



Technology Reviews

Team TL;DW

Anurag Agarwal

Mihir Gathani

Joobee Jung

Trisha Prasant

Background & Use Case



Our project:

TL;DW is a web application designed to provide users with a summary of YouTube videos along with recommendations for TED Talks and podcasts related to the video content.

Use Case:

- Our aim is to assist users in gaining a deeper understanding of YouTube content by offering curated supplementary resources.
- For example, if a user watches a video on climate change, the application would generate a summary of key points from the video and suggest TED Talks or podcasts that delve further into the subject matter.

Why use a Python Library?

- Our application needs a Python library to quickly create a transcript of a Youtube link based on the audio content of the video.

Python Package Choices



1. Youtube-transcript-api

- Developed by Jonas Depoix, YouTube-transcript-api is a Python library facilitating retrieval of transcripts from YouTube video links.
- Offers a straightforward approach without needing Selenium-based methods or API keys, simplifying access to both manually and automatically generated subtitles.
- Enables easy extraction of textual content from YouTube videos for tasks like transcription, analysis, and summarization, eliminating complex setups or dependencies.

2. BeautifulSoup4

- It is created by Leonard Richardson, and is a Python library tailored for web scraping and parsing HTML and XML documents.
- It is equipped with tools for navigating, searching, and modifying HTML page structures, facilitating extraction of relevant data from web pages.
- BeautifulSoup utilizes intuitive syntax, streamlining the process of extracting text, links, and other elements from web pages.

Package Comparison



Features	Youtube-transcript-api	Beautifulsoup4
Addressing requirements of project	Returns a list of dictionaries including subtitle text, start time, duration in seconds. Works for automatically generated subtitles.	Returns the raw text of HTML contents on the YouTube video page, including video subtitles, details, and description.
Compatibility with other elements	Needs to merge each subtitle into one string from each dictionary.	Needs to merge each string into one string.
Ease of use	Designed for extracting YouTube transcripts	Requires manual effort to extract Youtube data
Computational efficiency	Optimized for efficiently extracting YouTube transcripts using a single YouTube video URL.	Depending on the complexity of the HTML structure of the YouTube video page.
Software Bugs	11 open issues on github. Allow one transcript per language code.	18 open bugs. Improvement like adding type annotations.
Availability of examples	Available example for scraping YouTube transcript	Hard to find a youtube transcript scraping example
Release date	Jul 3, 2018.	Jan 20, 2014 (Initial version : 2004)

Our Choice - Youtube Transcript API



- **What we chose:**

We decided to go ahead with using the Youtube-transcript-api for extracting transcripts from Youtube videos.

- **Why we chose it:**

- The YouTube Transcript API automatically generates summaries of the video without any need for manual intervention, streamlining operations and ensuring a seamless process.
- The API returns the transcripts in dictionary format that enhances debugging capabilities, as well as makes it easier for the summarization module compared to HTML format provided by BeautifulSoup4.

Drawbacks/Remaining Concerns



- **Potential Drawbacks**

- The API crashes when it receives a video with no transcript and the user will be forced to try a different video.
- It collects no metadata about the video, which could help find a more appropriate podcast/ video.

- **Will you have to mitigate any drawbacks?**

- Initial exploration does not show any major drawbacks that we need to mitigate. However might consider using BeautifulSoup4 to get metadata in case of videos with no transcript.

- **What will you watch out for?**

- Consistencies in the format of data being returned.
- Any API call limitations, if they exist.

Package Demonstration : Youtube-transcript-api

MSDS Graduation 2023 Youtube Video

https://www.youtube.com/watch?v=0XYMXdfyl60&t=1460s&ab_channel=UWVideo

```
[11] !pip install youtube-transcript-api
```

```
Collecting youtube-transcript-api
  Downloading youtube_transcript_api-0.6.2-py3-none-any.whl (24 kB)
Requirement already satisfied: requests in /usr/local/lib/python3.10/dist-packages (from youtube-transcript-api) (2.31.0)
Requirement already satisfied: charset-normalizer<4,>=2 in /usr/local/lib/python3.10/dist-packages (from requests->youtube-transcript-api) (3.3.2)
Requirement already satisfied: idna<4,>=2.5 in /usr/local/lib/python3.10/dist-packages (from requests->youtube-transcript-api) (3.6)
Requirement already satisfied: urllib3<3,>=1.21.1 in /usr/local/lib/python3.10/dist-packages (from requests->youtube-transcript-api) (2.0.7)
Requirement already satisfied: certifi>=2017.4.17 in /usr/local/lib/python3.10/dist-packages (from requests->youtube-transcript-api) (2023.11.17)
Installing collected packages: youtube-transcript-api
Successfully installed youtube-transcript-api-0.6.2
```

```
from youtube_transcript_api import YouTubeTranscriptApi
```

```
video_id = '0XYMXdfyl60'
```

```
YouTubeTranscriptApi.get_transcript(video_id)
```

```
{'text': 'out into the world and a happy industry sponsor\nthat wants to work with our students.',
 'start': 2584.74,
 'duration': 5.58},
{'text': "In last nights event and tonight's event,\nI even see some joy.",
 'start': 2590.32,
 'duration': 5.37},
{'text': 'It would be a life to say -- between viruses,\npolitics, environmental issues they would',
```

Appendix: BeautifulSoup4 Demo

MSDS Graduation 2023 Youtube Video


https://www.youtube.com/watch?v=0XYMXdfyl60&t=1460s&ab_channel=UWVideo

✓ 7s [13] !pip install beautifulsoup4

Requirement already satisfied: beautifulsoup4 in /usr/local/lib/python3.10/dist-packages (4.12.3)

Requirement already satisfied: soupsieve>1.2 in /usr/local/lib/python3.10/dist-packages (from beautifulsoup4) (2.5)



✓ 0s  from bs4 import BeautifulSoup
import requests

```
video_url = 'https://www.youtube.com/watch?v=0XYMXdfyl60&t=1460s&ab_channel=UWVideo'  
response = requests.get(video_url)  
soup = BeautifulSoup(response.text, 'html.parser')  
transcript_data = soup.find('body')  
print(transcript_data.text)
```