**Project Report**

**On**

# Clip Master: Automated Video Highlights Using AI

*Submitted*
*In partial fulfillment*
*For the award of the Degree of*

**PG-Diploma in Artificial Intelligence (PG-DAI)**

**C-DAC, ACTS (Pune)**

| **Guided By:** | **Submitted By:** | |
|---|---|---|
| *Mr Prakash Sinha* | *Ameya Bhawsar* | *240340128003* |
| | *Shrinivas Jawade* | *240340128010* |
| | *Kunal Kurve* | *240340128012* |
| | *Manasi Malge* | *240340128013* |
| | *Pranav Gaddi* | *240340128019* |

**Centre for Development of Advanced Computing(C-DAC), ACTS (Pune- 411008)**

# Acknowledgment

This is to acknowledge our indebtedness to our Project Guide, **Mr Prakash Sinha.** C-DAC ACTS, Pune for his constant guidance and helpful suggestions for preparing this project Clip Master: Automated Video Highlights Using AI**.** We express our deep gratitude towards his inspiration, personal involvement, constructive criticism that he provided us, and technical guidance during this project.

We take this opportunity to thank the Head of the department **Mr. Gaur Sunder** for providing us with such a great infrastructure and environment for our overall development.

We express sincere thanks to **Mrs. Namrata Ailawar**, Process Owner, for their kind cooperation and extendible support towards the completion of our project.

We are pleased to express sincere and deep gratitude towards **Mrs. Risha P R (Program Head)** and **Mrs. Srujana Bhamidi** (Course Coordinator, PG-DAI) for their valuable guidance and constant support throughout this work and help to pursue additional studies.

Also, our warm thanks to **C-DAC ACTS Pune**, which provided us with this opportunity to carry out this prestigious Project and enhance our learning in various technical fields.

| | |
|---|---|
| *Ameya Bhawsar* | *240340128003* |
| *Shrinivas Jawade* | *240340128010* |
| *Kunal Kurve* | *240340128012* |
| *Manasi Malge* | *240340128013* |
| *Pranav Gaddi* | *240340128019* |

# ABSTRACT

This report presents a comprehensive solution for transcribing and summarizing video content using a combination of state-of-the-art deep learning models and natural language processing (NLP) techniques. The solution is implemented in Python and integrated into a Streamlit web application, enabling users to provide either a YouTube URL or a local video file for processing.

The system utilizes yt-dlp to download audio from online videos and ffmpeg to extract audio from local video files. The extracted audio is then transcribed using OpenAI's Whisper model, which supports transcription tasks with GPU acceleration. The transcribed text is processed to generate summaries using the BART (Bidirectional and Auto-Regressive Transformers) model, which is fine-tuned for summarization tasks.

The Streamlit interface allows users to interact with the tool seamlessly, displaying the combined summary and key points of the transcribed content. This tool is designed to help users quickly distill valuable information from lengthy video materials, making it particularly useful for educational, research, and content creation purposes.

In conclusion, this report documents the design and implementation of an efficient, user-friendly system for converting video content into summarized text, highlighting its practical applications in various domains.

# Table of Contents

| Sr. No. | Title | Page No. |
|---|---|---|
| | **Front Page** | **1** |
| | **Acknowledgment** | **2** |
| | **Abstract** | **3** |
| | **Table of Contents** | **4** |
| **1** | **Introduction** | **5** |
| 1.1 | Problem Statement | 6 |
| 1.2 | Objective and Specifications | 7 |
| **2** | **Literature Review** | **9** |
| **3** | **Requirements and Specifications** | **12** |
| 3.1 | Software Requirements | 12 |
| 3.2 | Hardware Requirements | 12 |
| **4** | **Methodology** | **13** |
| 4.1 | Flow Chart | 13 |
| 4.2 | Explanation | 14 |
| **5** | **Proposed System** | **15** |
| 5.1 | System Architecture | 15 |
| 5.2 | Whisper Model Architecture | 16 |
| 5.3 | BART Model Architecture | 17 |
| **6** | **Project Implementation** | **18** |
| 6.1 | Audio Extraction Module | 18 |
| 6.2 | Speech-to-Text Module | 18 |
| 6.3 | Text Summarization Module | 18 |
| 6.4 | User Interface Module | 19 |
| **7** | **Output** | **20** |
| **8** | **Conclusion** | **23** |
| **9** | **Future Scope** | **24** |
| **10** | **References** | **25** |

# 1. INTRODUCTION

In today’s digital age, video content has become one of the most prevalent forms of media, with vast amounts of information being disseminated through platforms like YouTube, educational websites, and online courses. However, the sheer volume of content can often make it challenging for users to extract valuable information efficiently. The need for tools that can automatically summarize lengthy video materials has become increasingly apparent, especially in domains such as education, research, and content creation.

## 1.1 The Project

### 1.1.1 Clip Master: Automated Video Highlights Using AI:

The project addresses this need by offering a comprehensive solution for transcribing and summarizing video content. This project leverages cutting-edge deep learning models and natural language processing (NLP) techniques to create a system that can effectively distill lengthy videos into concise, informative summaries. By doing so, it empowers users to quickly grasp the key points of a video without having to watch it in its entirety.

### 1.1.2 System:

The system is implemented in Python and seamlessly integrated into a user-friendly Streamlit web application. It allows users to input either a YouTube URL or a local video file, which is then processed to extract audio content. The audio is transcribed using OpenAI's Whisper model, a powerful tool designed for high-accuracy transcription tasks, even supporting GPU acceleration for enhanced performance. Once transcribed, the text is further processed using the BART (Bidirectional and Auto-Regressive Transformers) model, which has been fine-tuned specifically for summarization tasks.

