



Dissertation on

“Video Summarization”

Submitted in partial fulfilment of the requirements for the award of degree of

**Bachelor of Technology
in
Computer Science & Engineering**

UE18CS390A – Capstone Project Phase -2

Submitted by:

**Durjoy Ghosh
Shreya Deepak
Surabhi S
Vrushabh A Chougale**

**PES1201802124
PES1201801099
PES1201800849
PES1201801495**

Under the guidance of

Dr. Uma D
Professor
PES University

June - December 2021

**DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING
FACULTY OF ENGINEERING
PES UNIVERSITY**

(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India



PES UNIVERSITY
(Established under Karnataka Act No. 16 of 2013)
100ft Ring Road, Bengaluru – 560 085, Karnataka, India

FACULTY OF ENGINEERING

CERTIFICATE

This is to certify that the dissertation entitled

‘Video Summarization’

is a bonafide work carried out by

Durjoy Ghosh
Shreya Deepak
Surabhi S
Vrushabh A Chougale

PES1201802124
PES1201801099
PES1201800849
PES1201801495

In complete fulfilment for the completion of seventh semester Capstone Project Phase - 2 (UE18CS390A) in the Program of Study - Bachelor of Technology in Computer Science and Engineering under rules and regulations of PES University, Bengaluru during the period June. 2021 – December. 2021. It is certified that all corrections / suggestions indicated for internal assessment have been incorporated in the report. The dissertation has been approved as it satisfies the 7th semester academic requirements in respect of project work.

Signature
Dr. Uma D
Professor

Signature
Dr. Shylaja S S
Chairperson

Signature
Dr. B K Keshavan
Dean of Faculty

External Viva

Name of the Examiners

Signature with Date

1. _____

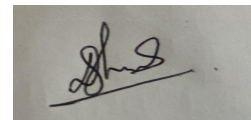
2. _____

DECLARATION

We hereby declare that the Capstone Project Phase - 1 entitled “**Video Summarization**” has been carried out by us under the guidance of Dr. Uma D, Profesor and submitted in partial fulfilment of the course requirements for the award of degree of **Bachelor of Technology in Computer Science and Engineering of PES University, Bengaluru** during the academic semester January – May 2021. The matter embodied in this report has not been submitted to any other university or institution for the award of any degree.

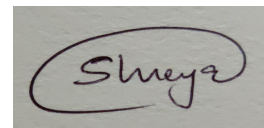
DURJOY GHOSH

PES1201802124



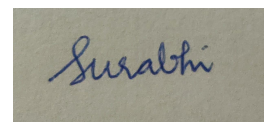
SHREYA DEEPAK

PES1201801099



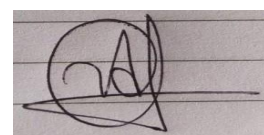
SURABHI S

PES1201800849



VRUSHABH A C

PES1201801495



ACKNOWLEDGEMENT

We would like to express our gratitude to Dr. Uma D, Department of Computer Science and Engineering, PES University, for her continuous guidance, assistance, and encouragement throughout the development of this UE18CS390A - Capstone Project Phase – 1.

We are grateful to the project coordinators, Prof. Kusuma KV and Prof. Sunitha R, for organising, managing, and helping with the entire process.

We take this opportunity to thank Dr. Shylaja S S, Chairperson, Department of Computer Science and Engineering, PES University, for all the knowledge and support we have received from the department. We would like to thank Dr. B.K. Keshavan, Dean of Faculty, PES University for his help.

We are deeply grateful to Dr. M. R. Doreswamy, Chancellor, PES University, Prof. Jawahar Doreswamy, Pro Chancellor – PES University, Dr. Suryaprasad J, Vice-Chancellor, PES University for providing to us various opportunities and enlightenment every step of the way. We are deeply grateful for the support from the bank employees and customers who have given us an opportunity to discuss their concerns and experience with us. Finally, this project could not have been completed without the continual support and encouragement I have received from my family and friends.

ABSTRACT

Video summarization is the process of distilling a raw video into a more compact form like video, audio, text, etc. without losing any information. Now, lecture videos have been made available to the students making it easier for them to go through the topics taught in class. But going through the whole video for one topic, when you just want to brush up the key points, is time-consuming. We have tried to make it easier for students by summarising the video in the form of slides along with the key points taught so that students can go through them and if they want an explanation of the topic they can go through the video clip associated with that slide. With the help of shot detection, we are taking a screenshot of the lecture right before there is a drastic change in the lecture. We have tried this method on short videos as well as long videos and have received good results. We have also included a search engine for topic-based searching of slides or video clips.

