Rudresh Choubey, 2025, 13:4
ISSN (Online): 2348-4098
ISSN (Print): 2395-4752

International Journal of Science, Engineering and Technology

An Open Access Journal

Chrome Extension: Video Summarizer

[1] Rudresh Choubey, [2]Mayank Kumar, [3]Ishita Sanya, [4]Nishi Priyanka Hembrom, [5]Asst.Prof.Kedar Pinniboniya

Department of Computer Science and Engineering, Parul Institute of Engineering and Technology, Vadodara, Gujarat, India.

Abstract- With the exponential growth of video content on platforms like YouTube, accessing key information efficiently has become a significant challenge. This paper proposes the design and implementation of a Chrome Extension integrated with a Flask-based API that automatically extracts YouTube video transcripts and generates concise summaries using Natural Language Processing (NLP). The system leverages AI-based summarization models to provide users with quick insights, thereby improving accessibility and saving time. The proposed framework operates in three stages: 1. Automated transcript retrieval using the YouTube Data API, 2. Preprocessing and re-finement of textual content through NLP pipelines, 3. Generation of abstractive or extractive summaries using transformer-based language models. The summarized outputs are then presented to users via a lightweight Chrome Extension interface, allowing seamless access without leaving the YouTube platform. This approach offers significant advantages for diverse user groups including students, researchers, and professionals who rely on video lectures, tutorials, or informational content for learning and decision making. By reducing information overload and minimizing the time required to grasp core ideas, the tool enhances productivity and accessibility. Experimental evaluations indicate that the generated summaries achieve high levels of relevance and coherence compared to raw transcripts, validating the utility of the system. Ultimately, this work demonstrates how AI-driven summarization can transform video-based learning and information retrieval into a more efficient and user-friendly experience.

Keywords - Video Summarization, Natural Language Pro- cessing, Flask API, Chrome Extension, YouTube API, AI Sum-marizer, Transcript Extraction.

I. INTRODUCTION

YouTube has become one of the largest sources of digital information, with over 500 hours of content uploaded every minute. While this growth provides vast learning opportunities, it also creates challenges for users who need to extract meaningful insights without spending hours watching videos. Traditional methods, such as manually searching or skipping through videos, are inefficient and often result in missing key points. Recent advances in Artificial Intelligence (AI) and Natural Language Processing (NLP) have enabled automated summarization of large text data, including video transcripts. This research proposes a YouTube Video Summarizer Chrome Extension integrated with a Flask-based API that automatically retrieves transcripts using the YouTube API and generates concise summaries. The system provides users with short, medium, or detailed summaries, depending on their needs, allowing faster understanding, improved accessibility, and ef-fective knowledge acquisition.

Problem Statement

Users frequently struggle to consume long-form video con- tent due to time constraints. While videos contain valuable insights, extracting the essence remains tedious. Conventional methods such as note-taking or manual skimming are ineffi- cient. Key problems include: (1) Time wasted watching long videos for limited key points. (2) Skipping through videos leads to missed insights. (3) Non-native speakers and hearing- impaired users face accessibility issues. Therefore, there is a pressing need for an AI-powered summarization system that can automatically generate concise, structured summaries of YouTube video transcripts, directly accessible within the platform's interface. Such a solution would save time, improve accessibility, and enhance the overall learning experience.

Project Objectives

The primary objective of this project is to develop an in- telligent YouTube Video Summarization system that improves accessibility and efficiency in consuming video content. The specific goals include: 1. Designing a Chrome Extension – To create a lightweight and user-friendly browser extension that integrates seamlessly with YouTube's interface, enabling users to generate and view summaries without leaving the platform. 2. Implementing a Flask-based Backend API – To develop a robust backend capable of extracting, processing, and managing video transcripts efficiently. The API serves as the communication bridge between the Chrome Extension, the YouTube API, and the AI summarization models. 3. Utilizing AI/NLP Summarization Models – To apply state-of- the-art Natural Language Processing techniques and AI-based summarization models that can accurately condense lengthy video transcripts into meaningful and coherent summaries. 4. Providing Customizable Summary Lengths – To allow users to select between short, medium, or detailed summaries based on their preference, ensuring flexibility for quick reviews as well as in-depth understanding. 5. Ensuring Multi-language Sup- port – To extend summarization capabilities beyond English by incorporating multi-language transcript handling, thereby enhancing accessibility for a global audience, including non-native speakers and learners from diverse linguistic back- grounds. Collectively, these objectives aim to deliver an ef- ficient, accessible, and intelligent tool that transforms video- based learning into a faster, more inclusive, and productive experience.

