A Project Report on

"YouTube Video Summarizer"

SUBMITTED TO

G H RAISONI COLLEGE OF ENGINEERING MANAGEMENT

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

SUBMITTED BY-

PRATIK JADE **TY AI-A70**

UNDER THE GUIDANCE OF

Prof. VAISHALI BAVISKAR

DEPARTMENT OF ARTIFICIAL INTELLIGENCE

G H RAISONI COLLEGE OF ENGINEERING MANAGEMENT, WAGHOLI,PUNE.

2022-23

CERTIFICATE

This is to certify that the minor project report entitles

"YouTube Video Summarizer"

Submitted By

PRATIK JADE **A70**

are the bonafide students of this institute and the work has been carried out by them under the supervision of **Prof. Vaishali Baviskar** and it is approved for the partial fulfilment of the requirement.

Prof. Vaishali Baviskar
Guide

Prof. Rachna Sable
Head of Department

Dr. R. D. Kharadkar
Campus Director
GHRCEM, Pune

ACKNOWLEDGEMENT

It gives us great pleasure and satisfaction in presenting this project report on “YouTube Video Summarizer”.

We are thankful to and fortunate enough to get constant encouragement, support and guidance from all Teaching staffs of AI Department which helped us in successfully completing our project work. Also, we would like to extend our sincere esteems to all staff in laboratory for their timely support.

We have furthermore to thank AI Department HOD Prof. R. Y. Sable and Guide Prof. Vaishali Baviskar to encourage us to go ahead and for continuous guidance. We would also like to thank our project team members who showedimmense patience and understanding throughout the project.

We would like to thank all those, who have directly or indirectly helped us for the completion of the work during this project.

1. Pratik Jade

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

ABSTRACT

YouTube Video summarizer has got a lot of scope in today's world. It highlights the important topics from the video. People spend a noticeable amount of time binge watching YouTube videos, be it for entertainment, education purposes, or getting some important information or exploring their interests. If you wish to find a video to get any important information about a topic, it is a very difficult task to achieve as most of the videos are filled with insignificant buffer material. In most of the cases, the overall intent is to obtain some form of quality information from the video. This project brings forward a video summarization system based on Natural Language Processing and Machine Learning to generalize YouTube video transcripts for abstractive text summarization without losing the main elements and content. This project focuses on to reducing the length of the script for the videos.

1. INTRODUCTION

YouTube is a video sharing platform, the second-most visited website, the second most used search engine, and is stronger than ever after more than 17 years of being online. YouTube uploads about 720,000 hours of fresh video content per day. The number of videos available on the web platform is steadily growing. It has become increasing easy to watch videos on YouTube for anything, from cooking videos to dance videos to motivational videos and other bizarre stuff as well. The content is available worldwide primarily for educational purposes. The biggest challenge while extracting information from a video is that the viewer has to watch the entire video to understand the context, unlike images, where data can be gathered from a single frame. If a viewer has low network speed or any other device limitation can lead to watch video with a low resolution that makes it blurry and hectic to watch. Also, in between advertisements are too frustrating. So, removing the junk at the start and end of the concerned video as well as skipping advertisements, and getting is summary to directly jump to your part of interest is valuable and time efficient.

This project focuses on to reducing the length of the script for the videos. Summarizing transcripts of such videos automatically allows one to quickly lookout for the important patterns in the video and helps to save time and effort to go through the whole content of the video. The most important part of this project will be its ability to string together all the necessary information and concentrate it into a small paragraph. Video summarization is the process of identifying the significant segments of the video and produce output video whose content represents the entire input video. It has advantages like reducing the storage space used for the video. This project will give an opportunity to have hands-on experience with state-of-the-art NLP technique for abstractive text summarization and implement an interesting idea suitable for intermediates and a refreshing hobby project for professionals.

The main idea is to be able to choose a small subset of the most significant facts from the entire collection and present it in a simple manner. A user-friendly layout As the amount of textual material on the internet expands, automatic text summarizing techniques are becoming increasingly common. It has the potential to be tremendously beneficial because more useful information may be read in less time. Putting together summaries.

Because machines must understand what humans have written and produce human-readable results, NLP is concerned with transcripts.

1.1 PROBLEM STATEMENT:

Throughout the day, an increasing number of video recordings are generated and shared on the Internet. It has become quite difficult to devote time to watching movies that may last longer than expected, and our efforts may be in vain if we are unable to extract useful information from them. Summarizing transcripts of such movies automatically allows us to rapidly spot essential patterns in the video, saving us time and effort from having to go through the entire content.

1.2 OBJECTIVES:

The objective of this project is to make a chrome extension to summarize the YouTube Video into short summary. Making a Chrome extension to quickly describe YouTube videos is the goal of this project. The main goal of the project is to create a comprehensive and effective web application that can deliver the experience. The web application is hosted on the local host.

