International Journal of Research in Engineering and Science (IJRES)
ISSN (Online): 2320-9364, ISSN (Print): 2320-9356
www.ijres.org Volume 11 Issue 5 ǁ May 2023 ǁ PP. 189-195

Youtube Transcript Summarizer

Siddhartha
SCSE
Galgotias University,
Gautam Budh nagar, Greater Noida

Prashu Pandey
SCSE
Galgotias University,
Gautam Budh nagar, Greater Noida

Ansh Saxena
SCSE
Galgotias University,
Gautam Budh nagar,Greater Noida

GUIDE- Anupam kumar sharma
Professor

Abstract—*Huge number of videos are being created and shared on the internet every day. In daily lifestyle, spending time on video is very difficult. First, we have to find correct video for our problems and from video if we don't correct solution, it turns problem into frustration which effects our health also. We only need relevant information from the video but sometimes for gaining the attention from the users, they upload misleading thumbnails, description, advertisement, etc. The number of creators is increasing rapidly from year to year, this directly impact the number of videos to be created. The creators in the greed of views might give misleading information rather than giving the correct information. In the videos, creators waste our time by promoting the different brands, repeating the phrase for subscribing the channel, share the video, etc. YouTube Transcript Summarizer is a chrome extension which summarizes the whole video in span of time. For user friendly interaction we created a button in the extension which displays summarized text of the current video running on the chrome browser.*

Keywords—*Flask API, Text Summarizer, HuggingFace Transformers, Chrome Extension, Web API*

Date of Submission: 26-04-2023 Date of acceptance: 06-05-2023

I. INTRODUCTION

YouTube videos are a huge source of information in today's time but learning from them is a tough task as it takes a plenty of time in grabbing the relevant information from the videos. Videos might contain unwanted and wasteful information which can skippable or cannot be so user has to see it. There number of YouTube users in 2023 is approximately 2.6 billion and number is rapidly increasing every day. On average, 20000 new videos are uploading on YouTube every hour and average video length is 12 minutes. The number of sharing videos is increasing exponentially and the user cannot waste their golden time in watching the videos and gain nothing from those videos.

Sometimes it become very difficult for watching video which may have longer duration of time than expected and sometimes efforts become futile if the video doesn't contain the correct information for which the user has searched. It might be frustrating and time consuming to find for the videos that contains the accurate information regarding our searched topic.

This summarizer is helpful for those users who want to accurate information rather than spending their time in watching the video. Summarizer describes the transcipt of the video in the text format so that user can get true information and solution for their problems.

Transcripts are an excellent source of information as they contain a textual representation of the audio in a video. Transcript summarization can be achieved through natural language processing (NLP) techniques that extract key phrases and sentences from the transcript. The generated summary can then be used to provide viewers with a quick overview of the video content.

It is chrome extension which we can download from chrome extensions store and for running this extension we have to click on the extension icon while the video is running in the background the pop up will show and then we have to click on the button with title summarizer then the extension will display the content of the video in the popup screen within a minute. That's how extension will save the time of users.

II. LITERTURE SURVEY

The use of YouTube transcript summarizers has gained attention from researchers in recent years due to the increasing amount of video content available on the platform. This section presents a literature survey of some of the previous works on YouTube transcript summarizers. 'Automated Video Summarization Using Speech Transcript' by Cuneyt M. Taskiran, Aronon Amir, Dulce B. Ponceleon, Edward J. Delph describes the compact representations of video data can enable efficient video browsing. They propose the method which summarizes the long video automatically. Their representations provide the user relevant information about the content with particular sequence examined while preserving the essentials of the content.[1]

'Video Summarization using NLP' by Sanjana R., Sai Gagana V, Vedhavati K R, Kiran K N proposes an automatic video summarization using Natural Language Processing (NLP) based algorithms. The increasing popularity of YouTube gave us the millions of video repository and hence there is an increase demand for good summarization algorithms to summarize various video without loss of any accurate information of the content. Their proposed system describes the YouTube video transcripts based on which summarized video is generated. [2].