Motivation

The rapid growth of Artificial Intelligence (AI) and Natural Language Processing (NLP) technologies has opened new pos- sibilities for automatic summarization, making it both feasible and effective in real-world applications. While existing text summarization tools have demonstrated significant success, most of them focus exclusively on written documents, articles, or reports. However, video-based content—such as lectures, tutorials, presentations, and webinars—remains largely under- explored despite its widespread use in education, professional training, and research. Users frequently struggle to identify and retain the essential points from long-form video content, lead- ing to information overload and inefficiency. Students often spend hours reviewing lectures, professionals must sift through lengthy presentations for decision-making, and researchers need quick access to insights from academic talks. Moreover, accessibility challenges exist for non-native speakers who may find spoken content difficult to follow, and for hearing- impaired users who rely on transcripts. These gaps motivated the development of a dedicated solution: a YouTube Video Summarizer that not only extracts and condenses transcripts but also presents information in a structured, user-friendly format. By leveraging AI powered summarization within a Chrome Extension, the project aims to transform video con- sumption into a more time-efficient, inclusive, and accessible learning experience for a diverse audience.

Technology Stack Overview

Frontend (Chrome Extension UI – React.js, HTML, CSS, JavaScript) The frontend is implemented as a Chrome Exten- sion, built using React.js to enable a modular, component- based architecture and efficient state management. React im- proves performance with its virtual DOM and provides a clean, dynamic, and reusable UI structure. HTML and CSS ensure a responsive design, while JavaScript (with React hooks and context APIs) manages interactivity, user input, and seamless communication with the backend. This allows users to request summaries in different lengths (short, medium, detailed) directly within the YouTube page, offering a modern and smooth user experience.

Backend (Python Flask Server) The backend is devel- oped using the Flask framework, chosen for its simplicity, scalability, and REST API capabilities. The server acts as the processing hub, handling transcript retrieval, pre-processing, and interaction with summarization models. It also ensures efficient communication between the Chrome Extension fron- tend and the AI models.

APIs (YouTube Data API) The system uses the YouTube Data API to fetch video transcripts automatically. This elim- inates the need for manual transcript uploads and ensures real-time extraction of spoken content. For videos without transcripts, fallback methods such as speech-to-text engines can be integrated.

AI Models (Hugging Face Transformers – BERT/GPT- based Summarizers) Advanced Natural Language Processing models, such as BERT and GPT-based summarizers from Hug- ging Face's Transformers library, are employed for generating summaries. These models support abstractive summarization, producing coherent and context-aware summaries instead of simple keyword extraction. This ensures that the generated summaries are both meaningful and accurate.

Integration (JSON-based Communication) Communica- tion between the Chrome Extension frontend (React-based) and the Flask backend is handled using lightweight JSON (JavaScript Object Notation). This ensures fast, structured, and secure data exchange, enabling real-time summary generation and delivery to the user interface.

II. LITERATURE SURVEY

Meeting and conversational summarization has emerged as a complex sub-field of natural language processing, with recent research targeting the challenges posed by long dialogues, noisy transcripts, and multi-speaker dynamics. A number of seminal works provide valuable insights into datasets, mod- eling architectures, and evaluation strategies for abstractive summarization. Four key studies are particularly relevant: the survey by Rennard et al. (2023), the HMNet model by Zhu et al. (2020), the QMSum benchmark by Zhong et al. (2021), and grounded summarization for podcasts by Song et al. (2022). Together, these contributions highlight the progress and remaining challenges in abstractive meeting and dialogue summarization.
Rennard et al. (2023) conducted a comprehensive survey on *abstractive meeting summarization*, distinguishing it from traditional document summarization by emphasizing the unique characteristics of conversational data. Meetings typi- cally involve long transcripts with multiple speakers, disfluen- cies, interruptions, and frequent automatic speech recognition (ASR) errors, which degrade the performance of conventional summarization models. The authors categorize prior research into three broad areas: data resources, modeling approaches, and evaluation techniques. They note the scarcity of large annotated meeting datasets, largely due to privacy concerns, and stress the need for better corpora to advance the field. On the modeling front, the survey outlines the use of hierarchical and dialogue-aware encoder–decoder architectures, as well as more recent applications of large pretrained language models. Evaluation emerges as a recurring challenge, with ROUGE often criticized for its inability to measure factual correct- ness or usefulness of summaries. Rennard et al. argue for developing richer evaluation metrics that consider consistency, accuracy, and summary utility. Their work not only synthesizes existing literature but also identifies future directions, includ- ing multimodal integration, controllable summary generation, and improved measures of faithfulness, thereby laying an important foundation for future research.