Some of the major objectives of this project are-

1. Generate a summary of a YouTube video in a concise and accurate manner.
2. Identify the key points and main ideas in a YouTube video.
3. Summarize a YouTube video in a way that is easy to understand.
4. Create chrome extension

1.3 SCOPE:

Many technical and instructional applications that create a lot of video and other multimedia are strong candidates for applying the video summarizing technique. These include the film business, the production of advertisements, data visualization, and match highlights from sporting events, which eliminate redundancy and lower computational and storage demands.

1. Research/Patents: This application can be used to extract key claims from research papers or patents, saving time and effort.
2. Quick Review: Students who wish to watch YouTube videos for their studies can simply receive a fast overview of the subject, read the video quickly, and choose

whether the video is appropriate for them or not.

3. Quick Notes: Students can use this software to generate notes from the video summary if they don't want to sit through lengthy lectures or have somehow missed class.
4. Customer feedback: Since consumers frequently provide detailed feedback on a given product, this program summaries their lengthy comments and may quickly determine whether they are positive or negative.
5. Hearing Impaired Person: This application is helpful for those who have trouble hearing because it allows them to view the entire video overview in just a few minutes without having to listen to it.

2. RELATED WORK

2.1 EXISTING SYSTEM/ Papers:

The Existing video summarization systems require strong prior technical knowledge. Machine learning based algorithms require high processing power. Summarizing video based on its subtitle is the fastest way of generating video summary, because dealing with text is easier and faster compared to raining various videos using machine learning models.

1. To get a summary, the user needs to open a new tab or visit another website application.
2. It is difficult to identify the correct application.
3. It is a time-consuming process
4. Slow method.
5. It is not user friendly.

Paper :

Youtube Transcript Summarizer Using Flask And Nlp

By P. Vijaya Kumari, M. Chenna Keshava, C. Narendra, P. Akanksha, K. Sravani

Introduction

Using the Flask framework, this backend takes API calls from the client and answers with a summary text response. This API can only be used with YouTube videos that have closed captions that have been properly prepared. The Summarizer is also available online, where users may make basic API calls and read the results ona webpage.This backend accepts API calls from the client and responds with a summary text response using the Flask framework. This API can only be used with YouTube videos that have been correctly prepared closed captions. Users can also utilizethe Summarizer online, where they can execute basic API calls and view the results on a webpage.

Work Flow

The steps are followed to accomplish the transcript information:

1. Client sends request to our flask backend server
2. Flask backend server asks for subtitles from the YouTube using YouTubeAPI.
3. YouTube sends the subtitles to our server and text summarization is done in our backend server
4. Client receives the summarized text

Limitations

1. Transcript cannot get from the videos without subtitle
2. Translated text other than English won't support text and pdf file formats because of encoding format

3. SYSTEM DESIGN

System analysis is the process of gathering and interpreting facts, diagnosing problems and using the information to recommend improvements on the system. System analysis is a problem-solving activity that requires intensive communication between the system users and system developers.

System analysis or study is an important phase of any system development process. The system is viewed as a whole, the inputs are identified and the system is subjected to close study to identify the problem areas. The solutions are given as a proposal. The proposal is reviewed on user request and suitable changes are made. This loop ends as soon as the user is satisfied with the proposal.

3.1 PROPOSED SYSTEM

In As the number of users of YouTube are increasing rapidly from year to year, this may directly impact the number of videos to be created. In greed for the number of views, the chance that the creators of the videos may give wrong information on the original content of the video is probably very high. This may waste the valuable time and resources of the user.