Millions of videos are created and shared on the repository platforms such as YouTube, Reddit, Instagram, etc. It is becoming challenging task to spend time on watching such videos, which may have longer duration. Sometimes efforts of watching the videos may go in vain if we are unable to extract our meaningful information from them but Summarizing transcripts of such videos can help us in extracting the meaningful information from the transcript of video. The YouTube summarization model is helpful in extracting the transcripts and generates the summarized version of it. The model automatically produces a summary containing important sentences and including all relevant information related to the original documentation. Abstractive approach generates a new word from input text making the task more difficult while Extractive approach extract the sentences and phrases from the input. [3]

In 2018, Nallapati et al. proposed a method for summarizing YouTube video transcripts using deep learning models. The method used a combination of convolutional and recurrent neural networks to extract relevant information from the transcript. The authors evaluated their method on a dataset of YouTube videos and reported competitive results compared to other summarization methods.[4]

In 2019, Nguyen et al. proposed a method for summarizing YouTube video transcripts using a hybrid approach that combines rule-based and machine learning techniques. The method used a set of rules to extract sentences from the transcript, which were then used to train a machine learning model to generate a summary. The authors evaluated their method on a dataset of TED Talks and reported competitive results compared to other summarization methods.[5]

In 2020, Zeng et al. proposed a method for summarizing YouTube video transcripts using a graph-based approach. The method used a combination of TF-IDF and Text Rank algorithms to extract key phrases from the transcript. The authors evaluated their method on a dataset of TED Talks and reported competitive results compared to other summarization methods.[6]

In 2021, Huang et al. proposed a method for summarizing YouTube video transcripts using a transformer-based language model. The method used a pre-trained transformer model to extract relevant information from the transcript. The authors evaluated their method on a dataset of educational videos and reported competitive results compared to other summarization methods.[7]

In conclusion, the literature survey suggests that YouTube transcript summarization has been approached using various techniques, including deep learning models, hybrid approaches, graph-based methods, and transformer-based language models. These methods have been evaluated on different datasets, including TED Talks, educational videos, and general YouTube videos, and have reported competitive results compared to other summarization methods.

III. PROPOSED SYSTEM

Most methods for video summarization do not use one of the most important sources of information in video sequence, the spoken text or the natural-language context. For the sequence like speeches, seminars and instructional programs does not have transcript we can obtain it by applying speech recognition on the audio and later we can it our summarizer. YouTube Transcript Summarizer is a tool that automatically generates the summary from the transcript of the video's audio. The model will involve developing and debugging of the different techniques and algorithms for natural language processing (NLP) and extraction of information as well as the implementation and testing on the large dataset of YouTube transcript. This model involves different API

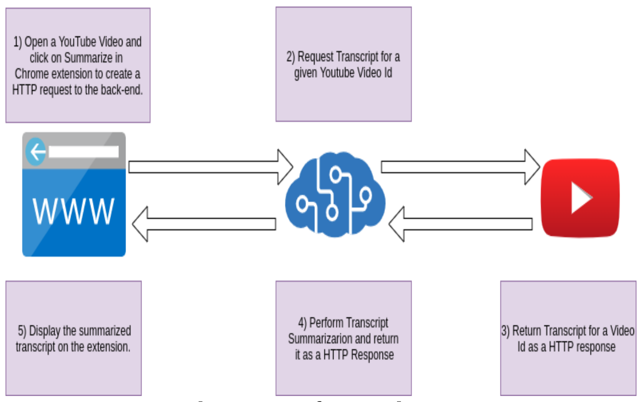
such as FLASK API for testing, Python API for getting YouTube video and use different languages and framework such as HTML, CSS and JavaScript for developing the extension for the web browser. The techniques used in text summarization is Natural Language Processing (NLP) analysis based on information-extraction techniques. This approach, utilizing artificial intelligence techniques, involves a comprehensive analysis of the source text's meaning to construct a source representation for a specific application. Then, a summary representation is generated using this source representation, and the summary text is produced. However, methods that rely on statistical processing to extract sentences for the summary often produce summaries that lack coherence. These methods also encounter the problem of dangling anaphors, which are pronouns, demonstratives, and comparatives such as "he," "this," and "more," that can only be understood by referring to an antecedent clause preceding the sentence in which these words appear. If the antecedent clause has not been selected for the summary, the use of anaphors may be confusing for the reader. Although NLP-based techniques generate better summaries, the knowledge base required for such systems is usually vast and complex. Furthermore, such systems are typically limited to a specific application domain and are challenging to generalize to other domains.
Natural language processing (NLP) is a field of computer science and artificial intelligence focused on enabling computers to understand and respond to human language, both written and spoken. This involves combining techniques from computational linguistics, statistical modeling, machine learning, and deep learning to process human language in the form of text or voice data, and to determine the intended meaning and sentiment behind it.

