VidBit - Video summarizer AI

Leveraging machine learning, NLP and Deep Learning

Submitted by :
Uzma Khatoon
(Roll no. : **223142-11-0046**)
Aparup Roy
(Roll no.: **223142-21-0045**)
Alapan Banerjee
(Roll no.:**223142-21-0049**)

Paper :**CMSA-CC-6-13-P**

Undergraduate Project Report
Submitted to the Department of Computer Science, Bangabasi College
University of Calcutta

SUPERVISOR CERTIFICATE

This is to certify that the project report on "**Video Summarizer AI Leveraging machine learning, NLP and Deep Learning**" in the bona fide record at the report is done by Uzma Khatoon, Aparup Roy, Alapan Banerjee as a partial fulfillment of the requirement for the Degree of Bachelor in Science(B.Sc. Hons.) in Computer Science from the Department of Computer Science, Bangabasi College under University of Calcutta.

This report has been prepared under my guidance and is a record of the bona fide word carried out successfully.

Uzma Khatoon(roll no. = 223142-11-0046)

Aparup Roy(roll no. = 223142-21-0045)

Alapan Banerjee(roll no. = 223142-21-0049)

Prof Saikat Sarkar
(Supervisor)

ACKNOWLEDGEMENT

No endeavor is ever truly solitary; this work stands on the shoulders of countless acts of guidance, patience and support. Acknowledging the almighty for keeping us safe and sound throughout, we would like to take this moment to thank every individual who has been the backbone of this project.

We are deeply indebted to our supervisor **Prof. Saikat Sarkar** for his invaluable guidance and expertise throughout this project. His profound knowledge of this subject matter, as well as the continuous support and encouragement have been truly inspiring and instrumental in shaping the direction of this project.

We are grateful to all the faculty members of the Department of Computer Science, Bangabasi College, Kolkata for their constant appreciation and encouragement without which the project might not have been a success.

Furthermore, we would like to express our heartfelt appreciation to our family members and our fellow friends for their unwavering support and motivation . The faith they had in us has been instrumental in overcoming the challenges and completion of this project.

This project would not have been possible without the collective support, guidance and contributions of all the individuals and organizations mentioned above. We are honoured and privileged to have had the opportunity to work on this project, and we sincerely appreciate everyone who has played a role in its successful completion.

TABLE OF CONTENTS

ABSTRACT

In a world overpowered with sprawling video content, the task of sitting through hours of footage to uncover its core message feels overwhelming, similar to searching a needle in a haystack. Here --- **VidBit** the Video Summary, Transcript and Clip Generator rises to the challenge offering an AI-powered solution that transforms videos into concise, accessible format with precision. This system employs Natural Language Processing (NLP) to generate succinct textual summaries, harnesses Deep Learning through models like Whisper for accurate transcription, and applies speech synthesis to create narrated clips that capture the video's essence. Buit with Python and supported by a Flask backend, the solution ensures efficient processing and user-interaction , automating the distillation of video content into multi-format outputs, this project enhances accessibility and utility, opening doors for applications in education, professional settings, and beyond, while exemplifying the fusion of narrative insight and technical innovation.

Keywords – Video summarization, textual summaries, Clip demonstration, NLP, Transformer, Deep Learning, Whisper, Education and Professional processes.

1. INTRODUCTION

1.1 Overview:

As the saying goes “time is money”, yet the rising tide of video content online often demands far too much of it, creating barriers that hinder its effective use. A major issue is the long time needed to sift through hours-long videos to find the main ideas, a struggle for students, professionals, or researchers who need quick access to insights. Another concern is that videos aren’t always easy to access, leaving people with hearing difficulties, those unfamiliar with the language, or anyone in a noisy place at a disadvantage, as most videos lack text to help them understand. Additionally, the extra clutter in videos making it tough to pinpoint the key points, much like looking for a needle in a haystack, leaving users lost in a sea of information without a simpler format. For news teams, the pressure to quickly turn breaking news into short updates for TV or social media requires both speed and accuracy, a task that can feel overwhelming. The effort of reusing videos for new purposes – like crafting training clips or promotional snippets – also takes significant work for educators, creators, or businesses. VidBit steps in to address these challenges to automatically generating short summaries, accurate transcripts and narrated clips, saving time, improving access for all, speeding up news updates, simplifying video reuse, and helping content reach more people.

1.2 Problem Statement:

A diverse group of individuals, including students attending lengthy and often dense lectures, hearing-impaired individuals who face barriers in accessing auditory content, and professionals engaged in extended business meetings, encounter significant challenges in efficiently processing and retaining critical information from video-based material. Students may struggle to keep up with fast-paced academic content, hearing-impaired individuals are limited by the lack of real-time auditory comprehension tools, and professionals often lack the time to review lengthy meeting recordings to extract actionable insights. These issues are compounded by the increasing volume of digital content, which overwhelms users and reduces their ability to focus on key takeaways, ultimately hindering learning, accessibility, and productivity. VidBit aims to address this multifaceted problem by leveraging an AI-driven solution that generates concise text summaries, accurate transcriptions, and relevant video clips from a variety of video sources. This tool will enable all users—students, hearing-impaired individuals, and professionals—to quickly access, understand, and utilize essential information on their devices, thereby enhancing educational outcomes, inclusivity, and workplace efficiency.

1.3 MOTIVATION:

The VidBit system is designed for integration into any device, offering seamless access to a wide range of users. It generates concise text summaries, accurate transcriptions, and relevant video clips from lengthy video content, including lectures, business meetings, and educational materials. Tailored to support students by efficiently extracting key points, assist hearing-impaired individuals with accessible information, and aid professionals by reducing the time needed to review extensive meeting recordings, the system delivers critical information customized to user needs on their devices. Additionally, an app uploads these summaries and transcriptions to a remote database, establishing a shared repository of accessible content for all VidBit users, thereby enhancing learning, inclusivity, and productivity.

1.4 DOMAIN DESCRIPTION:

The VidBit project operates within the domain of artificial intelligence (AI) and assistive technology, with a focus on enhancing accessibility and information processing across educational, professional, and inclusive communication contexts. It leverages natural language processing (NLP) and video analysis techniques to transform lengthy video content—such as lectures, business meetings, and educational tutorials—into concise, actionable formats, including text summaries, transcriptions, and video clips. The domain addresses the needs of diverse user groups, including students seeking efficient learning tools, hearing-impaired individuals requiring accessible content, and professionals aiming to optimize time management. This domain is increasingly relevant due to the rapid growth of digital content and the ongoing demand for inclusive, user-friendly solutions in education and workplace environments.

1.5 APPLICATION :

VidBit serves as a valuable tool for students to quickly grasp lecture highlights for study or revision. It supports hearing-impaired individuals by providing text-based access to video content in diverse settings. Professionals can utilize it to extract key insights from meetings, enhancing decision-making efficiency. Additionally, it aids in summarizing news by breaking down complex reports into digestible points and assists journalists in efficiently extracting critical details for their stories.

1.6 Main Idea

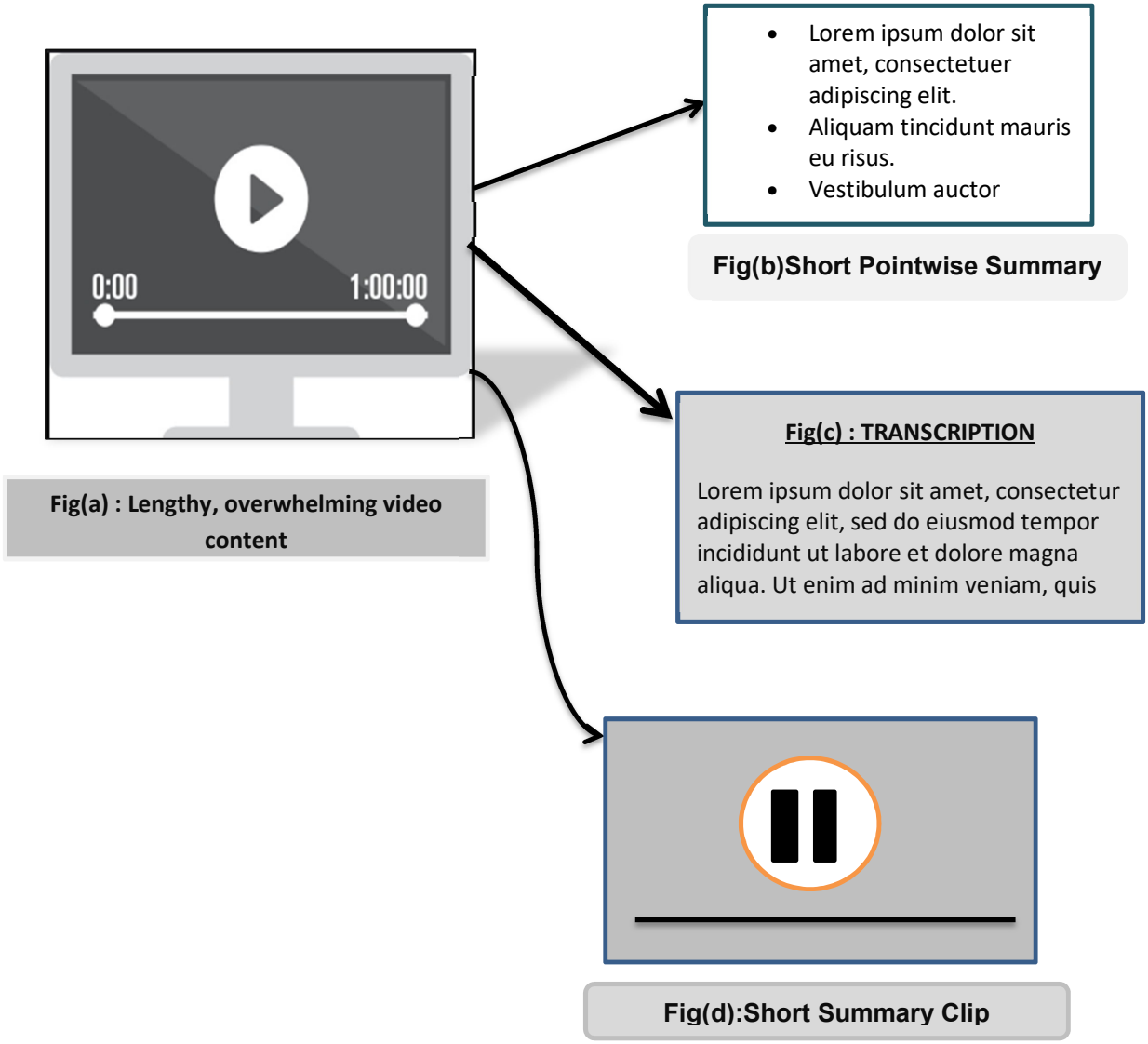

Fig 1: Shows an overview of the main objective of this project where in **Fig(a)** – a long, lengthy overwhelming video content with a long duration is transformed into three different components namely **Fig(b)** – a short pointwise summary which highlights the key points from the video, a transcript of the video as shown **in Fig(c)** and a short summary clip generated from the original video containing key points in **Fig(d).**

