CSE4022 Natural Language Processing

PROJECT REPORT

on

Video Summarization Using Transformer Models

Prepared by

Vinayak Agarwal (20BCT0318)

Harshita Rajput (20BCE0752)

Kartikeya Rawat (20BCE0641)

Dev Rishi (20BCE0965)

Dhruv Singh (20BCI0318)

VIT®

Vellore Institute of Technology

(Deemed to be University under section 3 of UGC Act, 1956)

School of Computer Science and Engineering

Vellore Institute of Technology, Vellore.

Introduction

A developing phenomena in today's fast-paced digital scene is video fatigue, which is caused by the sheer volume of video content that is available online. With a plethora of videos available to consumers, ranging from social media clips to instructional lectures, the time and focus needed to view each video through to the end have become valuable resources. As a result, users have a strong urge to skip the boredom and swiftly get the information they need without having to view protracted films. This need for conciseness is a result of the urge to be efficient as well as the realization that, in the era of information overload, time is a finite resource. As a result, there is a growing need for video summarizing techniques, which is indicative of a fundamental shift in user preferences toward digestible but informative material. This change recognizes the difficulties associated with content saturation and emphasizes the need for creative solutions, including video summary, to meet the changing needs and expectations of today's online audience.

The project aims to revolutionize the consuming experience by implementing state-of-the-art Transformer model in response to the deluge of internet video material. The work centers on video summarizing, which is the skill of condensing long recordings into brief but thorough summaries. The goal is to usher in a new era of effective and sophisticated video understanding by utilizing the power of Transformers, which are well-known for their achievements in natural language processing. This revolutionary method could completely change the way we interact with video in a variety of contexts in addition to solving the problem of content overload.

Abstract

A new era of ease and connectivity has been brought about by the widespread use of videos in the ever-changing environment of online content consumption. But this availability also presents a problem: consumers are becoming less willing to spend time watching long films. This condition, which is sometimes referred to as "video fatigue," captures the feeling of people who find it more and more tiresome to sort through an abundance of video content. As a result of users realizing this changing tendency, their behavior has markedly changed. These days, people are more likely to look for succinct yet insightful video summaries than to sit through the entire thing. This change in inclination is the result of a desire for efficiency, realizing that, in the midst of the flood of digital information, time is a limited resource. Video summarizing solutions are becoming more and more in demand as a result of this shifting terrain. Users long for the opportunity to quickly skim videos for the most important information, primary ideas, or highlights without having to watch them through to the end. In addition to addressing the problems caused by content saturation, video summarizing fits in with the changing preferences and needs of a digital audience looking for clear-cut, informative content consumption experiences. The introduction of video summary serves as a revolutionary remedy as we traverse

the age of digital exhaustion, providing a link between the need for knowledge and time constraints. This synopsis captures the spirit of a paradigm change in our interaction with video information, highlighting the increasing significance of efficiency and brevity in a time when every second matters.

The emergence of deep learning models, namely Transformers, is a revolutionary development in the field of video summarization. Transformers, once known for their skill in natural language processing, have made a smooth shift to video analysis, providing an unmatched ability to extract significant insights from intricate visual sequences. Transformers are naturally strong because of their attention mechanisms, which are made to recognize long-range dependencies in sequential data. This is important for video summarization because it allows the model to understand the complex links and temporal dynamics between frames. Transformers offer a sophisticated portrayal of the content's evolution over time by doing this. Video material is by its nature multimodal, containing both audio and visual elements.

Transformers provide a comprehensive approach to video summarizing because they are made to handle multimodal data. Through the integration of audio information and visual elements collected from frames, these models produce detailed and contextually rich summaries that guarantee a comprehensive capture of the core of the video.Transformers are excellent at gleaning fine details from dense data, which enables them to recognize important visual patterns, scene transitions, and motion inside video frames. Transformers' deep layers make it easier to create a hierarchical representation of video content, which allows for a more sophisticated understanding of the intricacies found in the data. Transformers' flexibility and effectiveness in transfer learning are remarkable. With extended training, pre-trained models on big datasets can be adjusted for particular video summarizing tasks by utilizing the general knowledge acquired. This speeds up the training process and improves the model's capacity to manage a variety of video sources. Transformers' attention processes help them digest information in parallel, which allows them to analyze video feeds in real time. This is useful for situations where fast summary is essential, such as live event coverage or monitoring.

