences based on word frequencies. Users have the flexibility to choose the summarization length—Small, Medium, or Large— tailoring the tool to their preferences. Furthermore, the project integrates translation capabilities using the translate library, allowing users to translate the summarized content into different languages. The current implementation supports English to Tamil translation. To enhance accessibility, the tool incorporates text-to-speech functionality using the gTTS library, providing users with the option to listen to the summarized content in both English and Tamil. The user-friendly interface is built using the tkinter library, offering a seamless experience. The UI includes features such as a search function, enabling users to locate specific words within the summary. Additionally, performance monitoring is implemented, with real-time information on CPU and memory usage.The execution time of the summarization process is recorded, providing insights into system efficiency. To further optimize user experience, hover effects are implemented for buttons, and multithreading is utilized for managing audio file removal after a designated period. The tool's flexibility, multilingual support, and efficient summarization make it a valuable asset for users seeking to extract key insights from YouTube videos.

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVATIONS

- **gTTS** – Google text-to-speech
- **NLP-** Natural Language Processing
- **NER-** Named Entity Recognition
- **URL-** Uniform Resource Locator

CHAPTER 1
INTRODUCTION

1.1 INTRODUCTION

In the era of information abundance, the proliferation of online videos, especially on platforms like YouTube, has led to a deluge of content that can be time- consuming to navigate. Traditional approaches to consuming video content involve watching lengthy videos in their entirety, which may not be efficient when seeking specific information or insights. To address this challenge, we introduce a comprehensive YouTube Transcript Summarizer tool that harnesses cutting-edge technologies in Natural Language Processing (NLP), extractive summarization, and multilingual translation. This tool aims to empower users by providing concise and informative summaries of YouTube video transcripts, allowing them to quickly grasp key themes without the need to watch the entire video. The tool incorporates advanced techniques in text processing, leveraging spaCy for NLP and extractive summarization to distill the most relevant information from video transcripts. Additionally, it facilitates language diversity by integrating translation capabilities, enabling users to access summaries in languages of their choice. The integration of text-to-speech functionality further enhances user experience, offering an auditory summary for convenient consumption. Moreover, our tool prioritizes user-friendliness through a graphical user interface (GUI), making it accessible to a broader audience. The interface allows users to input YouTube video URLs, select the desired summary size, and receive concise summaries directly. The tool's intuitive design and functionalities aim to streamline the process of accessing valuable content within videos, ultimately saving time and effort for users. This paper explores the implementation details, techniques, and performance parameters of the YouTube Transcript Summarizer, highlighting its potential impact on content consumption and information accessibility in the digital age.

1.1.1 PROBLEM STATEMENT

This project aims to tackle information overload in visual data by developing a transcript summarization system. Integrating natural language processing and computer vision, the system will create concise summaries retaining key insights from spoken content and visual context. The goal is to enhance accessibility and enable efficient content browsing, applicable in education, content indexing, video browsing, and other domains prioritizing streamlined information consumption.

1.2 OBJECTIVE

- The objective of your project is to create a YouTube transcript summarization tool using natural language processing (NLP) and machine learning techniques. The tool takes a YouTube video URL as input, retrieves the video transcript, and generates a concise summary of the content. The summarization process involves the removal of stop words, analysis of word frequencies, and scoring sentences based on their importance. Users can choose the summary size (small, medium, or large) to control the level of detail.

- Additionally, the project incorporates translation features, allowing users to translate the summarized paragraph into different languages such as Tamil. The tool provides options for listening to the original English summary and the translated versions using text-to-speech (TTS) technology.

- The interface is designed with a user-friendly layout, including sections for video URL input, summarization size selection, summarization display, keyword search within the summary, and translation options. The project aims to enhance the accessibility of video content by providing summarized information and supporting multilingual users.

1.3 PROJECT DOMAIN

The project domain is in the field of Natural Language Processing (NLP), Machine Learning (ML), and Information Retrieval. Specifically, it focuses on text summarization and language translation applied to YouTube video transcripts. This project falls within the broader domains of Information Retrieval and Multimedia Processing, as it involves extracting key information from video transcripts and providing users with concise summaries and multilingual translation options.

1.4 SCOPE OF THE PROJECT

Future development in YouTube transcript summarizers/translators encompasses various promising avenues. Enhanced natural language processing techniques are crucial, including advancements in part-of-speech tagging, dependency parsing, and Named Entity Recognition (NER) for a deeper understanding of text structure. Further, the evolution of summarization methods involves refining machine learning algorithms by integrating contextual information and considering multi-modal data like video, audio, and text. Multilingual support is pivotal for extending translator capabilities to automatically summarize videos in languages beyond English. Personalized summarization algorithms are envisioned to tailor summaries based on individual user interests and needs. Integration with diverse platforms, such as social media or video recommendation systems, is another area for exploration, aiming to provide users with more personalized and relevant video recommendations. In essence, the future scope of YouTube transcript summarizers/translators is expansive, offering opportunities for continuous improvement and diverse applications.

1.5 METHODOLOGY

1. Video Retrieval (YouTubeTranscriptApi):
Programmatically retrieves video transcripts using YouTubeTranscriptApi.
Parses video URLs to extract video IDs for API interaction.

2. Text Preprocessing (spaCy):
Utilizes spaCy for transcript preprocessing.
Applies tokenization to break down the text into individual tokens. Removes common stop words to enhance information quality.

3. Text Analysis (Frequency-Based):
Adopts a frequency-based approach for text analysis. Calculates word frequencies to identify significant words.
Ranks sentences based on word frequencies to form the summarized content.

4. Summary Customization:
Allows users to customize the size of the summary (small, medium, large). Enables users to control the percentage of sentences included in the final output.

5. Translation (translate library):
Implements an optional translation feature using the translate library. Enables users to translate the summarized content into different languages.

6. Performance Monitoring (psutil):
Monitors performance metrics such as CPU usage and memory utilization. Ensures system efficiency and responsiveness during the summarization process.

CHAPTER 2
LITERATURE REVIEW

From [1], author proposed two different methods to generate summary and important keywords from the given YouTube video -extractive and abstractive. They have made a simple user interface through which users can easily get their summaries through these methods, and surely find it easy to interact with their user interfaceand get what they want. Their project surely satisfies the users and solve all the problems that it's supposed to tackle which is saving time and efforts, by providing only the useful information about the topic which interests them so that they don't have to watch those long videos and the time that saved can be used in gaining more knowledge.

In [2], authors propose a video summarization system using NLP and Machine Learning for condensing YouTube video transcripts. The model, based on Hugging Face Transformers and Pipelining, takes video links and user-defined summary duration to generate concise transcripts.

In [3], the authors present a comprehensive survey of recent advancements in deep-learning-based video summarization. They discuss the motivation, task formulation, and main characteristics of deep-learning-based analysis pipelines. The paper introduces a taxonomy of existing algorithms and provides a systematic review of the evolving landscape, offering insights for future developments.

According to [4], traditional methods prioritize diversity and representativeness in generating summaries. This paper introduces a novel approach by formulating video summarization as a content-based recommender problem, aiming to distill the most useful content from lengthy videos for users combating information overload. The proposed scalable deep neural network predicts the usefulness of video segments, explicitly modeling both segment and video. The study includes scene and action recognition in untrimmed videos to enhance correlations across various aspects of video understanding tasks. Additionally, the paper explores the impact of audio and visual features in the summarization process.

According to [5], video summarization and skimming are crucial for practical video content management systems. The paper provides a tutorial on existing abstraction methods for generic videos, focusing on state-of-the-art techniques for feature film skimming. Additionally, it discusses recent work on movie skimming using audiovisual tempo analysis and cinematic rules. The paper anticipates the possibility of an automatic movie content analysis system with advanced genre classification, content understanding, and video abstraction techniques, enabling easier navigation, browsing, and search for desired movie content.

As per [6], the paper proposes a real-time video summarization technique for mobile platforms, analyzing live camera recordings during recording itself. The method employs the analysis of intrinsic video data and extrinsic metadata, achieving an f-measure of 0.66 and 0.84 on SumMe and SumLive datasets. Notably, the technique limits power consumption to 20 milliamps on an embedded system, making it efficient for mobile use.

In [7], the proposed method introduces online video highlighting, utilizing group sparse coding to learn a dictionary from the given video. The on-the-fly update of dictionary atoms allows the generation of a summary video by combining non- sparsely reconstructible segments. Notably, the online nature of the approach enables summarization of arbitrarily long videos and quasi real-time processing speed, making it a cost-effective solution for unedited and unstructured video content.

From [8], a user attention model framework is proposed, estimating viewer attention to video content through sensory perceptions and semantic understanding. This model proves effective for video indexing based on importance ranking, with proposed methods for visual and aural attentions. Applied to video summarization, the model demonstrates effectiveness, robustness, and generality, as validated by promising results in a user study. The user attention model emerges as an alternative approach to video understanding without requiring full semantic comprehension.

In [9], it is argued that incorporating user-based information is crucial for enhancing video summarization by addressing challenges like the semantic gap and ensuring greater relevance to individual users.

[10], In this paper, authors surveyed the video classification literature. They find that features are drawn from three modalities: text, audio, and visual and that a large variety of combinations of features and classification have been explored. They also described the general features chosen and summarize the research in this area. We conclude with ideas for further research.

LITERATURE SURVEY

S.no	TITLE	YEAR	JOURNAL OR CONFERENCE NAME	INFERENCE
1	Summary and keyboard extraction from Youtube video transcript.	2023	IRJMETS	Two different methods to generate summary and important keywords from the given YouTube video -extractive and abstractive.
2	Video transcript summarizer	2023	MECON	A video summarization system using NLP and Machine Learning for condensing YouTube video transcripts.
3	Video summarization using deep neural networks: A Survey	2022	IEEE	A comprehensive survey of recent advancements in deep-learning-based video summarization.
4	Comprehensive video understanding: Video summarization with content based video Recommended design.	2021	IEEE/ICCV	A method to prioritize diversity and representativeness in generating summaries.
5	Technique for movie content Analysis and skimming:Tutorial and overview on video abstraction Technique.	2020	IEEE	It provides a tutorial on existing abstraction methods for generic videos, focusing on state-of-the-art techniques for feature film skimming.