2. RELATED WORKS

- **NLP-Based Approaches for Automated Multimedia Summarization[1][2]3[4][5] :**
  Recent advancements in Natural Language Processing (NLP) have enabled efficient summarization of video and audio content, particularly from platforms like YouTube and online learning environments. By analyzing transcripts, captions, and metadata, these systems extract essential information using techniques such as text summarization, sentiment analysis, and topic modeling. Models like OpenAI's Whisper and ChatGPT have proven effective in transcribing and summarizing meetings, while algorithms like TF-IDF and Gensim help condense educational lectures. Some methods also integrate with IoT devices to support real-time transcription and contextual summarization. Collectively, these approaches aim to enhance accessibility, save time, and streamline content consumption across various domains.

- **Deep Learning Techniques for Video Summarization [6][7][8] :**
  Recent developments in video summarization leverage deep learning and NLP to extract and present essential content efficiently. Deep-learning-based approaches use neural network architectures to automatically identify informative segments of videos, with structured pipelines for feature extraction, classification, and summary generation. These methods are categorized through a systematic taxonomy and evaluated using standardized benchmarks, highlighting both current capabilities and future directions. Complementing this, NLP-driven methodologies employ speech recognition, keyword extraction (RAKE), sentence scoring, and timestamping to generate concise, navigable summaries—particularly useful in educational videos. Additionally, document embedding models using dynamic convolutional neural networks (DCNNs) offer powerful representations of text by preserving word and sentence order, enabling hierarchical semantic learning without extensive feature engineering. These integrated approaches advance video and text summarization by improving accuracy, semantic depth, and user accessibility.

- **CNN-Based ASR and Extractive NLP for Video Summarization [9]:**
  With the growing volume and length of online videos, there is an increasing demand for tools that can summarize content efficiently. This work presents a two-phase video summarization approach combining speech recognition and extractive text summarization. In the first phase, an Automatic Speech Recognition (ASR) system powered by Convolutional Neural Networks (CNN) converts spoken content into text transcripts. In the second phase, extractive summarization techniques are applied to distill key information from the transcripts. This method allows users to grasp the core message of videos without watching them in full, enhancing content accessibility and saving valuable time.

- **Multimodal and Language-Guided Approaches for Enhanced Summarization [10][11] :**
  Recent works leverage multimodal data and language models to improve summarization tasks. In healthcare, a framework using the MMQS dataset combines medical queries with visual imagery, employing CLIP and LLMs to generate context-rich, visually aware summaries for better clinical understanding. Similarly, the CLIP-It model unifies generic and query-focused video summarization by scoring video frames based on relevance to user queries or captions, achieving strong performance even in unsupervised settings. These approaches highlight the growing effectiveness of combining vision and language for domain-specific and customizable summarization.

- **Clustering-Based Approaches for Static and Efficient Video Summarization [12][13] :**
  To address the growing demand for concise video content, clustering-based techniques like VSCAN and Delaunay Triangulation have been explored for static video summarization. VSCAN enhances summary quality using a modified DBSCAN algorithm that considers both color and texture features, showing superior results compared to existing methods. Similarly, the Delaunay-based approach automatically clusters multidimensional frame data to generate low-redundancy summaries with fewer frames, requiring no manual parameter tuning. Both methods prove effective in content representation, with demonstrated performance on benchmark datasets like the Open Video Project.

- **Encoder-Based Approaches for Fine-Grained and Deep Feature Video Summarization [14][15] :**
  Recent unsupervised video summarization methods leverage encoder architectures to extract high-level semantic and motion-aware information. One approach introduces an online motion Auto-Encoder that captures object-level key motions from super-segmented clips for real-time summarization. Another uses multiple pre-trained CNNs with a Sparse Autoencoder to combine features, followed by a Random Forest classifier for key-frame selection, achieving high F-scores on VSUMM and OVP datasets. These encoder-driven techniques effectively produce compact, high-quality summaries across diverse video content without supervision.

3. METHODOLOGY

3.1 Problem Formulation

The main problem VidBit addresses is the difficulty users face in accessing key information from the growing volume of video content, such as academic lectures, professional meetings, and online media. Students, professionals, and individuals with hearing impairments encounter significant barriers, including time constraints, lengthy video durations, and accessibility challenges, such as hearing difficulties or environments unsuitable for audio playback.

Manual transcription and summarization of videos are time-consuming, prone to errors, and impractical for handling large amounts of content. While some automated tools exist, they often fail to deliver contextually accurate summaries or precise timestamps, making it challenging for users to efficiently locate and utilize critical information.

To detect and overcome the same we have followed the following steps and algorithms

3.1.1 Video and Audio Data Acquisition

The process begins with acquiring video content, such as YouTube videos or local files, which serve as the primary input. For videos requiring transcription, audio is extracted to facilitate speech-to-text conversion. The system ensures compatibility with various video formats and resolutions, capturing both visual and auditory elements essential for summarization. Preprocessing steps, such as audio normalization, are applied to enhance audio clarity, ensuring high-quality input for transcription.

3.1.1.1 Audio Seperation

The audio separation step is all about pulling the audio track out from the video, allowing us to zero in on the spoken content for a deeper analysis. This begins with extracting the audio as a standalone file, neatly separating it from the visual elements so we can focus entirely on the sounds. It's crucial that this process keeps all the important audio details intact, including every bit of speech, ensuring nothing gets lost as we move forward. Once separated, the audio is saved in a format that's perfect for the next stages, laying down a solid and clean foundation for the analysis to come.

3.1.1.2 Preprocessing

This stage is all about giving the audio a boost in quality to make sure the tasks that follow, like transcription, run smoothly and reliably. It starts with audio normalization, where we tweak the amplitude to a steady level, smoothing out any volume ups and downs to keep the clarity consistent throughout the track. Then, we tackle noise reduction by toning down those pesky background sounds, helping the main audio content stand out clearly for better understanding. Next up, we adjust the sample rate to a uniform pace, ensuring it plays nicely with the transcription tools and sidestepping any processing hiccups. Finally, we add some extra polish with quality enhancements, like filtering out distortions, to deliver a top-notch input that's primed for precise speech-to-text conversion.

3.1.2 Transcription and Segmentation

Transcription is performed using advanced speech-to-text models to convert video audio into text. The model generates timestamped transcript segments, capturing spoken content with precise start and end times. These segments are critical for aligning text with corresponding video moments. To improve transcription accuracy, the system accounts for variations in speech patterns, accents, and background noise. The transcript is further segmented into sentences using NLP techniques, such as sentence boundary detection, to create a structured text corpus suitable for summarization. It involves the following steps methodically :-

- Listening and Capturing: The process starts by carefully listening to the audio, picking up every word spoken and noting down the exact moments they begin and end, so we can link them back to the video timeline.

- Writing It Down: All the captured speech is turned into a flowing text, creating a rough draft of what was said, ready for the next stage.

- Breaking Into Sentences: The text is then split into clear, individual sentences, making it easier to work with by organizing it into bite-sized, meaningful chunks.

- Aligning with Video: Finally, each sentence is tied to its specific spot in the video, ensuring the written words match up perfectly with the visual moments they belong to

3.1.3 Extractive Summarization :

The system employs an extractive summarization approach to identify key sentences from the transcript that represent the video's core content. This process involves:

- **Sentence Relevance Scoring**: Sentences are encoded into semantic embeddings using a pre-trained language model. The embeddings capture contextual meaning, enabling the system to evaluate sentence importance based on their centrality to the overall content. Centrality is measured by computing average cosine similarities between sentence embeddings, identifying sentences that best represent the video's main ideas.
- **Diversity Selection**: To avoid redundancy, the system selects diverse sentences by setting a similarity threshold, ensuring that chosen sentences cover distinct aspects of the content. The selection prioritizes sentences with high centrality scores while maintaining variety, resulting in a concise set of key points.
- **Timestamp Alignment**: Selected sentences are mapped to their corresponding transcript segments, retaining their start and end timestamps. This alignment ensures that summary points are directly linked to specific video moments, facilitating accurate segment extraction.

The output is a set of timestamped key points, which collectively form a comprehensive yet condensed summary of the video.

3.1.4 Scene Detection and Segment Selection

To create a summarized video, the system detects scene changes in the original footage to identify visually distinct segments. A content-based scene detection algorithm analyzes frame-to-frame differences, marking transitions based on visual intensity changes. The algorithm's sensitivity is adjusted to balance the number of detected scenes with their relevance to the content.

Key points from the extractive summary are then matched to appropriate video segments. Segments corresponding to the timestamps of selected sentences are prioritized, with a duration constraint (e.g., 60 seconds maximum) to ensure brevity. The system calculates the total duration of selected segments to confirm that the summary video remains significantly shorter than the original, maintaining user-specified time constraints.

3.1.5 Summary Video Creation:

The selected video segments are concatenated to produce a cohesive summary video. The process involves:

- **Segment Extraction**: Individual video clips are extracted from the original footage based on their start and end timestamps. Each clip is re-encoded to ensure compatibility and optimize file size, using standard video codecs and resolutions (e.g., 1920x1080 at 25 fps).