TABLE OF CONTENTS

1.	INTRODUCTION	10
2.	PROBLEM STATEMENT	10
3.	LITERATURE REVIEW	10
	3.1. LECTURE VIDEO SUMMARIZATION USING SUBTITLES	10
	3.2. WHITEBOARD VIDEO SUMMARIZATION	10
	3.3. AUTOMATED SUMMARIZATION OF VIDEO LECTURES	10
	3.4. AUTO SUMMARIZATION OF AUTO-PRESENTATION	10
	3.5. IMPROVED ALGORITHM FOR VIDEO SUMMARIZATION	10
4.	PROJECT REQUIREMENT SPECIFICATION	10
	4.1. INTRODUCTION	10
	4.2. PRODUCT PERSPECTIVE	10
	4.3. FUNCTIONAL REQUIREMENTS	10
	4.4. EXTERNAL INTERFACE REQUIREMENTS	10
	4.5. NON-FUNCTIONAL REQUIREMENTS	10
5.	SYSTEM DESIGN	10
	5.1. INTRODUCTION	10
	5.2. CURRENT SYSTEM	10
	5.3. DESIGN CONSIDERATION	10
	5.4. HIGH LEVEL SYSTEM DESIGN	10
	5.5. DESIGN DESCRIPTION	10
	5.6. USER INTERFACE DIAGRAM	10
	5.7. SWIMLANE DIAGRAM	10
6.	IMPLEMENTATION AND CODE	10
7.	CONCLUSION	10

APPENDIX A : DEFINITION, ACRONYM AND ABBREVIATION

APPENDIX B : REFERENCES

APPENDIX C : RECORD OF CHANGE OF HISTORY

LIST OF FIGURES

Fig 0:	System Architecture
Fig 1 :	Video Splits
Fig 2 :	Extracting Images
Fig 3 :	Audio Splits
Fig 4 :	Speech Text
Fig 5 :	Text to Keyphrases
Fig 6 :	User Interface Diagram
Fig 7 :	Masterclass Diagram
Fig 8 :	User Interface Diagram 1
Fig 9 :	User Interface Diagram 2
Fig 10 :	User Interface Diagram 3
Fig 11 :	Splitting the video 1
Fig 12 :	Splitting the video 2
Fig 13 :	Video Splits
Fig 14 :	Producing Summary
Fig 15 :	API call
Fig 16 :	JSON to Raw text
Fig 17 :	Landing Page
Fig 18 :	Slides
Fig 19 :	Viewing the Full Video
Fig 20 :	Video Clip
Fig 21 :	Search Option

LIST OF TABLES

Table 1 : Record of change history

1. INTRODUCTION

Video summarization is a process of distilling a raw video into a more compact form like video, audio, text, etc. without losing any information. This can be done by selecting informative frames of the video and capturing relevant information.

Video can be summarised by subset selection: Key frame selection and Key Shot selection. Key frames selection is a method where isolated frames are selected and Key shots selection is a temporally continuous interval based selection. The expected outcomes of the video summarization should contain high priority elements and things from the video and the video summary should be free of repetition and redundancy.

There are two methods for video summarization - unsupervised and supervised methods. In unsupervised video summarization heuristic approaches are used to select the key frames. The methods used in unsupervised video summarization are key frame extraction and video skimming. Key frame extraction is a process where a set of frames are extracted from a video. Video Skimming is a method which makes the summarised video more continuous and decreases the number of abrupt changes of frames and cuts. In supervised both raw video and its summarised video is given for learning. The disadvantages of using supervised learning in summarization is the shortage of the summarised video dataset. To summarise a video, one has to sit through the whole video and select key frames from that video, which is a very time consuming process. The main aim is to produce a summarised video by selecting important parts of the video which represents the whole video.

Some of the applications of video summarization are Video Database Management, Consumer Video Analysis, CCTV Surveillance Video Summary, Sports Video Highlights Generation, Lecture Video Summarization and many more. The application that we are focusing on is lecture video summarization.

One of the most important aspects of e-learning is the digitization of the scholarly articles which includes recording lecture videos. In recent times there has been a surge in the amount of lecture videos being recorded by the universities with the pandemic being one of the

driving forces. These lecture videos enable the students to use the videos whenever they want to study a particular topic. Since the lecture videos are lengthy, students face difficulty going through the videos when they lack time or want to make a quick revision. The videos might contain content which is not relevant to the topic. The users will have to go through the entire video to search for the required sub topics which is time consuming. Quick revisions during the exams is also not possible with the whole video in place as it contains many things which are not so important. This project aims to summarize the lecture videos' content to save time for the user. This helps the students with their quick revision during exams, find particular subtopics of interest in the video and save ample time.

2. PROBLEM STATEMENT

The main objective of our system is to summarise the content of the lecture videos thereby helping the user to browse through the video easily without having to search for important subtopics thus saving the user's time. The system can be given any lecture video as the input where the audio is extracted from the video and converted to text and then text summarization is done. The timeline of the summarised text will be matched with the important snapshots taken.

The above method helps the user to go through the important subtopics in the summarised video easily without having to go through the entire video. The summarised video also helps in quick revision for the users.