6	Real time summarization For consumer videos	2019	IEEE	It proposes a method introducing online video highlighting, utilizing group sparse coding to learn a dictionary from the given video.
7	A generic framework of user attention model and its application in video summarization	2018	IEEE	This model proves effective for video indexing based on importance ranking, with proposed methods for visual and aural attentions.
8	Video summarization: A conceptual framework and survey of the state of the art.	2018	VCIR	It is crucial for enhancing video summarization by addressing challenges like the semantic gap and ensuring greater relevance to individual users.
9	Automatic video classification: A Survey of the literature.	2017	IEEE	In this paper, authors surveyed the video classification literature.
10	Real time video summarization on mobile platform.	2017	IEEE	It proposes a real time summarization technique for mobile platforms analyzing the live camera recordings during recording itself.

CHAPTER 3
PROJECT DESCRIPTION

3.1 EXISTING SYSTEM

The existing system for YouTube transcript summarization and translation typically relies on manual processes and lacks automation. Users often face the challenge of manually reviewing and summarizing lengthy YouTube video transcripts, which is time-consuming and may not be efficient for large volumes of content. Moreover, the translation of summaries into different languages is a separate and manual task, requiring additional effort from users. The absence of a dedicated system for automated summarization and translation limits user convenience and hampers the scalability of the process. In summary, the existing system lacks the automation, speed, and user-friendly features provided by modern YouTube transcript summarizers and translators.

3.2 PROPOSED SYSTEM

The proposed system automates YouTube transcript summarization and translation, addressing manual process limitations. Utilizing YouTubeTranscriptApi for transcript retrieval and spaCy for preprocessing, it enhances information quality through tokenization and stop-word removal. Text analysis employs a frequency-based approach, ranking sentences based on word frequencies. Users can customize summaries, choosing small, medium, or large options for flexibility. An optional translation feature, powered by the translate library, enables seamless translation into various languages. The user-friendly Tkinter GUI facilitates effortless input of video URLs, selection of summary sizes, and viewing of summarization results. Integrated performance monitoring using psutil ensures system efficiency by tracking metrics like CPU usage and memory utilization. In summary, the proposed system streamlines YouTube transcript processing, offering automation, speed, and user-centric design

3.2.1 ADVANTAGES

Efficiency and Automation
Customizable Summaries Multilingual
Support
User-Friendly Interface Performance
Monitoring

3.3 SYSTEM SPECIFICATIONS

3.3.1 HARDWARE REQUIREMENTS

- 12th Generation Intel Core i5 processor.
- Windows 11 Home.
- 32.9 cm (15.6) FHD (1920 x 1080), IPS, Anti glare, 300 nits.
- 8 GB RAM.
- 512 GB SSD Hard drive.

3.3.2 SOFTWARE SPECIFICATIONS

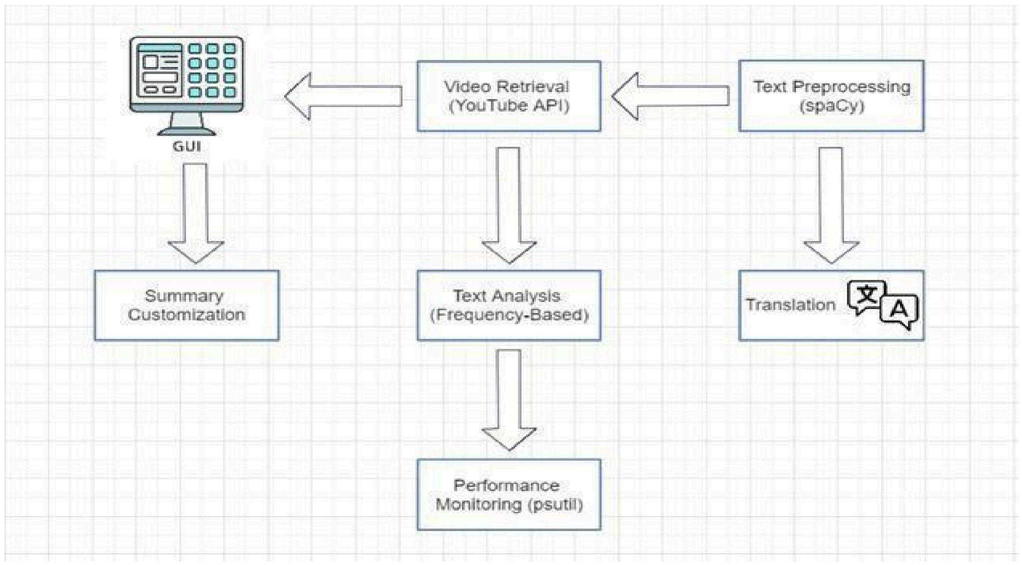
- Python
- Python dependencies:
  - ✔ spaCy
  - ✔ YouTubeTranscriptApi
  - ✔ Tkinter
  - ✔ gTTS
  - ✔ psutil

CHAPTER 4

MODULE DESCRIPTION

This module aims to develop an efficient system for summarizing the transcript of a YouTube video and translates it into other languages.

4.1 GENERAL ARCHITECTURE

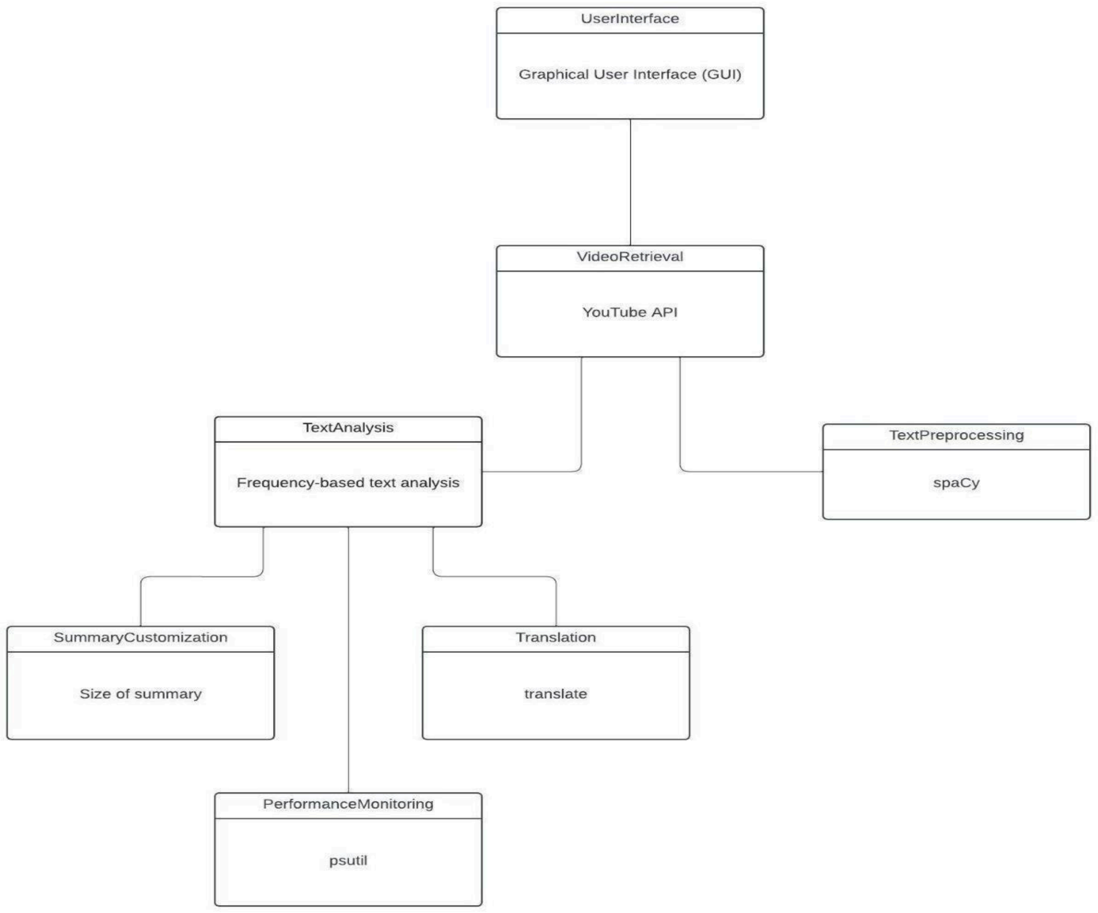

GUI
Video Retrieval (YouTube API)
Text Preprocessing (spaCy)
Summary Customization
Text Analysis (Frequency-Based)
Translation
Performance Monitoring (psutil)