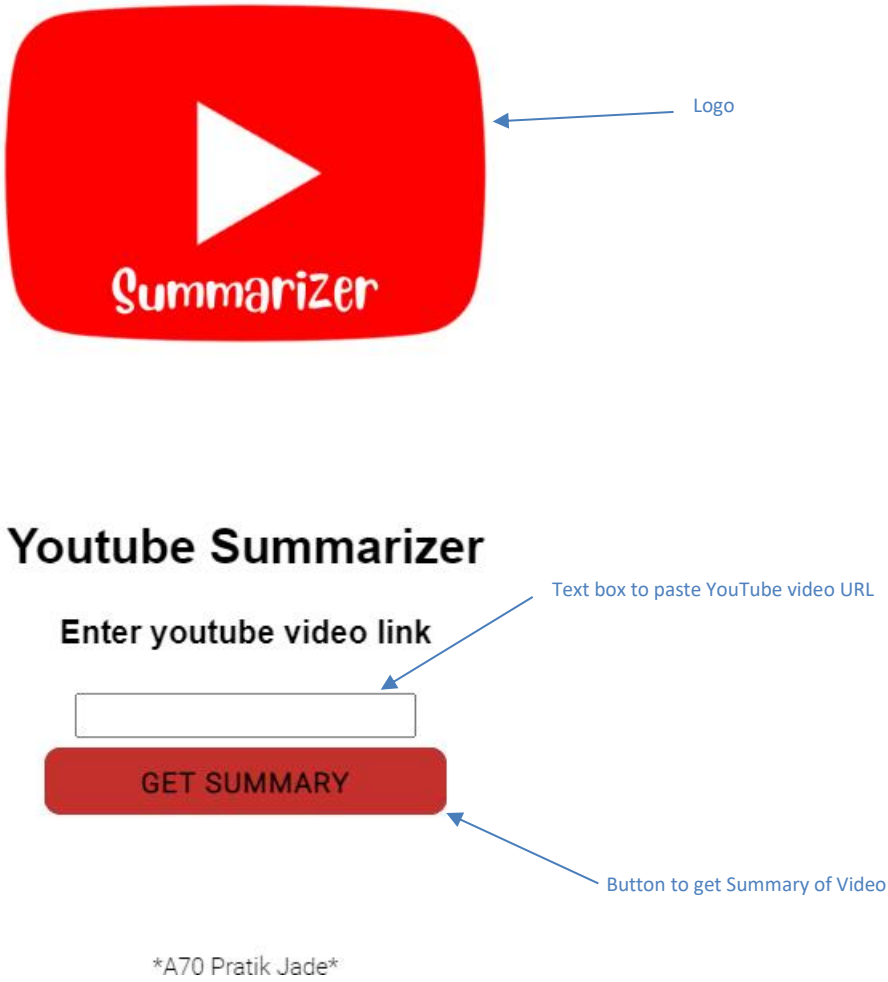
To further improve the user interaction with the summarizer Chrome extension is used for user-friendly interaction which consists of a summarize button. On clicking this button, the Chrome extension displays the summarized text of the current YouTube video running on the Google Chrome web browser.

There are 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 interface and 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.

3.2 DESIGN

- **CHROME EXTENSION**

  New user needs to install chrome extension to use it. Go to the website chrome://extensions and make sure developer mode is turned on from the top right-hand corner. Then select on Load unpacked and select the folder containing the manifest file that we just created. This creates our extension.



- **YouTube video URL**

  A user needs to copy YouTube video URL and paste in chrome extension.

- **Get Summary**

  User can click on get summary button to get summary of video.

3.3 FUNCTIONAL REQUIREMENTS:

- The system requires to Host local server in order to use software.
- The system requires to turn on developer mode in Web browser to add extension
- The system requires to give permission to access storage
- The system did not require any person info and any order permission

3.4 BLOCK DIAGRAM:

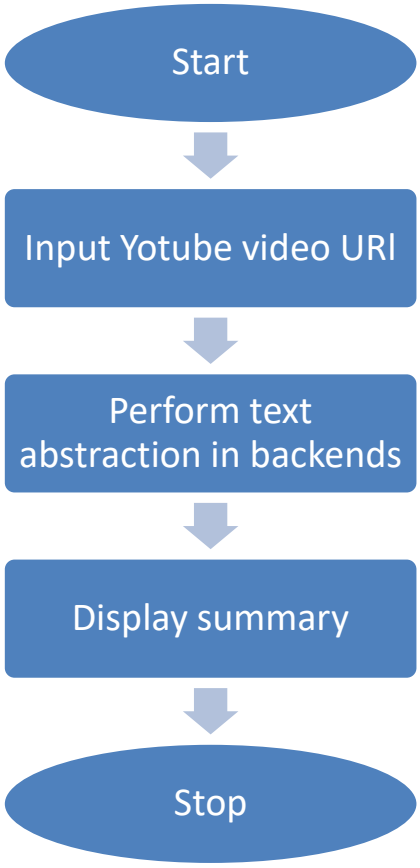


Figure 3.1: Block diagram of YouTube Summarizer

4. METHODOLOGY

NLP stands for Natural Language Processing, which is a part of Computer Science, Human language, and Artificial Intelligence. It is the technology that is used by machines to understand, analyze, manipulate, and interpret human's languages. It helps developers to organize knowledge for performing tasks such as translation, automatic summarization, Named Entity Recognition (NER), speech recognition, relationship extraction, and topic segmentation.

This project basically aims at providing clean and concise summary of the YouTube videos that the user don't want to waste their time at. This project uses popular python libraries like:

1. Flask
2. Nltk
3. Heapq
4. re
5. youtube_transcript_api
6. Http
7. transformers
8. Json

1. **Flask :** Flask is a web framework that provides libraries to build lightweight web applications in python. It is developed by Armin Ronacher who leads an international group of python enthusiasts (POCCO). It is based on WSGI toolkit and jinja2 template engine. Flask is considered as a micro framework.

2. **Nltk :** NLTK is a toolkit build for working with NLP in Python. It provides us various text processing libraries with a lot of test datasets. A variety of tasks can be performed using NLTK such as tokenizing, parse tree visualization, etc.

