

A Project Report
on
YOUTUBE TRANSCRIPT SUMMARIZER

*Submitted in partial fulfillment of the
requirement for the award of the degree of*

B.Tech SCSE



(Established under Galgotias University Uttar Pradesh Act No. 14 of 2011)

**Under The Supervision of
Ms Suman Devi
Assistant Professor**

Submitted By
19SCSE1010612- TRIPTI

**SCHOOL OF COMPUTING SCIENCE AND ENGINEERING
DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA
INDIA
DECEMBER, 2021**



**SCHOOL OF COMPUTING SCIENCE AND
ENGINEERING
GALGOTIAS UNIVERSITY, GREATER NOIDA**

CANDIDATE'S DECLARATION

We hereby certify that the work which is being presented in the thesis/project/dissertation, entitled "YOUTUBE TRANSCRIPT SUMMARIZER" in partial fulfillment of the requirements for the award of the Btech submitted in the School of Computing Science and Engineering of Galgotias University, Greater Noida, is an original work carried out during the period of october, 2021 to December, 2021, under the supervision of Ms Suman Devi, Assistant professor, Department of Computer Science and Engineering/Computer Application and Information and Science, of School of Computing Science and Engineering , Galgotias University, Greater Noida

The matter presented in the project has not been submitted by me for the award of any other degree of this or any other places.

Tripti,19SCSE1010612

This is to certify that the above statement made by the candidates is correct to the best of my knowledge.

Ms Suman Devi
Assistant Professor

CERTIFICATE

The Final Thesis/Project/ Dissertation Viva-Voce examination of Tripti:19SCSE1010612 has been held on _____ and his/her work is recommended for the award of B.Tech

Signature of Examiner(s)

Signature of Supervisor(s)

Signature of Project Coordinator

Signature of Dean

Date: 22nd December 2021

Place: Greater Noida

ABSTRACT

People are watching YouTube videos daily which can be educational, documentary or of any genre with longer length to which maybe most of the people don't have time to watch completely. Enormous number of video recordings are being created and shared on the Internet throughout the day. It has become really difficult to spend time watching such videos which may have a longer duration than expected and sometimes our efforts may become futile if we couldn't find relevant information out of it

Think about how much time can be saved if someone narrated the whole summary of the video to us in shorter length of paragraphs . Summarizing transcripts of such videos automatically allows us to quickly lookout for the important patterns in the video and helps us to save time and effort to go through the whole content of the video.

In this project, we will be targeting to make one such creating Chrome Extension which will make a request to backend API where it will perform NLP and respond with a summarized version of a YouTube transcript. . Summarizing transcripts of such videos automatically allows us to quickly lookout for the important patterns in the video and helps us to save time and effort to go through the whole content of the video.

Table of Contents

Title	Page No.
Candidates Declaration	I
Abstract	III
Contents	IV
List of Table	V
List of Figures	VI
Acronyms	VII
LITERATURE REVIEWS	8
PROBLEM FORMULATION	11
REQUIRED TOOLS	12
COMPLETE WORK PLAN OUT	13
REFERENCES	

List of Figures

S.No.	Title	Page No.
1	User scenario diagram	13
2	Work flow diagram	15

Acronyms

NLP	Natural Language Processing
API	Application Programming Interface
REST	Representational state transfer

1. LITERATURE REVIEWS

YouTube is an American free to use online video sharing and social media platform launched in February 2005. It is currently one of the biggest video platforms where its users watch more than 1 billion hours of videos every day. Closed captions are the text derived from the video which are intended for adding more details (such as dialogues, speech translation, non-speech elements) for the viewer. They are widely used to understand video without understanding its audio.