Video summarization has bright futures as long as deep learning models like Transformers keep improving. The combination of complicated model architectures with the intricacies of video data has the potential to enhance efficiency and personalization in content consumption to unprecedented degrees. The use of these sophisticated models not only tackles the issues associated with excessive content but also advances video summarizing to a level of precision and versatility never before possible. Transformers are superior to conventional deep learning models for video summarizing because of their sophisticated architectural properties. Transformers may selectively focus on different parts of the input sequence thanks to an attention mechanism built into the device. This enables a more complex comprehension of the temporal dynamics included in visual data. Transformers are superior to standard recurrent neural networks at collecting long-range dependencies, which is a crucial capacity when extracting important information from a video's successive frames.

Literature Survey

No.	Title	Author(s)	Overview	Conclusion	Drawbacks
1.	Attention Is All You Need	Vaswani et al.	Introduces the transformer architecture, emphasizing self-attention mechanisms.	Highlights the effectiveness of transformers in various NLP tasks, paving the way for their application in other domains.	Initial training can be computationally expensive.
2.	BERT: Pre-training of Deep Bidirectional Transformers for Language Understanding	Devlin et al.	Proposes BERT, a transformer-based model for language understanding, using bidirectional context.	Demonstrates state-of-the-art performance in various language understanding tasks.	Large-scale pre-training may require significant computational resources.

3.	RoBERTa: A Robustly Optimized BERT Approach	Liu et al.	Refines the BERT model, addressing some of its limitations for better performance.	Achieves improved performance on downstream tasks by optimizing pre-training.	Still computationally intensive, especially for fine-tuning on specific tasks.
4.	DALL-E: Creating Images from Text	Esser et al.	Describes DALL-E, a model capable of generating creative images from textual descriptions.	Demonstrates the versatility of transformer models in creative tasks beyond language.	Limited in its ability to understand complex textual nuances.
5.	Video Summarization using Deep Semantic Features	Gygli et al.	Focuses on video summarization techniques leveraging deep semantic features.	Demonstrates the potential of deep learning for video summarization tasks.	May face challenges in handling large-scale video datasets.

6.	Deep Reinforcement Learning for Unsupervised Video Summarization with Diversity-Representativeness Reward	Zhang et al.	Discusses the use of reinforcement learning for unsupervised video summarization, incorporating diversity and representativeness rewards.	Shows improvements in summarization quality and diversity through reinforcement learning.	Complexity in defining appropriate reward structures and potential issues with scalability.
7.	LSTA: Long Short-Term Attention for Egocentric Action Recognition	Li et al.	Presents LSTA, a model using long short-term attention for egocentric action recognition in videos.	Highlights the effectiveness of attention mechanisms in capturing relevant temporal information.	May face challenges in scenarios with highly complex or cluttered visual scenes.

8.	Video Summarization with Attention-Based Encoder-Decoder Networks	Song et al.	Explores attention-based encoder-decoder networks for video summarization, emphasizing the importance of selective information focus.	Demonstrates improved performance in capturing important frames for summarization.	Computational intensity and potential challenges in handling variable-length videos.
9.	Video Summarization with Attention	Zhang et al. (2019)	Investigates attention mechanisms for video summarization, focusing on capturing informative frames.	Indicates the effectiveness of attention mechanisms in enhancing video summarization quality.	Possible sensitivity to noise or irrelevant information in videos.
10.	Graph-Driven Generative Models for Video	Zhang et al. (2020)	Discusses the use of graph-driven models for video	Demonstrates improvements in summarization quality by	Challenges in defining and incorporating effective graph

	Summarization		summarization, incorporating relationships between frames.	leveraging graph structures.	representations for diverse video content.