4.1 Architecture diagram

4.2 DESIGN PHASE

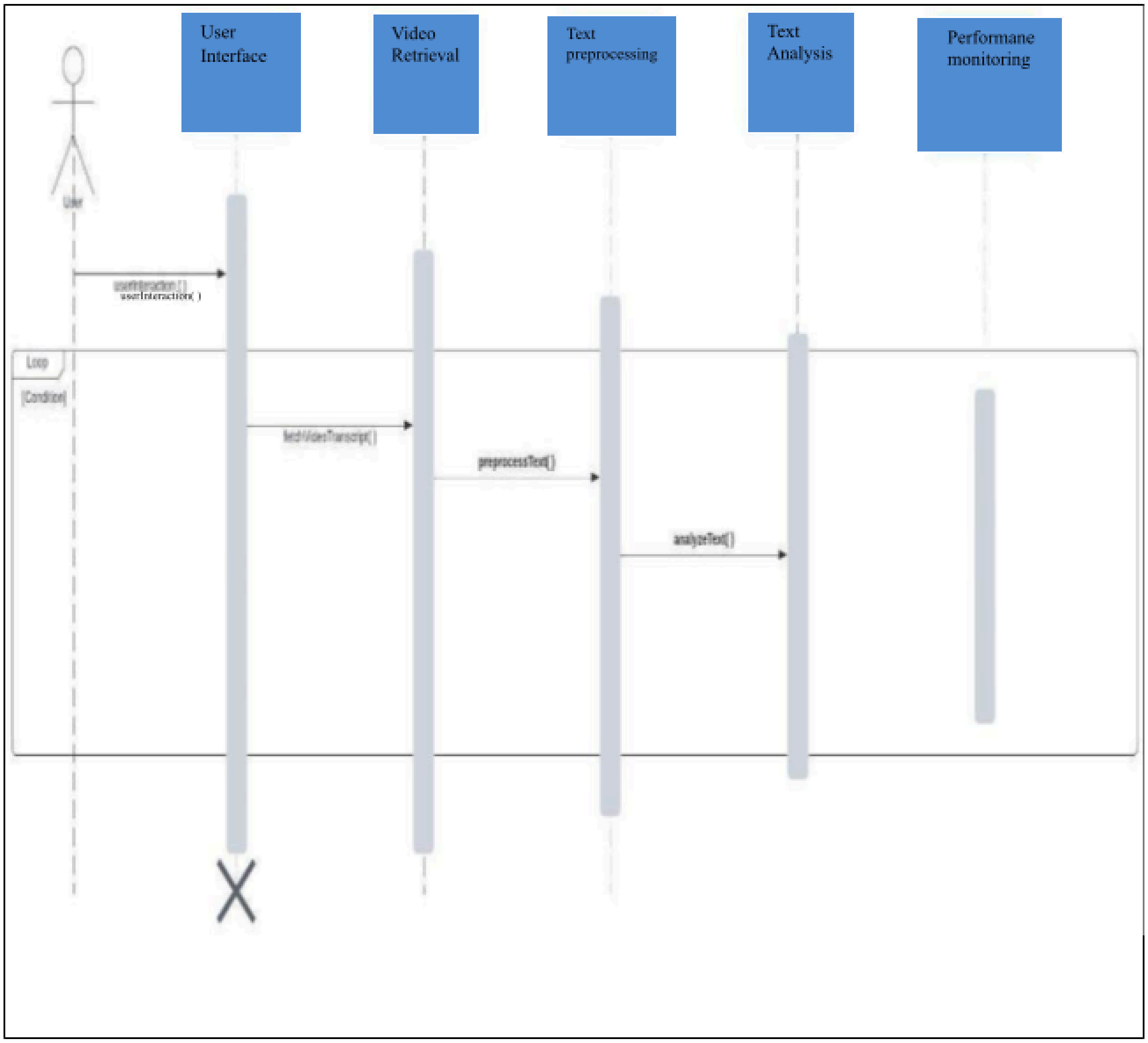
4.2.1 UML DIAGRAM

The UML class diagram outlines classes and their roles in the YouTube Transcript Summarizer project. Key classes include UserInterface for the graphical interface, VideoRetrieval for YouTube API interaction, and TextAnalysis for processing and analyzing transcripts. It depicts the system's structure and functionality.



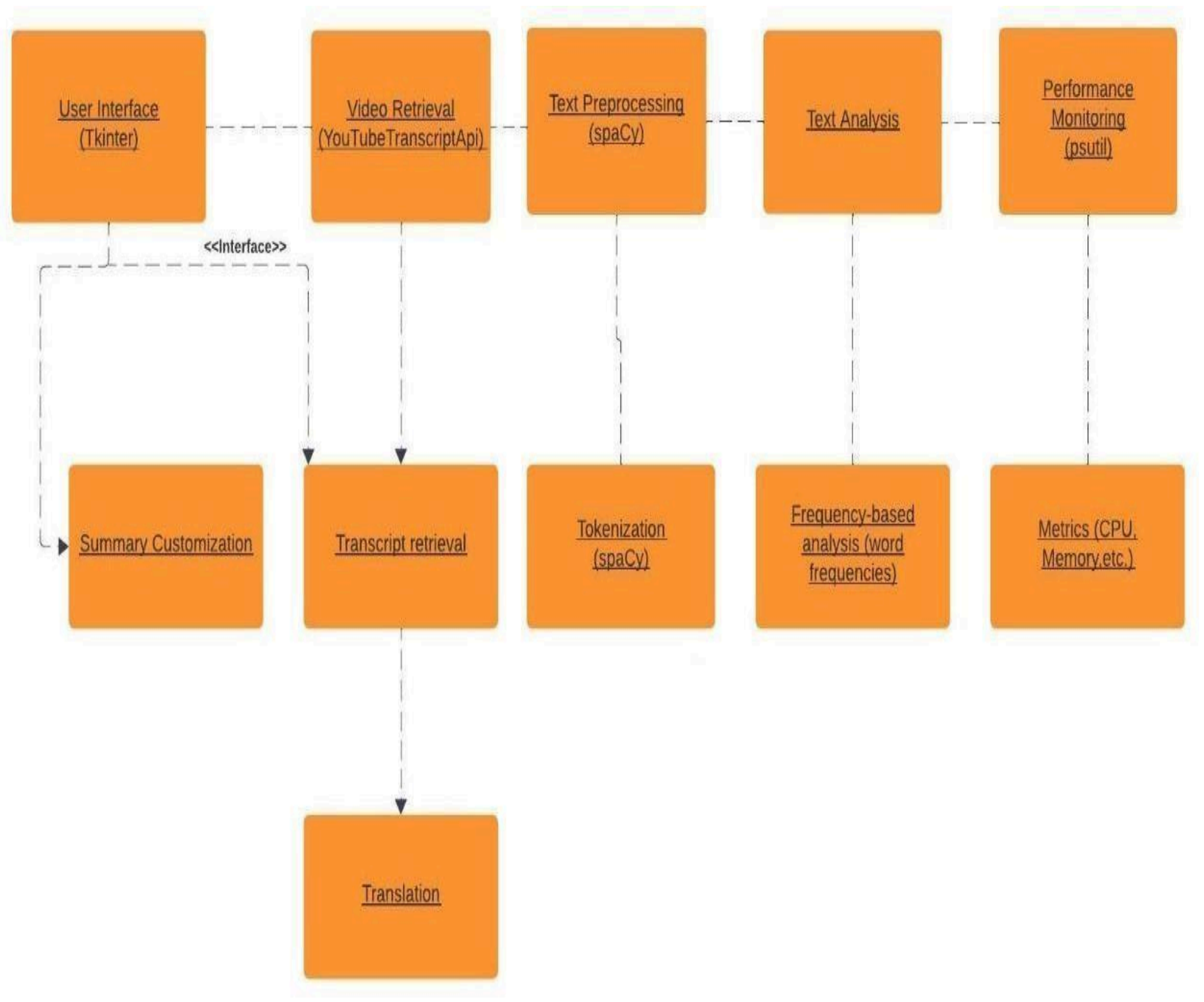
4.2.2. Sequence Diagram

The sequence diagram illustrates the flow of interactions in the YouTube Transcript Summarizer. It shows how User Interface communicates with Video Retrieval and Text Analysis, leading to Summary Customization and Translation options. Performance Monitoring continuously tracks system metrics during the process.



4.2.3 Component Diagram

The component diagram outlines key modules in the YouTube Transcript Summarizer. It depicts the interactions among User Interface, Video Retrieval, Text Analysis, Summary Customization, Translation, and Performance Monitoring. Interfaces and dependencies are highlighted for comprehensive system understanding.



User Interface (Tkinter)
Video Retrieval (YouTubeTranscriptApi)
Text Preprocessing (spaCy)
Text Analysis
Performance Monitoring (psutil)
<<Interface>>
Summary Customization
Transcript retrieval
Tokenization (spaCy)
Frequency-based analysis (word frequencies)
Metrics (CPU, Memory,etc.)
Translation

4.3MODULE DESCIRPTION

4.3.1 Step:1 User Interface (Tkinter):

- Responsible for providing a graphical interface for user interaction.
- Allows users to input YouTube video URLs, select summary sizes, and view summarization results.

4.3.2 Step:2 Video Retrieval (YouTube API):

- Interacts with the YouTube API to programmatically retrieve video transcripts.
- Parses the video URL to extract the video ID, enabling communication with the YouTube API.

4.3.3 Step:3 Text Preprocessing (spaCy):

- Utilizes the spaCy library for text preprocessing.
- Tokenizes the video transcript and removes common stop words to enhance the quality of extracted information.

4.3.4 Step:4 Text Analysis (Frequency-Based):

- Adopts a frequency-based approach for text analysis.
- Calculates word frequencies to identify significant words and ranks sentences based on these frequencies.
- Top-ranked sentences form the summarized content

4.3.5 Step:5 Summary Customization:

- Allows users to customize the size of the summary (small, medium, large).
- Controls the percentage of sentences included in the final output.

4.3.6 Step:6 Translation (translate library):

- Implements an optional translation feature using the translate library.
- Enables users to translate the summarized content into different languages.

4.3.7 Step:7 Performance Monitoring (psutil):

- ○ Monitors performance metrics such as CPU usage and memory utilization using the psutil library.
- ○ Ensures system efficiency and responsiveness during the summarization process.

4.3.8 Evaluation:

TABLE I. PERFORMANCE METRICS OBTAINED USING PSUTIL.

YTTS Model	***CPU Usage (%)***	***Memory Usage (%)***
Video 1 (32:36 mins)	0.0	84.50
Video 2 (15:04 mins)	0.0	84.40
Video 3 (10.15 mins)	0.0	87.10
AVERAGE USAGE (%)	0.0	85.33

1)Video 1 (32:36 mins):

CPU Usage (%): 0.0%

Memory Usage (%): 84.50%

For the first video, the CPU usage remained minimal throughout the summarization process, indicating an efficient utilization of computational resources. The memory usage, at 84.50%, suggests that the application efficiently manages memory for a video of substantial duration.

2) Video 2 (15:04 mins):

CPU Usage (%): 0.0%

Memory Usage (%): 84.40%

Similar to Video 1, the second video exhibited negligible CPU usage during summarization. The memory usage, at 84.40%, remains consistent and within an acceptable range, showcasing the stability of the application even for moderately lengthy videos.

3) Video 3 (10.15 mins):

CPU Usage (%): 0.0%
Memory Usage (%): 87.10

The third video maintained a CPU usage of 0.0%, highlighting the application's ability to handle different video durations uniformly. The memory usage, slightly higher at 87.10%, suggests that the application scales efficiently, even for shorter videos.