3. LITERATURE REVIEW

3.1 Lecture Video Summarization Using Subtitles [1]

- Ramamohan Kashyap Abhilash, Choudhary Anurag, Vaka Avinash and D.Uma

In this paper, first, the content of the elements is extracted from the video and these frames are mixed together to produce the summarised video. Text summarization is done by audio extraction from the lecture video and subtitles are added. Then, the summary of the video is obtained. Here, four modules are used to obtain the summarised video - conversion module, preprocessing module, text summarization module and video summarization module.

The use of the conversion module is to generate subtitles to the video. Here, from the video, audio is extracted and is processed. The speech obtained is converted to text by using Google speech to text conversion. Punctuations are added to the subtitles by noting down the difference in time between the utterances of the speaker. The output obtained is a file having subtitles which is used as the input in the preprocessing module. The preprocessing module involves removal of special characters from the subtitles and the tokenization of subtitles takes place to form sentences. Also, the time stamps are extracted from the subtitles. In the text summarization module, the vocabulary set is built and TFIDF (Term Frequency-Inverse Document Frequency) scores were calculated for the words.

The TFIDF scores for the sentences are calculated by adding some amount of weightage to the keywords. The text summary is generated by using the results obtained from these scores. The last module uses the timestamps of sentences in the text and corresponding sub clips are created to generate a summarised video. The evaluation of the summarised video is done by using ROGUE (Recall-Oriented Under-257 study for Gisting Evaluation) toolkit. It is observed that the scores obtained with punctuations are higher than the scores obtained without punctuations, thereby increasing the efficiency of the summarised video. Some improvements can be made in the proposed method to increase its efficiency when punctuations are used. This can be done with the help of binary classification of the pauses between the utterances of the user and the parts of the video that can be removed. This results in a smooth transition of video frames and a meaningful summarised video can be obtained.

3.2 Whiteboard Video Summarization Via Spatio-Temporal Conflict Minimization [2]

- K. Davila and R. Zanibbi

This paper focuses on the extraction and summarization method of the content which is handwritten. The method proposed in the paper is divide-and-conquer segmentation which is performed where the regions of conflict are analysed with content at a larger scale. The lecture video is split recursively into parts containing fewer or zero conflicts. The components which are connected to each other are said to be in conflict with each other if overlapping occurs in the space occupied by them present in the whiteboard but it exists at different time intervals. This occurs when the content on the whiteboard is erased and new content is written in the same area. The method mentioned in the paper allows video summarization by using fewer keyframes. A keyframe is capable of combining different parts of the content which was never present on the whiteboard at the same time but might have occupied other areas and can be displayed as a single image to give better compression rates thereby delivering shorter summaries.

The lecture videos are summarized by using lesser frames which are based on reducing conflicts between regions having content. A spatio-temporal index is used to navigate through lecture videos based on the content written on the whiteboard. The dataset consists of lecture videos that are labeled and are used for training and testing of latest approaches for the above method or problems related to this method. The efficiency of the method is evaluated by using four baselines - binary, whiteboard segmentation, ground truth based whiteboard segmentation, and reconstructed. Whiteboard segmentation is a method used to identify the whiteboard region and involves removal of connected components that are not present in this region.

The connected components are matched between ground truth and summary key-frames having time intervals that overlap each other for each summary. For each pair of frames, a translated summary of the key-frame is obtained which increases pixel-wise recall and then evaluation of pixel-wise recall and precision of overlapping connected components is performed. The metrics for global recall metrics are calculated by considering different

connected components present in the summaries. The global precision is calculated by considering all the components present in the frames. Finally, we observe that the binary frames obtained the highest recall but had a lesser precision because of all the non-content connected components.

In conclusion, the method proposed in the paper for binary frames reconstruction and the procedures for background removal are efficient when the precision of content extracted from lecture videos is increased after binarization.

The limitation in the paper is that the proposed method is not efficient when the lectures are recorded using multiple cameras. To solve this issue, more efficient methods are required for automatic fine tuning of the parameters.

3.3 Automated Summarization of Lecture Videos [3]

- Anusha Vimalaksha Abhijit Prekash Siddarth Vinay and NS Kumar

In this paper, they have tried to design a tool that tackles problems such as storage archiving large numbers of videos for a prolonged time, navigating among the different sub topics taught in class and helping the students by summarising the videos so as to give the essential information without watching the whole lecture. They planned to design it such that when a lecture video is given as input it would split it into multiple parts, i.e. each video consists of a single topic, wr.t the times provided and makes sure that the video does not split in the middle of a word and reduce the total length of the video by the process of transcription and summarization.

They thought of three methods to split the videos into concise parts. One is to collect data about the start and end times of a topic through crowdsourcing where students could record it when attending classes and these statistics could be turned into raw data which when fed to the program would provide an output. Second was to acquire a device which would be in the teachers' control that produces a high frequency sound that is inaudible to the human ear that indicates the end of a topic. And the final idea was to introduce a black screen that would

obstruct the camera's view for a short period of time to indicate the end of the topic. However the second and third ideas were not feasible and impractical respectively and they decided to go along with the first idea that is crowdsourcing.