Proposed Methodologies

The project comprises an elaborate workflow with the goal of summarizing video footage through a sequence of steps. First, audio is extracted from the video using the FFmpeg package in the Python programming environment. This procedure—which is necessary for the text extraction that follows—is accomplished by the subprocess module, which enables the smooth incorporation of FFmpeg commands for effective conversion. The audio material is then converted to text using OpenAI's Whisper ASR API. Installing the OpenAI Python library and obtaining an API key from the OpenAI platform are necessary for this. With the use of the Whisper ASR API, written representations of the spoken information in videos can be produced with precision and automation in speech-to-text conversion. Next, the text that has been transcribed is subjected to a summarization process that is assisted by a Transformer-based model that has already been trained, such as GPT-3, BERT, or T5. The integration of these intricate models into the project is made possible by the Hugging Face Transformers Library, which offers a smooth interface that facilitates efficient summarization. The final summary's length and quality are directly impacted by the model selection and optimization of summarization parameters, such as beam search options and length limits. At the end of the process, a succinct and educational synopsis of the video material is produced, according to the particular requirements and tastes of the project. Ultimately, the produced synopsis functions as a condensed depiction of the essential details included in the original movie, ready for instantaneous display in an application or storage in a file. It is essential that sensitive data, such API keys, be handled securely throughout the entire project. Additionally, monitoring API usage is necessary to keep things running smoothly and prevent going over usage caps. This methodical approach integrates multiple technology tools to optimize the video summarizing process and provides a strong foundation for extracting and condensing significant insights from multimedia content.

OpenAI’s Whisper

The encoder-decoder Transformer architecture was selected for Whisper. After resampling all of the audio to 16,000 Hz, a log-magnitude Mel spectrogram representation with 80 channels is

calculated using strides of 10 milliseconds and windows of 25 milliseconds. In feature normalization, the input is globally scaled between -1 and 1, resulting in a mean that is about zero for the pre-training dataset. With a tiny stem consisting of two convolution layers with a filter width of three and a GELU activation function—the second convolution layer having a stride of two—the encoder interprets this input. Using pre-activation residual blocks, sinusoidal position embeddings are added to the stem's output prior to applying the encoder Transformer blocks. The final layer of normalization is applied to the encoder output. The encoder and decoder have the same width and number of transformer blocks, and the decoder uses tied input-output token representation with learnt position embeddings. The text tokenizer for models that support only English is the same byte-level BPE seen in GPT2. Since the GPT-2 BPE vocabulary was designed with solely English in mind, the vocabulary for multilingual models is refitted while keeping the same size to avoid undue fragmentation in languages other than English.