- **Clip Concatenation**: The extracted clips are combined into a single video file using a concatenation protocol that preserves audio-video synchronization. The system validates the output to ensure no errors, such as corrupted frames or audio desync, occur during processing.

- **Storage Management**: Temporary files generated during extraction and concatenation are stored in a designated directory and removed after processing to optimize disk space usage.

The resulting summary video is stored in a user-specified location, providing a condensed and precise representation of the original content which now contains key elements along with audio and visual elements.

3.1.6 Output Storage and Accessibility :

The system generates multiple outputs to enhance accessibility and usability. Timestamped key points are saved as a text file, formatted to include sentence text and corresponding timestamps for easy reference. A JSON file is also created to store comprehensive response data, including the video link, title, full transcript, timestamped summary, and summary video path. These outputs cater to diverse user needs, such as text-based review for hearing-impaired individuals or quick video summaries for time-constrained professionals.

3.1.7 Validation and Optimization:

The system is validated using a variety of video inputs, including educational lectures, documentaries, and professional recordings, to ensure robustness across content types. Performance is evaluated based on metrics such as summary coherence, transcription accuracy, and video segment relevance. User feedback and test results guide iterative improvements, such as refining sentence selection criteria or optimizing scene detection thresholds.

Optimization focuses on reducing processing time and resource usage, enabling real-time or near-real-time operation on standard hardware. The system is designed to scale, supporting integration into broader platforms, such as educational tools or assistive technology devices, to maximize its impact.

3.1.8 Some Important Methodologies Used :

WORD EMBEDDING :

Word embedding is a technique used in natural language processing to represent words as numerical vectors that capture their meaning and context. This allows computers to understand how words relate to each other — for example, understanding that "king" and "queen" are related in meaning. In the context of building a video summarizer AI, word embeddings are crucial. When a video is processed, the AI typically extracts a transcript using speech recognition. That transcript can be long and unstructured, so word embeddings help the AI understand which parts are important by capturing the deeper meaning of the words and sentences, not just surface-level keywords. This allows the AI to generate a meaningful short text summary that accurately reflects the key events or topics in the video. But it doesn't stop there — if the AI is also tasked with generating a short highlight video to match the summary, word embeddings are used again to link the summary sentences with the actual video content. Using models like CLIP, both text and video segments are converted into vectors, allowing the AI to compare and find the most relevant scenes. For example, if the summary says, "the cat jumps on the table," the AI can search the video for a clip that matches that scene even if the visual data doesn't explicitly contain the exact words. This whole process ensures that the AI not only understands what's being said but can also visually show it, making the summaries much more informative and engaging. In short, word embeddings act as the intelligence layer that connects spoken words, written summaries, and visual content in a seamless and meaningful way.

Word embedding is important in video summarization and clip generation AI because it helps the system understand the **meaning behind words** in transcripts or summaries. This allows the AI to identify **key moments**, **match text to video scenes**, and create more accurate, relevant summaries—both in text and visuals. Without embeddings, the AI would rely on simple keyword

matching, missing the deeper context needed for high-quality summaries and clips.

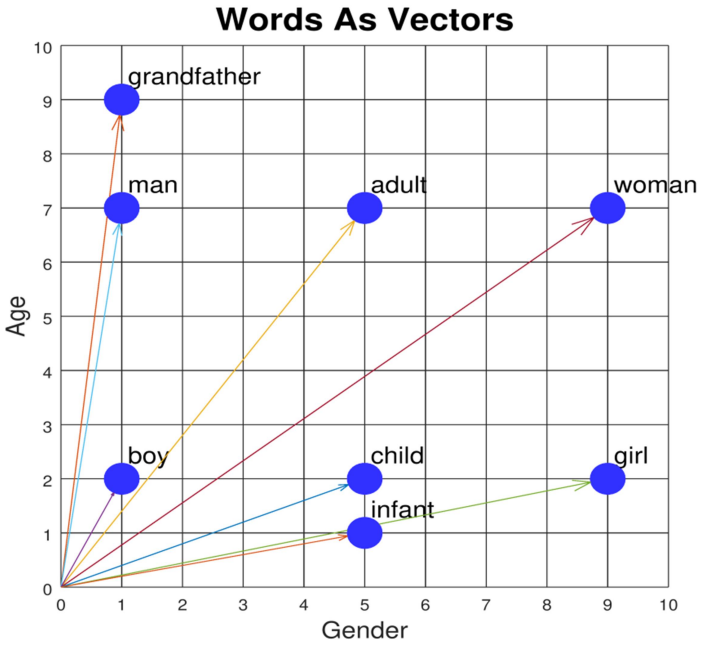

Fig 2 : An illustration of word embedding – words are treated as vectors and the relationship between them is mapped (*Source : www.cs.cmu edu*)

HOW VIDBIT LEVERAGES WORD EMBEDDING

The VidBit project, designed for sharing and discovering short-form video content, leverages word embeddings to generate pointwise summary transcripts and corresponding summary video clips directly from video inputs . For example, when a user uploads a video featuring a skateboarding session at a park, VidBit processes any associated text, such as an automatically generated transcript or user-provided captions (e.g., “skateboarder performs tricks at the park”), using word embeddings. These embeddings map words like “skateboarder,” “tricks,” and “park” to numerical vectors that capture their semantic relationships, recognizing their contextual similarity to related terms like “stunts” or “skateboarding.” By analyzing these relationships, VidBit creates a concise pointwise summary transcript, such as:

- Skateboarder executes dynamic tricks.
- Action takes place at an outdoor park. This transcript distills the video’s key moments into clear, digestible points. Simultaneously, the system uses the semantic insights from word embeddings to identify and extract relevant video segments, such as clips showing the skateboarder performing tricks, to generate a summary video clip that visually corresponds to the transcript. This dual output ensures users can quickly

understand the video's core content through both text and visuals, streamlining content discovery and enhancing the overall experience on the VidBit platform

.

ENCODER – DECODER TRANSFORMER :

Transformers are a type of deep learning architecture that have revolutionized the field of natural language processing (NLP) in recent years. They are widely used for tasks such as language translation, text classification, sentiment analysis, and more.

The transformer architecture was first introduced in a 2017 paper by Google researchers Vaswani et al. called "Attention Is All You Need"[16]. This paper proposed a novel approach to NLP tasks that relied solely on the self-attention mechanism, a type of attention mechanism that allows the model to weigh the importance of different words in a sentence when encoding it into a fixed-size vector representation.

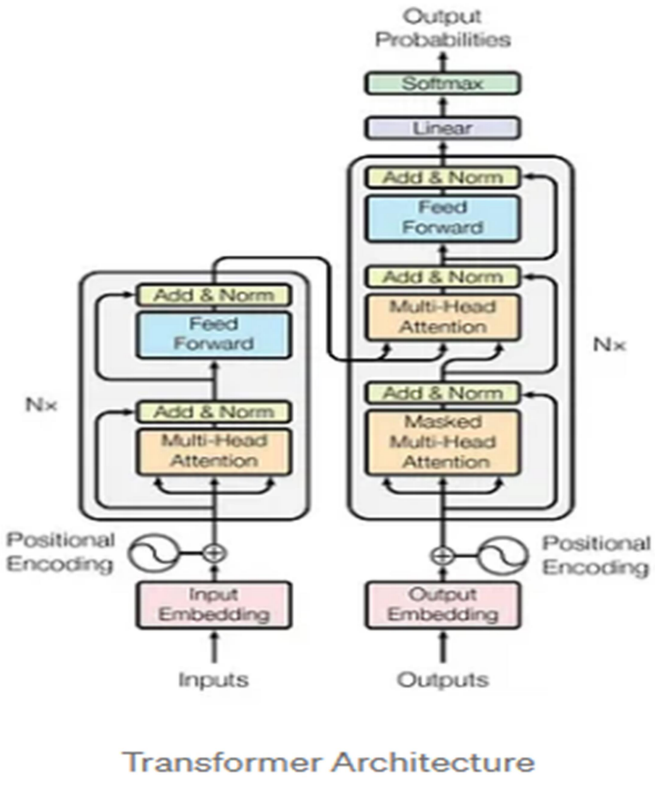

Fig 3: Transformer Architecture (*Source : www.medium.com*)

GENERAL OVERVIEW OF THE ARCHITECTURE :

- **Encoder -** Reads and understands the full input text, capturing the meaning and relationships between words.
- **Decoder -** Uses that understanding to generate a shorter version — the summary — word by word.
- **Attention mechanism -** Helps the decoder focus on the most important parts of the input while generating the summary.

The transformer architecture is composed of an encoder and a decoder, each of which is made up of multiple layers of self-attention and feedforward neural networks. The self-attention mechanism is the heart of the transformer, allowing the model to weigh the importance of different words in a sentence based on their affinity with each other.

In addition to self-attention, the transformer also introduces positional bias, which allows the model to keep track of the relative positions of words in a sentence. This is important because the order of words in a sentence can significantly impact its meaning**.**

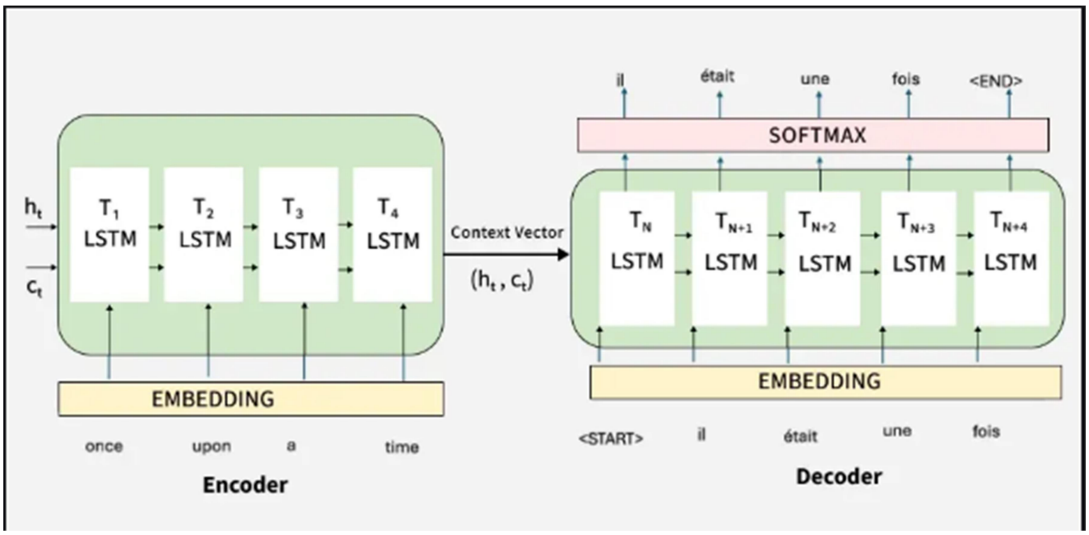

Fig 4: Encoder - Decoder Model (*Source : geeksforgeeks*)

TRANSFORMER ENCODER :

The transformer encoder architecture is used for tasks like video summarization, where the model must generate a concise summary of a video by understanding its content. The encoder takes in a sequence of visual and/or audio features extracted from the video frames and produces a fixed-size vector representation of the entire video, which can then be used to generate a summary.