### 1.1.3 Output:

The result is a streamlined process where users can easily interact with the tool to obtain a summarized version of the video’s content, including key points and essential information. This not only saves time but also enhances the accessibility of information contained within video formats.

This report delves into the technical details, design considerations, and practical applications of the Clip Master system. It documents the implementation of the system, highlighting the integration of advanced AI models and their role in transforming raw video data into valuable, summarized text. Through this project, we aim to demonstrate the significant impact that AI-driven tools can have on improving information retrieval and content consumption in the modern digital landscape.

## 1.2 Problem Statement

As video content continues to proliferate across digital platforms, users are increasingly challenged by the need to efficiently extract relevant information from lengthy videos. Whether in educational settings, research, or content creation, individuals often face the time-consuming task of watching entire videos to identify key points and valuable insights. This process is not only labour-intensive but also impractical, especially when dealing with large volumes of content.

Traditional methods of summarizing videos, such as manually reviewing and annotating, are inefficient and prone to human error. Furthermore, the lack of automated tools capable of handling diverse video formats and generating accurate, concise summaries exacerbates this challenge. As a result, there is a growing demand for a solution that can automatically transcribe and summarize video content, making it easier for users to access and understand the core information without having to engage with the full video.

The **Clip Master: Automated Video Highlights Using AI** project addresses this critical need by providing an AI-driven system that can quickly and accurately transcribe and summarize videos. By leveraging advanced deep learning models and natural language processing techniques, this solution offers a practical and efficient way to distill lengthy video content into concise summaries, thereby enhancing the accessibility and usability of video-based information.

## 1.3 Objective and Specification

### 1.3.1 Objective

The primary objective of the project is to develop an efficient, user-friendly system capable of automatically transcribing and summarizing video content. This system aims to:

1. **Automate the Transcription Process**: Convert spoken content in videos into accurate textual transcriptions using advanced deep learning models.

2. **Generate Concise Summaries**: Employ natural language processing techniques to produce summaries that capture the essential points and key insights from the transcribed text.

3. 3. **Enhance Information Accessibility**: Provide users with a tool that allows them to quickly access and understand the core information in lengthy videos, reducing the time and effort required for content consumption.

4. 4. **Integrate into a User-Friendly Interface**: Develop a Streamlit web application that allows users to interact with the system seamlessly, whether by providing a YouTube URL or a local video file for processing.

5. **Support Multiple Video Formats**: Ensure the system can handle various video formats, whether online or offline, and process them efficiently to extract audio and generate summaries.

### 1.3.2 Specification

To achieve the objectives outlined, the project involves the following key components:

1. **Video Input Handling**:

   **Online Videos**: Utilize yt-dlp to download and extract audio from YouTube videos.

   **Local Videos**: Use ffmpeg to extract audio from video files stored locally on the user's device.

2. **Audio Transcription**:

   **Whisper Model**: Implement OpenAI's Whisper model to transcribe the extracted audio into text. This model supports GPU acceleration for faster and more accurate transcription.

3. **Text Summarization**:

   **BART Model**: Employ the BART (Bidirectional and Auto-Regressive Transformers) model, fine-tuned for summarization tasks, to generate concise summaries from the transcribed text.

4. **User Interface**:

   **Streamlit Web Application**: Develop a Streamlit-based web interface that allows users to easily interact with the system. The interface should support input of both YouTube URLs and local video files, and display the generated summaries and key points.

5. **Performance Optimization**:

   **GPU Acceleration**: Ensure that the system is optimized for performance, particularly by leveraging GPU acceleration during the transcription process to handle large video files efficiently.

6. **Application of Results**:

   **Summarization Output**: Provide users with a summarized version of the video content, including the most critical points and insights, in a format that is easy to read and understand.

# 2. Literature Survey

## 1. Introduction to Video Summarization and Transcription

The rapid growth of video content on digital platforms has driven significant research and development in the fields of video summarization and transcription. These areas are critical for enhancing the accessibility and usability of video content across various domains, including education, entertainment, and business. Video summarization aims to distill long-form videos into concise representations that capture the essence of the content, while transcription involves converting spoken dialogue in videos into text. Both tasks are essential for improving information retrieval and enabling users to quickly extract valuable insights from video materials.

## 2. Traditional Approaches to Video Summarization

Early approaches to video summarization relied heavily on visual analysis, focusing on keyframes, scene changes, and shot boundaries to create condensed versions of videos. These techniques often involved low-level image processing and heuristics to identify significant segments of a video. For instance, efforts by Hanjalic and Zhang (1999) introduced an algorithm that uses color histograms to detect scene boundaries and select representative frames, which were then assembled into a video summary. While effective in certain scenarios, these methods were limited by their reliance on visual cues, which often failed to capture the semantic content of the video.

Subsequent advancements introduced the concept of video abstraction, which combined both keyframe extraction and dynamic content analysis to produce more meaningful summaries. However, these methods still faced challenges in accurately representing the narrative structure and thematic elements of the video, leading to the development of more sophisticated techniques incorporating audio and textual information.

## 3. Emergence of Deep Learning in Video Processing Summarization

The advent of deep learning has revolutionized video summarization and transcription, enabling the development of models that can understand and interpret complex patterns in multimedia data. Convolutional Neural Networks (CNNs) and Recurrent Neural Networks (RNNs) have been particularly influential in this regard. CNNs have been employed to extract spatial features from video frames, while RNNs, particularly Long Short-Term Memory (LSTM) networks, have been used to model temporal dependencies across video sequences.