There are several tasks associated with NLP, including speech recognition (converting voice data into text), part of speech tagging (determining the part of speech of a word or text based on context), word sense disambiguation (determining the correct meaning of a word with multiple meanings), co-reference resolution (identifying when two words refer to the same entity), sentiment analysis (extracting attitudes, emotions, sarcasm, and other subjective qualities from text), and natural language generation (putting structured information into human language).

Overall, NLP aims to give computers the ability to understand and use language much like humans do, which has many potential applications in areas such as customer service, chatbots, voice assistants, and more.

i. System Design and Analysis

In this project we get transcript/subtitle for given YouTube video using a Python API, using HuggingFace transformers,
Perform the text summarization. To expose the summarization service to the client we will build a Flask backend REST API. By developing the chrome extension which will utilize the backend API display summarized to the user.


1) Open a YouTube Video and click on Summarize in Chrome extension to create a HTTP request to the back-end.
2) Request Transcript for a given Youtube Video Id
WWW
5) Display the summarized transcript on the extension.
4) Perform Transcript Summarizarion and return it as a HTTP Response
3) Return Transcript for a Video Id as a HTTP response
Figure 1: System Architecture of YouTube Transcript Summarizer

The project follows a clear flowchart as shown in Figure 1. Firstly, the user opens a YouTube video and clicks on the "summarize" button in the chrome extension. This initiates a HTTP request to the back-end of the system. Subsequently, the request is made to access the transcripts using the YouTube video ID obtained from the URL. The response to this request will be a transcript of the video in JSON format.

Once the transcripts are obtained in text format, the system performs transcript summarization, which involves reducing the length of the transcript while retaining the most important information. Finally, the summarized transcript is displayed on the extension.

1.1 BACK END

APIs have revolutionized the way applications are built and there are numerous examples of APIs being used in different applications. To set up our API, we begin by creating a back-end application directory with an app.py file. This file is initialized with a basic Flask RESTful Boilerplate. We then create a virtual environment to isolate the location where all the dependencies will reside. Once the virtual environment is activated, we use pip to install the necessary dependencies, including Flask, YouTube_Transcript_API, and transformers. It is important to ensure that the content is original and not plagiarized to maintain its integrity.[8]

1.2 GET TRANSCRIPTS

In this module, we will utilize a Python API to obtain transcripts/subtitles for a specified YouTube video. The API is capable of working with automatically generated subtitles, translating subtitles, and does not require a headless browser like other Selenium-based solutions. In app.py, we define a function that takes the YouTube video ID as an input parameter and returns the parsed full transcript as the output. Since we receive the transcript in JSON format with text, start, and duration attributes, we only extract the text data from the response and return the transcript as a single string. This process allows us to obtain the complete transcript of the video.[9]

1.3 PERFORM TEXT SUMMARIZATION

Text summarization refers to the task of condensing longer text into a shorter summary while preserving the key information and meaning of the original text. There are two main approaches used for text summarization: extractive summarization and abstractive summarization. Extractive summarization involves identifying important sentences and phrases from the original text and outputting only the necessary parts, while abstractive summarization involves generating a completely new text that is shorter than the original text, often using encoder-decoder models like Bart or T5.

For this project, 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, a function is created that accepts the YouTube transcript as input and returns the summarized transcript as output. To perform the summarization, a tokenizer and a model are instantiated from the checkpoint name. The T5-specific prefix "summarize:" is added to the transcript that needs to be summarized. The PreTrainedModel.generate() method is then used to generate the summary.

1.4 REST API ENDPOINT

The next step is to define the resources that will be utilized in the implementation of this backend service. As this is a straightforward application with only a single endpoint, the only resource we need to define is the summarized text.

In app.py, we create a Flask API Route with a GET HTTP Request method and a URI of http://[hostname]/api/summarize?youtube_url=#{url}. We then extract the YouTube video ID from the YouTube URL obtained from the query parameters. After that, we generate the summarized transcript by executing the transcript generation function and the transcript summarizer function. Finally, we return the summarized transcript with an HTTP Status OK and handle HTTP exception as required.[10]

1.5 CHROME EXTENSION

Chrome Extensions are software programs designed to enhance and customize the browsing experience of users. These extensions are built on web technologies like HTML, CSS, and JavaScript. To create a Chrome extension for our project, we need to create an application directory with the necessary files, including a manifest.json file, which is used to load the extension in the browser.