Unlike the encoder-decoder architecture, the transformer encoder is only concerned with the input sequence and does not generate any output sequence. It applies self-attention mechanism to the input tokens, allowing it to focus on the most relevant parts of the input for the given task.

- The **encoder** takes the full input text (like a paragraph or article).
- It breaks the text into tokens (words or subwords), then:

- Applies **self-attention** to understand relationships between words.
- Builds a **contextual representation** of the whole text.

WORKING OF ENCODER WITH REFERENCE TO VIDBIT :

- **Video Processing:**
  - A video is a sequence of frames (images).
  - First, we extract feature**s** from each frame using a CNN or Vision Transformer (ViT**)** — this turns each frame into a numerical vector.

- **Positional Encoding:**
  - Since transformers don't inherently understand order, we add **positional encodings** to these frame vectors.
  - This helps the model know the **temporal order** of frames (e.g., beginning vs end of video).

- **Self Attention Layers :**

  The encoder applies **self-attention** across all frame vectors:

  - This allows each frame to attend to (i.e., understand its relation to) all other frames.
  - Important for learning which parts of the video are **contextually important**.

- **Feed Forward and Layer Norm :**

  - After attention, the vectors go through a feedforward network for deeper understanding.
  - Layer normalization and residual connections help with stable learning.

- **Final Encoder Output:**
  - The encoder outputs a sequence of contextual frame embeddings
- These embeddings are passed to the **decoder** to generate.

TRANSFORMER DECODER :

In video summarization, the transformer decoder architecture plays a crucial role in generating coherent textual summaries from visual content. It takes a fixed-size vector representation of the video and typically produced by an encoder and generates a sequence of words or sentences that describe the video's key moments, with each part of the summary conditioned on what has already been generated**.**

WORKING OF DECODER WITH REFERENCE TO VIDBIT :

The decoder in VidBit's encoder-decoder transformer uses the encoder's contextual frame embeddings to generate textual summaries (e.g., "A chef prepares pasta"), visual summaries (e.g., keyframes of key actions), or both. The following steps demonstrate the work of decoder :-

- **Input to Decoder**

  the decoder takes the encoder's output (e.g., 120 contextual frame embeddings for a 2-minute video) and an initial input, like a start token () for text or a query vector for keyframes.

- **Masked Self Attention**

  The decoder uses masked self-attention to process its output sequence (e.g., generated text) sequentially, ensuring each word or keyframe score only attends to prior elements. This ensures textual summaries are grammatically coherent or or keyframes are selected in a logical order

- **Encoder Decoder Attention**

  The decoder queries the encoder's frame embeddings to focus on relevant video parts. This maps video content to outputs: words like "prepares" attend to frames of cooking actions for text summaries, or high attention scores select keyframes of key scenes (e.g., stirring a pot).

- **Feed Forward Neural Network and Layer Normalization**

  A feedforward neural network (FFN) refines each output element (word or frame score), with layer normalization and residual connections stabilizing training

➢ **Output Generation**

The output summary is generated .

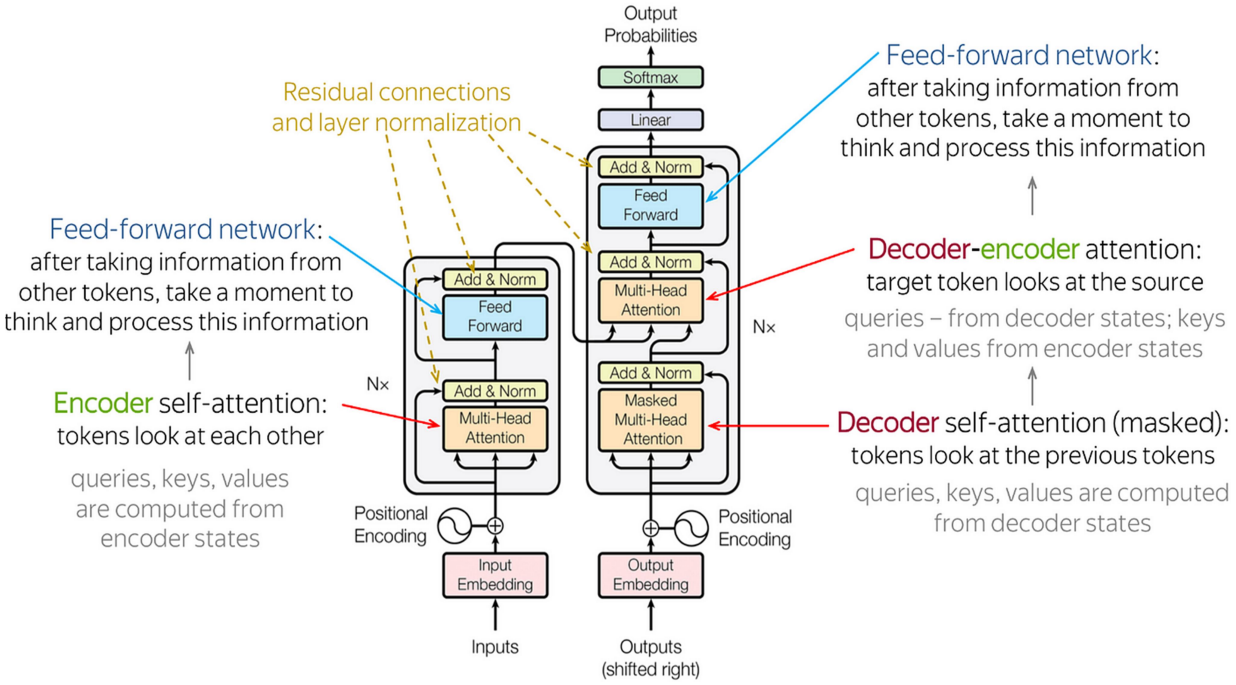

Fig 5 : An explanation of Encoder Decoder Transformer (*Source : AI mind)*

SELF ATTENTION :

Self-attention is a technique used in NLP that helps models to understand relationships between words or entities in a sentence, no matter where they appear. It is a important part of transformers model which is used in tasks like translation and summary generation.

Self-attention helps us understand how frames relate over time (e.g., a door opening leading to a new scene), ensuring summaries reflect the video's storyline. It assigns higher attention scores to salient frames, so we can focus on critical content (e.g., a chef serving a dish) while ignoring redundant frames (e.g., static shots).

MECHANISM

This mechanism captures long-range dependencies by calculating attention between all words in the sequence and helping the model to look at the entire sequence at once. Unlike traditional models that process words one by one it helps the model to find which words are most relevant to each other helpful for tasks like translation or text generation. Here's how the self-attention mechanism works:

1. Input Vectors and Weight Matrices: Each encoder input vector is multiplied by three trained weight matrices W(Q), W(K), W(V) to generate the key query and vector values
2. Query-Key Interaction: Multiply the query vector of the current input by the key vectors from all other inputs to calculate the attention scores.
3. Scaling Scores: Attention scores are divided by the square root of the key vector's dimension(*dk*) usually 64 to prevent the values from becoming too large and making calculations unstable.
4. Softmax Function: Apply the softmax function to the calculated attention scores to normalize them into probabilities.
5. Weighted Value Vectors: Multiply the softmax scores by the corresponding value vectors.
6. Summing Weighted Vectors: Sum the weighted value vectors to produce the self-attention output for the input.

Above procedure is applied to all the input sequences. Mathematically self-attention matrix for input matrices (Q, K, V) is calculated as :