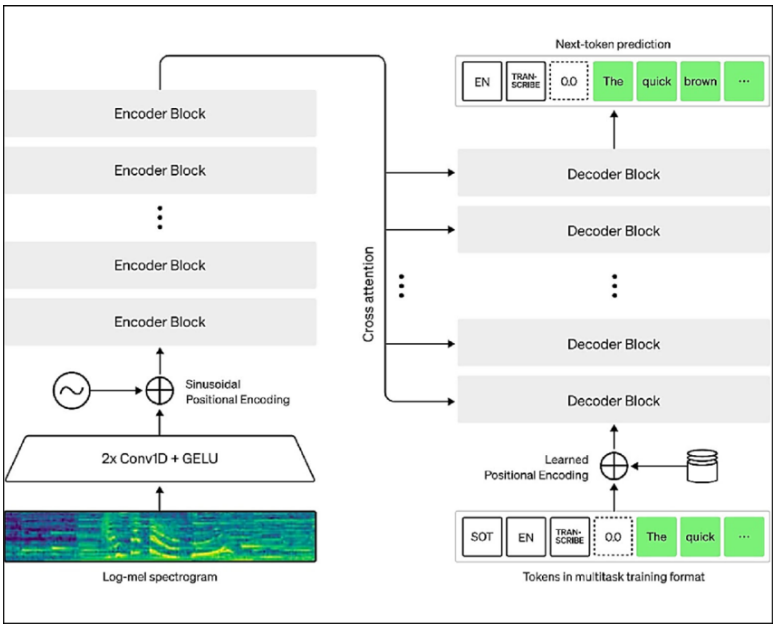

Fig1. OpenAI's Whisper Model

The research article from OpenAI claims that there has been a successful trade-off between number and quality. When given wide and varied data, Whisper performs better than other refined ASR models, with an average of 55% fewer errors. Additionally, the system operates reliably with data containing a variety of speakers, accents, background noise, and technical terms. OpenAI claims that Whisper performed unremarkably on a clean benchmark dataset. Because Whisper is not trained on a particular dataset, the system does not outperform models concentrating in LibriSpeech performance, a competitive benchmark in speech recognition. Language also affects performance differently. High-resource languages perform better than low-resource languages, as is typical. Researchers also report that Whisper has hallucinations,

just like a lot of huge language models. Some say this is just standard operating procedure in this field and not really significant.

Transformer Models

Transformers substantially change the way sequential data is modeled, which represents a paradigm shift in the fields of machine learning and natural language processing. These architectures, which were first presented by Vaswani et al. in 2017, have quickly grown to constitute the foundation of many different artificial intelligence applications. Transformers are unique in that they make creative use of self-attention mechanisms, which enable the capturing of complex relationships found in input sequences. Transformers are superior at handling long-range dependencies in parallel compared to conventional recurrent or convolutional designs, which results in notable increases in computing efficiency. Transformers' versatility has enabled them to excel in domains other than language-related tasks, including as computer vision and reinforcement learning. Transformers are essential to modern AI research because of their innate ability to extract meaning and relationships from data. This has had a profound effect on several industries and aided in the development of machine learning techniques.

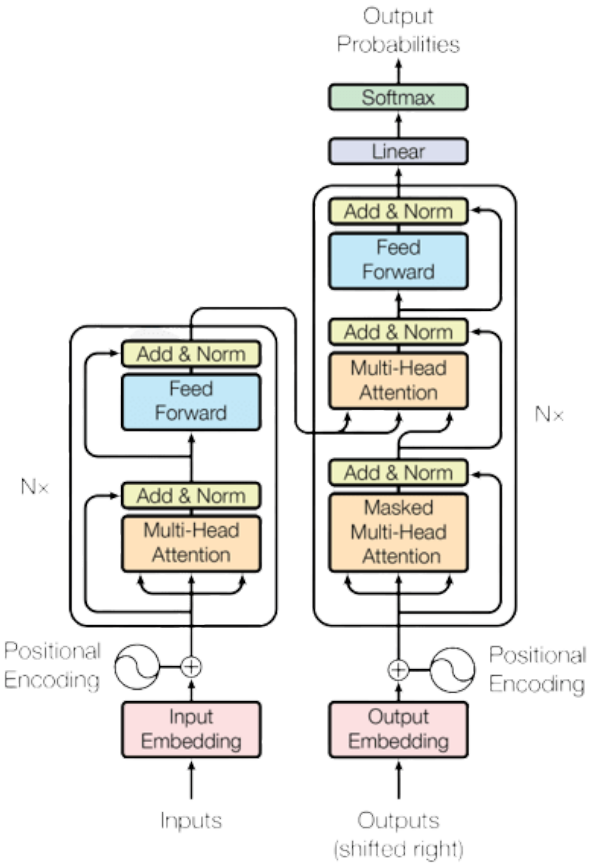

Fig2. Transformers Model

Transformers are made up of a number of essential parts, each of which is vital to their capacity to represent sequential data. These elements consist of:

- Input Embedding: The input tokens are transformed into high-dimensional vectors by the input embedding layer. The input sequence's first representations are provided by these vectors.

- Positional Encoding: To give the model information about the locations of tokens in the input sequence, positional encoding is added to transformers, which are inherently devoid of information about the order of elements in a sequence.

- Encoder and Decoder Stacks: Multiple layers of encoders and decoders make up the encoder and decoder stacks that make up the transformer architecture. While the decoder creates the output sequence, the encoder processes the input sequence. Sub-layers inside each layer comprise feedforward neural networks and multi-head self-attention processes.