To load the extension in the browser, we need to turn on developer mode in the Chrome extensions page, and then select "Load unpacked" and choose the folder containing the manifest file. After this, the extension is created and can be reloaded every time changes are made.[11]

1.6 USER INTERFACE FOR EXTENSION POPUP

User interface is an essential aspect to ensure that users can interact with the pop-ups of the Chrome extension. There are several types of user interfaces that a Chrome extension can provide, but in this project, we will enable the User Interface for a Popup by adding the line below to page action in the manifest file.

In the popup.html file, we include the popup.css file for styling and popup.js file to enable user interaction with the HTML elements. We add a button element named "Summarize" which emits a click event when clicked. We also add a div element to display the summarized text received from the backend REST API Call. In the popup.css file, we provide appropriate CSS styling to the HTML elements, such as the button and div, for a better user experience.

It is important to note that the use of appropriate CSS styling can significantly improve the user experience of a web page or application by enhancing the look and feel of the interface, making it more intuitive and easier to navigate.[12]

1.7 DISPLAY SUMMARIZED TEXT

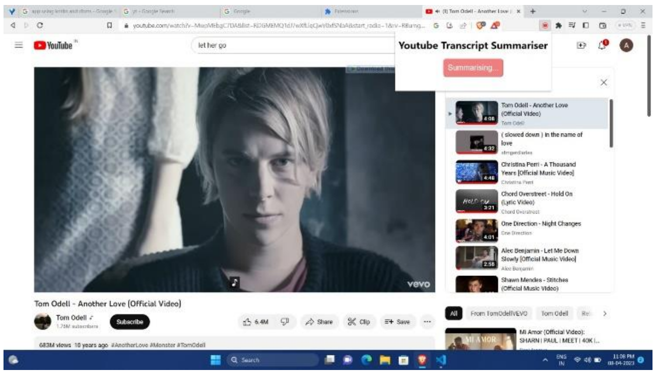
To enable interaction between the extension and backend server, we need to add functionality to make HTTP REST API Calls.
In popup.js, we attach an event listener to the Summarize button with the event type "click" and pass an anonymous callback function.
In the callback function, we use the chrome.runtime.sendMessage method to send an action message to contentScript.js to generate the summary.
We also add an event listener, chrome.runtime.onMessage, to listen for message results from contentScript.js, which will execute the outputSummary callback function.
In the callback function, we use JavaScript to programmatically display the summary in the div element.

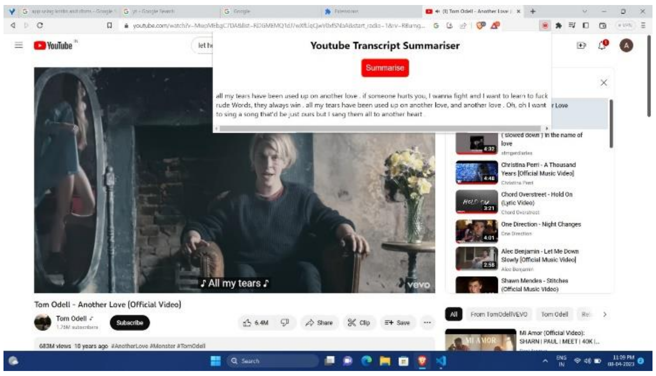
We also need to inject the content script contentScript.js into a particular page and execute the script automatically.

In contentScript.js, we add an event listener chrome.runtime.onMessage to listen to the message generator, which will execute the generate Summary callback function.
In the callback function, we extract the URL of the current tab, make a GET HTTP request using the XML HTTP Request Web API to the backend, and receive the summarized text as a response.
Then, we send an action message result with the summary payload using chrome.runtime.sendMessage to notify popup.js to display the summarized text.[13]

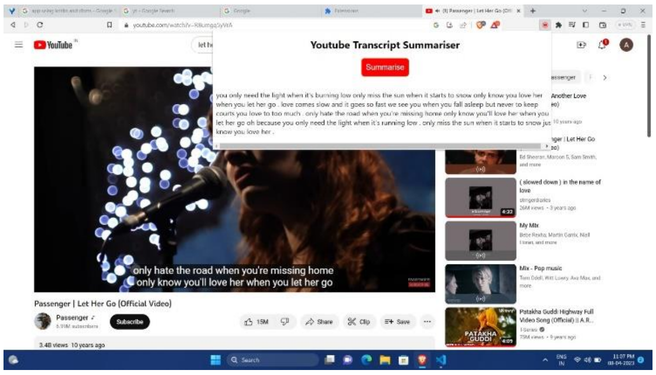
IV. RESULTS

The results of the project are shown below:





all my tears have been used up on another love . if someone hurts you, I wanna fight and I want to learn to fuck rude Words, they always win . all my tears have been used up on another love, and another love . Oh, oh I want to sing a song that'd be just ours but I sang them all to another heart .