3. **Heapq :** This module provides an implementation of the heap queue algorithm, also known as the priority queue algorithm. Heaps are binary trees for which every parent node has a value less than or equal to any of its children. This implementation uses arrays for which heap[k] <= heap[2*k+1] and heap[k] <= heap[2*k+2] for all k, counting elements from zero. For the sake of comparison, non-existing elements are considered to be infinite. The interesting property of a heap is that its smallest element is always the root, heap [0].

4. **Re :** The Python "re" module provides regular expression support. In Python a regular expression search is typically written as: match = re. search (pat, str) The re.search() method takes a regular expression pattern and a string and searches for that pattern within the string.

5. **youtube-transcript-api :** Python provides a large set of APIs for the developer to choose from. Each and every service provided by Google has an associated API. Being one of them, YouTube Transcript API is very simple to use provides various features. This is a python API which allows you to get the transcript/subtitles for a given YouTube video. It also works for automatically generated subtitles, supports translating subtitles and it does not require a headless browser, like other selenium based solutions do!

6. **Http :** The http or Hyper Text Transfer Protocol works on client server model. Usually the web browser is the client and the computer hosting the website is the server. IN python we use the requests module for creating the http requests. It is a very powerful module which can handle many aspects of http communication beyond the simple request and response data. It can handle authentication, compression/decompression, chunked requests etc.

7. **Transformers :** PyTorch-Transformers is a library of state-of-the-art pre-trained models for Natural Language Processing (NLP). Transformers provides APIs and tools to easily download and train state-of-the-art pretrained models. Using pretrained models can reduce your compute costs, carbon footprint, and save you the time and resources required to train a model from scratch. These models support common tasks in different modalities, such as:

    1. Natural Language Processing: text classification, named entity recognition, question answering, language modeling, summarization, translation, multiple choice, and text generation.
    2. Computer Vision: image classification, object detection, and segmentation.
    3. Audio: automatic speech recognition and audio classification.
    4. Multimodal: table question answering, optical character recognition, information extraction from scanned documents, video classification, and visual question answering.

Transformers support framework interoperability between PyTorch, TensorFlow, and JAX.

This provides the flexibility to use a different framework at each stage of a model's life; train a model in three lines of code in one framework, and load it for inference in another. Models can also be exported to a format like ONNX and TorchScript for deployment in production environments.

8. **JSON:** JSON JavaScript Object Notation is a format for structuring data. It is mainly used for storing and transferring data between the browser and the server. Python too supports JSON with a built-in package called json. This package provides all the necessary tools for working with JSON Objects including parsing, serializing, deserializing, and many more. Let's see a simple example where we convert the JSON objects to Python objects and vice-versa.

In this project we get transcripts/subtitles for a given YouTube video Id using a Python API, perform text summarization on obtained transcripts using HuggingFace transformers, build a Flask backend REST API to expose the summarization service to the client and develop a chrome extension which will utilize the backend API to display summarized text to the user.

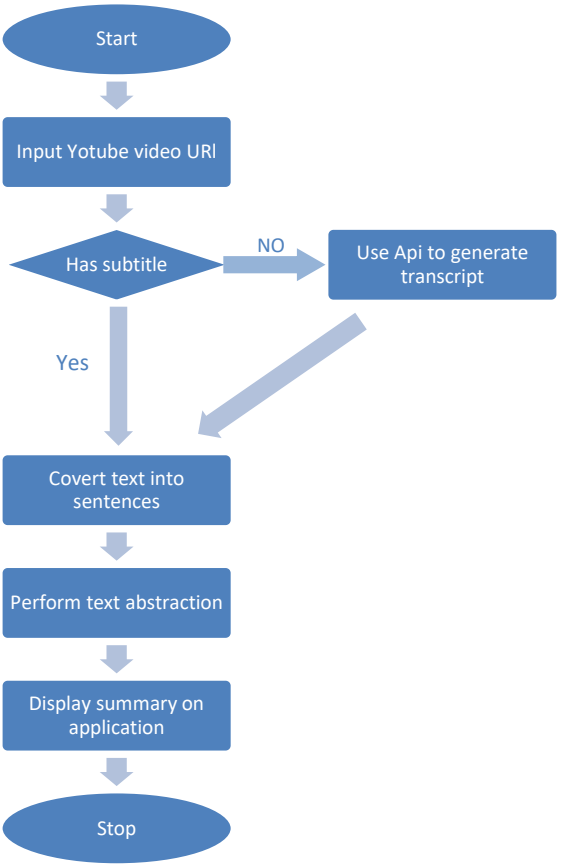


Figure 4.1 System Architecture of YouTube Summarizer

According to Figure 1, initially we open a YouTube video, enter video URL and click the button summarize in the chrome extension. This will create a HTTP request to the back-end. Then the request to access the transcripts will be made with YouTube video ID which was taken from the URL. The response will be a transcript of that video in json format. After getting the transcripts in the text format the system performs Transcript Summarization. Finally, it displays the summarized transcript on the extension.