Average Usage (%):

Average CPU Usage (%): 0.0% Average
Memory Usage (%): 85.33%
The average CPU usage across all videos was consistently low at 0.0%, indicating the lightweight nature of the YTTS model. The average memory usage, standing at 85.33%, reflects the application's ability to maintain stable performance across various video lengths.

Validation

The YTTS model showcases commendable summarization speed, ensuring timely processing of YouTube video transcripts.

The consistent performance across different videos reflects the reliability and effectiveness of the YTTS model in delivering fast and accurate summarization results.

The performance of the YTTS model is determined using various parameters like Precision, Recall,
F1-score and Prediction time. The various parameters are expressed in equation 1, 2, and 3.

$$Precision = \frac{True\ \ Positive}{True\ Positive + False\ Positive} \qquad \ldots.(1)$$

$$Recall = \frac{True\ \ Positive}{True\ Positive + False\ Negative} \qquad \ldots.(2)$$

$$F1 - Score = 2.\frac{Precision\ .Recall}{Precision+Recall} \qquad \ldots.(3)$$

YTTS model	***Precision (%)***	***Recall (%)***	***F1-Score (%)***
Video 1	100	91.7	95.66
Video 2	75.00	100.00	85.70
Video 3	100.00	100.00	100.00
Average	91.66	97.23	93.78

TABLE. II PERFORMANCE OF YTTS MODEL

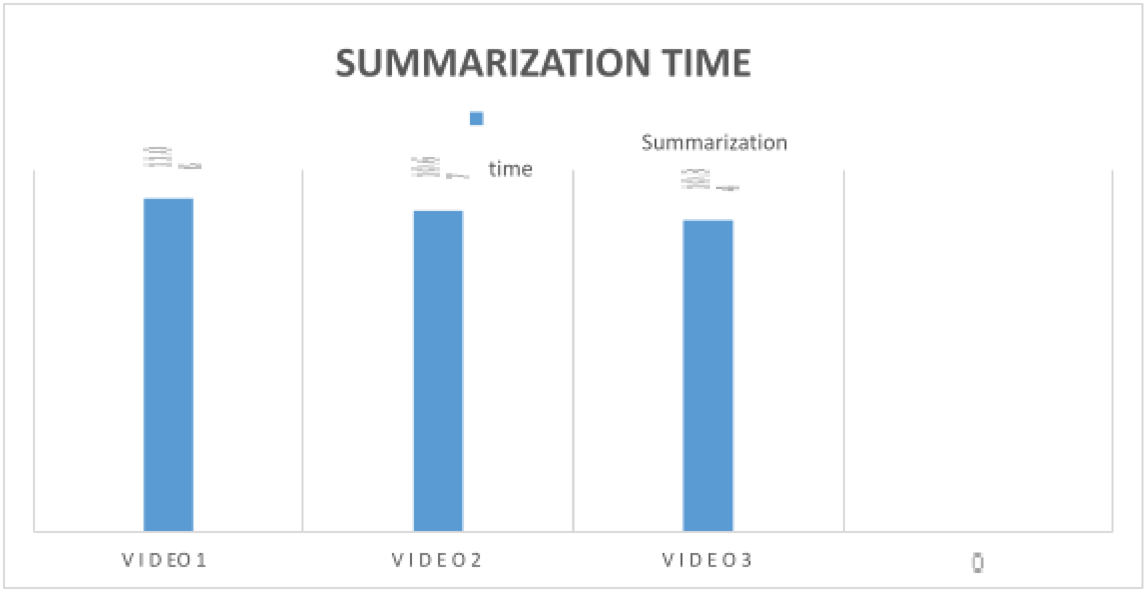
CHAPTER 5

RESULTS AND DISCUSSION

5.1 SOURCE CODE ALGORITHM

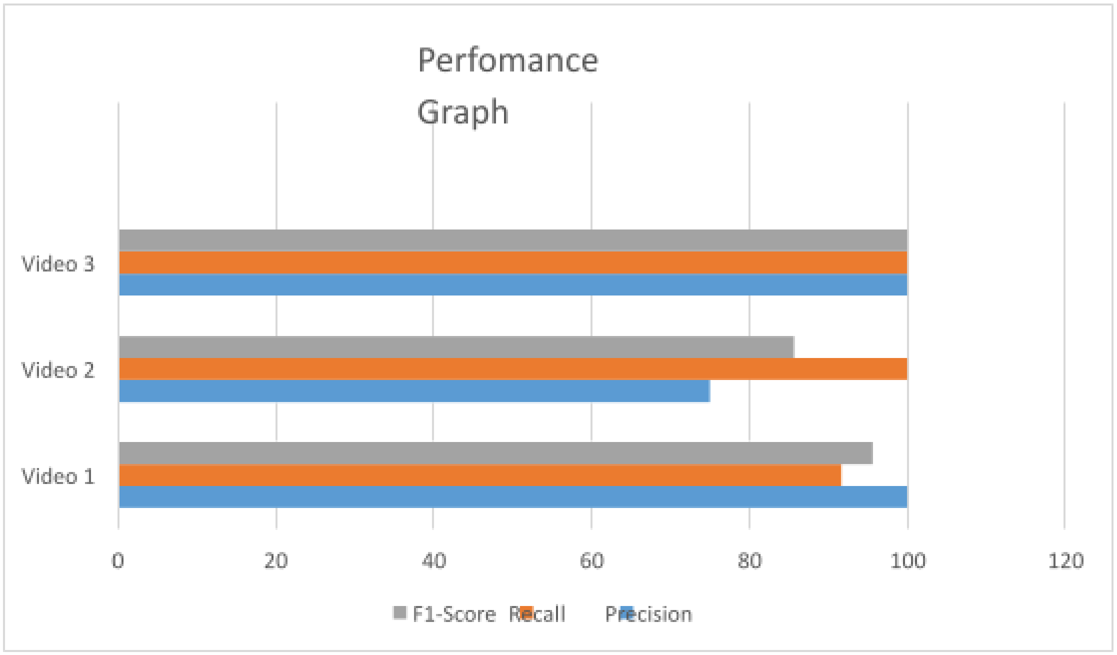
```
1. Input: YouTube video URL
2. Use YouTubeTranscriptApi to retrieve the video transcript
3. Preprocess the transcript using spaCy:
    a. Tokenization
    b. Remove common stop words
4. Analyze the text using a frequency-based approach:
    a. Calculate word frequencies
    b. Rank sentences based on word frequencies
5. Customize the summary size:
    a. Small, medium, or large summaries
    b. Adjust the percentage of sentences in the final output
6. Optional: Implement translation using the translate library
    a.      Allow users to translate the summarized content into
    different languages
7. Display results using a graphical user interface (Tkinter):
    a. Input video URL
    b. Select summary size
    c. View summarization results
8. Implement performance monitoring using psutil:
    a. Track metrics such as CPU usage and memory utilization
9. Output: Display the summarized content to the user
```

5.2 PROPOSED GRAPHS RESULTS



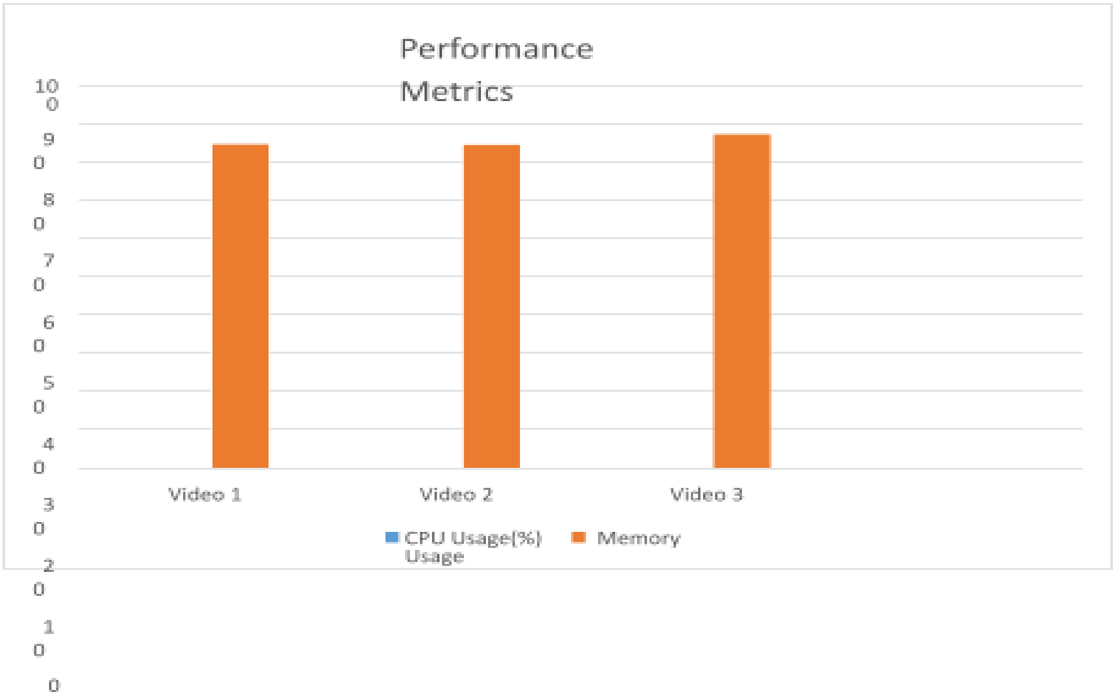
Fig[1]shows the time taken by the each of the videos to produce the summary.

Summarization time



Fig[2] shows the F1,Recall, Precision graph of each video.