They used multiple functions in the open source Python Library Pydub such as AudioSegment and Silence modules to improve the accuracy and efficiency in splitting the videos. They then compressed the obtained parts of the video by removing prolonged periods of unnecessary silences and limited the lectures to its key points in three stages : transcription of the video, summarization of the same and mapping the summary to the original video. They have done this by using an online transcription service called ListenByCode for transcription, the python library Sumy for summarization and for mapping they have used the time stamps provided.

Hence this tool performs two main tasks that is splitting the lecture videos and then summarising it. They found two limitations in this tool: one is topic overlapping and the other is that some of the important points in the original video are cut out in the summary video.

3.4 Auto Summarization of Audio-Video Presentations [4]

- Liwei He, Elizabeth Sanocki, Anoop Gupta, and Jonathan Grudin

In this paper, they talk about how digital media content is becoming pervasive around the world. And how even the companies are uploading videos of internal seminars on the net and academies are recording the lectures so that the students can view it whenever they want by uploading it on their personalised websites or on some other site. They discuss the different sources that can be used to obtain automated summaries. They are namely: video channel, audio channel, speaker's actions and end-users' actions. In their project they use only three out of the four sources : the audio channel, speaker's actions and the end-users' actions. There are two major types of sources in the audio channel that are the spoken text and the pitch, intensity, pause and other prosody information in the audio. They use the pitch activity of the speaker as one of the ways to summarise the video based on how the pitch changes when a new topic starts or when the speaker emphasises on important points.

Summarization based on the speaker's actions is using their hand gestures, slide changes or how long they stay on a particular slide where the longer they stay on a particular slide emphasizes the importance of the slide. Summarization based on the end-users' actions is based on user count for a particular part of the video where the video is divided by change in slides. The increase or drop in user count determines the importance of the slide and this is calculated by finding the difference between the average user count on the current slide and the average user count on the previous slide. They try to make sure that the summaries meet the following four properties, the 4 C's, namely – Conciseness, Coverage, Context and Coherence.

They used three algorithms to summarise the videos, the first uses information in slide transition information only, the second algorithm tries to detect emphasised regions in speech by pitch activity analysis and the third algorithm (SPU) uses all three sources of information i.e., the slide transitions, pitch activity, and user-access patterns.

The slide-transition-based summary uses the heuristics that the slide transitions indicate a change in the topic, and that relative time spent by the speaker on a slide indicates its relative importance. The pitch-based summary uses the algorithm proposed by Arons to identify emphasis in speech. SPU combines the information sources used by the above two algorithms and combines information from user-access patterns. The heuristics used in SPU is if there is a surge in the user-count of a slide relative to the previous slide, it likely indicates a topic transition. Also, the slide doesn't become interesting if the user counts for that slide falls. They then asked the speaker of that video to mark the important points and summarize the lecture and create a small quiz based on that particular lecture. They chose four videos based on different topics and performed one summarization method on one video for the four different summarization algorithms that are author-based, slide-based, pitch-based and SPU.

They then decided to test this with human subjects and selected a few people who don't know anything about the topic of the videos, or that the videos were summarised using computer algorithms, and asked them to answer the quiz set by the speaker and then made them watch the four summarised videos. Then they made them answer the quiz again. They then compared the quiz before they watched the summarised video and the quiz after they watched the summarised video and also compared the results of the quiz of the different videos. They

noticed that the author based summarised video quiz fared better than the other three methods but they were not far behind. The subjects were surprised when they came to know that the videos were summarised based on computer algorithms. Based on the properties of clarity, conciseness, and coherence, the computer-based methods were rated particularly poor on coherence. The subjects complained that the summaries “jumped topics” and were “frustrating because it would cut out or start in the middle [of a topic],” and so forth.

They concluded that they didn’t find any major differences between users’ preferences for the three computer based methods that lead them to believe that the simpler methods (slide-based and pitch-based) may be preferable for now. They also saw that users’ tolerance to computer generated summaries increased. And they deduced that the user community might be ready to accept the current generation summarization methods even though there are many improvements to be made.

3.5 An Improved Algorithm for Video Summarization – A Rank Based [5]

- Manasa Srinivas, M. M. Manohara Pai and Radhika M. Pai

In this paper, the video summary is obtained by selecting important keyframes from the video by the process of video skimming. The keyframes are extracted by identifying the change in the video, clustering of low level features and objects. The keyframes are chosen in such a way that they represent the entire video and can provide the summary with just a few frames.

The proposed method is divided into three steps for selecting keyframes. In the first step, the scores are calculated for each frame. In the second step, the scores are combined and based on the outcome, keyframes are selected. In the third stage, the duplicate frames are removed. The scores are calculated by considering the following factors- quality, static attention, temporal attention and representativeness.