We spend a noticeable amount of our weekly time watching YouTube videos, be it for entertainment, education, or exploring our interests. In most cases, the overall intent is to obtain some form of information from the video. We were seeking a solution to increase the efficiency of this "information extraction" process as YouTube's speed adjustment option is the only relevant tool. Enormous number of video recordings are being created and shared on the Internet throughout the day. It has become really difficult to spend time watching such videos which may have a longer duration than expected and sometimes our efforts may become futile if we couldn't find relevant information out of it. Summarizing transcripts of such videos automatically allows us to quickly lookout for the important patterns in the video and helps us to save time and effort to go through the whole content of the video. Various organizations today, be it online shopping, private sector organizations, government, tourism, and catering industry, or any other institute that offers customer services, are all concerned to learn their customer's feedback each time their services are utilized. Now, consider that these companies are receiving an enormous amount of feedback and data every single day. It becomes quite a tedious task for the management to analyze each of these data points

and come up with insights. However, we have reached a point in technological advancements where technology can help with the tasks and we ourselves do not need to perform them. One such field that makes this happen is Machine Learning. Machines have become capable of understanding human language with the help of NLP or Natural Language Processing. Today, research is being done with the help of text analytics. One application of text analytics and NLP is Text Summarization. Text Summarization Python helps in summarizing and shortening the text in the user feedback. It can be done with the help of an algorithm that can help in reducing the text bodies while keeping their original meaning intact or by giving insights into their original text.

This project will give us 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. This project will allow us to have hands-on experience with state-of-the-art NLP techniques for abstractive text summarization and implement an interesting idea suitable for intermediates and a refreshing hobby project for professionals.

REST is a set of architectural constraints, not a protocol or a standard. API developers can implement REST in a variety of ways. When a client request is made via a RESTful API, it transfers a representation of the state of the resource to the requester or endpoint. This information, or representation, is delivered in one of several formats via HTTP: JSON (Javascript Object Notation), HTML, XML, Python, PHP, or plain text. JSON is the most generally popular file format to use because, despite its name, it's language-agnostic, as well as readable by both humans and machines. Something else to keep in mind: Headers and parameters are also important in the HTTP methods of a RESTful API HTTP request, as they contain important identifier information as to the request's metadata, authorization, uniform resource identifier (URI), caching, cookies, and more. There are request headers and

response headers, each with their own HTTP connection information and status codes. Representational state transfer (REST) is a software architectural style that was created to guide the design and development of the architecture for the World Wide Web. REST defines a set of constraints for how the architecture of an Internet-scale distributed hypermedia system, such as the Web, should behave. The REST architectural style emphasises the scalability of interactions between components, uniform interfaces, independent deployment of components, and the creation of a layered architecture to facilitate caching components to reduce user-perceived latency, enforce security, and encapsulate legacy systems.

2. **PROBLEM FORMULATION:**

Youtube is an online video download, upload, and stream platform which encounters enormous viewers on daily basis. People often come across long videos of long length like an hour or so and sometimes it is not feasible for them to watch the whole video as it is time taking and less efficient for their situation. In that case we always want if someone could just recite a summary of the video. Here is exactly what we are doing. Using natural language processing we can summarize the transcript of the video. Further producing the summary of the video.

3. **REQUIRED TOOLS:**

- Processor: minimum 2.0GHz
- RAM: 4 GB
- Hard disk: 100 GB
- Input device: standard keyboard
- Software: visual studio
- OS: windows 7 and above, linux, mac
- Languages: Python, Flask, HTML,CSS.

4. COMPLETE WORK PLAN :

YouTube Video Transcript Summarization over Flask:

This back-end uses Flask framework to receive API calls from the client and then respond with the summarized text response. This API can work only on those YouTube videos which have well-formatted closed captions in it. The same backend also hosts a web version of the Summarizer to make those API calls in simple way and show the output within the webpage.

Pre-requisite Knowledge:

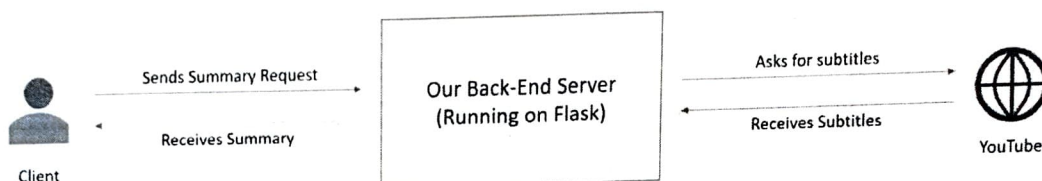
YouTube is an American free to use online video sharing and social media platform launched in February 2005. It is currently one of the biggest video platforms where its users watch more than 1 billion hours of videos every day.