BACK END: APIs changed the way we build applications, there are countless examples of APIs in the world, and many ways to structure or set up our APIs. We create a back-end application directory containing files named as app.py. We initialize app.py with basic Flask RESTful BoilerPlate. We then create a virtual environment to isolate the location where everything resides. By activating the newly created virtual environment we install the dependencies like Flask, youtube_transcript_api, transformers using pip

GET TRANSCRIPTS: After taking the video link from the user, the next part is to get the transcripts on video. Now it will check whether the given video has subtitles available or not. If the Given video has subtitles in it then we can simply use the python library called YouTube_Transcript_Api which is used to extract the transcripts from the given video. Now what if the given video doesn't have subtitles in it so in this case It will again YouTube Api to generate subtitles, supports translating subtitles, and does not require a headless browser like other Selenium-based solutions do. In app.py, we create a function which will accept YouTube video id as an input parameter and return parsed full transcript as output. As we get the transcript in the json format with text, start and duration attributes we only need text so we parse the data from the response to return the transcript in whole string format

TEXT SUMMARIZATION: Now this is the main phase of the project where the whole project depends upon. This phase basically includes the text summarization.
There are mainly two types of summarization techniques:-

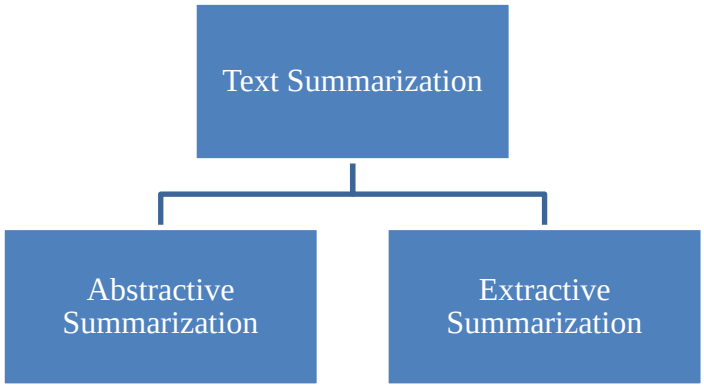


Figure 4.2: Text Summarization methods

1. **Extractive Summarization:** The extractive approach involves picking up the most important phrases and lines from the documents. It then combines all the important lines to create the summary. So, in this case, every line and word of the summary actually belongs to the original document which is summarized.

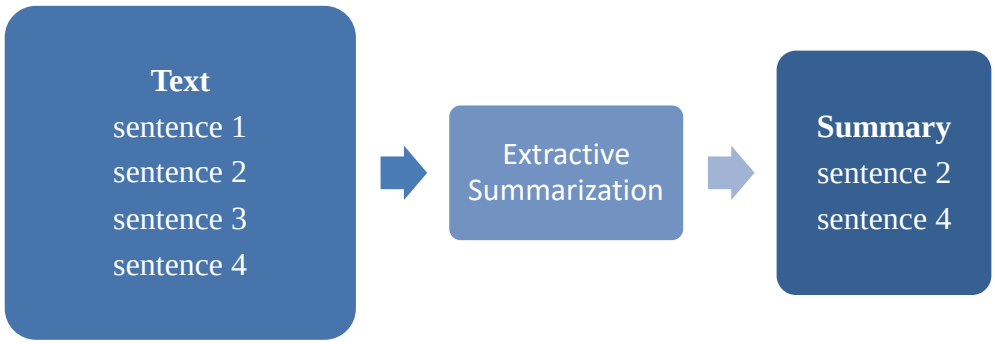


Figure 4.3: Extractive Summarization model

2. **Abstraction-based Summarization:** The abstractive approach involves summarization based on deep learning. So, it uses new phrases and terms, different from the actual document, keeping the points the same, just like how we actually summarize. So, it is much harder than the extractive approach. This is another text summarization technique which is a new start of art method because it generates the sentences in a newly formed way. It will basically reproduce the sentence which is more clean and concise than the original sentence and in a most human readable form. This is better than extractive summarization techniques basically This model produces a completely different text that is shorter than the original, it generates new sentences

in a new form. In this project, we will use transformers for this approach. In this system, we will use the HuggingFace transformers library in Python to perform abstractive text summarization on the transcript obtained from the previous step. In app.py