Building upon such challenges, Zhu et al. (2020) proposed HMNet (Hierarchical Network for Abstractive Meeting Sum- marization), a model that directly addresses the structural complexity of meeting data. Unlike flat sequence-to-sequence models that treat transcripts as linear text, HMNet introduces a hierarchical encoder–decoder design that captures utterance- level representations and discourse-level context simultane- ously. This dual-level modeling allows it to effectively process lengthy and noisy transcripts, including overlapping speech and interruptions common in meetings. A notable innovation is the use of **cross-domain pretraining*, where HMNet is first trained on large-scale news summarization datasets and then fine-tuned on limited meeting corpora such as AMI and ICSI. This transfer-learning approach significantly boosts perfor- mance, demonstrating the utility of leveraging out-of-domain data to overcome scarcity of annotated meetings. Experimental results confirm HMNet's superiority over baseline transformer and polysynaptic models. While the model achieved strong performance under ROUGE metrics, the authors echoed con- cerns about evaluation, noting that lexical overlap does not always reflect factual accuracy or coherence. Overall, HMNet represents a milestone by combining hierarchical modeling with pretraining to generate more coherent and context-aware abstractive summaries.

While HMNet focuses on architecture, Zhong et al. (2021) address the problem from a *dataset and task-design perspec- tive* by introducing *QMSum*, a benchmark for query-based, multi-domain meeting summarization. Unlike traditional ap- proaches that aim to produce generic meeting summaries, QMSum is designed to mirror real-world use cases where users often seek answers to specific questions such as “What decisions were made?” or “What are the next steps?”. The dataset includes over 1,800 query-summary pairs derived from 232 meetings spanning academic, product, and committee domains. Each instance includes human-written queries and summaries, enabling systematic evaluation of a model’s ability to generate focused, user-relevant outputs. Baseline experi- ments with sequence-to-sequence models revealed significant performance gaps compared to human summaries, particularly in grounding responses within long transcripts and handling distributed factual information. The benchmark thus provides both a resource and a research challenge, steering the field toward query-focused and task-oriented summarization that aligns closely with user needs.

Extending beyond meetings, Song et al. (2022) investigated *abstractive grounded summarization of podcast transcripts*, which introduces even greater challenges due to informal language, cultural references, and reliance on external knowl- edge. The authors propose grounding mechanisms that ensure generated summaries remain factually aligned with source transcripts, thereby reducing hallucination, a common flaw in abstractive models. Their study contributes a large-scale dataset of podcast transcripts paired with human-written sum- maries, serving as a valuable resource for the broader task of dialogue summarization. Experiments with pretrained lan- guage models demonstrated that grounded approaches outper- form conventional abstractive methods, producing summaries that are both coherent and factually faithful. Importantly, the authors argue for moving beyond ROUGE as an evaluation standard, highlighting the need for metrics that better capture factual consistency and human-judged quality. By focusing on podcasts, their work extends summarization research into new conversational domains while maintaining relevance to meeting summarization.

In summary, these four papers collectively illustrate the*evolution of abstractive summarization research*. The sur- vey by Rennard et al. provides a high-level synthesis and identifies gaps, HMNet demonstrates the benefits of hier- archical modeling with cross-domain pretraining, QMSum introduces a benchmark for query-focused summarization, and Song et al. emphasize grounded generation for factual accuracy. Together, they showcase how progress in datasets, modeling techniques, and evaluation frameworks is gradually addressing the complexities of conversational summarization. Nonetheless, challenges such as privacy in data collection, long-context modeling, and faithful evaluation remain open problems. These works not only provide theoretical and methodological contributions but also chart a path forward for practical applications in summarizing meetings, podcasts, and other long-form dialogues.