The quality of the video depends on the quality of each independent frame present in the video and the quality score is calculated by using measures like colorfulness, brightness, edge distribution, hue count and contrast. The method proposed to calculate colorfulness is called histogram intersection. In this method, histogram is built for the original image and

hypothetical image for every RGB channel. The histograms built have been normalised to unit length. Also, monotonic mapping is calculated between the two histograms by using the proposed method. The brightness of the image is calculated by converting the image into HSV color space and computing the average value score. Later, edge distribution, hue count and contrast of the image is calculated.

Static attention is a method of choosing the salient region present in the frames and the frames having a high saliency value is chosen while selecting the keyframes. Region based contrast and image signature are the methods used to identify the salient region in the frames and the average score is calculated for these methods and the value obtained becomes the user's attention to the frame. We also observe that the pixel values change in the adjacent frames which are used to calculate motion information throughout the frames known as temporal attention. The method used to select the most important frames is called k-medoids clustering. Representativeness score is calculated by the method mentioned and is applied to the frames representing the video, also known as medoid points. Uniformity is a feature that is used to ensure that there is a uniform motion in the video without any jumps.

The algorithm proposed in the paper consists of two methods- keyframes selection and removal of duplicate frames. In keyframe extraction, scores are calculated for every individual frame by the methods mentioned above and weights are assigned to the frames based on their importance. Standard deviation of these frames are calculated and the frames having higher standard deviation are chosen and an average sum of these scores are calculated. Also only the top fifty percent of the frames are selected in order to avoid bad frames. If there are two identical frames then the frames having a higher score is selected and the duplicate frame having lower score is eliminated.

The proposed method is evaluated by calculating precision, recall and f-measure. It is observed that the keyframes selected by using the methods mentioned above include almost all the frames present in the ground truth when compared to other methods. The representativeness feature had a better outcome among other features. Also the f-measure computed by the proposed method was better than the value obtained by comparing with other methods.

4. PROJECT REQUIREMENTS SPECIFICATIONS

4.1 Introduction

4.1.1. Project Scope

We are developing a system for video summarization. This system can be utilised for summarizing video lectures. Since lecture videos are pretty lengthy, it is very difficult for the students to surf through the videos to search a particular topic. And also quick revision during the exams is also not possible with the whole video in place as it contains many things which are not so important. The summarization technique can help the students to find a particular topic in the whole video very easily. Since we are summarizing the whole video into a slide show with the most important snapshots of the lecture helps the students with a quick revision. This system can be given any lecture video, which it extracts the audio from and converts it to text and then summarises the text. The timeline of the summarised text will be matched with the important snapshots taken.

Some of the limitations of the system would be improper videos with a lot of disturbance. This could result in decreased efficiency. And if any topics other than the syllabus discussed in the video can also pose a threat to the system, as it will fail to summarise it along with everything else discussed in the class.

4.2 Product Perspective

Nowadays there is a huge amount of data produced in the form of videos. It is not possible to go through/monitor all videos for a human. That is why summarization of videos is required. There have been trials of video summarization on CCTV footage. Going through hours of CCTV footage would be a tiresome job, hence the summarization of the same can be helpful. We can keep track of the frames where

there has been maximum change or disturbance. This might help us in finding what we want without having to go through the whole footage.

Apart from CCTV footage video summarization techniques have also been employed in generating the highlights of a sports game automatically. This summarization takes into account the audio from the viewers of the game. Whenever there is a sudden change in the sound of the collective audience indicates that something important might have happened in the game and doing so we can select the particular frame. With some of such continuous frames highlights of a particular sports game can be generated automatically.

4.2.1. Product Features

The Video Summarization software will allow the users to get a summarized version of a video lecture in the form of a slideshow. The slideshows will contain snapshots of relevant frames and summarized text of the lecture. The users can also navigate through the slides with the help of keywords.

4.2.2. Operating Environment

For Desktops:

- RAM: 2 GB
- Internet / LAN connection
- Operating System: Windows/Ubuntu
- Keyboard and Mouse
- Space: 80 GB Hard Drive
- Core: Pentium IV MHZ or more
- Cache: 512kb
- Necessary Python Modules
- Necessary APIs

4.2.3. General Constraints, Assumptions and Dependencies

A few things assumed are as follows:

- The lecture video is in the English language spoken in a fluent manner.
- The audio output is clear.
- There is an active internet connection
- Since the application uses a web interface, it is assumed that the user has a web browser.

4.2.4. Risks

The Video Summarization software will take into account the most salient topics into consideration, overlooking a few topics which may be important to the user. The system might produce bad results if the input video is not a lecture video.

4.3. Functional Requirements

Validity tests on inputs:

1. A valid input to the system would be a video. If the input is not in the video format, then it can not be processed and hence it will be an invalid input.
2. Even if the algorithm generates the output for any type of input video, the output may not be proper, or it may contain major chunks of the video which are of no interest to us.