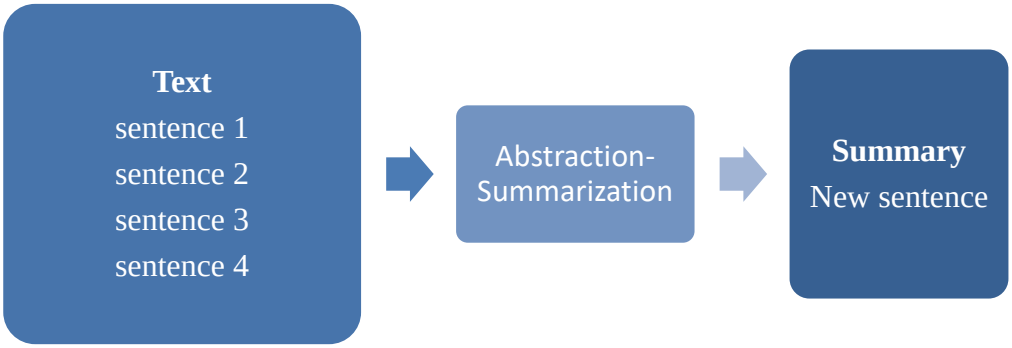


Figure 4.4: Abstractive Summarization model

At first, we have to use the Abstractive Summarization technique to summarize the text. In which we have used Hugging Face Transformer for summarization using a pipeline. Hugging Face Transformer uses the DistilBERT model for summarization as DistilBERT is nothing more than a smaller version of the BERT technique developed by the hugging Face company. In this procedure firstly extracting the captions or say subtitles through python API for a particular ID of a YouTube video. In the very next step using the hugging face transformer for summarizing the obtained transcript of the YouTube video ID and for exposing the summarized version of the transcript is to create a Flask backend REST API. So that users can get the required summarized content of a YouTube video. For general usage also infused this algorithm into the chrome extension so that everyone can have the access to it. Also developed chrome extension will use back-end API to display a summarized version of the text to the user.

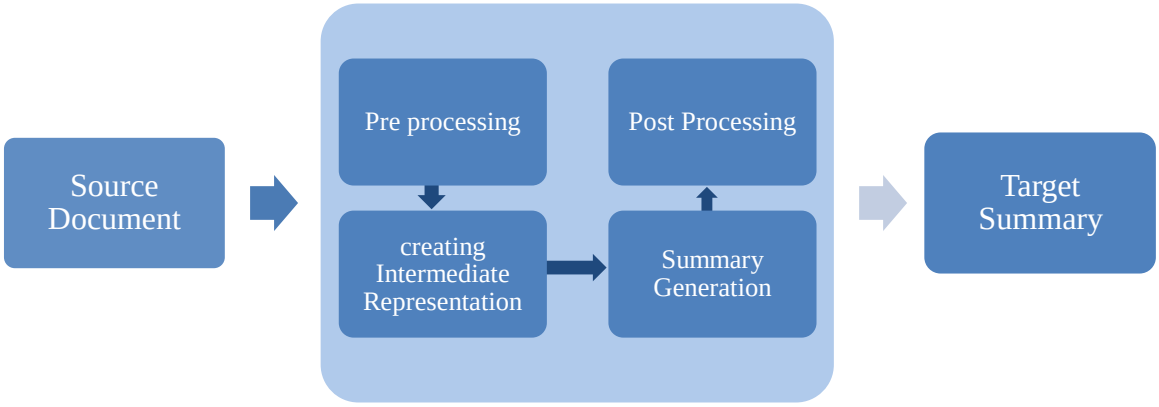


Figure 4.5: Architecture of Abstractive Text Summarization.

CHROME EXTENSION: Chrome Extensions are the software programs that customize and enhance the browsing experience. They enable users to enhance Chrome functionality experience and behavior to individual preferences. They are built on the web using web technologies such as HTML, CSS and JavaScript. In this step, we create a chrome extension application directory containing essential files required.

We need to create a manifest.json file for loading our extension in the browser. Go to the website chrome://extensions and make sure developer mode is turned on from the top right-hand corner. Then select on Load unpacked and select the folder containing the manifest file that we just created. This creates our extension. We'll need to reload the extension every time we make a change in the extension.

Finally converting the whole project into a chrome extension using Flask Framework and one can easily deploy it on cloud platforms like Heroku, AWS, etc.

5. SYSTEM REQUIREMENTS

Hardware requirements:

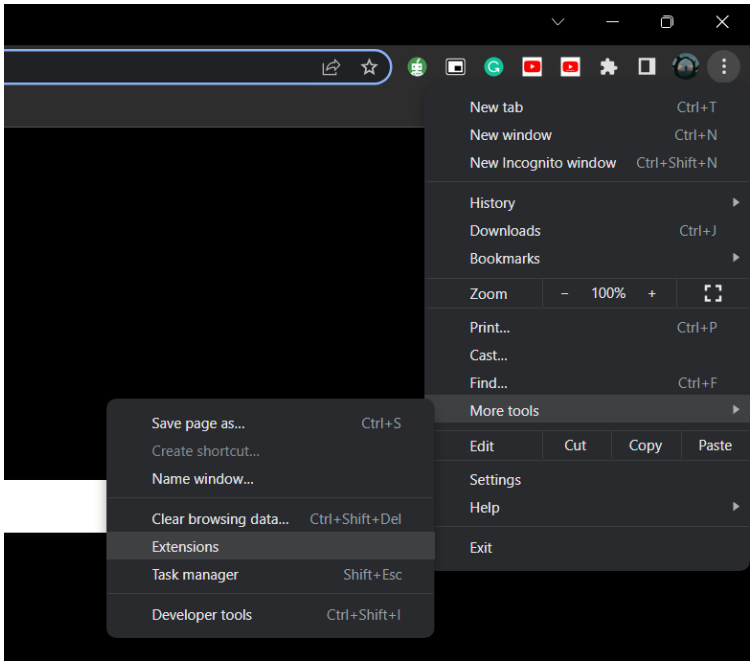
- 1 gigahertz (GHz) or faster 32-bit (x86) or 64-bit (x64) processor
- 1 gigabyte (GB) RAM (32-bit) or 2 GB RAM (64-bit)
- 1 GB or more available hard disk space
- DirectX 9 graphics device with WDDM 1.0 or higher driver
- Internet access