III. METHODOLOGY

Overview

The proposed system is designed to automatically gener- ate abstractive summaries of YouTube videos by leveraging Natural Language Processing (NLP) techniques, pretrained transformer-based models, and data acquisition through the YouTube Data API and YouTubeTranscriptApi. The methodol- ogy integrates transcript extraction, preprocessing, summariza- tion, and evaluation into a single pipeline. The key components include:
Transcript acquisition from YouTube. Text preprocessing and cleaning. Summarization using the BART-large CNN model. Post-processing and meta-summarization of long transcripts. Summary evaluation with statistical and qualitative mea- sures. The architecture emphasizes robustness, handling noisy transcripts, chunking long inputs, caching transcripts for ef- ficiency, and ensuring summaries maintain coherence and factual accuracy.

Tools and Frameworks

Programming Language: Python was chosen for its exten- sive support in machine learning, NLP, and API integration. Hugging Face Transformers: Provides pre-trained state-of- the-art NLP models including BART. The pipeline abstraction simplifies model loading and inference.

PyTorch: Acts as the backend framework for the BART model. YouTubeTranscriptApi: Used to extract publicly available transcripts directly from YouTube videos. YouTube Data API v3: Serves as a fallback method to fetch video captions if transcript extraction fails. dotenv: For secure API key management by loading envi- ronment variables from .env files. Logging and Caching: A custom logger tracks pipeline activity, and cache stores transcripts to reduce redundant API calls. This ecosystem of tools ensures modularity, reproducibility, and efficient deployment of the summarization pipeline.

Transcript Acquisition

The first step in the pipeline is extracting transcripts. Two approaches are integrated to maximize coverage: Primary Method – YouTubeTranscriptApi Fetches machine-generated or human-annotated transcripts if available. Provides text in small segments (time-stamped entries). These segments are concatenated into a continuous tran- script string. Fallback Method – YouTube Data API Triggered when transcripts are disabled or unavailable. The pipeline queries the videos endpoint to validate video availability. The captions endpoint is used to retrieve subtitle data where possible.

Captions are merged into a single transcript string. Video ID Extraction and Validation A regex-based parser extracts video IDs from different YouTube URL formats. Validation ensures IDs conform to the standard 11-character format. Caching Retrieved transcripts are cached using a lightweight key- value store. Repeated requests for the same video reuse cached tran- scripts, reducing API load. This dual strategy ensures resilience against missing or disabled transcripts.

Preprocessing

The raw transcripts are often noisy, containing fillers, dis- fluencies, and irregular sentence boundaries. Preprocessing ensures clean, model-ready input. Cleaning Extra whitespaces are collapsed. Special characters are removed using regex patterns, pre- serving punctuation like . , ! ? -. Sentence Segmentation Transcripts are split into sentences using regex rules. Empty or trivial fragments are discarded.

Chunking for Model Compatibility

BART-large CNN supports a maximum input length of 1024 tokens. The text is chunked into segments that fit within this constraint. Token length is estimated using a heuristic (4 characters per token). Chunks are constructed dynamically to avoid truncation, ensuring semantic completeness. The result is a set of clean, coherent text chunks optimized for summarization.

Summarization Model 5.1 Choice of Model

The summarization component uses BART-large CNN from Hugging Face (facebook/bart-large-cnn). BART (Bidirectional and Auto-Regressive Transformers) is a sequence-to-sequence model combining BERT-like encoders with GPT-like decoders. It excels in abstractive summarization due to its ability to capture bidirectional context and generate fluent text.

Pipeline Implementation

The Hugging Face pipeline(”summarization”) is employed with PyTorch backend. Lazy loading ensures the model is only instantiated when required, optimizing memory usage.

Hyperparameters include:

max length can dynamically adjusted 50–200 tokens de- pending on input size. min length set between 20–50 tokens to enforce meaningful summaries. do sample False ensures deterministic outputs suitable for factual summarization. 5.3 UML Diagrams To gain a clear understanding of the system’s architecture and behavior, we designed multiple UML diagrams. These diagrams help in visualizing how different components of the system interact with each other and with external users. Use Case Diagram:

Represents the high-level functionalities of the system and illustrates the interactions between the user, the Chrome extension interface, and the summarization services. Class Diagram: Defines the core classes of the system such as TranscriptFetcher, Summarizer, CacheMan- ager, and UserInterface, along with their attributes, methods, and relationships. Sequence Diagram: Demonstrates the step- by-step flow of operations, starting from the user request to the Chrome extension, transcript retrieval, summarization processing, and finally returning the generated summary to the user. Together, these UML diagrams provide both structural and behavioral perspectives, ensuring that the system design is robust, modular, and easy to extend in future enhancements.