For video summarization, deep learning models such as the Deep Summarization Network (DSN) have shown promise by leveraging both visual and textual information to generate summaries that are not only concise but also contextually relevant. These models are trained on large datasets of videos, learning to identify key segments that best represent the overall content. The integration of attention mechanisms further enhances the ability of these models to focus on the most critical aspects of the video, improving the quality of the summaries generated.

## 4. Natural Language Processing in Video Transcription and Summarization

Natural Language Processing (NLP) has played a pivotal role in advancing the state of video transcription and summarization. Early NLP-based approaches to transcription involved the use of Hidden Markov Models (HMMs) and Gaussian Mixture Models (GMMs) for automatic speech recognition (ASR). These models, while ground-breaking at the time, were constrained by their reliance on handcrafted features and limited by the availability of labeled data.

The introduction of deep learning into NLP, particularly through models like Transformer-based architectures, has led to significant improvements in ASR and text summarization. Transformer models, such as BERT (Bidirectional Encoder Representations from Transformers) and BART (Bidirectional and Auto-Regressive Transformers), have demonstrated exceptional performance in a variety of NLP tasks, including language modeling, text classification, and summarization.

BART, in particular, has been fine-tuned for summarization tasks and has achieved state-of-the-art results on several benchmarks. It combines the strengths of both autoregressive and bidirectional transformers, making it highly effective for generating coherent and contextually relevant summaries from transcribed text. This model's ability to understand the semantic content of the text and generate concise summaries aligns perfectly with the goals of the Clip Master project.

## 5. The Role of Whisper Model in Transcription

OpenAI's Whisper model represents a significant advancement in the field of automatic speech recognition. It is designed to transcribe spoken content into text with high accuracy, leveraging large-scale datasets and powerful neural network architectures. The Whisper model is particularly notable for its ability to handle diverse accents, noisy environments, and multiple languages, making it a versatile tool for transcription tasks.

The use of the Whisper model in the Clip Master project addresses many of the limitations of traditional ASR systems. Its support for GPU acceleration further enhances its performance, allowing for faster processing times, which is crucial when dealing with lengthy video content. The integration of Whisper into the system ensures that the transcriptions generated are not only accurate but also produced promptly, facilitating the subsequent summarization process.

## 6. Streamlit: Bridging the Gap Between Users and AI Models

The final piece of the Clip Master project involves the integration of these advanced AI models into a user-friendly interface. Streamlit, an open-source Python library, has emerged as a popular tool for creating interactive web applications that allow users to engage with machine-learning models. Its simplicity and flexibility make it an ideal choice for developers looking to deploy AI-driven tools quickly.

Streamlit's ability to handle real-time inputs and outputs ensures that users can seamlessly interact with the Clip Master system. Whether uploading a local video file or providing a YouTube URL, the interface guides users through the process, displaying the generated transcriptions and summaries in a readable format. This ease of use is critical for broadening the accessibility of the tool, allowing users with varying levels of technical expertise to benefit from the capabilities of the system.

## 7. Conclusion

The convergence of deep learning, natural language processing, and user-friendly interfaces has paved the way for the development of sophisticated tools like Clip Master. By leveraging state-of-the-art models such as Whisper and BART, the project addresses the growing need for efficient video summarization and transcription, making it easier for users to extract valuable insights from video content. The integration of these technologies into a Streamlit application further enhances the system's usability, ensuring that it can be readily adopted across various domains where video content is prevalent.

This literature review highlights the key technological advancements that have contributed to the development of the Clip Master system, providing a foundation for understanding the project's significance and its potential impact on improving video content consumption.

# 3. Requirements and Specifications.

## 3.1 Software Requirements:

**Python Libraries:**

- **Streamlit (streamlit):** For building and running the web-based user interface.
  Installation: pip installs streamlit

- **yt-dlp (yt-dlp):** For downloading audio from YouTube videos.
  Installation: pip install yt-dlp

- **Whisper (whisper):** For transcribing audio to text.
  Installation: pip install git+https://github.com/openai/whisper.git

- **Transformers (transformers):** Provides access to the BART model for text summarization.
  Installation: pip install transformers

- **NLTK (nltk):** For tokenizing text into sentences, used in the text chunking process.
  Installation: pip install nltk

- **Subprocess (built-in):** A built-in Python module for running system commands (used for ffmpeg).
  No installation is needed as it's included with Python.

