INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor : 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 01, January 2024, pp : 53-56

COMPARISON OF VARIOUS YOUTUBE TRANSCRIPT SUMMARIZER TECHNIQUES

Tripti Sharma[1], Neetu Anand[2]

[1]Professor, IT Department, MSIT, New Delhi, India.

[2]Associate Professor, Department of Computer Application, MSI, New Delhi, India.

ABSTRACT

In this paper we aim to create a hassle-free user experience through which a user can summarize any youtube video and save it and use it again at any given point in time. In this paper we will compare the existing youtube transcript summarizer techniques. In any instance, there is a staggering number of video recordings that are available on the internet, and also more are being created at the same time as well. That effort can be in vain if we find it hard to find time to watch longer-than-expected movies, and if useful information can't be extracted from them. Automatically summarizing movie transcripts like this can quickly identify important patterns in your videos, saving time and effort by not having to review the entire content. This work uses a flask server for text transcription. Then natural language processing (NLP) is used to summarize the transcript. About the front end, it is built using react. The website has all the necessary features which a user may require such as saving the notes for review, and a folder structure to organize similar notes. Also, we have compared the different algorithms usedto summarize text on the server as well based on some metrics such as time taken, cosine values, and more.

Keywords: Summarization, Natural Language Processing, Hugging-Face Transformers, LSA Algorithm, Text Rank Algorithm

1. INTRODUCTION

About 500 hrs of video content is being shared on youtube at any passing minute. The number of active YouTube users is about 2.5 billion users in 2022 and is growing rapidly year by year. According to a Google survey, nearly a third of YouTube viewers in India watch videos on their mobile phones and spend more than 48 hours a month on the website [1]. That is all fine if you are using youtube for entertainment purposes but it gets tricky if you want to learn something or just get an overview of a topic very quickly but the content available is in hours. That's the exact problem that can be solved using automatic summarization tools [2]. Searching for videos that contain the information you're looking for can be thwarting and long-delayed. For example, there are several videos on the Internet talking about a specific topic, but it is difficult to know what the speaker is trying to convey to the audience without watching the entire video. That's why an automatic summarization tool is the need of the hour. The tool we have used uses NLP (natural language processing). In natural language processing, there are two types of summarizations: abstractive and extractive. In extractive summarization we take sentencesas it is from the original text which seems relevant and more important, no modification is done. But, in abstractive summarization, it tries to guess the meaning of the sentences and creates its own words and sentences for the important information. We are using the abstractive summarization method in this work. To implement these algorithms, we use python as our primary language. Python has various packages that are very useful [3]. Access to YouTube content is now simplified through APIs in Python libraries such as B.Video transcripts, etc. We use this to our advantage to directly access video transcripts, aggregate them and display them to the user. There is also another algorithm that is already modeled in python known as the hugging face transformer. That too is also used in this work for comparison.

2. METHODOLOGY

As a videotape summarization system, it brings powerful expertise. Algorithms based on machine learning models require high processing power. Recapturing a videotape based on cut lines is the easiest way to summarize a videotape. It's easier and faster to work with text than it is to train colorful videos. Machine Learning models can be used to achieve this.

YouTube Video:

The variety of YouTube videos includes short films, music and audio clips, spot movies, images, audio recordings, commercial film camps, live channels, vlogs and other content from popular YouTubers. Over 1 billion hours of videotapes are watched on YouTube every day. Therefore, YouTube videos were considered as data for the proposed algorithmto add videotapes. The YouTube Paraphrase API retrieves slogans from a specific videotape using links [5]. Video downloading process is not convenient and requires a lot data and storage. For this, you must first copy the url of the video and paste the url to the YouTube video downloader website. This loading system takes time. Pytube is a very lightweight and powerful Python library used to seamlessly download videos.

INTERNATIONAL JOURNAL OF PROGRESSIVE RESEARCH IN ENGINEERING MANAGEMENT AND SCIENCE (IJPREMS)

e-ISSN : 2583-1062

Impact Factor : 5.725

www.ijprems.com
editor@ijprems.com

Vol. 04, Issue 01, January 2024, pp : 53-56

YouTube module object by passing the YouTube videotape link as a parameter created bythe Pytube library. It also supports the correct videotape length and resolution. The train names may be for Stoner's convenience.

(a) Latent Semantic Analysis Algorithm:

It is an unmonitored method to natural language processing. It is an extractive summarization algorithm that extracts functions from judgments that cannot be addressed altogether. It is a natural language processing technique, specifically distributed semantics, that organizes related terms related to a document, allowing us to understand the differences between a set of documents and the terms they contain. Analyze relationships. LSA assumes that words having similar meanings appear in similar parts of text (distribution hypothesis). The document word count matrix (rows representing different words and columns representing each document) is generated from large text and maintained using Singular Value Decomposition (SVD) [6]. while reducing thenumber of lines. Similar structure between columns. Documents are compared by cosine similarity between any two columns. Values close to 1 represent very similar documents, values close to 0 represent very different documents

Working of LSA:

1. Concept co-occurrence matrix: The matrix has dimensions of (vocabulary size) (vocabulary size). It represents the frequency of words coming together in the dataset. A matrix helps in understanding sentences that belong together. The similarity between two different aggregates is computed using cosine similarity between aggregate matrices.
2. Singular Value Decomposition: SVD decomposes a matrix into three different matrices namely orthogonal column matrix, orthogonal row matrix, and singular matrix.
3. Set Selection: Use the SVD results to select the significant set using various algorithms.