$$\text{Attention}(Q, K, V) = \text{softmax}((QK^T / \sqrt{dk}))V.$$

3.2 FLOWCHART DEMONSTRATION

SYSTEM ARCHITECTURE / WORKFLOW

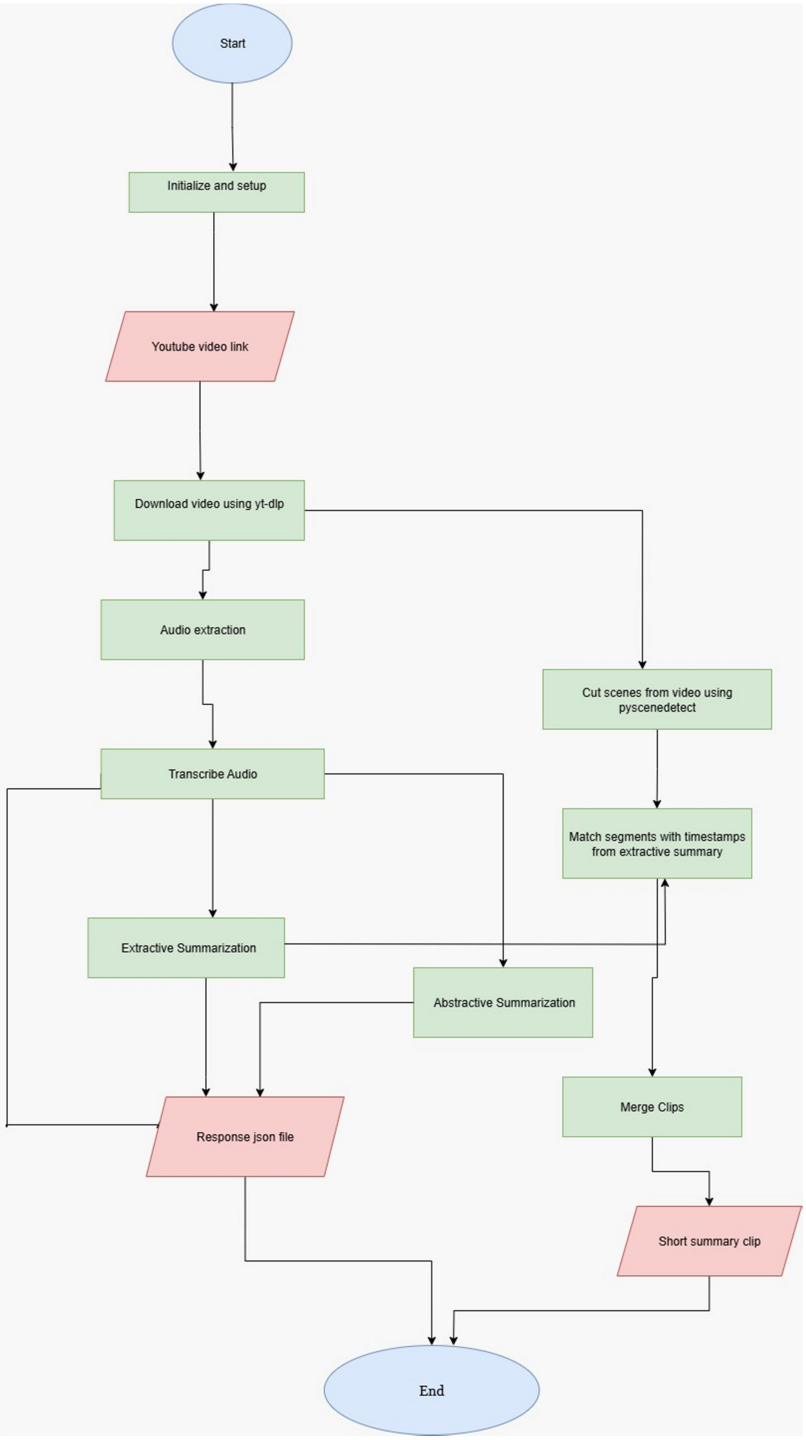

Fig 6 : Flowchart depicting the working of the Video Summarizer VidBit model

4.IMPLEMENTATION

4.1 Runtime Environment:

VidBit is developed to operate within Google Colab, a free, browser-based platform tailored for machine learning, data science, and computational tasks. Google Colab eliminates the need for local hardware setup by providing access to virtual machines (VMs) equipped with substantial computational resources. This makes VidBit accessible to users without high-end local hardware, as all processing occurs in the cloud. It leverages Google's free computing resources, including CPUs, GPUs, and TPUs, to deliver efficient and scalable performance for users. It typically requires 8–50 GB of RAM (free tier offers ~12.7 GB, Colab Pro/Pro+ up to 52 GB), 4–40 GB of GPU VRAM depending on workload, and 10–50 GB of disk space for datasets (free tier provides ~112 GB).

The main tools , APIs, and Libraries used to develop the Video Summariser Web Application are as follows :

- 4.1.1 **Fast Whisper** : In this project, the FastWhisper module, built on the Faster Whisper library, serves as a core component for high-performance automatic speech recognition (ASR), enabling the transcription of audio from video files into accurate, timestamped text. It utilizes an optimized implementation of OpenAI's Whisper model, leveraging techniques like model quantization and efficient inference to deliver fast, resource-efficient processing on Google Colab.

  FastWhisper extracts audio tracks from videos and transcribes spoken content into text, supporting a wide range of languages, dialects, and accents, while robustly handling challenges like background noise, overlapping speech, or variable audio quality. Its functionality includes generating precise word-level or segment-level timestamps, which is critical for applications like automated subtitle creation, enabling synchronized captions for videos.

  FastWhisper's model variants (e.g., tiny, base, medium, or large) allow users to balance transcription accuracy and computational cost , depending on the model size and audio length. For example, the base model might use ~4 GB RAM for a 10-minute video, while the large model could require ~10 GB for longer or complex audio. Users can further customize FastWhisper's behavior, such as enabling language detection for multilingual videos or fine-tuning transcription parameters, enhancing its versatility for diverse use cases in the VidBit ecosystem.

- 4.1.2: **yt- dlp** : The yt-dlp module serves as a critical component for downloading and extracting video and audio content from online platforms, primarily YouTube, but also supporting numerous other websites.

A powerful, open-source fork of youtube-dl, yt-dlp enables VidBit to fetch high-quality video and audio streams, metadata, and subtitles with high reliability and flexibility.

Its core functionality includes downloading videos in various formats and resolutions (e.g., MP4, WebM, up to 4K or 8K when available), extracting audio as standalone files (e.g., MP3 or WAV), and retrieving metadata like video titles, descriptions, and thumbnails. It also supports advanced features such as downloading entire playlists, extracting specific segments of videos, or accessing region-restricted content via proxy configurations. Within VidBit, yt-dlp facilitates seamless content acquisition for further processing, such as feeding videos into FastWhisper for transcription or other analysis pipelines. Resource-wise, yt-dlp is lightweight, typically requiring 1–4 GB of RAM for most tasks and minimal GPU usage, though disk space needs , based on video size and quality. Its efficiency and extensive format support make it an essential tool for enabling VidBit's video processing workflows.

- 4.1.3: **FastAPI** : The FastAPI module serves as a high-performance, asynchronous web framework for building and deploying APIs to facilitate efficient communication between VidBit's backend and client applications. Written in Python, FastAPI enables the creation of robust, scalable RESTful APIs that handle requests for processing video data, retrieving results, or managing user interactions. Its core functionality includes defining endpoints to accept inputs (e.g., video URLs or processing parameters), processing requests asynchronously for low-latency responses, and delivering outputs like processed video metadata or analysis results in JSON format. FastAPI's automatic generation of OpenAPI documentation simplifies testing and integration, allowing users to interact with VidBit's services via tools like Swagger UI. It supports data validation, dependency injection, and seamless integration with databases or other services, making it ideal for managing VidBit's dynamic workloads.

- 4.1.4 : **PyTorch** : The PyTorch module serves as a powerful, open-source machine learning framework for building and training deep learning models to process and analyze video or audio data.

  PyTorch provides a flexible, dynamic computational graph that enables the creation of neural networks for tasks such as feature extraction, video classification, or semantic analysis of content within the project. Its core functionality includes constructing and optimizing models (e.g., convolutional or transformer-based architectures) to process video frames or embeddings, leveraging GPU acceleration for efficient training and inference. PyTorch's extensive library of pre-built modules, such as torchvision for image/video processing or torchaudio for audio handling,

supports tasks like frame sampling or audio feature extraction. It also facilitates model training with automatic differentiation and optimizers, enabling the project to learn patterns for summarization or content understanding. Resource-wise, PyTorch's requirements depend on model complexity and data size, typically needing 4–16 GB of RAM for lightweight tasks. PyTorch's versatility and optimization capabilities make it a cornerstone for the project's deep learning-driven video summarization workflows.

- 4.1.5: **Sentence Transformer** : In the video summarizer project, the Sentence Transformers module, built on the Hugging Face Transformers library, is utilized to generate high-quality text embeddings for semantic understanding and analysis of textual content derived from video or audio data. It leverages pre-trained transformer models (e.g., BERT, RoBERTa, or DistilBERT) fine-tuned for tasks like sentence similarity, semantic search, or text clustering, enabling the project to process transcribed text (e.g., from audio) or metadata for summarization and content analysis. Its core functionality includes encoding text into dense vector representations, which capture semantic meaning, allowing for tasks such as identifying key themes in video transcripts, grouping similar content, or generating concise summaries based on semantic relevance. The module supports efficient batch processing and can run on both CPU and GPU, with GPU acceleration enhancing performance for large datasets.

  By providing robust semantic text processing, Sentence Transformers empowers VidBit to deliver context-aware, high-quality summarization and content insights, enhancing user interaction with video data.

- 4.1.6: **PySceneDetect** : In the VidBit project, the PySceneDetect module serves as a specialized tool for detecting scene changes in video content, enabling the project to segment videos into meaningful shots or scenes for further analysis or summarization. Built as a Python library, PySceneDetect analyzes video frames to identify transitions, such as cuts, fades, or dissolves, using algorithms like content-aware detection (based on pixel intensity changes) or threshold-based detection. Its core functionality includes splitting videos into discrete scenes, generating timestamps for scene boundaries, and exporting these segments for downstream processing, such as feeding specific clips into other VidBit components for transcription or semantic analysis. This capability is crucial for tasks like identifying key moments in a video, creating highlight reels, or organizing content for summarization by focusing on significant scene changes. Its precise scene detection enhances the project's ability to structure and process video content intelligently, making it an essential component for creating coherent and contextually relevant summaries or analyses.

- 4.1.7: **FFmpeg** : the FFmpeg module serves as a versatile, open-source multimedia processing tool that handles a wide range of video and audio manipulation tasks, significantly enhancing the project's ability to preprocess and prepare content for summarization and analysis.

  FFmpeg's core functionalities include extracting audio tracks from videos, converting between various video and audio formats (e.g., MP4 to WAV), trimming or splitting video clips, resizing or rescaling video resolutions, and applying filters like cropping or frame rate adjustments. It also supports muxing, demuxing, and encoding, enabling VidBit to standardize input files for compatibility with other modules or extract specific segments for processing.

  FFmpeg contributes by preparing video and audio inputs for tasks like transcription or scene detection, ensuring seamless integration with other components by providing clean, optimized media streams. For instance, it can extract high-quality audio for transcription or trim videos to focus on key scenes, streamlining the summarization pipeline. Its robust command-line interface allows for precise control over media processing, making FFmpeg indispensable for VidBit's ability to handle diverse video formats and deliver polished, structured content for downstream analysis and summarization workflows.