The basic sequence of operation would be as follows:

1. A video lecture will be taken as the input for pre-processing. In this step the video recorded before the actual class started and video recorded after the class ended will be trimmed off. In this way an initial stage of pre-processing can be done.
2. Once the preprocessing is done then the audio from the video will be extracted which will be converted to text format for summarization. Techniques of text summarization will be applied on the text generated from the audio.
3. Apart from this, snapshots from the video will be taken from the video. Snapshots based on the frame difference will be taken. These snapshots taken from the video and the summarized text will be used to output the slideshow of the summary generated.

Error handling and recovery:

1. Since each step in the operation is interdependent on each other we can not proceed from one step to another if there is any error in the previous step.
2. Hence if there is any error in the text summarization step, then we can not move forward as this error might be magnified later in the process.
3. The output of the text summarization and the snapshots taken from the video are equally important to the final output, therefore any error in the process should be handled then and there only.

4.4. External Interface Requirements

4.4.1. User Interfaces

The user interface will include a simple web design. The user can upload videos/text of lectures as an input. The backend software aims to give a ppt slideshow as an output to the user. The user can navigate through the slides and search for a particular topic inside the slides by typing a keyword in the search bar.

4.4.2. Hardware Requirements

For Desktops:

RAM: 1 GB

Space: 80 GB Hard Drive

Core: Pentium IV MHZ or more

Cache: 512kb

4.4.3. Software Requirements

Operating System: Windows 10/8/7 (including 64-bit).

Python 3.5.1 or higher.

Internet Browser supporting JavaScript and Adobe Flash.

Audio Extractor : To extract the audio from the input video.

Text Converter : To convert the audio to the text we will need this software.

Summarizer : To summarize the text we will need the text summarizer.

4.4.4. Communication Interfaces

Since the user has to upload the video to the web, the system will use FTP. The upload of the video does not require any type of streaming and all, therefore FTP would be the preferred protocol.

4.5. Non-Functional Requirements

4.5.1. Performance Requirement

With a well formatted video lecture with minimum disturbance the system should be able to perform decently. A well formatted video should be converted to well formatted ppt with important snapshots from the video. The system should be able to leave out the parts of the video which are not important.

4.5.2. Safety Requirements

The user must make sure that the video he/she is using is a free-to-use/licensed video which is not copyrighted by any person or organisation.

4.5.3. Security Requirements

Since any user can upload the video for summarization, the particular video should be accessible by a user only if he/she uploads it. This is a security requirement. A user can not access any other videos other than the one they have uploaded.

5. System Design

5.1. Introduction

We are developing a system for video summarization. This system can be utilized for summarizing video lectures. Since lecture videos are pretty lengthy, it is very difficult for the students to surf through the videos to search a particular topic. And also quick revision during the exams is also not possible with the whole video in place as it contains many things which are not so important. The summarization technique can help the students to find a particular topic in the whole video very easily. Since we are summarizing the whole video into a slide show with the most important snapshots of the lecture helps the students with a quick revision. This system can be given any lecture video, which it extracts the audio from and converts it to text and then summarizes the text. The timeline of the summarized text will be matched with the important snapshots taken.

Some of the limitations of the system would be improper videos with a lot of disturbance. This could result in decreased efficiency. And if any topics other than the syllabus discussed in the video can also pose a threat to the system, as it will fail to summarize it along with everything else discussed in the class.

5.2. Current System

The main problem in currently available video summarization systems is the continuation of the summarized data. Some of the applications tried to solve this by making use of Support Vector Machines. SVMs are used to understand the importance of each sentence and find the relationship between different sentences. In this way the decision of the present sentence is based on the decision of the previous sentences. And this helped solve the problem of continuation of the data in the output.

In some applications low level features like extracted text from the video are also used. But this system faces a major drawback when the videos are not clear.

Disturbance in the video data affects such systems a lot since most of the features are being taken from the video data and not from the audio data.

In some systems video frames are also used. Using these video frames moving objects can be detected. Video surveillance applications make use of video frames in an efficient way. These video frames can capture the important features of the video based on the object movement. But this method finds its application nowhere other than video surveillance systems.

5.3. Design Considerations

5.3.1. Design Goals

- Most of the already existing systems make use of the video frames from the video which mostly contains the slides of the lecture. Since these slides can be shared with the students beforehand, they will be of little help for students.
- Hence instead of using the video frames we are extracting the audio from the lecture and using the subtitles from the audio, which will be used for summarization.
- From the subtitles extractive and informative text summarization will be performed. Hence, we have chosen Latent Semantic Analysis as our text summarization algorithm. The timeline of the words/sentences will be extracted while transcribing the audio file.
- In this way we can combine both video frames and the audio to prepare the summarized video.