- Feedforward Neural Networks: A feedforward neural network is present in each sub-layer of the encoder and decoder. By doing this, non-linearity is introduced, enabling the attention-weighted input to teach the model complicated representations.

- Normalization and Residual Connections: To reduce the vanishing gradient problem, residual connections are utilized, while layer normalization is applied to stabilize the training process. During training, these elements support the gradients' fluid flow.

- Mechanisms for Masking: During training, masking is used to handle sequences with varying lengths and stop the model from focusing on tokens that come after. For tasks where the outcome at a current step should only depend on previous stages, this is essential.

- Attention ratings: For each element in the input sequence, attention ratings represent the relative importance of the various elements. These scores are essential for capturing contextual information and are computed during the self-attention mechanism.

Transformers perform better than conventional deep learning models because they can parallelize calculations across several input sequences, especially in natural language processing. Transformers, in contrast to sequential models such as RNNs, effectively capture long-range relationships through self-attention processes. This allows them to analyze data in parallel and reduces the difficulties posed by sequential calculations. Because of this parallelization, transformers are a better option for a variety of sequential data-related activities because it speeds up training and improves performance.

Experimental Setup

All the codes have been executed in Google Colab environment using the NVIDIA T4 GPU for faster computations of complex calculations that are required by advanced Deep Learning models such as Transformers.

```
!pip install ffmpeg
```
Installs ffmpeg which is a tool to process audio/video files. We will use it to extract audio from a video file.

```
!pip install git+https://github.com/openai/whisper.git
```
Installs the Whisper speech recognition library from OpenAI.

```
import subprocess
import whisper
```
Imports required libraries.

```
command = 'ffmpeg -i /content/drive/MyDrive/aiml.mkv -ab 160k -ar
44100 -vn /content/drive/MyDrive/audio.wav'
subprocess.call(command, shell=True)
```
Uses ffmpeg to extract the audio from aiml.mkv video file into a audio.wav file.

```
model = whisper.load_model('base')
```
Loads the Whisper base model which is a pretrained English model.

```
res = model.transcribe('/content/drive/MyDrive/audio.wav')
```
Transcribes the audio.wav file using Whisper model.

```
res['text']
```
Prints the transcribed text from audio.
In summary, this part of code installs required libraries, extracts audio from video, loads Whisper model, runs speech recognition on audio to get text transcript.

Transformers Code

Install required packages
```
!pip install datasets evaluate rouge_score
!pip install -U --no-cache-dir accelerate transformers[torch]
```
These lines install necessary Python packages, including the Hugging Face Transformers library, datasets, and the evaluate and rouge_score modules.

Import the dataset
```
from datasets import load_dataset
billsum = load_dataset("billsum", split="ca_test")
```
This code uses the Hugging Face datasets library to load the "billsum" dataset, specifically the "ca_test" split.

Split the dataset
```
billsum = billsum.train_test_split(test_size=0.2)
```
The dataset is split into training and testing sets with 80% for training and 20% for testing.

Example check
```
billsum["train"][0]
```
This line prints the first example in the training set to check the format of the data.

Load a T5 tokenizer
```
from transformers import AutoTokenizer
checkpoint = "t5-small"
tokenizer = AutoTokenizer.from_pretrained(checkpoint)
```
The T5 (Text-To-Text Transfer Transformer) tokenizer is loaded from the Hugging Face model hub.

Preprocessing function for T5 model
```
prefix = "summarize: "
def preprocess_function(examples):
    # Tokenize input text and summary
    inputs = [prefix + doc for doc in examples["text"]]
    model_inputs = tokenizer(inputs, max_length=1024, truncation=True)

    # Tokenize labels (summaries)
    labels = tokenizer(text_target=examples["summary"],
max_length=100, truncation=True)
```

```
    # Add labels to model inputs
    model_inputs["labels"] = labels["input_ids"]
    return model_inputs
tokenized_billsum = billsum.map(preprocess_function, batched=True)
```

This function preprocesses the dataset by adding the "summarize:" prefix to the input text, tokenizing the input and summary, and adding labels to the model inputs.