- 4.1.8: **MoviePy** : In the video summarizer project, the MoviePy module functions as a Python-based video editing library that streamlines the manipulation and processing of video and audio content, significantly enhancing the project's ability to create polished summaries and edited outputs. MoviePy's core functionalities include programmatic video editing tasks such as cutting, concatenating, and merging video clips, applying transitions (e.g., fades or slides), adding text overlays or subtitles, and adjusting video properties like resolution, frame rate, or aspect ratio. It also supports audio manipulation, such as syncing audio tracks or applying effects, and can render videos in various formats (e.g., MP4, AVI). Within the project, MoviePy contributes by enabling the creation of concise summary videos by stitching together key scenes (e.g., those identified by scene detection) or overlaying generated subtitles onto video segments. Its intuitive API allows for seamless integration with other components, facilitating tasks like producing highlight reels or exporting processed clips for user-facing outputs. By offering flexible, scriptable video editing, MoviePy empowers the project to deliver visually cohesive and professionally edited summaries tailored to the analyzed content.

5. RESULT

5.1 Qualitative Results :

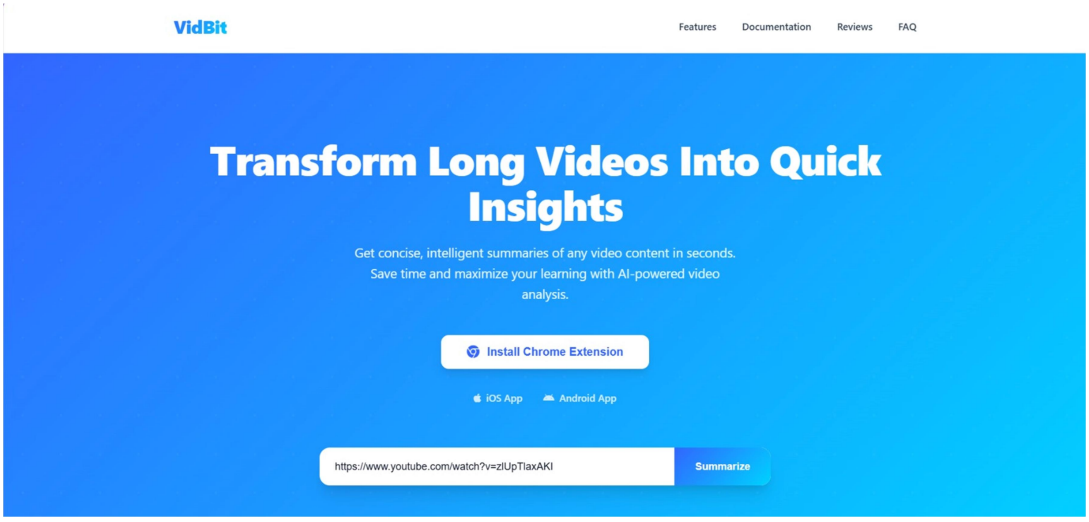

Fig 7 (i) : The home page of the VidBit – Video Summarizer AI

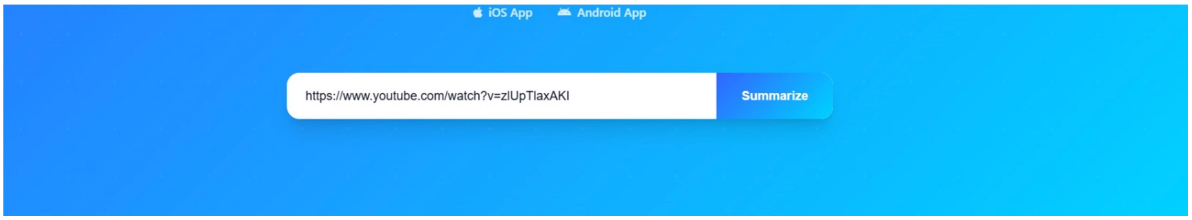

Fig 7 (ii) : Entering the Video link on the tab for summarization

Fig 7(i) and 7(ii) shows the home page of the VidBit – Video Summarizer AI webpage

RESULTS AFTER SUMMARIZATION :

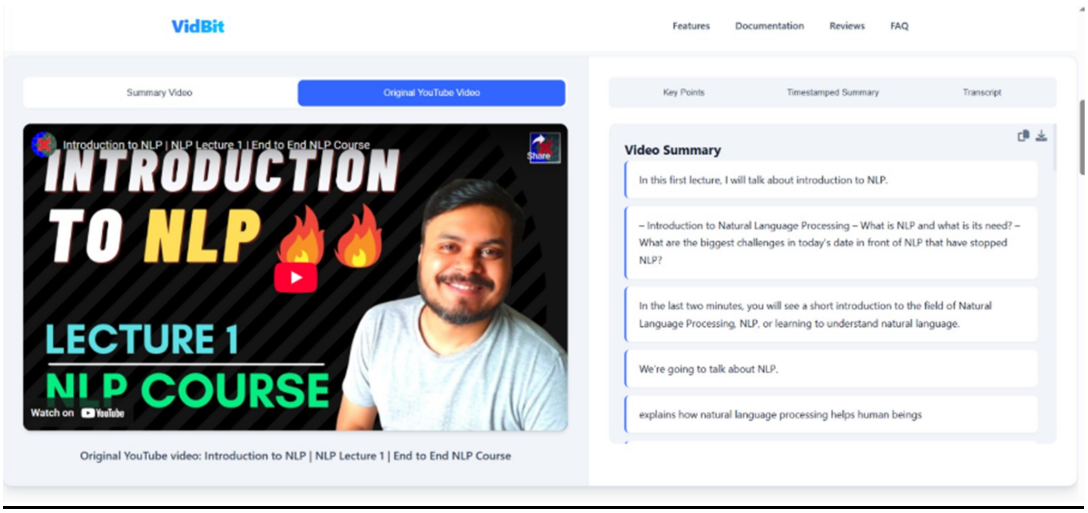

Fig 8(i) : The corresponding pointwise summary generated .

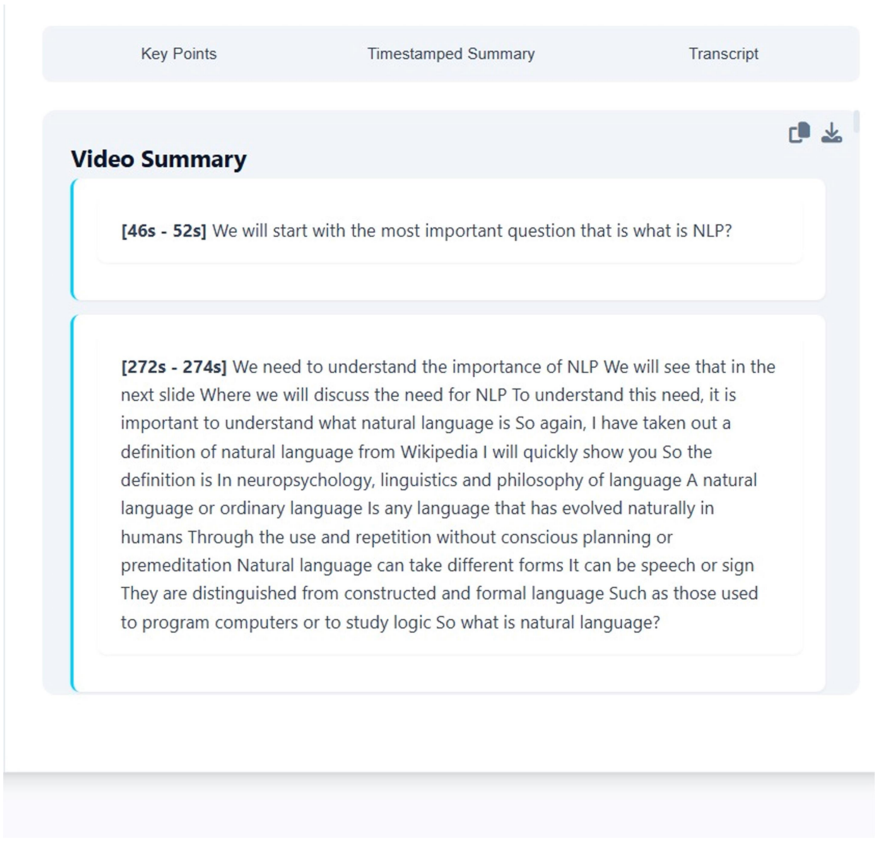

Fig 8(ii) : The text summary with time-stamps.