5.3.2. Architecture Choices

This architecture was chosen because the use of only video frames will only be useful for creating slides/ppts as a summary. But, slides/ppts are already prepared by institutes for the students. Hence, we decided to make use of both video frames and the audio output of the video to make a summarized video. This will contain more

information and will be of more value to the users. The use of audio files along with the video frames also helps us in having a continuation in the summarized output.

Getting a more informative output comes at the cost of execution time. Since there is so much computation going on, taking into account the audio, frames, text, summarization text, timelines, etc generation the output will require a lot of time.

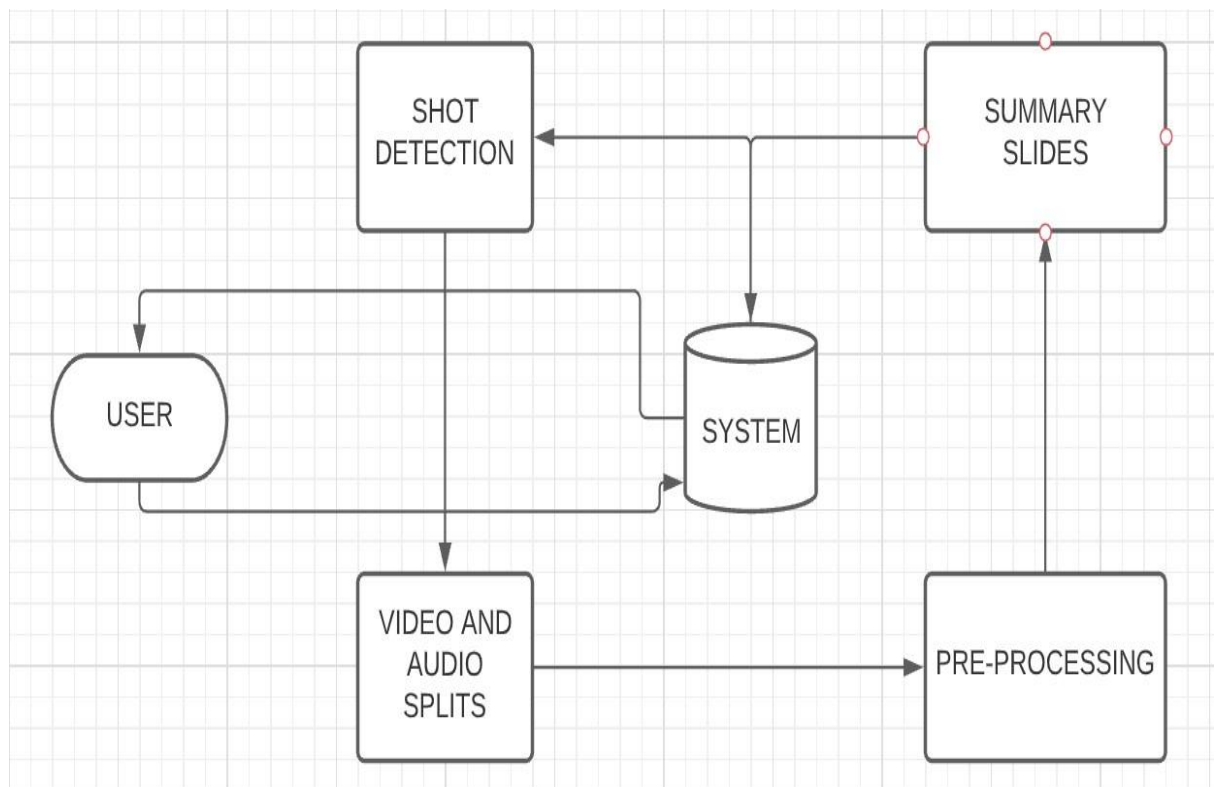


Fig. 0

5.3.2. Constraints, Assumptions and Dependencies

The following are assumed:

- It is assumed that the lecture video is in the English language
- The audio from the video is clear so that it can give accurate results.
- The user has an active internet connection

The operating environment should have the following configuration:

- 2 GB RAM
- Operating System: Windows/Ubuntu
- Keyboard and Mouse
- Available space of at least 15 GB
- Core: Pentium IV MHZ or more

- Cache: 512kb

The user must have the following software packages installed: Tensorflow, NumPy, MoviePy, PyDub, Nltk, etc.

5.4. High Level System Design

5.4.1 Splitting the video

We wish to split the video in a manner so as to separate the sub-topics taught in the lecture. If we can capture the moment in the video where the professor moves on to the next slide, we can say that the professor has moved to the next sub-topic. It is this change in slide that we wish to detect. Hence, we split the video wherever a shot transition takes place. An additional benefit of detecting shots is that we will also be finding out when the camera focus changes from the professor (and the teaching board) to the slides. This helps us capture content written on the black board as well.