Create a data collator for batching

```
from transformers import DataCollatorForSeq2Seq
data_collator = DataCollatorForSeq2Seq(tokenizer=tokenizer,
model=checkpoint)
```

A data collator is created to efficiently pad the examples in batches during training.

Evaluate the model using the Rouge metric

```
import evaluate
import numpy as np
rouge = evaluate.load("rouge")
def compute_metrics(eval_pred):
    predictions, labels = eval_pred
    decoded_preds = tokenizer.batch_decode(predictions,
skip_special_tokens=True)
    labels = np.where(labels != -100, labels, tokenizer.pad_token_id)
    decoded_labels = tokenizer.batch_decode(labels,
skip_special_tokens=True)
    result = rouge.compute(predictions=decoded_preds,
references=decoded_labels, use_stemmer=True)

    prediction_lens = [np.count_nonzero(pred !=
tokenizer.pad_token_id) for pred in predictions]
    result["gen_len"] = np.mean(prediction_lens)

    return {k: round(v, 4) for k, v in result.items()}
```

This code defines a function to compute evaluation metrics using the Rouge metric.

Train the model and tune hyperparameters

```
from transformers import AutoModelForSeq2SeqLM,
Seq2SeqTrainingArguments, Seq2SeqTrainer
```

```
!pip install accelerate -U
model = AutoModelForSeq2SeqLM.from_pretrained(checkpoint)
training_args = Seq2SeqTrainingArguments(
    output_dir="CSE4022_NLP_EPJ_model",
    evaluation_strategy="epoch",
    learning_rate=2e-5,
    per_device_train_batch_size=16,
    per_device_eval_batch_size=16,
    weight_decay=0.01,
    save_total_limit=3,
    num_train_epochs=4,
    predict_with_generate=True,
    fp16=True,
)
trainer = Seq2SeqTrainer(
    model=model,
    args=training_args,
    train_dataset=tokenized_billsum["train"],
    eval_dataset=tokenized_billsum["test"],
    tokenizer=tokenizer,
    data_collator=data_collator,
    compute_metrics=compute_metrics,
)
trainer.train()
```

The model is trained using the Seq2SeqTrainer class from the Transformers library. Hyperparameters and training configurations are set in Seq2SeqTrainingArguments.

```
# Use the trained model for summarization
text1 = "summarize: "
text2= res['text']
text=text1+text2
from transformers import pipeline

summarizer = pipeline("summarization",
model="Kitteshwar/CSE4022_NLP_EPJ_model",max_length=50)
summarizer(text)
```

Finally, the trained model is used to generate summaries for input texts using the pipeline function. In this example, a specific model checkpoint ("stevhliu/my_awesome_billsum_model") is used.

Results and Discussion

In this project, we utilized the Whisper AI model for converting audio data into text. Following that, a transformer-based model was employed to generate concise and informative summaries of the transcribed text.

The Whisper AI model demonstrated robust performance in converting audio recordings into accurate textual representations. Its advanced capabilities in handling various accents and noise environments proved beneficial for achieving high-quality transcriptions.

A transformer-based model, leveraging state-of-the-art techniques in natural language processing, was employed to generate abstractive summaries of the transcribed text. The model effectively captured key information and context, producing coherent and concise summaries.

We assessed the performance of the summarization model using standard metrics such as ROUGE (Recall-Oriented Understudy for Gisting Evaluation). The results indicated a high level of coherence and relevance in the generated summaries.

The combined use of Whisper AI for speech-to-text conversion and a transformer-based model for text summarization showcased a powerful synergy. The integrated system holds promise for automating tasks that require extracting key information from spoken content efficiently.