Multi-Stage Summarization
For transcripts exceeding 5000 words, each chunk is sum- marized independently. Generated chunk summaries are concatenated. If the combined summary exceeds model limits, a meta- summary is produced by re-summarizing the concatenated summaries. This hierarchical approach balances completeness with brevity.

Post-Summarization Processing
After generating summaries, the system provides additional processing for usability: Error Handling Edge cases like empty inputs, excessively short transcripts (¡20 words), or failed summarization attempts return user- friendly error messages. Summary Statistics The system computes key metrics: Original word count. Summary word count. Compression ratio (Number of chunks processed.
These statistics provide insights into the efficiency of the summarization process.

Evaluation 7.1 Intrinsic Evaluation
Quantitative Metrics: ROUGE (Recall-Oriented Understudy for Gisting Evaluation) can be used in future experiments to compare generated summaries against reference summaries.
Compression Ratio: Already implemented in the system, it provides a numerical measure of summary conciseness.

Extrinsic Evaluation
Human Judgments: Evaluators can assess summaries for fluency, coherence, factual accuracy, and informativeness. Task-Oriented Evaluation: Since summaries target video transcripts, their effectiveness can be measured by user sat- isfaction and retrieval of key video insights.

Robustness and Error Mitigation
The methodology accounts for multiple failure points: Transcript Missing: Switch to YouTube Data API fallback. Long Texts: Apply chunking and hierarchical summariza- tion. Noise in Input: Regex-based cleaning and sentence segmen- tation. API Latency: Caching transcripts to reduce repeated calls. This ensures the pipeline remains reliable across a wide variety of video types and transcript qualities.

System Workflow
User provides YouTube URL or video ID. Video ID is extracted and validated. Transcript is fetched via YouTubeTranscriptApi or YouTube Data API. Transcript is cached for efficiency. Preprocessing cleans and chunks the text. BART-large CNN generates abstractive summaries for each chunk. Chunk summaries are merged; a meta-summary is produced if necessary. Final summary and statistics are returned to the user.

Advantages of the Approach
Scalability: Handles videos of varying lengths through chunking and meta-summarization. Robustness: Dual transcript-fetching strategy minimizes failure cases. State-of-the-art NLP: Employs BART, a transformer model well-suited for abstractive summarization. User-Centered Design: Generates concise, readable sum- maries rather than raw transcripts. Extendability: Can be adapted to support other summariza- tion models (e.g., T5, Pegasus) or evaluation metrics.

Limitations and Future Enhancements
Factuality Concerns: While BART produces fluent text, it can occasionally hallucinate facts. Grounding techniques (e.g., retrieval-augmented summarization) could improve accuracy. Evaluation Gap: Automated evaluation relies mainly on ROUGE; richer metrics such as BERTScore or human assess- ments should be integrated. Multimodal Inputs:

Current system relies solely on text transcripts; incorporating audio and video cues may improve emphasis detection and context. Domain Adaptation: Fine-tuning on conversational datasets (e.g., AMI or QMSum) could yield better meeting or dialogue summarization performance.