Closed captions are the text derived from the video which are intended for adding more details (such as dialogues, speech translation, non-speech elements) for the viewer. They are widely used to understand video without understanding its audio.

Use case Scenario:

YouTube has very large number of videos which has transcripts. Summarization would be especially helpful in the cases where videos are longer in length and different parts might have varying importance. In this sense, Summarization of the

video might be useful in saving the viewer's time. It will help in improving user productivity since they will focus only on the important text spoken in video.



There are four endpoints:

(Root Endpoint): It displays a general purpose introductory webpage and also provides links to web summarizer and API information.

(Web Summarizer Endpoint): It displays the web version of the summarizer tool. The webpage has input elements and a summarize button. After clicking summarize, the API is called and the response is displayed to the user.

(API Description Endpoint): The webpage at this endpoint describes basic API information in case you would like to use it.

(API Endpoint): This endpoint is for **API purposes only**. That is why, the response type of the **GET Request** at this endpoint is in JSON format.

Sending request to our API:

The query (or API request) to our backend can be made using following three variables only. They are:

- **id** : Video ID of the YouTube Video. Each video has its own unique ID in its URL. For example, 9No-FiEInLA is the Video ID in <https://www.youtube.com/watch?v=9No-FiEInLA>.

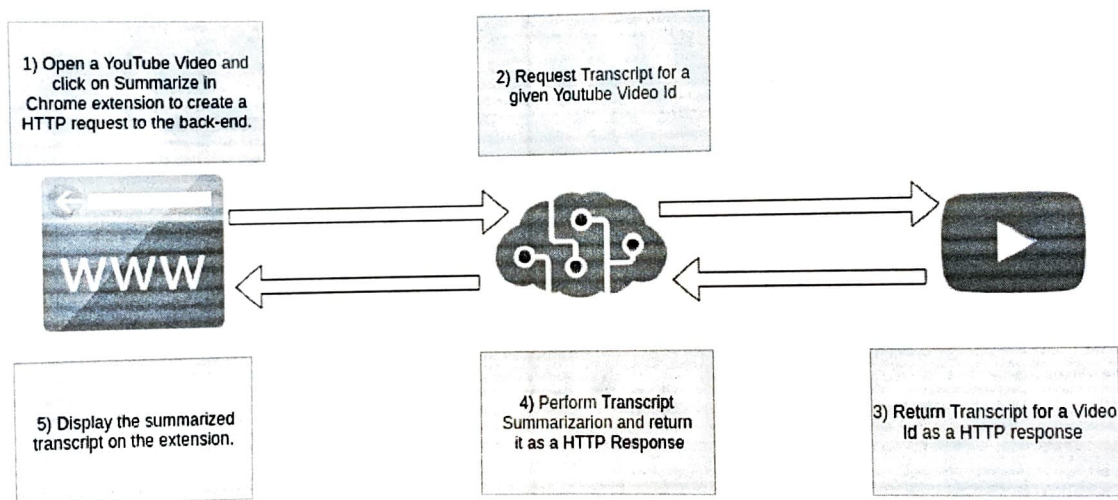
- **choice** : Algorithm Choice for the summarizing the Transcript. There are only six accepted values in this variable.

These choices are written along with algorithm names as follows:

- gensim-sum : Text Rank Algorithm Based using Gensim
- spacy-sum : Frequency Based Approach using Spacy.
- nltk-sum : Frequency Based Summarization using NLTK.
- sumy-lsa-sum : Latent Semantic Analysis Based using Sumy.
- sumy-luhn-sum : Luhn Algorithm Based using Sumy.
- sumy-text-rank-sum : Text Rank Algorithm Based using Sumy.

Receiving request from our API

Once you send a successful API request, our server will take that request and process it. After successful processing, the server will send back the relevant response to the made request. The response sent is always in the JSON Format



IMPLEMENTATION:

So far the code for app has been written and is yet to be debugged.