Software requirements:

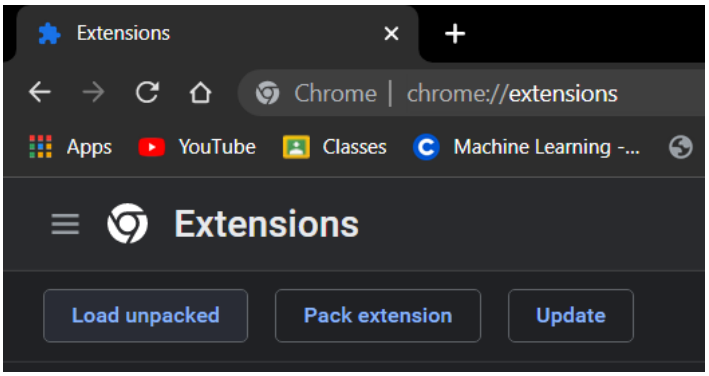
- Operating system: Windows 7+ / Mac OS 10.12 +
- Web Browser like Google Chrome, Microsoft edge, Brave, or any which Support Chrome Extensions.
- Python 3.5+, HTML, JavaScript,
- Any Python IDE/ Code Editor

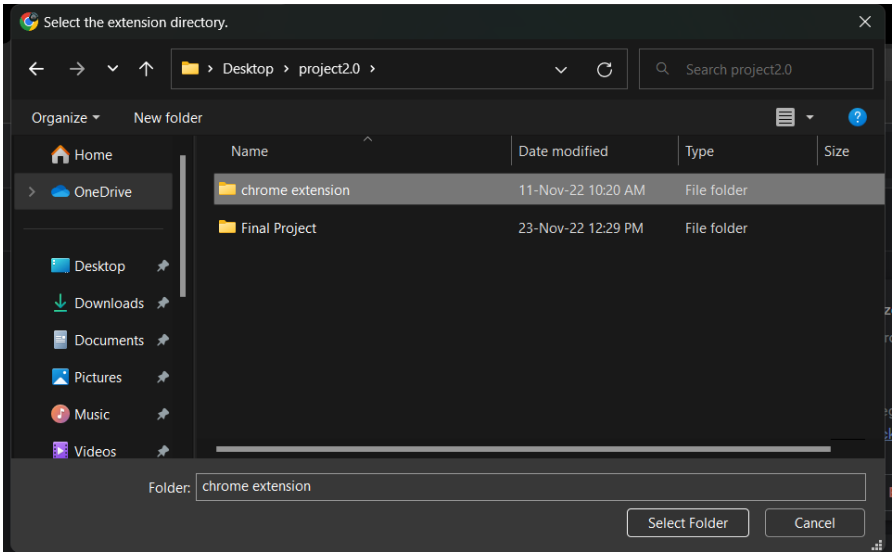
6. RESULTS

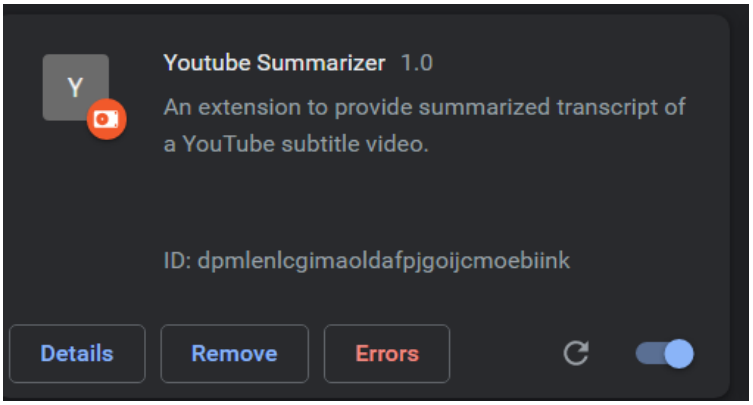
Result 6.1: Screenshot of command prompt running backend app.py