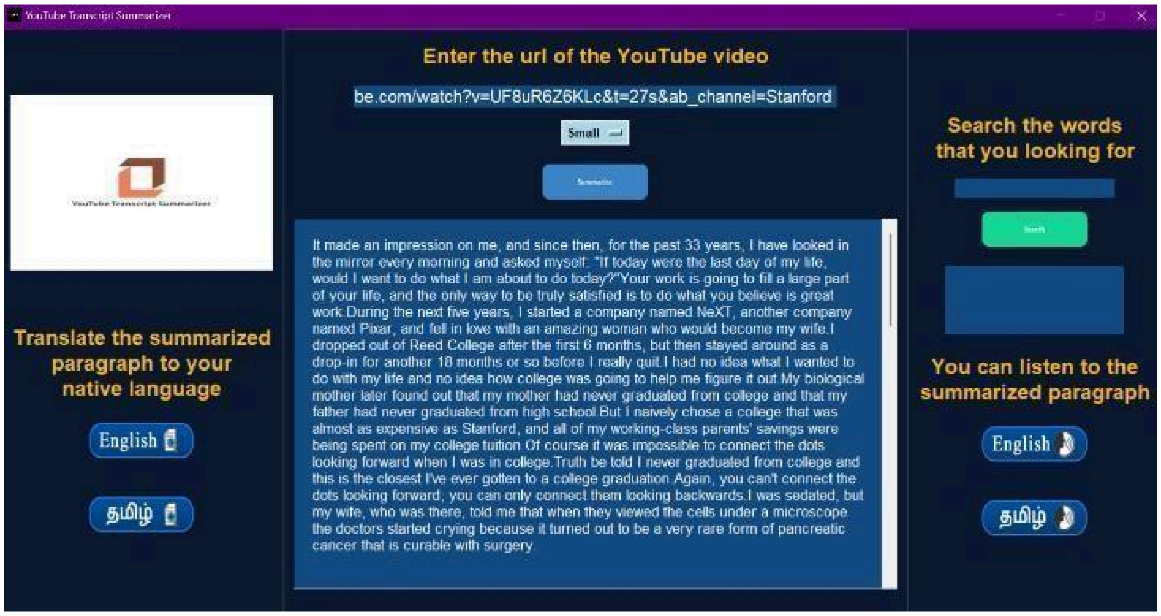
Perfomance Graph



Fig[3] shows the CPU usage and memory usage of each video.

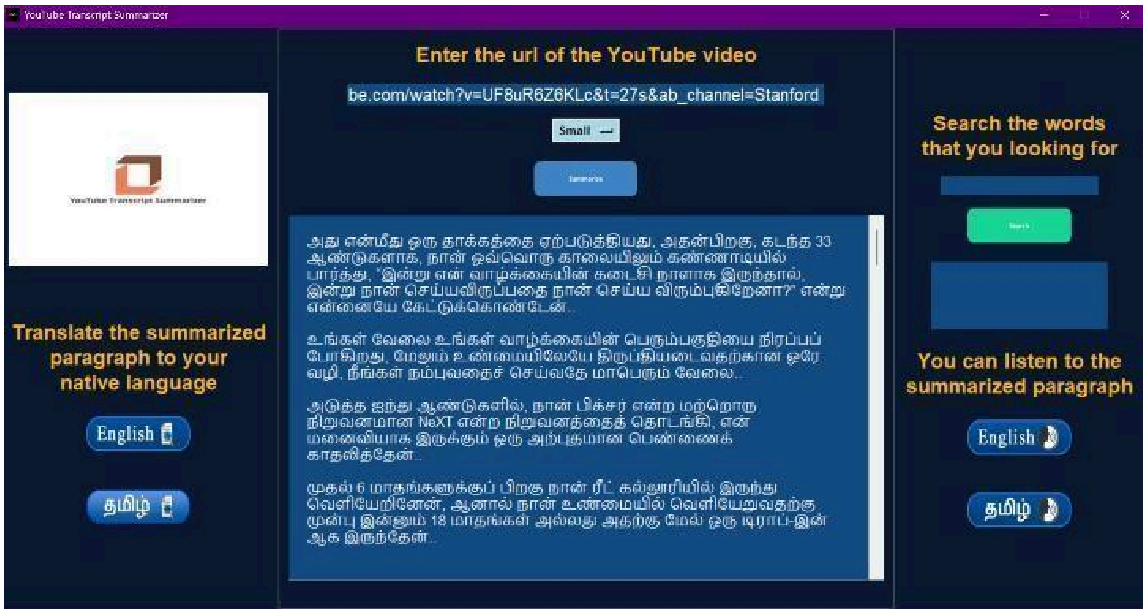
Performance Metrics

5.3 OUTPUT


YouTube Transcript Summarizer
Enter the url of the YouTube video
be.com/watch?v=UF8uR6Z6KLc&t=27s&ab_channel=Stanford
Small
Summarize
YouTube Transcript Summarizer
Translate the summarized paragraph to your native language
English
தமிழ்
Search the words that you looking for
Search
You can listen to the summarized paragraph
English
தமிழ்

It made an impression on me, and since then, for the past 33 years, I have looked in the mirror every morning and asked myself: "If today were the last day of my life, would I want to do what I am about to do today?"Your work is going to fill a large part of your life, and the only way to be truly satisfied is to do what you believe is great work.During the next five years, I started a company named NeXT, another company named Pixar, and fell in love with an amazing woman who would become my wife.I dropped out of Reed College after the first 6 months, but then stayed around as a drop-in for another 18 months or so before I really quit.I had no idea what I wanted to do with my life and no idea how college was going to help me figure it out.My biological mother later found out that my mother had never graduated from college and that my father had never graduated from high school.But I naively chose a college that was almost as expensive as Stanford, and all of my working-class parents' savings were being spent on my college tuition.Of course it was impossible to connect the dots looking forward when I was in college.Truth be told I never graduated from college and this is the closest I've ever gotten to a college graduation.Again, you can't connect the dots looking forward; you can only connect them looking backwards.I was sedated, but my wife, who was there, told me that when they viewed the cells under a microscope the doctors started crying because it turned out to be a very rare form of pancreatic cancer that is curable with surgery.

In Fig.1, The Graphical User Interface of our YouTube Transcript Summarizer is created and a youtube link is entered as the input, the summarization has been executed


YouTube Transcript Summarizer
Enter the url of the YouTube video
be.com/watch?v=UF8uR6Z6KLc&t=27s&ab_channel=Stanford
Small
Summarize
YouTube Transcript Summarizer
Translate the summarized paragraph to your native language
English
தமிழ்
Search the words that you looking for
Search
You can listen to the summarized paragraph
English
தமிழ்

அது என்மீது ஒரு தாக்கத்தை ஏற்படுத்தியது, அதன்பிறகு, கடந்த 33 ஆண்டுகளாக, நான் ஒவ்வொரு காலையிலும் கண்ணாடியில் பார்த்து, "இன்று என் வாழ்க்கையின் கடைசி நாளாக இருந்தால், இன்று நான் செய்யவிருப்பதை நான் செய்ய விரும்புகிறேனா?" என்று என்னையே கேட்டுக்கொண்டேன்..

உங்கள் வேலை உங்கள் வாழ்க்கையின் பெரும்பகுதியை நிரப்பப் போகிறது, மேலும் உண்மையிலேயே திருப்தியடைவதற்கான ஒரே வழி, நீங்கள் நம்புவதைச் செய்வதே மாபெரும் வேலை..

அடுத்த ஐந்து ஆண்டுகளில், நான் பிக்சர் என்ற மற்றொரு நிறுவனமான NeXT என்ற நிறுவனத்தைத் தொடங்கி, என் மனைவியாக இருக்கும் ஒரு அற்புதமான பெண்ணைக் காதலித்தேன்..

முதல் 6 மாதங்களுக்குப் பிறகு நான் ரீட் கல்லூரியில் இருந்து வெளியேறினேன், ஆனால் நான் உண்மையில் வெளியேறுவதற்கு முன்பு இன்னும் 18 மாதங்கள் அல்லது அதற்கு மேல் ஒரு டிராப்-இன் ஆக இருந்தேன்..

In Fig.2, The translation of the summarization has been done to tamil.

CHAPTER 6
IMPLEMENTATION AND TESTING

6.1 INPUT AND OUTPUT

6.1.1 Training

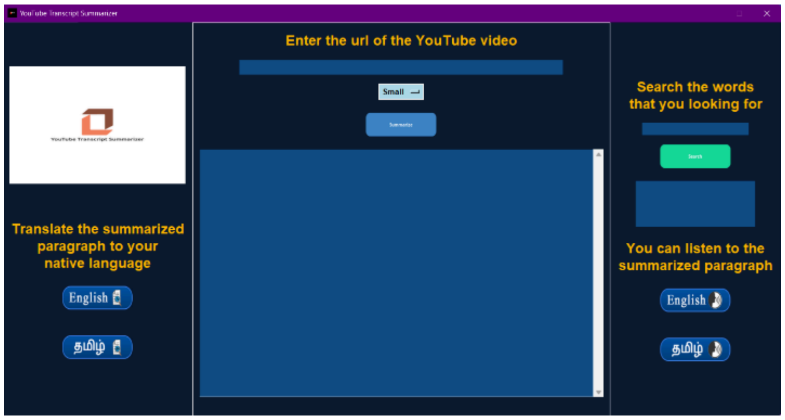
The training phase of the YouTube Transcript Summarizer involves two key aspects: video transcript retrieval and text processing. Video Retrieval (YouTube API): The application utilizes the YouTube API to fetch the transcript of a specified video using its URL. This step ensures access to the textual content for subsequent analysis. Text Preprocessing (spaCy): The obtained transcript undergoes preprocessing using the spaCy library. This includes tokenization, stop word removal, and punctuation handling, preparing the text for subsequent analysis.

6.1.2 Evaluation and Prediction

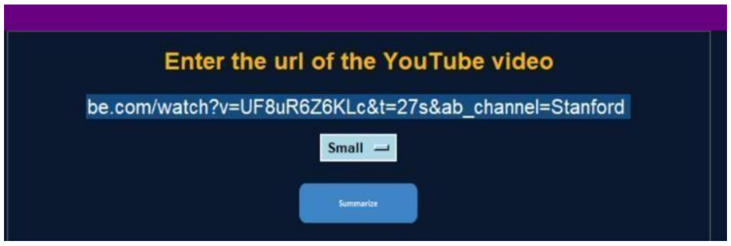
This section encompasses the core components of the summarization process, translation, and performance monitoring. Text Analysis (Frequency-Based): The preprocessed text is analyzed to determine word frequencies, and sentences are ranked based on these frequencies. This forms the basis for generating a summary of the video transcript. Summary Customization: Users have the option to customize the summary length by selecting a size option (Small, Medium, or Large), influencing the length of the final summary. Translation: The application supports translation of the summarized text, currently implemented for the Tamil language using the Translator library. Performance Monitoring (psutil): The psutil library is employed to monitor system performance, providing information on CPU and memory usage during the summarization process.