We have used libraries like

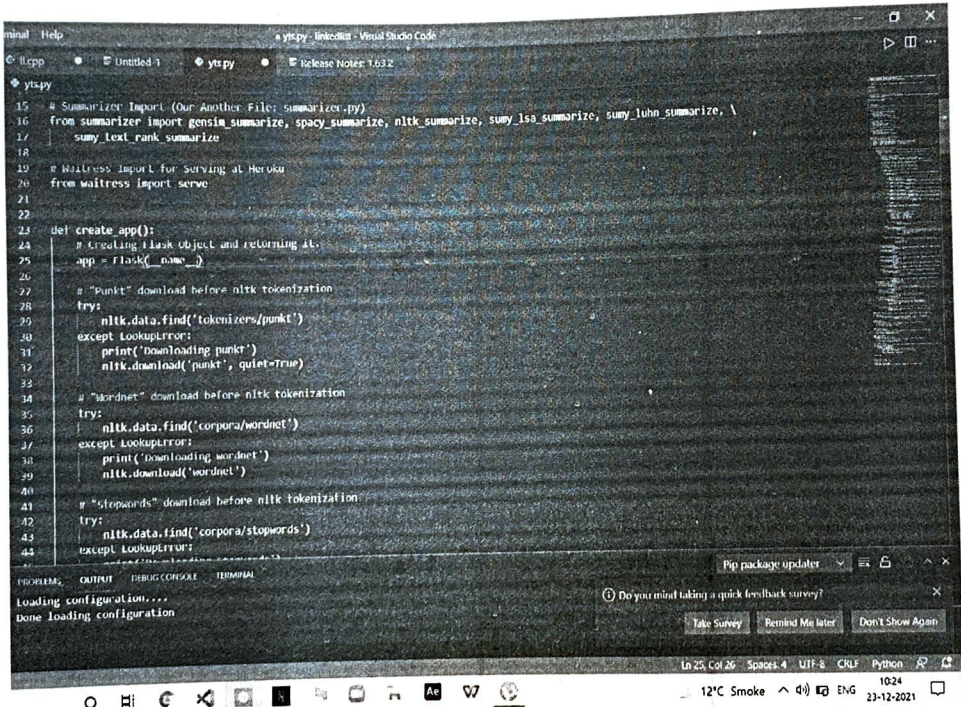
Numpy

Flask

Panda

Etc

The code is in DOM format:



```
15 # Summarizer Import (Our Another File: Summarizer.py)
16 from summarizer import gensim_summarize, spacy_summarize, nltk_summarize, sumy_lsa_summarize, sumy_luhn_summarize, \
17    sumy_text_punk_summarize
18
19 # waitress import for serving at Heroku
20 from waitress import serve
21
22
23 def create_app():
24     # Creating Flask Object and returning it.
25     app = flask(__name__)
26
27     # "punkt" download before nltk tokenization
28     try:
29         nltk.data.find('tokenizers/punkt')
30     except LookupError:
31         print('downloading punkt')
32         nltk.download('punkt', quiet=True)
33
34     # "wordnet" download before nltk tokenization
35     try:
36         nltk.data.find('corpora/wordnet')
37     except LookupError:
38         print('downloading wordnet')
39         nltk.download('wordnet')
40
41     # "stopwords" download before nltk tokenization
42     try:
43         nltk.data.find('corpora/stopwords')
44     except LookupError:
```

PROBLEMS: OUTPUT DEBUG CONSOLE TERMINAL
Loading configuration...
Done loading configuration

Pip package updater
Do you mind taking a quick feedback survey?
Take Survey Remind Me later Don't Show Again