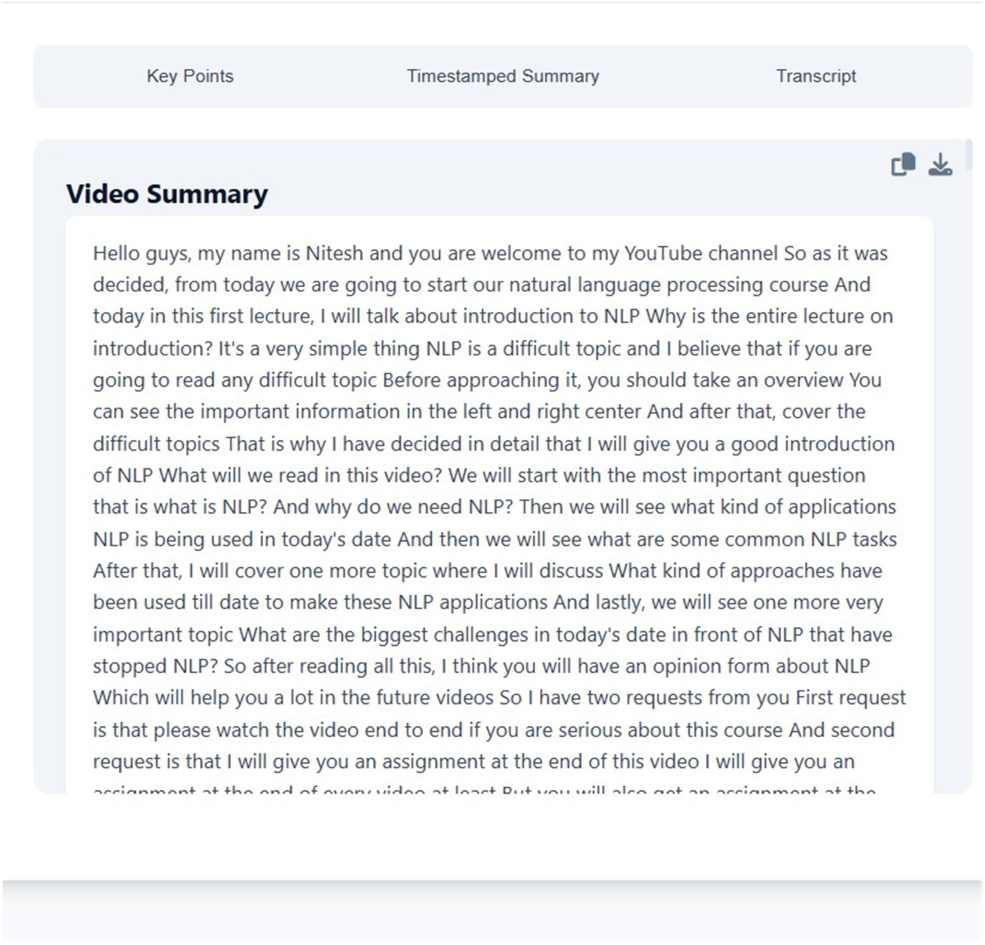

Fig 8(iii) : The whole text transcript for the entire content

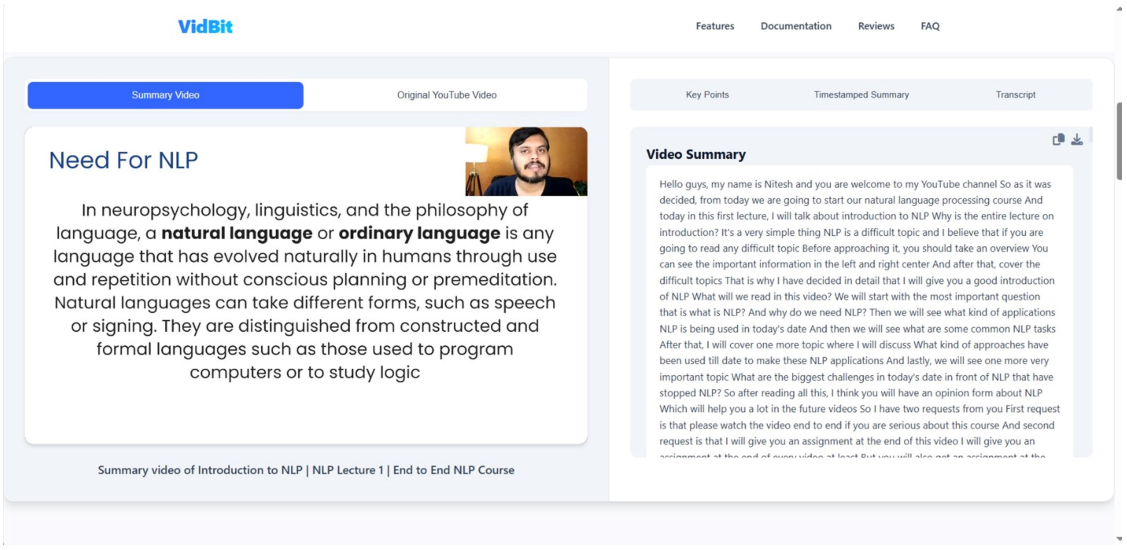

Fig 8(iv) : The clip generated from the original video containing the summary of the original content

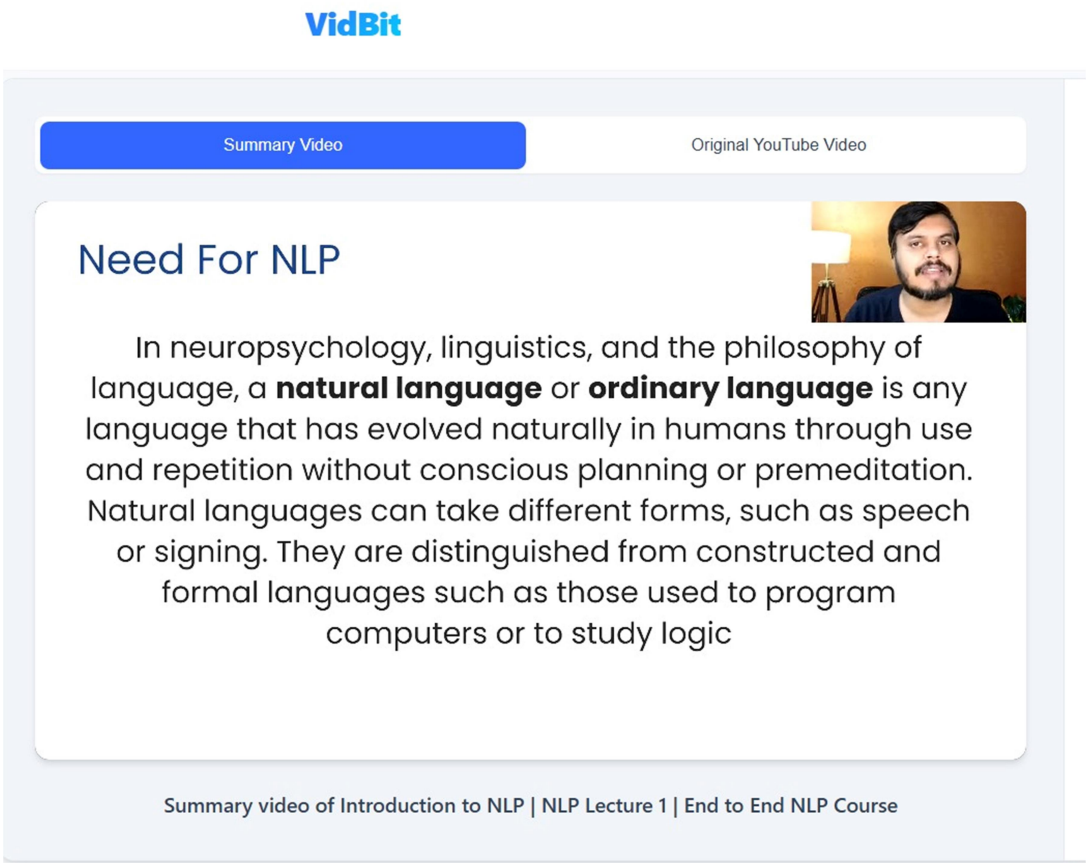

Fig 8 (v) : The detailed view of the summary video

[Fig 8(i) depicts the pointwise text summary generated highlighting the keypoints of the video 8(ii) illustrates 8(i) summary with time stamps. 8(iii) shows the transcript for the whole video content 8(iv) and 8(v) demonstrate the summary video clip generated from the original video]

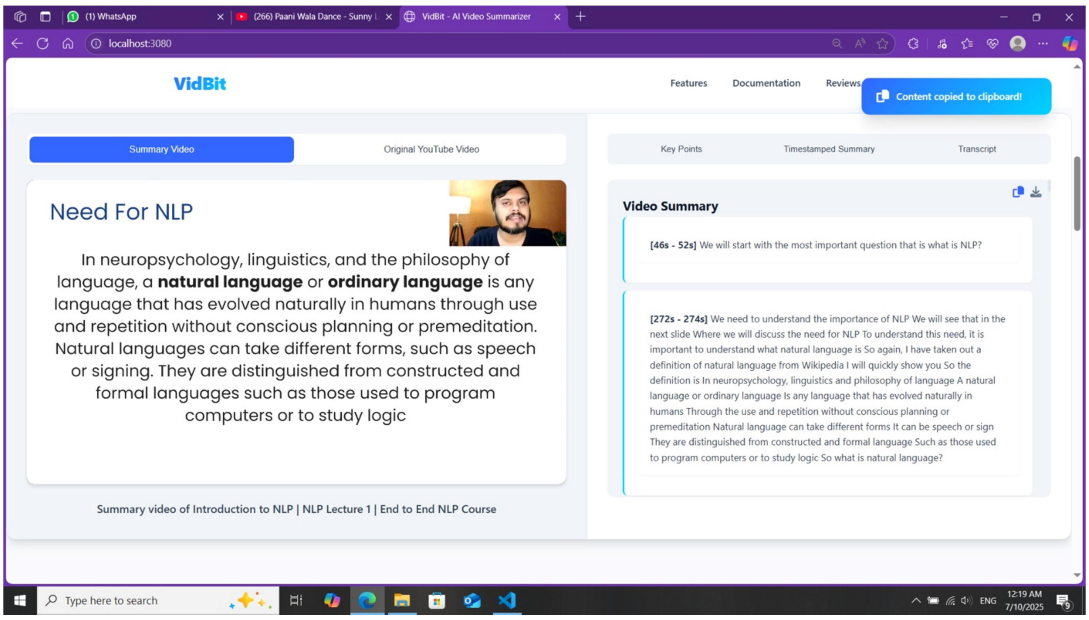

Fig 9(i) : shows that the summary points and the transcript are copied to the clipboard and can be stored locally.