**Others:**

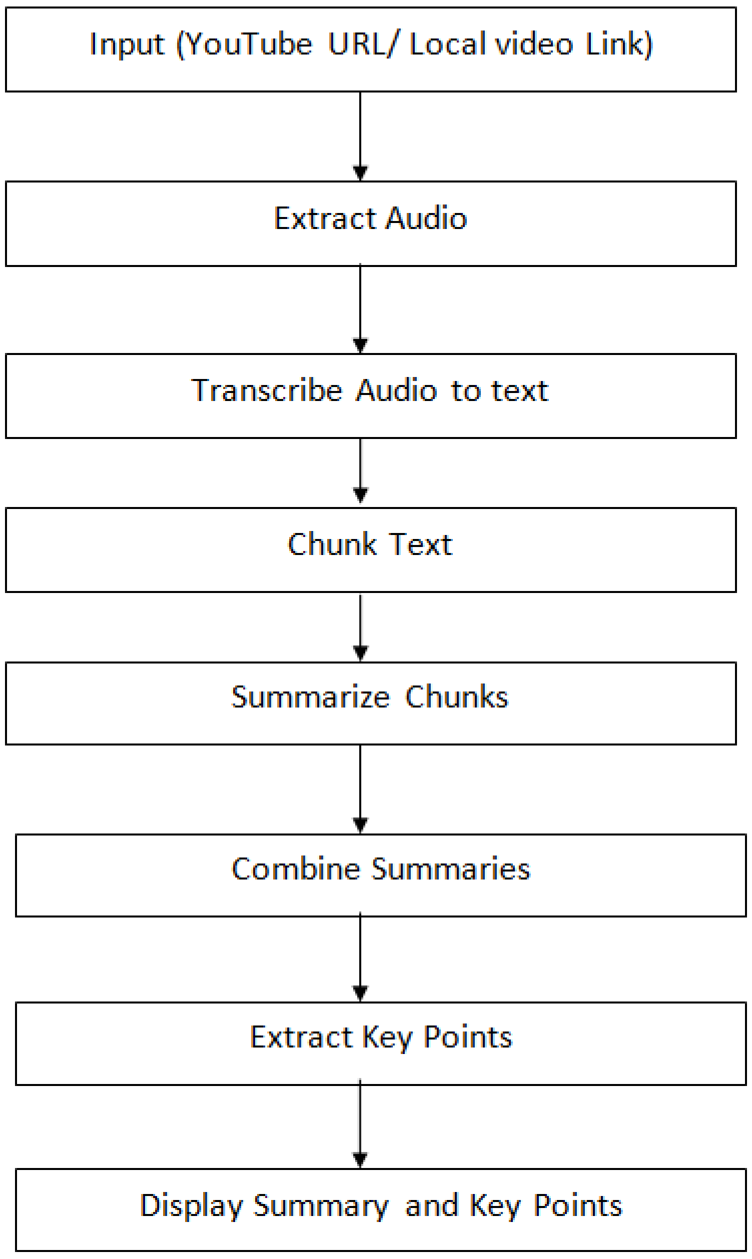
- Jupyter Notebook / Jupyter Lab
- Google Colab

## 3.2 Hardware Requirements:

- **OS:** Windows 11, MAC OS, Ubuntu
- **RAM:** 16 GB
- **Graphic Card:** NVIDIA RTX 3050
- **GPU:** 6 GB and more
- **CPU:** 64-bit operating processing system with 2.40GHz.

# 4. Methodology

## 4.1 Flow Chart



## 4.2 Explanation

1. **Input Handling:**

   The system accepts both YouTube URLs and local video files, ensuring versatility in input sources.

2. **Audio Extraction:**

   Efficient audio extraction is performed using yt_dlp for online videos and ffmpeg for local files.

3. **Speech-to-Text Conversion:**

   The whisper model transcribes the extracted audio to text with high accuracy, handling multiple languages and accents.

4. **Text Chunking and Summarization:**

   The text is chunked and processed by the BART model to produce concise summaries, maintaining the context of the original content.

5. **Key Points Extraction:**

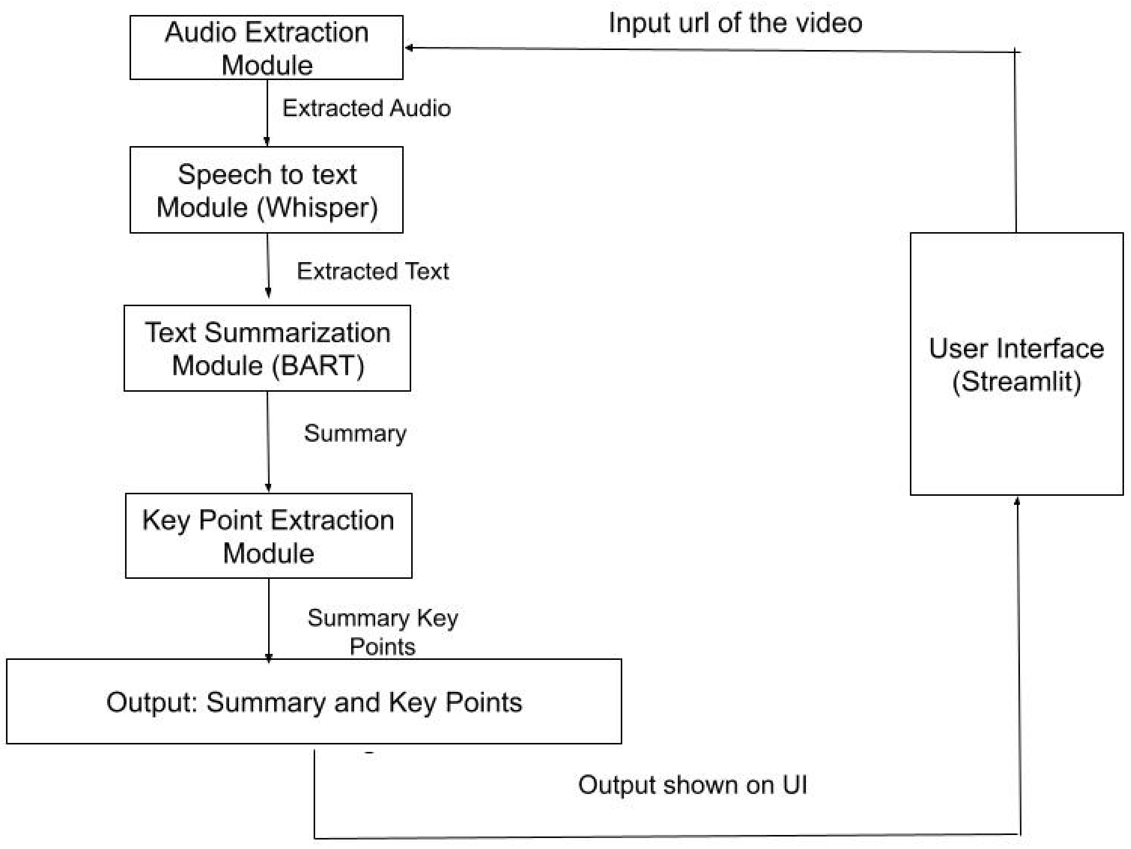
   Critical information is distilled from the summary, highlighting the most important aspects of the content.