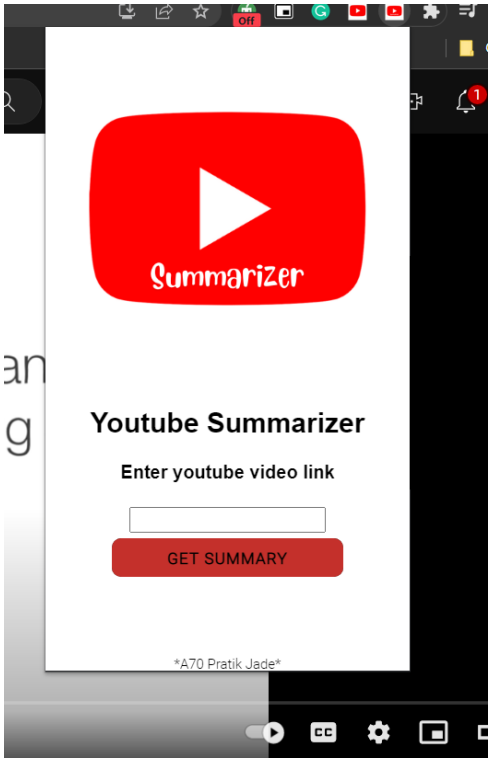
Result 6.2: extensions settings



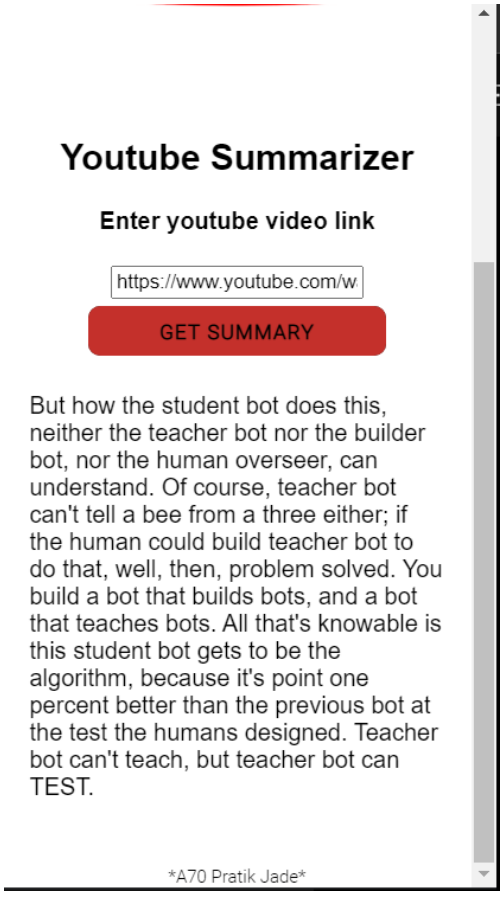




Result 6.3: How to add Extension in Browser



Result 6.4: YouTube Summarizer Chrome Extension.



Result 6.5: Screenshot of output summarized text from YouTube Video

Table 6.1: Results of different Summarize Video

Video Length	Sentences	Sentences Summary
5 Minutes	89	5
9 Minutes	`179	5
30 Minutes	610	6

7. CONCLUSION

This project video summarization has attracted considerable interest from researchers and as a result, various algorithms and techniques have been proposed. This project YouTube summarizer in which, the system takes the input YouTube video from the Chrome extension of the Google Chrome browser when the user clicks on the summarize button on the chrome extension web page, and access the transcripts of that video with the help of python API. The accessed transcripts are then summarized with the transformers package. Then the summarized text is shown to the user in the chrome extension web page.

This project helps the users a lot by saving their valuable time and resources. This helps us to get the gist of the video without watching the whole video. It also helps the user to identify the unusual and unhealthy content so that it may not disturb their viewing experience.

This project also ensures great user interface experience in finding out the summarized text as chrome extensions have been used. This helps in getting the summarized text without any third-party applications.

REFERENCES

[1] Hank Liao, Erik McDermott, Andrew Senior“Large scale deep neural network acoustic modeling with semi-supervised training data for YouTube video transcription” December 2013.

[2] Gaurav Sharma, Shaba Parveen Khan, Shivanshu Sharma, Syed Ubed Ali “Summarizer For Easy Video Assessment” Volume: 3 Issue:04-April-2021

[3] Atluri Naga, Laggisetti Valli, JahnaviDuvru “Video Transcript Summarizer” Issue: 11 March 2022

[4] AniquaDilawari, Muhammad usmanghani khan.“Abstractive Summarization of Video Sequences” IEEE Access, 2019.

[5]. 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.

[6]. 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.

[7] https://journalppw.com/index.php/jpsp/article/view/9886/6457

[8] https://huggingface.co/transformers/

[9] https://atmamani.github.io/blog/building-restful-apis-with-flask-in-python/

[10] https://pypi.org/project/youtube-transcript-api/

[11] https://developer.chrome.com/docs/extensions/mv2/