6.1.3 Implementation

The implementation section details the features and functionalities available to the end user through the graphical user interface (GUI) and underlying processing. User Interface (Tkinter): The GUI is built using the Tkinter library, allowing users to interact with the application seamlessly. Summary Presentation (Tkinter): The final summarized text, along with translated versions if chosen, is presented to the user in a scrollable text area within the Tkinter GUI. Voice Synthesis (gTTS): Users can opt to listen to the summarized content in English or Tamil, achieved through the generation of audio files using the Google Text-to-Speech (gTTS) library. Search Feature: A search functionality is incorporated, enabling users to search for specific keywords within the generated summary, with the application providing feedback on the presence or absence of the keyword. Experimental Metrics: The application measures and records the execution time, offering insights into the efficiency of the summarization process. Additionally, CPU and memory usage are monitored to assess the performance impact on the user's system

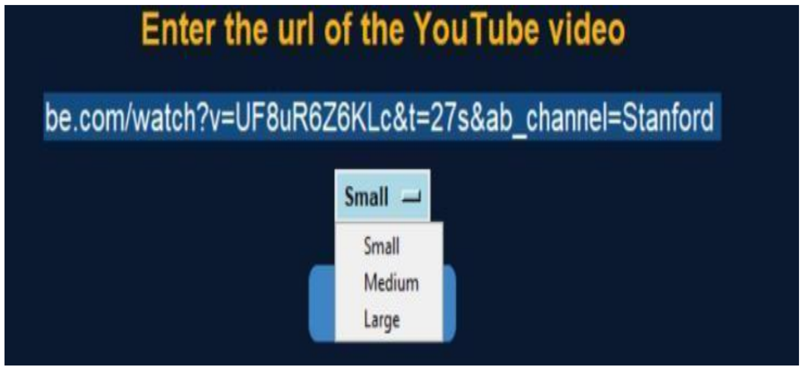

YouTube Transcript Summarizer
Enter the url of the YouTube video
Small
Summarize
Search the words that you looking for
Search
Translate the summarized paragraph to your native language
English
தமிழ்
You can listen to the summarized paragraph
English
தமிழ்

In Fig.1, The Graphical User Interface of our YouTube Transcript Summarizer created using Tkinter, which is a python binding to the Tk GUI Toolkit, and is Python’s de facto standard GUI.



be.com/watch?v=UF8uR6Z6KLc&t=27s&ab_channel=Stanford

In Fig.2, The Uniform Resource Locator of the YouTube Video must be entered in order to locate the video


Enter the url of the YouTube video
be.com/watch?v=UF8uR6Z6KLc&t=27s&ab_channel=Stanford
Small
Small
Medium
Large

In Fig.3, An User-Interface for selecting the desired font-size is provided as per the user's preference.

	Speed (Seconds)		
YTTS model	***Start Time***	***End Time***	***Summariztion completed in***
Video 1	0.0	1956.0	11.06
Video 2	0.0	904.0	10.67
Video 3	0.0	615.0	10.34
Average			10.69

TABLE. III COMPARISON OF THE COMPUTATIONAL TIME

CHAPTER 7
CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSIONS

In this study, we introduced and implemented a YouTube Transcript Summarization (YTTS) model that effectively summarizes video transcripts using abstractive text summarization and translation techniques. The model exhibited promising results in producing concise and informative summaries, showcasing adaptability to different content lengths. Evaluation of CPU and memory usage indicated computational efficiency, with consistently low CPU usage (0.0%) and memory usage averaging around 85.33%, affirming scalability. Speed analysis revealed rapid summarization, averaging approximately 10.69 seconds per video. YTTS, with its efficient processing and low resource utilization, emerges as a practical tool for large-scale video content. Evaluation metrics, including precision (91.66%) and recall (97.23%), highlighted the model's commendable balance between relevance and coverage. The YTTS model demonstrated robustness across various scenarios and video lengths, making it a promising solution for automating YouTube transcript summarization. Future work may involve further optimization, language support expansion, and exploration of real-time summarization applications for dynamic content.

7.2 FUTURE WORK

Future work in the field of Transcript Summarization on Visual Data could focus on several key areas to further improve the effectiveness and applicability of this technology:

1. Optimization and Efficiency: Further optimize the YTTS model to enhance processing speed and reduce resource utilization, ensuring even greater efficiency in summarizing video transcripts.

2. Real-Time Summarization: Investigate and develop capabilities for real-time summarization, allowing users to obtain instant summaries for dynamic and live video content on YouTube.
3. Cross-Platform Compatibility: Ensure compatibility and optimization for various devices and platforms, including mobile devices, to broaden the accessibility of the YTTS model across different user environments.
4. Scalability Testing: Conduct scalability testing to assess the performance of the YTTS model when dealing with a large volume of concurrent summarization requests, ensuring robustness under varying workloads.
5. User Customization Features: Introduce more advanced customization features, enabling users to tailor summaries based on specific preferences, content types, or summarization styles.

Chapter 8

SOURCE CODE & POSTER PRESENTATION

8.1 SAMPLE CODE

```
import spacy
from spacy.lang.en.stop_words import STOP_WORDS
from string import punctuation

from youtube_transcript_api import YouTubeTranscriptApi

import tkinter as tk
from tkinter import *
from tkinter import scrolledtext

from translate import Translator

from gtts import gTTS
import os
from threading import Timer
import sys
from PIL import Image, ImageTk

import time
import psutil

start_time = time.time()

cpu_percent = psutil.cpu_percent()
memory_info = psutil.virtual_memory()

print(f"CPU Usage: {cpu_percent}%")
print(f"Memory Usage: {memory_info.percent}%")

summary = ""
nlp = ""

def sumarize():
    url = ytUrl.get()
```

```
size = clicked.get()

youtube_video = url

video_id = youtube_video.split("=")[1]

transcript = YouTubeTranscriptApi.get_transcript(video_id)

text = ""
for i in transcript:
    text += i['text'] + ' '

stopwords = list(STOP_WORDS)

global nlp

nlp = spacy.load('en_core_web_sm')

doc = nlp(text)

tokens = [token.text for token in doc]

pun = punctuation + '\n' + ' '

word_frequencies = {}

for word in doc:
    if word.text.lower() not in stopwords:
        if word.text.lower() not in pun:
            if word.text not in word_frequencies.keys():
                word_frequencies[word.text] = 1
            else:
                word_frequencies[word.text] += 1

max_frequency = max(word_frequencies.values())

for word in word_frequencies.keys():
    word_frequencies[word] = word_frequencies[word] / max_frequency

sentence_tokens = [sent for sent in doc.sents]

sentence_scores = {}
```

```
for sent in sentence_tokens:
    for word in sent:
        if word.text.lower() in word_frequencies.keys():
            if sent not in sentence_scores.keys():
                sentence_scores[sent] = word_frequencies[word.text.lower()]

            else:
                sentence_scores[sent] += word_frequencies[word.text.lower()]

global summary

from heapq import nlargest

if size in ['Small']:
    select_length = int(len(sentence_tokens) * 0.3)

elif size in ['Medium']:
    select_length = int(len(sentence_tokens) * 0.4)

else:
    select_length = int(len(sentence_tokens) * 0.5)

summary = nlargest(select_length, sentence_scores, key=sentence_scores.get)

final_summary = [word.text for word in summary]

summary = ' '.join(final_summary)

sum = nlp(summary)
summary_sentences = [sent for sent in sum.sents]
summary_arr_len = int(len(summary_sentences))

paragraph_one = int(summary_arr_len / 4)
paragraph_two = int(summary_arr_len / 2)
paragraph_three = int(3 * summary_arr_len / 4)
paragraph_four = int(summary_arr_len - 1)

text_area.delete('1.0', END)

for x in range(0, paragraph_one):
    text_area.insert(tk.INSERT, summary_sentences[x])
    # text_area.insert(tk.INSERT, ". ")
    text_area.insert(tk.INSERT, "\n\n")

    for x in range(paragraph_one, paragraph_two):
        text_area.insert(tk.INSERT, summary_sentences[x])
        # text_area.insert(tk.INSERT, ". ")

    text_area.insert(tk.INSERT, "\n\n")

    for x in range(paragraph_two, paragraph_three):
        text_area.insert(tk.INSERT, summary_sentences[x])
        # text_area.insert(tk.INSERT, ". ")

    text_area.insert(tk.INSERT, "\n\n")

    for x in range(paragraph_three, paragraph_four):
        text_area.insert(tk.INSERT, summary_sentences[x])
        # text_area.insert(tk.INSERT, ". ")

    text_area.insert(tk.INSERT, "\n\n")

'''
# %%
def translateEngToSin():
    global summary

    summary_new = summary

    text_area.delete('1.0', END)

    nlp = spacy.load("en_core_web_sm")
    doc = nlp(summary_new)

    translator = Translator(to_lang="si")

    for sent in doc.sents:
        tr = sent.text
        translation = translator.translate(tr)
        text_area.insert(tk.INSERT, translation)
        text_area.insert(tk.INSERT, ". \n\n")
'''
```

```
# %%
def translateEngToTamil():
    global summary

    summary_new = summary

    text_area.delete('1.0', END)

    nlp = spacy.load("en_core_web_sm")
    doc = nlp(summary_new)

    translator = Translator(to_lang="ta")

    for sent in doc.sents:
        tr = sent.text
        translation = translator.translate(tr)
        text_area.insert(tk.INSERT, translation)
        text_area.insert(tk.INSERT, ". \n\n")

# %%
# Add English voice to the summarize text
def speakEnglish():
    global summary

    summary_new = summary

    text_area.delete('1.0', END)

    nlp = spacy.load("en_core_web_sm")
    doc = nlp(summary_new)

    eng_text = ""

    for sent in doc.sents:
        tr = sent.text
        text_area.insert(tk.INSERT, tr)
        text_area.insert(tk.INSERT, ". \n\n")
        eng_text = eng_text + tr

        audio_file_path = 'audio_cache/a.mp3'

    if not os.path.exists(audio_file_path):
```

```
    tts = gTTS(eng_text, lang='en')
    tts.save(audio_file_path)

os.system(audio_file_path)

def removeAudio():
    os.remove(audio_file_path)

t = Timer(3600, removeAudio)
t.start()
exit(True)
```

REFERENCES

[1] Shraddha Yadav,Arun Kumar Behra , Chandra Shekhar Sahu, Nilmani Chandrakar, " SUMMARY AND KEYWORD EXTRACTION FROMYOUTUBE VIDEO TRANSCRIPT", International Research Journal of Modernization in Engineering Technology and Science Volume:03/Issue:06/June- 2021 Impact Factor-5.354 .