6. **Output Delivery:**

   Summaries and key points are presented in an organized manner, enabling quick understanding and review.

# 5. Proposed System

## 5.1 System Architecture

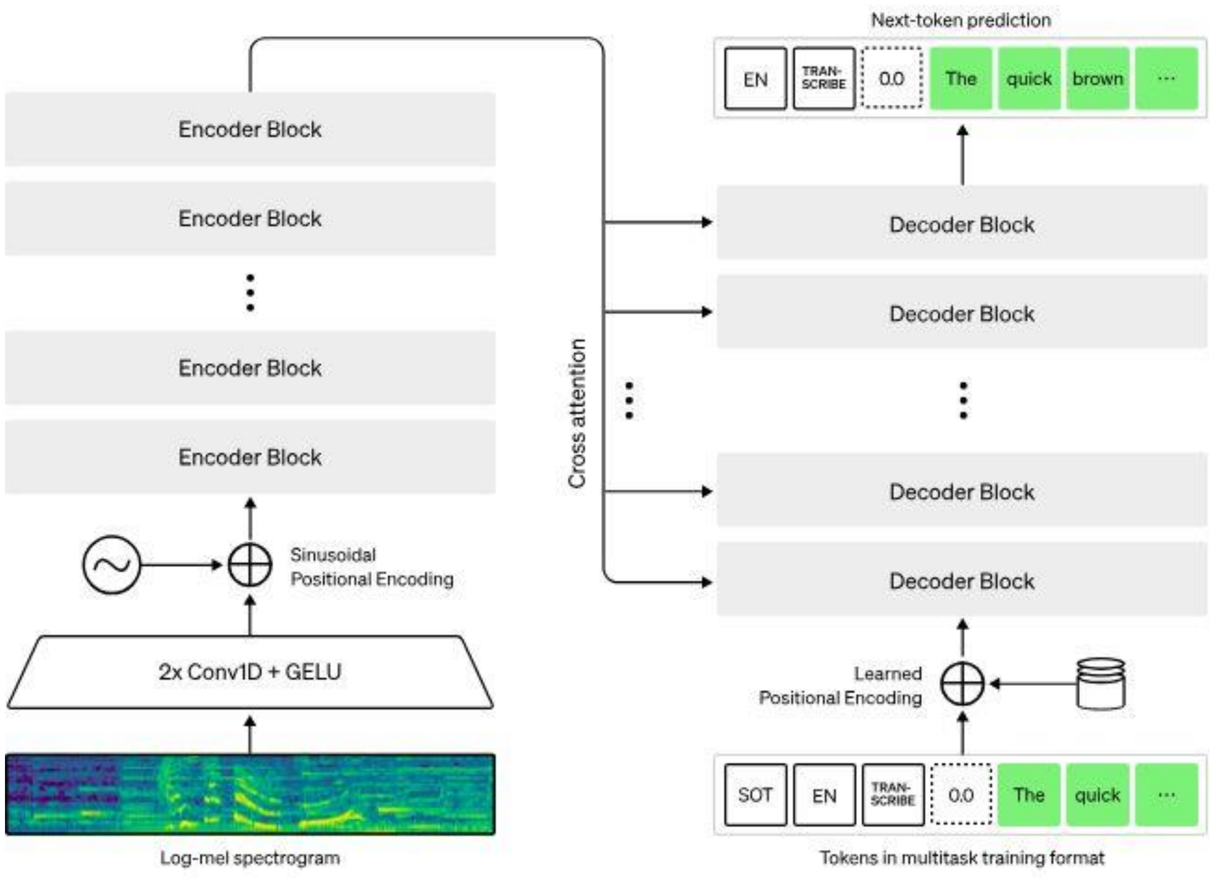

Audio Extraction Module
Input url of the video
Extracted Audio
Speech to text Module (Whisper)
Extracted Text
Text Summarization Module (BART)
User Interface (Streamlit)
Summary
Key Point Extraction Module
Summary Key Points
Output: Summary and Key Points
Output shown on UI

**Workflow:**

First, the user can enter the URL of the YouTube video or provide any local file path present to him. Then that URL will be processed by yt-dlp if the given URL is of YouTube and if it's a local file then it will be processed by ffmpeg respectively.

The audio is extracted and then given to Whisper for transcribing. Then that text will be passed to BART large CNN model for generating a summary and then from that summary key points are extracted and displayed on the UI.

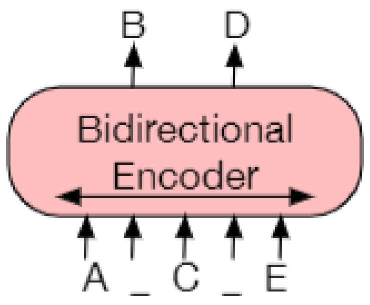
## 5.2 Whisper model Architecture



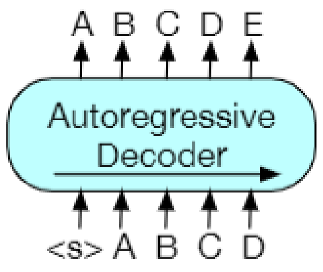
Whisper is an automatic speech recognition (ASR) system trained on 680,000 hours of multilingual and multitask supervised data collected from the web. We show that the use of such a large and diverse dataset leads to improved robustness to accents, background noise, and technical language. Moreover, it enables transcription in multiple languages, as well as translation from those languages into English. We are open-sourcing models and inference code to serve as a foundation for building useful applications and for further research on robust speech processing.