Ln 25, Col 26 Spaces: 4 UTF-8 CRLF Python 1024
12°C Smoke ENG 23-12-2021


```
terminal - Help
yts.py - LinkedIn - Visual Studio Code
yts.py
198 @app.route('/web/')
199 def summarize_web():
200     # We are at web.html, online input boxes are there to summarize the given video URL.
201     # Displaying web.html to the end user
202     return render_template('web.html')
203
204 @app.route('/api/')
205 def summarize_api_info_route():
206     # Since we have two end points inside root, we are closing root endpoint.
207     # Displaying root.html to the end user
208     return render_template('api.html')
209
210 @app.before_request
211 # Before Request function, we are redirecting any HTTP requests to HTTPS especially on heroku environment.
212 def enforce_https_in_heroku():
213     if 'ONNO' in os.environ:
214         if request.headers.get('X-Forwarded-Proto') == 'http':
215             url = request.url.replace('http://', 'https://', 1)
216             code = 301
217             return redirect(url, code=code)
218
219 return app
220
221
222 if __name__ == '__main__':
223     # Running flask Application
224     # app.run()
225     flask_app = create_app()
226     serve(flask_app, host='0.0.0.0', port=80, debug=False, url_scheme='https')
```

```
terminal - Help
yts.py - LinkedIn - Visual Studio Code
yts.py
71
72 # Pre-check if the summary will have at least one line
73 select_length = int(num_sent_text * (int(percent) / 100))
74
75 # Summary will have at least 1 line. Proceed to summarize.
76 if select_length > 0:
77
78     # Condition satisfied for summarization, summarizing the formatted text based on choice.
79     if num_sent_text > 1:
80
81         # Summarizing formatted text based upon the request's choice
82         if choice == "gensim-sum":
83             summary = gensim_summarize(formatted_text,
84                                       percent) # Gensim library for TextRank based Summary.
85         elif choice == "spacy-sum":
86             summary = spacy_summarize(formatted_text,
87                                       percent) # Spacy Library for frequency-based summary.
88         elif choice == "nltk-sum":
89             summary = nltk_summarize(formatted_text,
90                                     percent) # NLTK Library used for frequency-based summary.
91         elif choice == "sumy-lsa-sum":
92             summary = sumy_lsa_summarize(formatted_text,
93                                         percent) # Sumy for extractive summary using LSA.
94         elif choice == "sumy-luhn-sum":
95             summary = sumy_luhn_summarize(formatted_text,
96                                           percent) # Sumy Library for TF-IDF Based Summary.
97         elif choice == "sumy-text-rank-sum":
98             summary = sumy_text_rank_summarize(formatted_text,
99                                                percent) # Sumy for Text Rank Based Summary.
100         else:
101             summary = None
```

REFERENCES

- [1]. M. G. Christel, M. A. Smith, R. Taylor, and D. B. Winker, "Evolving video skims into useful multimedia abstractions," Proceedings of the ACM Computer-Human Interface Conference (CHI'98), April 18-23 1998, Los Angeles, CA, pp. 171-178.
- [2]. L. He, E. Sanocki, A. Gupta, and J. Grudin, "Comparing presentation summaries: Slides vs. reading vs. listening," Conference on Human Factors in Computing Systems (CHI 2000), April 1-6 2000, The Hague, Netherlands, pp. 177-184.
- [3]. S. Srinivasan, D. Ponceleon, A. Amir, B. Blanchard, and D. Petkovic, "Engineering the web for multimedia," Proceeding of the Web Engineering Workshop (WEBE), WWW-9, May 15-19 2000, Amsterdam, Netherlands.
- [4]. A. W. M. Smeulders, M. Worring, S. Santini, A. Gupta, and R. Jain, "Content-based image retrieval at the end of the early years," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 22, no. 12, pp. 1349-1380, December 2000.
- [5]. M. M. Yeung and B.-L. Yeo, "Video visualization for compact presentation and fast browsing of pictorial content," IEEE Transactions on Circuits and Systems for Video Technology, vol. 7, no. 5, pp. 771-785, October 1997.
- [6]. Y. Taniguchi, A. Akutsu, and Y. Tonomura, "Panorama excerpts: Extracting and packing panoramas for video browsing," Proceedings of the ACM Multimedia, November 9-13 1997, Seattle, WA, pp. 427-436.
- [7]. N. Vasconcelos and A. Lippman, "Bayesian modeling of video editing and structure: Semantic features for video

summarization and browsing," Proceedings of IEEE International Conference on Image Processing, October 4-1998, Chicago, IL.