(b) Text Rank Algorithm:

With the concept of text rank algorithm, we organize the most important sentences in the text and draw conclusions. For the AutoSum task, TextRank models each document as a graph, using sentences as nodes [7]. This function calculates the similarity of sentences and forms boundaries between them. This function is used to weight the edges of a graph, andthe more similar the sets are, the more important the edges between sets in the graph are. TextRank determines the similarity of two sentences based on their common content. This overlap is simply calculated as the number of common lexical tokens between them divided by the length of each, avoiding the promotion of long sentences.

Working:

1. Concatenate all the text in the article.
2. Next, break the text into individual sentences.
3. The next step is to find the vector representations of the individual sentences.
4. Resemblance between sentence vectors is computed and stored in a matrix.
5. The similarity matrix is transformed into a graph with sentences as vertices and similarity as edges, and sentence ranks are calculated.
6. Finally, a certain number of top sentences constitute the final summary of the text.

(c) Hugging-Face:

Hugging Face Transformer uses an abstract summarization approach in which the model unfolds new sentences in new forms, generating unique texts that are shorter than the original sentences, just like humans do. Hugging Face is an Artificial Intelligence community and Machine Learning platform [8]. We aim to democratize NLP by giving data scientists, AI practitioners, and engineers instant access to over 20,000 pre-trained models based on the state-of-the-art Transformer architecture [9]. These models are applicable to:

- **Text:** In the text over100 languages to perform tasks such as classification,information extraction, question answering, generation, generation, and translation.
- **Speech** to recognize and differentiate audio or speeches.
- **Vision** to classify images or detect objects and segmentations**.**
- **Tabular data** for regression and classification problems.
- **Reinforcement** Learning Transformer.

Working:

1. Import pipelines from Transformers. This imports the pipeline functionality so you can easily use different pre-trained models.
2. Read articles stored in text files.

3. Initialize and configure the summary pipeline and generate summaries using BART.
4. Print summary text.

3. COMPARISON

a. Cosine Similarity: Cosine similarity is a measure of resemblance between two nonzero vectors. Computed as the angle between these vectors (this is the same as the dot product). This work uses cosine similarity and nltk toolkit modules to calculate cosine similarity of summary_text from original_text.

Functions used:

nltk. tokenize:

Used for tokenization. Tokenization is the process of breaking large amounts of text into smaller pieces called tokens. Word tokenize(X) splits the given sentence X into words and returns a list.

nltk. corpus:

This program uses it to get a list of stop words. Stop words which are commonly used words for the program, such as "the", "a", "an", "in".

Table 1.1: Execution Time & Cosine Similarity Of Different Algorithms

Text SummarizationAlgorithms	Video Id	Execution Time(in seconds)	Cosine SimilarityScore
Latent SemanticsAnalysis	**tznztJVsW9E**	**0.25**	**0.839**
Text Rank	**tznztJVsW9E**	**0.006**	**0.839**
Hugging Face	**tznztJVsW9E**	**133.68**	**0.569**

4. RESULT

Table 1.1 suggests the effects of the proposed set of rules used to achieve video summaries the usage of video subtitles. So the processing time of the results depends on the length of input tape and the total time desired by the user. According to the results obtained from Table 1, the similarity scores of the LSA and Text Rank algorithms are comparable and higher than the hugging face algorithm. Hugging Face also takes longer to run than his two other algorithms. In this work, we have shown the results of all three algorithms along with the similarity score and execution time.

5. CONCLUSION

This work aims to summarize YouTube videos and evaluate the best algorithms for summarization in terms of running time and cosine similarity.

The Python library sumy returns a textbook document summary for a set of rules given as arguments. A large number of summarization algorithms are available using this library. So on the research and the result obtained, it can be decide that the best text summarization algorithm amongall the three algorithms , is the Text Rank algorithm implemented using Sumy (NLP method) in Python.

6. FUTURE SCOPE

There are other algorithms which can also be used to summarize the transcript of a Youtube video using other metric as well. This paper provides the summarization through three summarization algorithms namely Latent Semantic Analysis(LSA), TextRank & Hugging Face using transformers. This work can be extended in future for transcribing videos without subtitles.

7. REFERENCES

[1] Dictionary.com. (n.d.). Summary definition & meaning. Dictionary.com. Retrieved March 15, 2022, from https://www.dictionary.com/browse/summary

[2] Mehta, P., & Majumder, P. (2019). Introduction. From Extractive to Abstractive Summarization: A Journey, 1–9.

[3] https://www.sciencedirect.com/topics/computer-science/text-summarization

[4] Desarda, A. (2020). Working with hugging face Transformers and TF 2.0. Medium, https://towardsdatascience.com/working-with-hugging-facetransformers-and-tf-2-0-89bf3 5e3555a

[5] Joshi, P. (2022, April 5). What are transformers in machine learning. What Are Transformers In Machine Learning. Retrieved March 15, 2022, from https://prateekjoshi.substack.com/p/what-aretransformers-in-machine

[6] Sanjana R, Sai Gagana V, Vedavathi K R, KiranK N. (2021). Video Summarization using NLP International Research Journal of Engineering and Technology (IRJET). Vol 08, No. 08. pp. 3672-3675

[7] Zhou, K., Qiao, Y., & Xiang, T. (2018). Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward. In Proceedings of the AAAI Conference on Artificial Intelligence, Vol. 32, No. 1.

[8] Bhandare, M. K., Chigare, A. A., Patil, U. U., & Sangle, S. B. (2022). YOUTUBE TRANSCRIPT SUMMARIZER. International Research Journal of Modernization in Engineering Technology and Science Vol. 04, No. 03.

[9] Porwal, K., Srivastava, H., Gupta, R., Pratap Mall, S. and Gupta, N. (2022). Video Transcription and Summarization using NLP. Available at SSRN 4157647.