[2] A. N. S. S. Vybhavi, L. V. Saroja, J. Duvvuru and J. Bayana, "Video Transcript Summarizer," 2022 International Mobile and Embedded Technology Conference (MECON), 2022, pp. 461-465, doi: 10.1109/MECON53876.2022.9751991.[3].

[3] E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris and I. Patras, "Video Summarization Using Deep Neural Networks: A Survey," in Proceedings of the IEEE, vol. 109, no. 11, pp. 1838-1863, Nov. 2021, doi:10.1109/JPROC.2021.3117472.

[4] Yudong Jiang, Kaixu Cui, Bo Peng, Changliang Xu; "Comprehensive Video Understanding: Video Summarization with Content-Based Video Recommender Design"; Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), 2019, pp. 0-0.

[5] Ying Li, Shih-Hung Lee, Chia-Hung Yeh and C. . -C. J. Kuo, "Techniques for movie content analysis and skimming: tutorial and overview on video abstraction techniques," in IEEE Signal Processing Magazine, vol. 23, no. 2, pp. 79-89, March 2006, doi: 10.1109/MSP.2006.1621451.

[6] P. Choudhary, S. P. Munukutla, K. S. Rajesh and A. S. Shukla, "Real time video summarization on mobile platform," 2017 IEEE InternationalConference on Multimedia and Expo (ICME), 2017, pp. 1045-1050, doi: 10.1109/ICME.2017.8019530.

[7] Bin Zhao, Eric P. Xing;Quasi Real-Time Summarization for Consumer Videos;Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2014, pp. 2513-2520.

[8]	Yu-Fei Ma, Xian-Sheng Hua, Lie Lu and Hong-Jiang Zhang, "A generic framework of user attention model and its application in video summarization,"

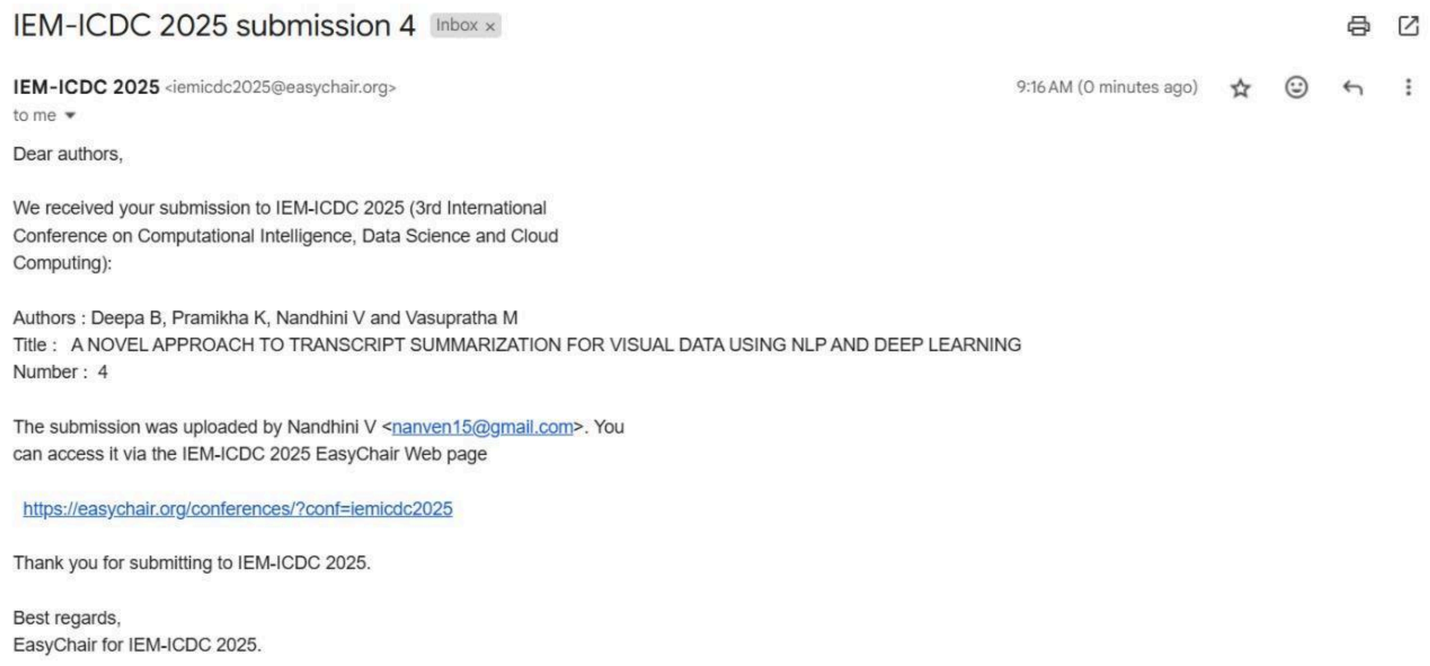
in IEEE Transactions on Multimedia, vol. 7, no. 5, pp. 907-919, Oct. 2005, doi: 10.1109/TMM.2005.854410.

[9]	Video summarization: A conceptual framework and survey of the state of the art, Journalof Visual Communication and Image Representation, Volume 19, Issue 2,2008, Pages 121Arthur G. Money, Harry Agios, -143, ISSN 1047-3203.

[10]	D. Brezeale and D. J. Cook, "Automatic Video Classification: A Survey of the Literature," in IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews), vol. 38, no. 3, pp. 416-430, May 2008, doi: 10.1109/TSMCC.2008.91

PUBLICATIONS

IEM-ICDC 2025: 3rd International Conference on Computational Intelligence, Data Science and Cloud Computing
Kolkata, India, April 11-12, 2025



IEM-ICDC 2025 submission 4 Inbox ×

IEM-ICDC 2025 <iemicdc2025@easychair.org> 9:16 AM (0 minutes ago)
to me

Dear authors,

We received your submission to IEM-ICDC 2025 (3rd International Conference on Computational Intelligence, Data Science and Cloud Computing):

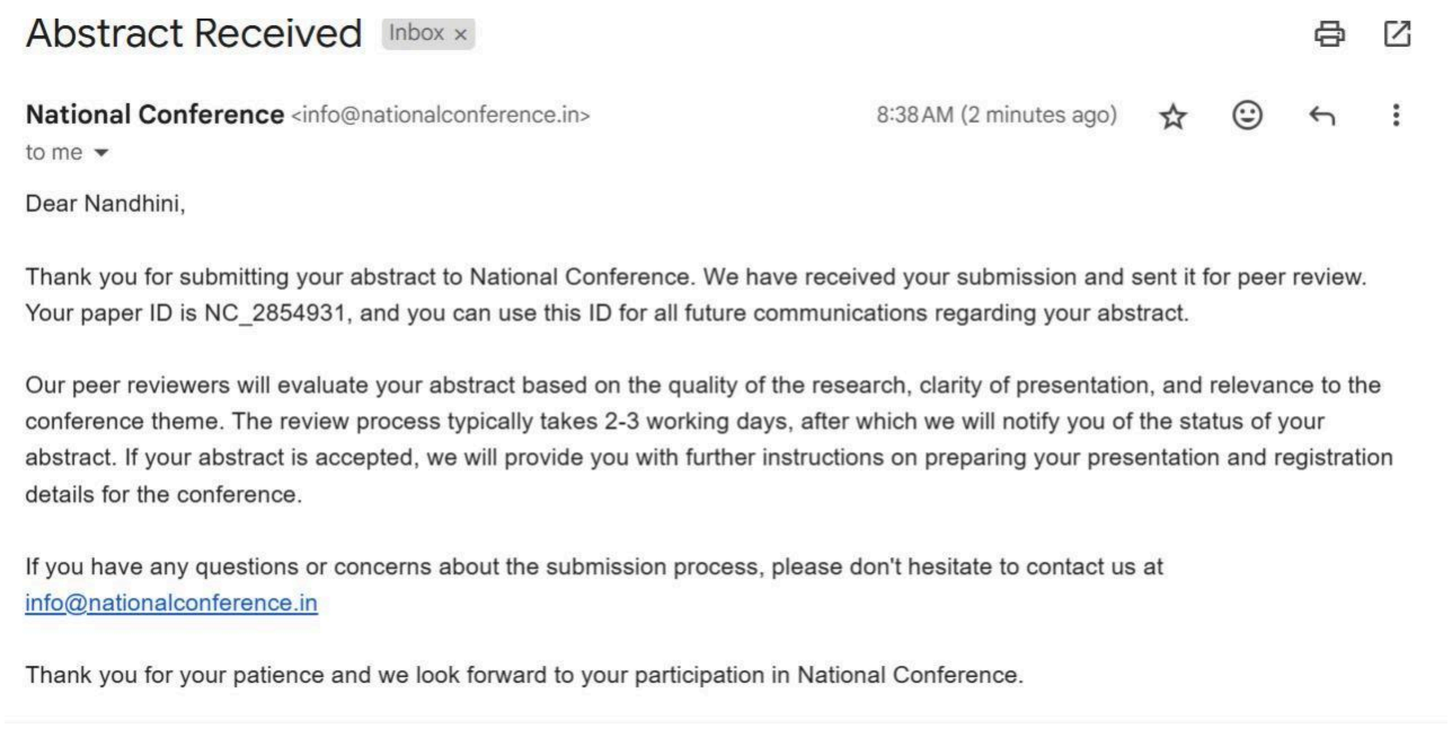
Authors : Deepa B, Pramikha K, Nandhini V and Vasupratha M
Title : A NOVEL APPROACH TO TRANSCRIPT SUMMARIZATION FOR VISUAL DATA USING NLP AND DEEP LEARNING
Number : 4

The submission was uploaded by Nandhini V <nanven15@gmail.com>. You can access it via the IEM-ICDC 2025 EasyChair Web page

https://easychair.org/conferences/?conf=iemicdc2025

Thank you for submitting to IEM-ICDC 2025.