The Whisper architecture is a simple end-to-end approach, implemented as an encoder-decoder Transformer. Input audio is split into 30-second chunks, converted into a log-Mel spectrogram, and then passed into an encoder. A decoder is trained to predict the corresponding text caption, intermixed with special tokens that direct the single model to perform tasks such as language identification, phrase-level timestamps, multilingual speech transcription, and to-English speech translation.

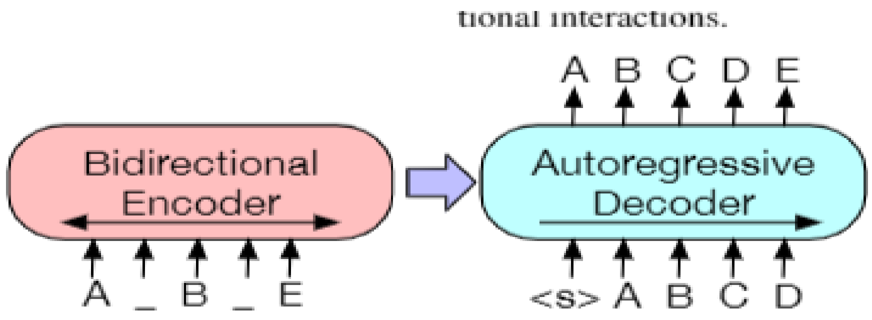
## 5.3 BART Model Architecture



1. BERT model



2. GPT model



3. BART model

1. **BERT:** Random tokens are replaced with masks, and the document is encoded bidirectionally. Missing tokens are predicted independently, so BERT cannot easily be used for generation.
2. **GPT:** Tokens are predicted auto-regressively, meaning GPT can be used for generation. However, words can only condition in a leftward context, so they cannot learn bidirectional interactions
3. **BART:** Inputs to the encoder need not be aligned with decoder outputs, allowing arbitrary noise transformations. Here, a document has been corrupted by replacing spans of text with mask symbols. The corrupted document (left) is encoded with a bidirectional model, and then the likelihood of the original document (right) is calculated with an autoregressive decoder. For fine-tuning, an uncorrupted document is input to both the encoder and decoder and we use representations from the final hidden state of the decoder.

# 6. Project Implementation

## 6.1 Audio Extraction Module

- **Input Sources:**
  The system accepts both online video sources (e.g., YouTube URLs) and local video files.

- **Technology Used:**
  yt_dlp library for downloading audio from online videos.
  ffmpeg for extracting audio tracks from local video files.

- **Process Flow:**
  The system first detects whether the input is a URL or a file path.
  Based on the input type, the appropriate method is called to extract the audio and save it in a suitable format for transcription.

## 6.2 Speech-to-Text Module

- **Model Selection:**
  The Whisper model by OpenAI is chosen for its high accuracy in transcribing spoken language into text, with support for multiple languages and translation.

- **GPU Utilization:**
  The model is deployed on a GPU to enhance the speed and efficiency of the transcription process, making it suitable for handling large audio files.

- **Translation Support:**
  The system is capable of translating the transcribed text into English if the original audio is in a different language, broadening the system's applicability.

## 6.3 Text Summarization Module

- **Model Selection:**
  The BART model (Facebook/bart-large-CNN) is used for its robust performance in text summarization, leveraging its transformer-based architecture.

- **Chunking Mechanism:**
  To manage long transcriptions, the text is divided into smaller chunks based on token limits, ensuring that the model processes the text efficiently.

- **Summary Generation:**
  Each chunk is summarized independently, and the individual summaries are then combined to form a coherent overall summary of the video content.

- **Key Points Extraction:**
  The system extracts key points from the summary by analyzing sentence structures, helping users quickly identify the most critical information.

## 6.4 User Interface Module

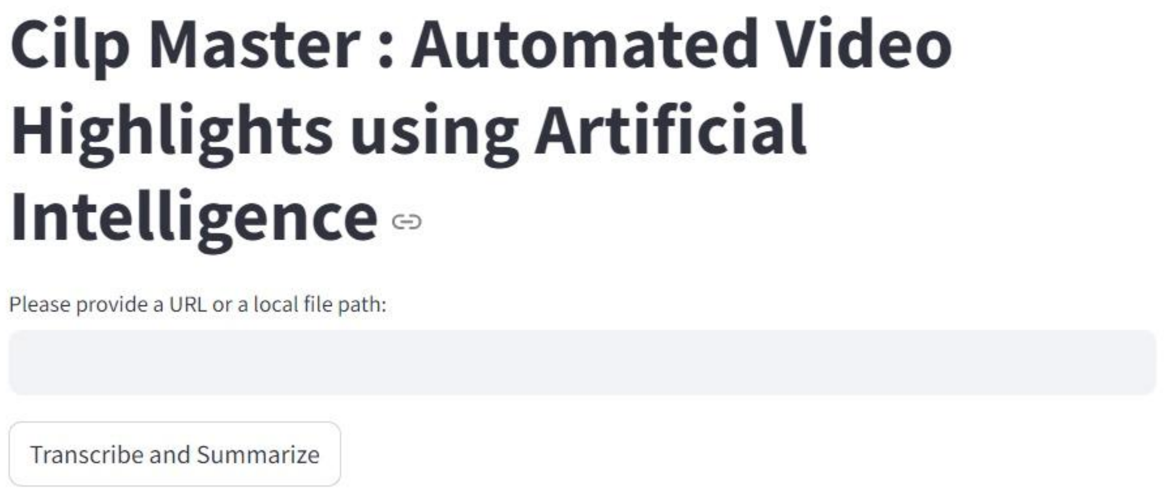
- **Streamlit Integration:**
  The system is wrapped in a user-friendly web interface built with Streamlit, allowing users to interact with the system easily.