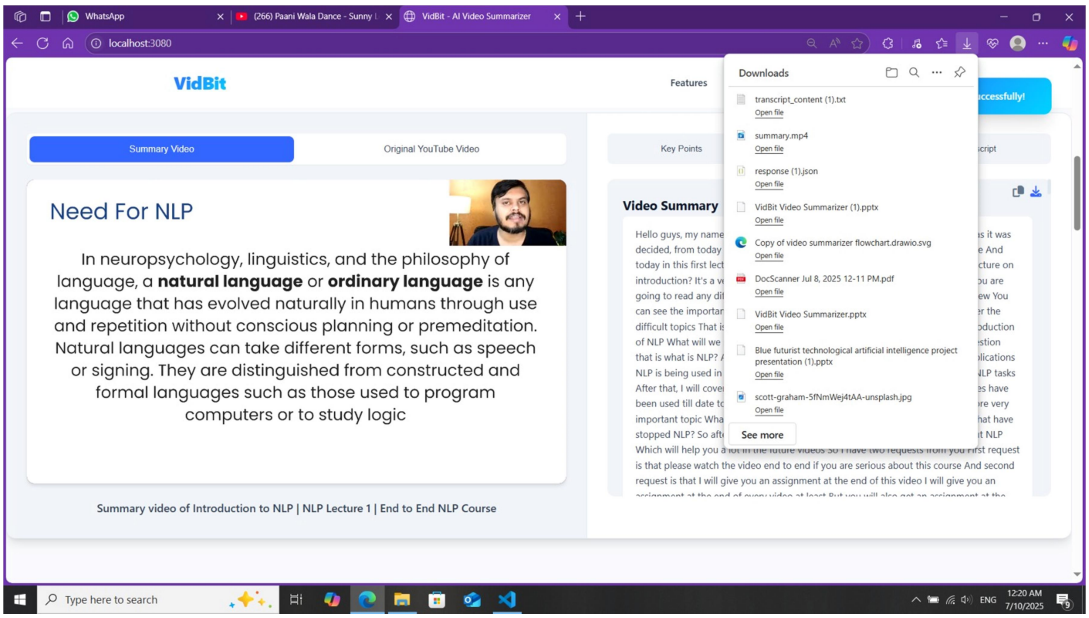

Fig 9(ii) : shows the summary video downloaded as an mp4 file and the transcript content saved as a txt file

5.2 QUANTITATIVE RESULTS :

ACCURACY OF THE SYSTEM :

5.2.1 The Whisper WER Error:

The Whisper WER (Word Error Rate) is a metric used to evaluate the accuracy of transcriptions produced by the Whisper automatic speech recognition (ASR) model, which is leveraged in the FastWhisper module of your video summarizer project. WER measures the difference between the transcribed text and a reference (ground truth) transcript by calculating the percentage of errors, including substitutions (replacing a correct word with an incorrect one), insertions (adding extra words), and deletions (missing words). It is computed as:

WER = (S + I + D) / N

Where ,

- S is number of substitutions
- I is number of insertions
- D is number of deletions
- N is total number of words in the reference transcript

A lower WER indicates higher transcription accuracy. For example, a WER of 10% means 10 out of 100 words were incorrect. In your project, Whisper's WER reflects how accurately FastWhisper transcribes video audio, impacting downstream tasks like summarization or subtitle generation. Factors affecting Whisper's WER include audio quality, background noise, accents, or model size (e.g., tiny vs. large), with larger models generally achieving lower WERs due to better performance but requiring more computational resources.

Dataset size	English WER (↓)	Multilingual WER (↓)	X→En BLEU (↑)
3405	30.5	92.4	0.2
6811	19.6	72.7	1.7
13621	14.4	56.6	7.9
27243	12.3	45.0	13.9
54486	10.9	36.4	19.2
681070	**9.9**	**29.2**	**24.8**

Fig 10 : WER error rate of Whisper Model

Performance metrics for the model across varying dataset sizes: English WER ranges from 9.9% to 30.5%, multilingual WER from 29.2% to 92.4%, and X-to-English BLEU from 0.2% to 24.8%. Larger datasets (up to 681070) show improved accuracy and translation quality.

5.2.2 Accuracy of the model :

	Train (hard_extend)	Evaluate(hard)	Evaluate(Self-designed datasets)
BERT	×	Loss: 0.69 Accuracy: 0.5	Loss: 0.69 Accuracy: 0.48
BERT + bidirectional LSTM	×	Loss: 0.69 Accuracy: 0.51	Loss: 0.69 Accuracy: 0.5
BERT + bidirectional LSTM(Fine tuning)	Loss: 0.51 Accuracy: 0.79	Loss: 0.66 Accuracy: 0.65	1Loss: 0.79 Accuracy: 0.52
all-MiniLM-L6-v2	×	Accuracy: 0.82 ROC AUC: 0.81 Precision: [0.5 0.74 1.] Recall: [1. 0.95 0.]	Accuracy: 0.44 ROC AUC: 0.44 Precision: [0.5 0.44 1.] Recall: [1. 0.47 0.]
all-MiniLM-L6-v2(Fine tuning)	Loss：0.66 Accuracy: 0.94	Loss：0.67 Accuracy: 0.82 ROC AUC: 0.89 Precision: [0.55 0.91 1.] Recall: [1. 0.91 0.]	Loss：0.74 Accuracy: 0.44 ROC AUC: 0.76 Precision: [0.55 0.87 1.] Recall: [1. 0.64 0.]

Fig 11 : all-MiniLM-L6-v2 accuracy achieves the highest accuracy (0.94) demonstrating its effectiveness for text processing tasks.

6 . LIMITATIONS AND FUTURE SCOPE

The VidBit project, designed to process and summarize video content, faces several limitations that impact its current capabilities and reliability.

One significant constraint is the dependency on the quality and diversity of input data, where poor audio clarity, varied accents, or incomplete video metadata can lead to inaccurate transcriptions and summaries, affecting overall performance.

Additionally, the system's reliance on cloud-based resources means it is subject to external limitations such as runtime restrictions, potential resource unavailability, or bandwidth issues, which can disrupt processing for large or complex video files.

Scalability poses another challenge, as handling a high volume of videos or real-time processing may strain the available computational resources, potentially leading to delays or incomplete results.

Moreover, the accuracy of the output is influenced by the training data of the underlying technologies, which may not fully account for niche languages, dialects, or evolving content trends, introducing biases or gaps in understanding.

Looking to the future, the scope of VidBit is promising, with opportunities to enhance its adaptability by incorporating more diverse datasets to improve multilingual support and handle varied audio-visual conditions. Future iterations could explore integration with advanced real-time processing techniques to enable live summarization, catering to dynamic content creation environments.

There is also potential to expand its functionality to include personalized summarization based on user preferences, making it more tailored and user-centric. Furthermore, leveraging translation system to transcript and summarise videos from a given language to another language and also seeing the transcript and summary in different languages.

7. CONCLUSION

The VidBit project has been a remarkable journey, marked by both innovation and challenge. This initiative has given rise to a powerful tool that redefines how we engage with video content, delivering concise and meaningful summaries with impressive accuracy, despite the complexities of varied data quality and limited resources.

Developed with a strong foundation in cloud-based processing, VidBit demonstrates significant potential for growth.

As it continues to evolve, this AI-driven video summarizer is poised to play a pivotal role in the digital media landscape, streamlining content consumption for educators, creators, and audiences alike, and unlocking new possibilities for information access and engagement.

8. REFERENCES

[1] NLP-Driven Video Transcription: A Comprehensive Survey and Innovative Office Meeting Automation , **Adithya, T., et al, "Multi Lingual Video Summarizer using Natural Language Processing," in *2024 5th International Conference for Emerging Technology (INCET)*, 2024.**

[2] Video Based Transcript **Summarizer** for Online Courses using **Natural Language Processing, K. Kulkarni, R. Padaki, "Video Based Transcript Summarizer for Online Courses using Natural Language Processing," in *2021 IEEE International Conference on Computation System and Information Technology for Sustainable Solutions (CSITSS)*, 2021, pp. 1–5.**

[3] Video transcript **summarizer**, A. N. S. S. Vybhavi, L. V. Saroja, J. Duvvuru and J. Bayana, "Video Transcript Summarizer," *2022 International Mobile and Embedded Technology Conference (MECON)*, Noida, India, 2022, pp. 461-465, doi: 10.1109/MECON53876.2022.9751991.

[4] P. Nagaraj, V. Muneeswaran., B. Rohith, B. Sai Vasanth, G. Veda Varshith Reddy and A. Koushik Teja, "Automated Youtube Video Transcription To Summarized Text Using Natural Language Processing," *2023 International Conference on Computer Communication and Informatics (ICCCI)*, Coimbatore, India, 2023, pp. 1-6, doi: 10.1109/ICCCI56745.2023.10128375.

[5] Alqurni, J.S., Alsmadi, M.K., Alfagham, H. *et al.* Streamlining Video Summarization with NLP: Techniques, Implementation, and Future Direction. *SN COMPUT. SCI.* 6, 110 (2025). https://doi.org/10.1007/s42979-024-03591-w

[6]E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris and I. Patras, "Video Summarization Using Deep Neural Networks: A Survey," in *Proceedings of the IEEE*, vol. 109, no. 11, pp. 1838-1863, Nov. 2021, doi: 10.1109/JPROC.2021.3117472.]

[7] M. Rochan, L. Ye, Y. Wang, "Video summarization using fully convolutional sequence networks," in *Proceedings of the European conference on computer vision (ECCV)*, 2018, pp. 347–363.

[8] Denil, M., et al. "Modelling, visualising and summarising documents with a single convolutional neural network," in *arXiv preprint arXiv:1406.3830*, 2014.

[9]Video Summarization using Speech Recognition and Text Summarization
T Tyagi, L Dhari, Y Nigam, R Nagpal

2023 4th International Conference for Emerging Technology (INCET), 2023

[10] M. Narasimhan, A. Rohrbach, T. Darrell. "Clip-it! language-guided video summarization," in *Advances in neural information processing systems*, vol. 34, pp. 13988–14000, 2021.

[11] Ghosh, A., et al, "Clipsyntel: clip and llm synergy for multimodal question summarization in healthcare," in *Proceedings of the AAAI Conference on Artificial Intelligence*, 2024.

[12] P. Mundur, Y. Rao, Y. Yesha. "Keyframe-based video summarization using delaunay clustering," in *International journal on digital libraries*, vol. 6, pp. 219–232, 2006.

[13] K. Mahmoud, M. Ismail, N. Ghanem, "Vscan: an enhanced video summarization using density-based spatial clustering," in *Image Analysis and Processing–ICIAP 2013: 17th International Conference, Naples, Italy, September 9-13, 2013. Proceedings, Part I 17*, 2013, pp. 733–742.

[14] Zhang, Y., et al. "Unsupervised object-level video summarization with online motion auto-encoder," in *Pattern Recognition Letters*, vol. 130, pp. 376–385, 2020.

[15] Nair, M., & Mohan, J. (2021). Static video summarization using multi-CNN with sparse autoencoder and random forest classifier*. Signal, Image and Video Processing, 15(4), 735–742.*

[16]Attention Is All You Need

Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N. Gomez, Lukasz Kaiser, Illia Polosukhin