Best regards,
EasyChair for IEM-ICDC 2025.



Abstract Received

National Conference <info@nationalconference.in>

8:38 AM (2 minutes ago)

to me

Dear Nandhini,

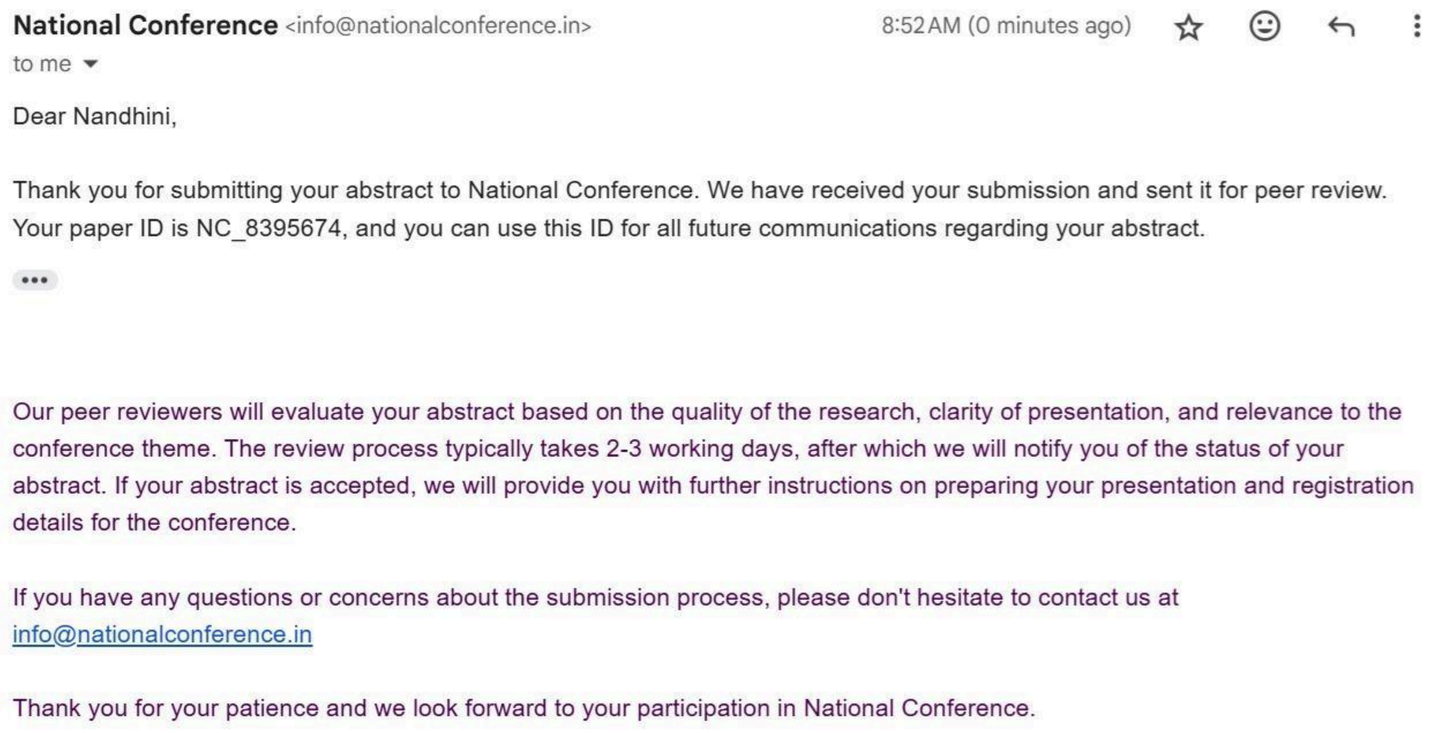
Thank you for submitting your abstract to National Conference. We have received your submission and sent it for peer review. Your paper ID is NC_2854931, and you can use this ID for all future communications regarding your abstract.

Our peer reviewers will evaluate your abstract based on the quality of the research, clarity of presentation, and relevance to the conference theme. The review process typically takes 2-3 working days, after which we will notify you of the status of your abstract. If your abstract is accepted, we will provide you with further instructions on preparing your presentation and registration details for the conference.

If you have any questions or concerns about the submission process, please don't hesitate to contact us at info@nationalconference.in

Thank you for your patience and we look forward to your participation in National Conference.

National Conference on Advanced Computer Science and Information Technology (NCACSI)
29th December 2024



National Conference <info@nationalconference.in> 8:52 AM (0 minutes ago)
to me

Dear Nandhini,

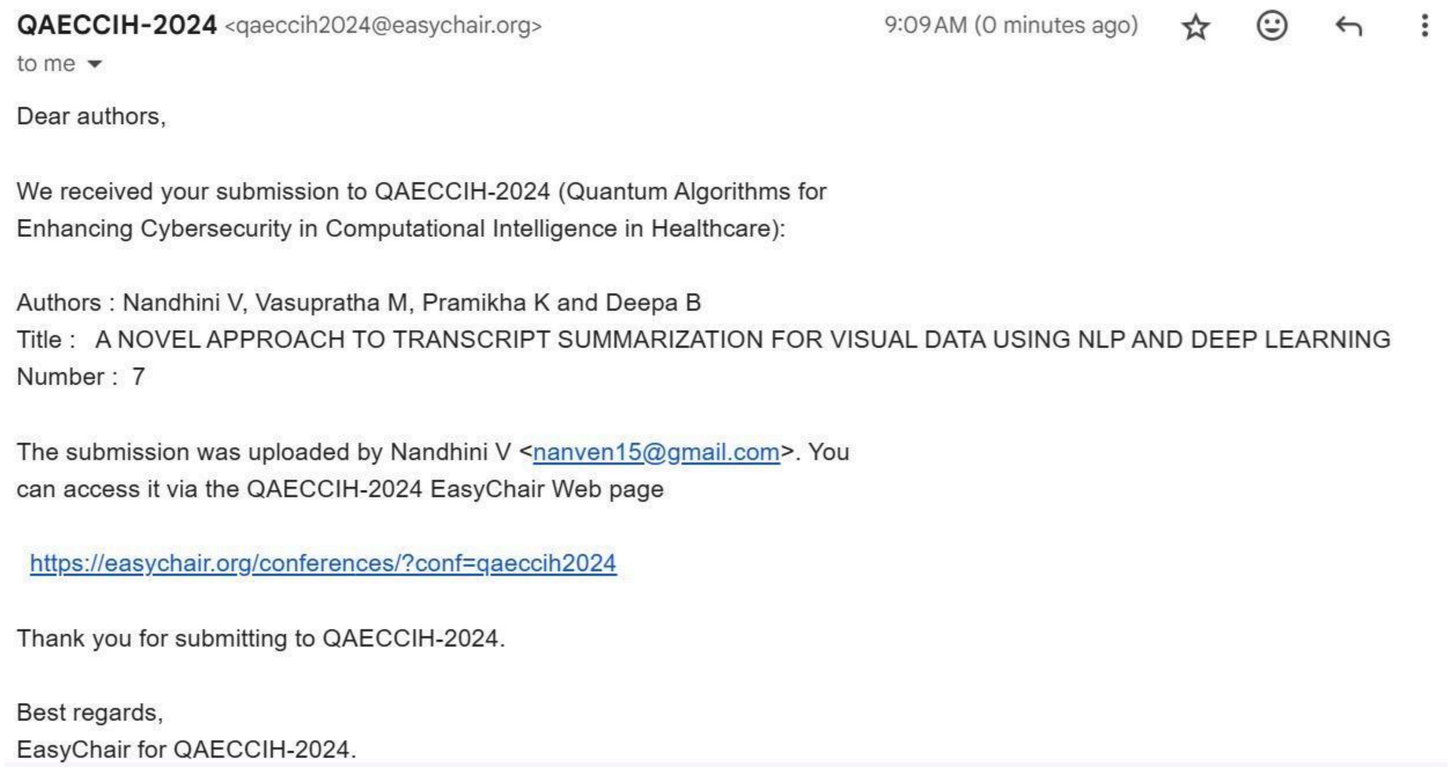
Thank you for submitting your abstract to National Conference. We have received your submission and sent it for peer review. Your paper ID is NC_8395674, and you can use this ID for all future communications regarding your abstract.

Our peer reviewers will evaluate your abstract based on the quality of the research, clarity of presentation, and relevance to the conference theme. The review process typically takes 2-3 working days, after which we will notify you of the status of your abstract. If your abstract is accepted, we will provide you with further instructions on preparing your presentation and registration details for the conference.

If you have any questions or concerns about the submission process, please don't hesitate to contact us at info@nationalconference.in

Thank you for your patience and we look forward to your participation in National Conference.

Quantum Algorithms for Enhancing Cybersecurity in Computational Intelligence in Healthcare
India, May 20-22, 2025



QAECCIH-2024 <qaeccih2024@easychair.org> 9:09 AM (0 minutes ago)
to me

Dear authors,

We received your submission to QAECCIH-2024 (Quantum Algorithms for
Enhancing Cybersecurity in Computational Intelligence in Healthcare):

Authors : Nandhini V, Vasupratha M, Pramikha K and Deepa B
Title : A NOVEL APPROACH TO TRANSCRIPT SUMMARIZATION FOR VISUAL DATA USING NLP AND DEEP LEARNING
Number : 7

The submission was uploaded by Nandhini V <nanven15@gmail.com>. You
can access it via the QAECCIH-2024 EasyChair Web page

https://easychair.org/conferences/?conf=qaeccih2024

Thank you for submitting to QAECCIH-2024.

Best regards,
EasyChair for QAECCIH-2024.