Youtube Transcript Summariser
Summarise
Passenger | Let Her Go (Official Video)
Subscribe
only hate the road when you're missing home
only know you'll love her when you let her go

you only need the light when it's burning low only miss the sun when it starts to snow only know you love her when you let her go . love comes slow and it goes so fast we see you when you fall asleep but never to keep courts you love to too much . only hate the road when you're missing home only know you'll love her when you let her go oh because you only need the light when it's running low . only miss the sun when it starts to snow jus know you love her .

V. CONCLUSION

The proposed project is a YouTube transcript summarizer that utilizes a Google Chrome extension to access the transcript of a video and summarize it using Python API and transformers package. When the user clicks the summarize button on the Chrome extension webpage, the system accesses the transcript and generates a summary that is then displayed on the extension webpage.

This project offers a significant time-saving solution for users by providing a summary of the video without the need to watch the entire video. It also helps users to identify any inappropriate or harmful content before watching the video.

Furthermore, the project offers an excellent user interface experience by using Chrome extensions. Users can easily obtain the summarized text without having to copy and paste the video URL into third-party applications or terminals.

VI. FUTURE SCOPE

The field of YouTube transcript summarization is still evolving, and there is a lot of potential for future research and development. Some possible future scope for YouTube transcript summarizers are :

1 Multi-lingual support: As YouTube is a global platform with content in various languages, there is a need for transcript summarizers to support multiple languages. Future research can focus on developing transcript summarizers that can summarize transcripts in different languages.

2 Real-time summarization: Real-time summarization can be useful for live events or news broadcasts. Future research can focus on developing summarizers that can summarize the transcript in real-time.

3 Incorporating user feedback: User feedback can be used to improve the quality of the summary generated by the transcript summarizer. Future research can focus on developing summarizers that can incorporate user feedback to improve the accuracy and relevance of the summary.

4 Image and video analysis: The use of image and video analysis techniques can improve the accuracy and relevance of the summary generated by the transcript summarizer. Future research can focus on developing summarizers that can analyze images and videos to generate more informative summaries.

5 Summarization based on user preferences: Different users may have different preferences for the type of summary they want. Future research can focus on developing summarizers that can generate personalized summaries based on the user's preferences.

6 Evaluation metrics: There is a need for better evaluation metrics to measure the quality of the summaries generated by the transcript summarizer. Future research can focus on developing new evaluation metrics that can provide more insights into the quality of the summary.

In conclusion, there is a lot of potential for future research and development in the field of YouTube transcript summarization. The development of new techniques and technologies can lead to more accurate and relevant summaries, making it easier for users to extract information from YouTube videos.

REFERENCES

[1]. 'Automated Video Summarization Using Speech Transcript' by Cuneyt M. Taskiran, Aronon Amir, Dulce B. Ponceleon, Edward J. Delph

[2]. "Digital video Summarization Techniques", Ashenafi Workie, Rajesh Sharma, Yun Koo Chun

[3]. S. Tharun, R. Kranthi Kumar, P. Sai Sravanth, G. Srujan Reddy, B. Akshay, "Survey on Abstractive Transcript Summarization of YouTube Videos", in International Journal of Advanced Research in Science, Communication and Technology (IJARSCT)

[4]. Nallapati, R., Zhou, B., Gulcehre, C., & Xiang, B. (2017). Summarunner: A recurrent neural network based sequence model for extractive summarization of documents. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 31, No. 1).

[5]. Nguyen, T. T., Nguyen, M. Q., Nguyen, L. T., & Nguyen, H. N. (2019). A hybrid approach for summarizing youtube video transcripts. Information Processing & Management, 56(6), 1444-1459.

[6]. Zeng, J., Wei, F., & Liu, S. (2020). Learning to summarize from human feedback on summary prototypes. In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics (pp. 5641-5647).

[7]. Huang, X., Shi, Y., Xiong, W., & Zhang, J. (2021). EduSum: A large-scale dataset and neural model for automated educational video summarization. In Proceedings of the 2021 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies (pp. 452-462).

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

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

[10]. https://medium.com/swlh/parsing-rest-api-payload-and-query-parameters-with-flask-better-than-marshmallow-aa79c889e3ca

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

[12]. https://developer.mozilla.org/en-US/docs/Web/API/XMLHttpRequest/Using_XMLHttpRequest