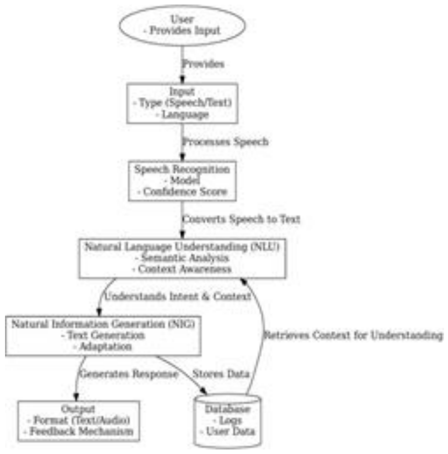


Fig. 1. User Flow Diagram

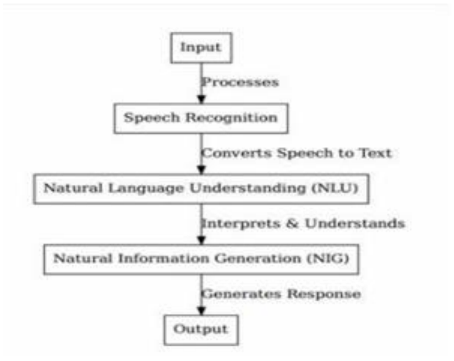
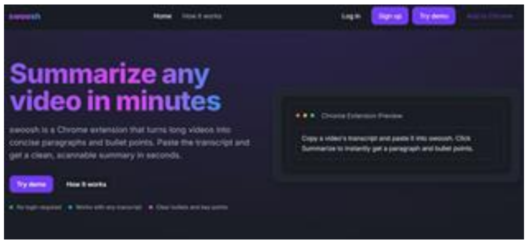


Fig. 2. Flow Chart Diagram

IV. RESULTS

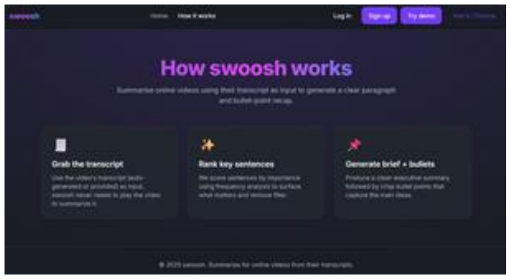
The summarizer extension demonstrates significant benefits in terms of time-saving, accessibility, and usability for students and professionals. Our project's results and implementations are given below:



swoosh is a Chrome extension that turns long videos into concise paragraphs and bullet points. Paste the transcript and get a clean, scannable summary in seconds.

Copy a video's transcript and paste it into swoosh. Click Summarize to instantly get a paragraph and bullet points.

Fig. 3. Home Page



Summarise online videos using their transcript as input to generate a clear paragraph and bullet-point recap.

© 2025 swoosh. Summaries for online videos from their transcripts.

Fig. 4. Basic Site Working

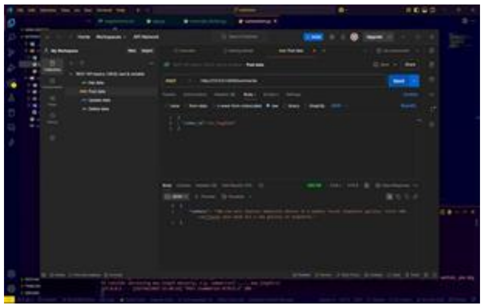
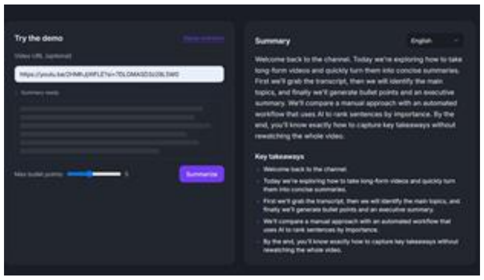


Fig. 5. Model: facebook/bart-large-cnn



Welcome back to the channel. Today we're exploring how to take long-form videos and quickly turn them into concise summaries. First we'll grab the transcript, then we will identify the main topics, and finally we'll generate bullet points and an executive summary. We'll compare a manual approach with an automated workflow that uses AI to rank sentences by importance. By the end, you'll know exactly how to capture key takeaways without rewatching the whole video.

Key takeaways

- Welcome back to the channel.
- Today we're exploring how to take long-form videos and quickly turn them into concise summaries.
- First we'll grab the transcript, then we will identify the main topics, and finally we'll generate bullet points and an executive summary.
- We'll compare a manual approach with an automated workflow that uses AI to rank sentences by importance.
- By the end, you'll know exactly how to capture key takeaways without rewatching the whole video.

Fig. 6. Basic Output

V. CONCLUSION

In conclusion, The Video Summarizer Extension provides a practical solution to the growing demand for efficient video content consumption. By combining YouTube API transcripts with AI-powered NLP summarization, the system delivers concise summaries that enhance learning, research, and accessibility. Despite limitations, the extension lays the ground- work for future innovations in multi-platform, voice-enabled, and context-aware video summarization. Practical Solution – Addresses the challenge of consuming long video content efficiently by providing quick and meaningful summaries. Integration – Combines YouTube API transcripts with AI- powered NLP summarization for accurate and concise out- puts. Outcome – Delivers summaries that enhance learning, research, and accessibility for students, professionals, and gen- eral users. Limitations – Dependent on transcript availability, model accuracy, and lacks tone/visual interpretation.