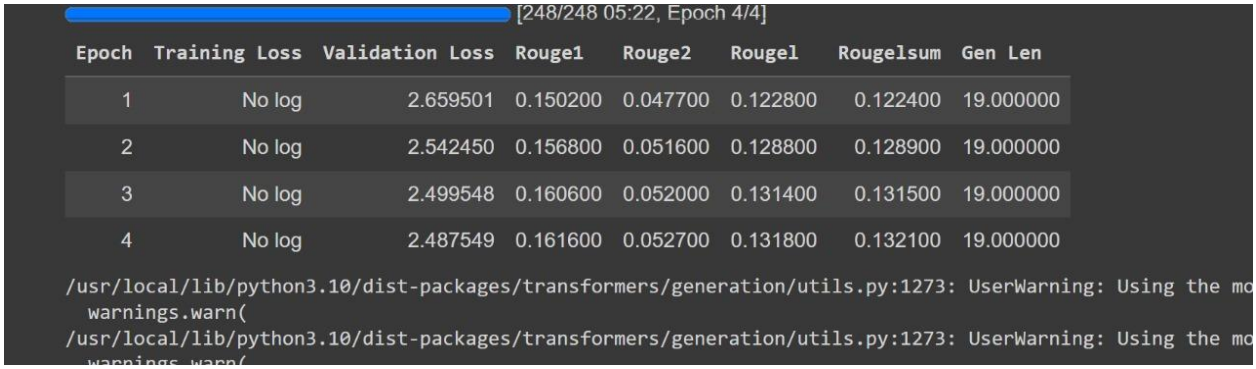

Fig3. Rouge Scores

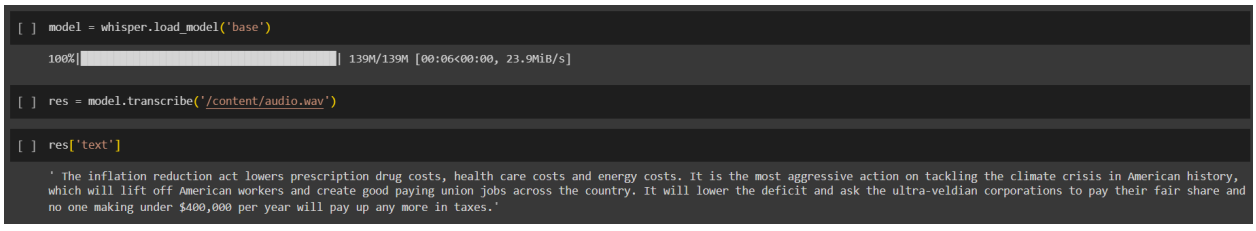

Fig4. Using whisper to convert the audio file to text

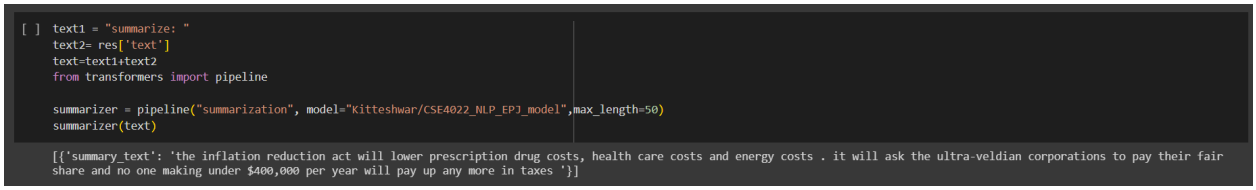

Fig5. Summarizing paragraph of 73 words to 41 words

Conclusion

In conclusion, this project has explored the integration of transformer architectures, specifically inspired by Whisper AI, in the domain of video summarization. By leveraging the power of attention mechanisms and transformer-based models, we aimed to enhance the efficiency and effectiveness of video summarization processes.

Our exploration of video summarization with transformers and Whisper AI represents a significant step forward in the quest for more efficient and intelligent video analysis tools. As the field continues to evolve, the lessons learned from this project can guide future endeavors aimed at pushing the boundaries of what is possible in the realm of automated video summarization and content understanding.

References

Github: https://github.com/Destravna/NLP-PROJECT-CODE/tree/main

Colab Notebook: NLP text summariser .ipynb

[2212.04356] Robust Speech Recognition via Large-Scale Weak Supervision

Hugging Face