- **Input Handling:**
  Users can input a YouTube URL or upload a local video file through the interface.
  A single button click initiates the entire process, from audio extraction to summarization, providing a seamless user experience.

- **Output Display:**
  The summarized text and key points are displayed in an organized manner, enabling users to review the content effectively.

# 7. Output

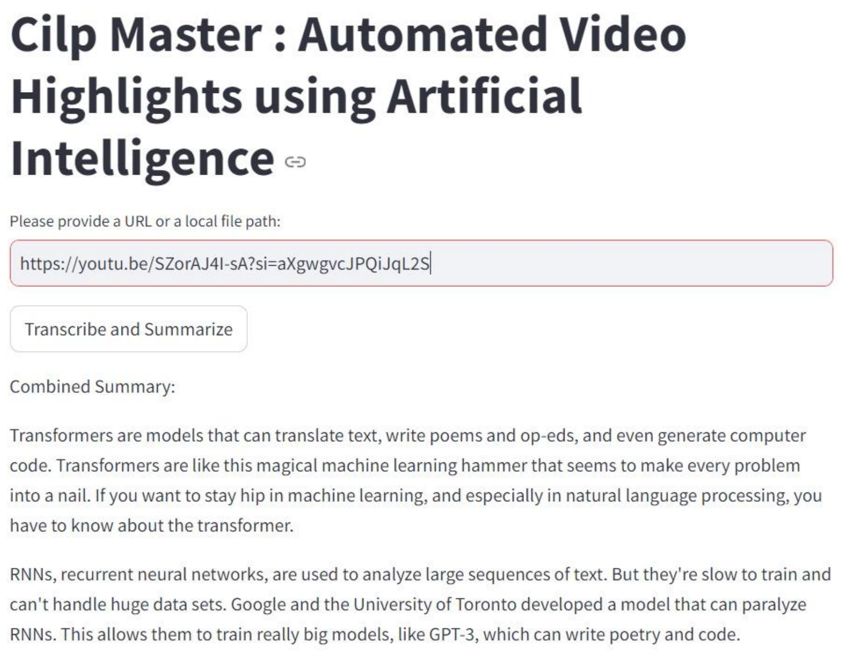
## 7.1 The user can provide a link or path of the video



Cilp Master : Automated Video Highlights using Artificial Intelligence

Please provide a URL or a local file path:

## 7.2 Summary is displayed of the video



Cilp Master : Automated Video Highlights using Artificial Intelligence

Please provide a URL or a local file path:

https://youtu.be/SZorAJ4I-sA?si=aXgwgvcJPQiJqL2S

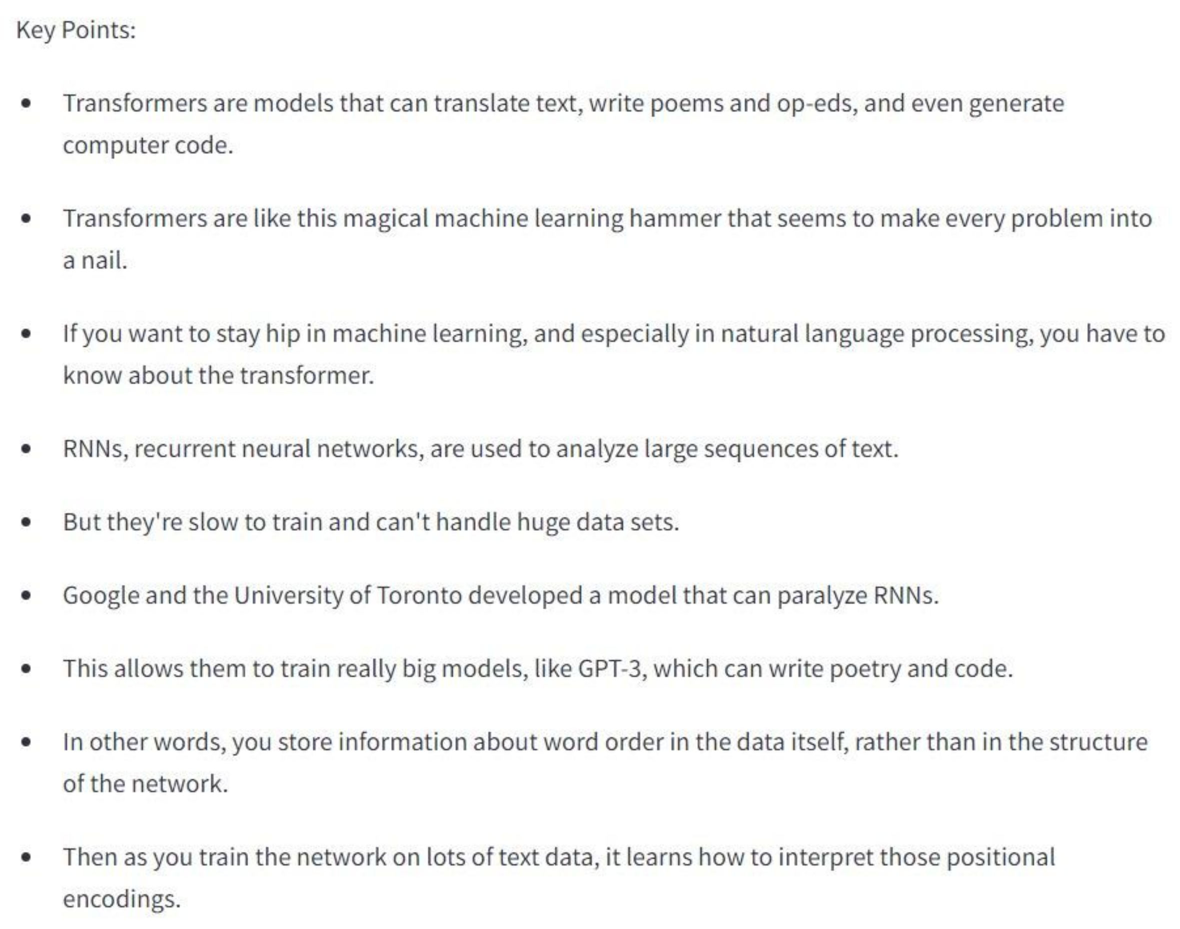
Transcribe and Summarize

Combined Summary:

Transformers are models that can translate text, write poems and op-eds, and even generate computer code. Transformers are like this magical machine learning hammer that seems to make every problem into a nail. If you want to stay hip in machine learning, and especially in natural language processing, you have to know about the transformer.

RNNs, recurrent neural networks, are used to analyze large sequences of text. But they're slow to train and can't handle huge data sets. Google and the University of Toronto developed a model that can paralyze RNNs. This allows them to train really big models, like GPT-3, which can write poetry and code.

## 7.3 Key highlights of the summary are displayed



Key Points:

- Transformers are models that can translate text, write poems and op-eds, and even generate computer code.
- Transformers are like this magical machine learning hammer that seems to make every problem into a nail.
- If you want to stay hip in machine learning, and especially in natural language processing, you have to know about the transformer.
- RNNs, recurrent neural networks, are used to analyze large sequences of text.
- But they're slow to train and can't handle huge data sets.
- Google and the University of Toronto developed a model that can paralyze RNNs.
- This allows them to train really big models, like GPT-3, which can write poetry and code.
- In other words, you store information about word order in the data itself, rather than in the structure of the network.
- Then as you train the network on lots of text data, it learns how to interpret those positional encodings.

# 8. Conclusion

In conclusion, this project successfully demonstrates an efficient and scalable system for video-to-text transcription and summarization by integrating state-of-the-art technologies such as OpenAI's Whisper model and Facebook's BART model. The combination of these tools with Streamlit for an intuitive user interface allows for seamless extraction, transcription, and summarization of audio from video content, addressing the growing demand for accessible and summarized media. The use of ffmpeg for audio extraction, yt-dlp for video downloading, and advanced NLP techniques for summarization ensures that the system is both robust and versatile. This project not only highlights the potential of AI in automating complex tasks but also opens up opportunities for further enhancements, such as real-time processing and multilingual support. Overall, this implementation sets a solid foundation for future developments in media content processing and analysis.

# 9. Future Scope

1. **Real-Time Transcription and Summarization:**
   Implement real-time audio transcription and summarization to enable live processing of video content, which can be valuable for streaming platforms and live broadcasts.

2. **Multilingual Support:**
   Extend the system to support transcription and summarization in multiple languages, making it accessible to a broader global audience and applicable in multilingual environments.

3. **Integration with Content Management Systems (CMS):**
   Develop plugins or APIs to integrate the system with popular content management systems, enabling seamless transcription and summarization directly within content creation platforms like WordPress or YouTube.

4. **User-Customizable Summaries:**
   Allow users to customize the level of detail in summaries, providing options for brief overviews, medium-length summaries, or detailed explanations depending on user needs.

5. **Sentiment Analysis and Key Insights Extraction:**
   Incorporate sentiment analysis and key insights extraction into the summarization process to provide users with a deeper understanding of the content, including emotional tone and important themes.

6. **Mobile Application Development:**
   Develop a mobile application version of the system to allow users to access transcription and summarization features on the go, making the system more accessible and convenient.

7. **Chatbot Integration:**
   Integrate a conversational AI chatbot on top of the system to allow users to interact with and query the transcribed and summarized content more dynamically and interactively.

# 10. References

- Radford, A., Kim, J. W., Xu, T., Brockman, G., McLeavey, C., & Sutskever, I. (2021). *Robust Speech Recognition via Large-Scale Weak Supervision*.
  OpenAI. Available at: https://github.com/openai/whisper

- Lewis, M., Liu, Y., Goyal, N., Ghazvininejad, M., Mohamed, A., Levy, O., Stoyanov, V., & Zettlemoyer, L. (2020). *BART: Denoising Sequence-to-Sequence Pre-training for Natural Language Generation, Translation, and Comprehension*.
  In Proceedings of the 58th Annual Meeting of the Association for Computational Linguistics.
  Available at: https://arxiv.org/abs/1910.13461

- Streamlit Inc. (2022). *Streamlit: The fastest way to build and share data apps*.
  Available at: https://streamlit.io/

- yt-dlp Developers. (2021). *yt-dlp: A youtube-dl fork with additional features and fixes*.
  Available at: https://github.com/yt-dlp/yt-dlp

- FFmpeg Developers. (2022). *FFmpeg: A complete, cross-platform solution to record, convert and stream audio and video*.
  Available at: https://ffmpeg.org/

- **facebook/bart-large-cnn https://huggingface.co/facebook/bart-large-cnn**

- Python Software Foundation. (2023). *Python Language Reference, version 3.10*.
  Available at: https://docs.python.org/3/
  *subprocess* module documentation:
  Available at: https://docs.python.org/3/library/subprocess.html

- Open Ai whisper https://openai.com/index/whisper/