Acknowledgement
I would like to thank Mr. Kedar Pinniboyina Assistant professor of department of CSE for his guidance during work on the implementation of these techniques and while writing this paper.

REFERENCES

1. Virgile Rennard, Guokan Shang, Julie Hunter, Michalis Vazirgiannis. Abstractive Meeting Summarization: A Survey. Transactions of the Association for Computational Linguistics (TACL), 2023.
2. Chenguang Zhu, Ruochen Xu, Michael Zeng, Xuedong Huang. A Hi- erarchical Network for Abstractive Meeting Summarization with Cross-Domain Pretraining (HMNet). Findings of EMNLP, 2020.
3. Ming Zhong, Da Yin, Tao Yu, Ahmad Zaidi, Mutethia Mutuma, Rahul Jha, Ahmed Hassan Awadallah, Asli C¸ elikyilmaz, Yang Liu, Xipeng Qiu, Dragomir Radev. QMSum: A New Benchmark for Query-based Multi-domain Meeting Summarization. NAACL, 2021.
4. Kaiqiang Song, Chen Li, Xiaoyang Wang, Dong Yu, Fei Liu. Towards Abstractive Grounded Summarization of Podcast Transcripts. ACL, 2022.
5. Harsh Vartakavi, Saurabh Garg, Zafar Rafii. Audio Summarization for Podcasts. EUSIPCO (EURASIP), 2021.
6. Yang Liu, Mirella Lapata. Text Summarization with Pretrained Encoders. EMNLP-IJCNLP, 2019.
7. Shraman Palaskar, Florian Metze, et al. Multimodal Abstractive Sum- marization for How2 Videos. ACL (Short), 2019.
8. Shraman Palaskar, Ramon Sanabria, Florian Metze. MAST: Multimodal Abstractive Summarization with Trimodal Hierarchical Attention. arXiv preprint, 2020.
9. Iz Beltagy, Matthew E. Peters, Arman Cohan. Longformer Encoder- Decoder (LED) for Document Summarization. arXiv preprint, 2020.
10. Bharat Arora, et al. GPT2MVS: Using GPT-2 for Multimodal Video Summarization. arXiv preprint, 2021.
11. Spyros Barbakos, et al. Unsupervised Transcript-Assisted Video Sum- marization and Highlight Detection. arXiv preprint, 2025.
12. Nima Sadri, Bohan Zhang, Bihan Liu. MeetSum: Transforming Meeting Transcript Summarization using Transformers!. arXiv preprint, 2021.
13. Tengchao Lv, Lei Cui, Momcilo Vasilijevic, Furu Wei. VT-SSum: A Benchmark Dataset for Video Transcript Segmentation and Summariza- tion. arXiv preprint, 2021.
14. Xiachong Feng, Xiaocheng Feng, Bing Qin, Xinwei Geng. Dialogue Discourse-Aware Graph Model and Data Augmentation for Meeting Summarization. IJCAI (originally arXiv), 2021.
15. Bo He, Jun Wang, Jielin Qiu, Trung Bui, et al. Align and Attend: Multimodal Summarization with Dual Contrastive Losses (A2Summ). CVPR, 2023.
16. Sangwoo Cho, et al. StreamHover: Live stream Transcript Summariza- tion and Annotation. arXiv preprint, 2021.
17. Bin Zhao, Maoguo Gong, Xuelong Li. Hierarchical Multimodal Trans- former to Summarize Videos. arXiv preprint, 2021.
18. Iz Beltagy, Matthew E. Peters, Arman Cohan. Longformer: The Long- Document Transformer. arXiv preprint, 2020.
19. Jayesh Chavan, Satvik Bhabal, Srimanta Ghosh. Transcript Summarizer of YouTube Videos Using Deep Learning. Journal of Artificial Intelli- gence Research Advances (JoAIRA), 2024.
20. ang Deng, Jiawei Gao, Jianmei Nie, Haiyun Chen, Nitesh V. Chawla, Chenliang Li. Aspect- based Meeting Transcript Summarization: A Two-Stage Approach with Weak Supervision. arXiv preprint, 2023.