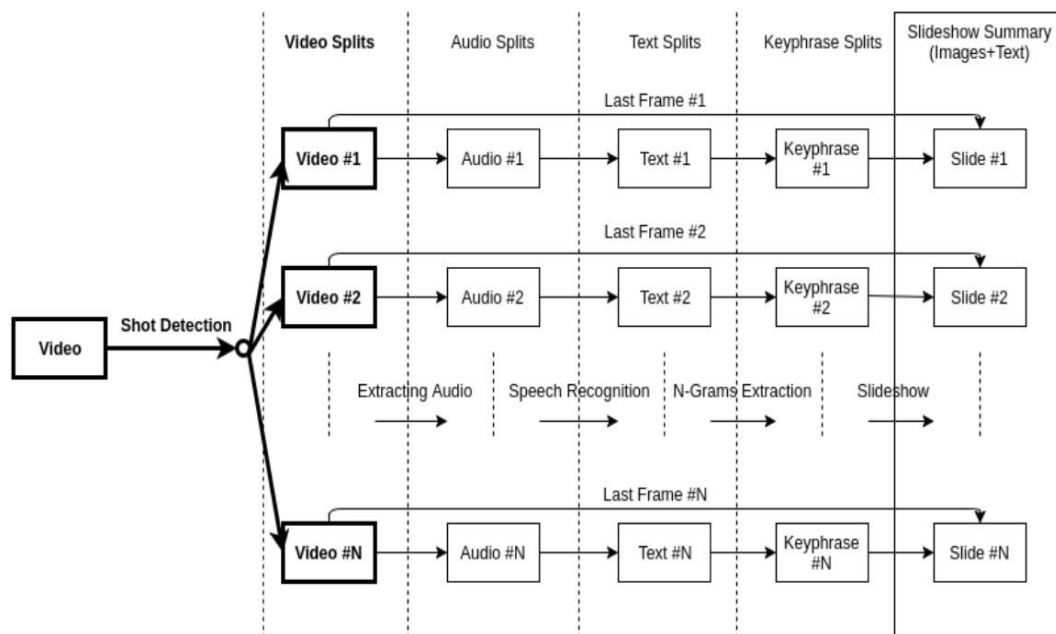


Fig. 1

5.4.2 Extracting the images

During the lecture, professors generally design their slides in such a way that each point in the slide appears one by one as they speak. We are not really interested in these intermediate, incomplete slides. From each split video, we extract the last image frame of that particular shot. We can safely say that the last frame will not contain any of the intermediate incomplete slides, but only the complete one. For extracting the last frame in the video, we count the total number of frames in the video, loop over all the frames until we reach the required frame using OpenCV.

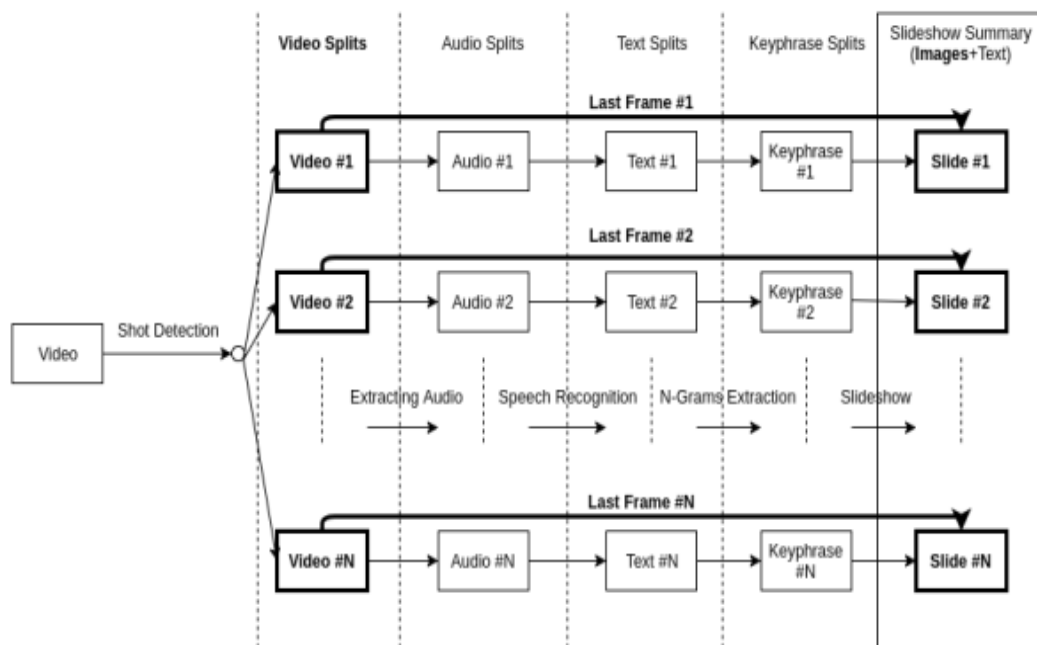


Fig. 2

5.4.3 Extracting the audio

We now extract audios from each video split. This is necessary to find out what the teacher is saying during the video lecture. To separate audio from the video, we use online tools

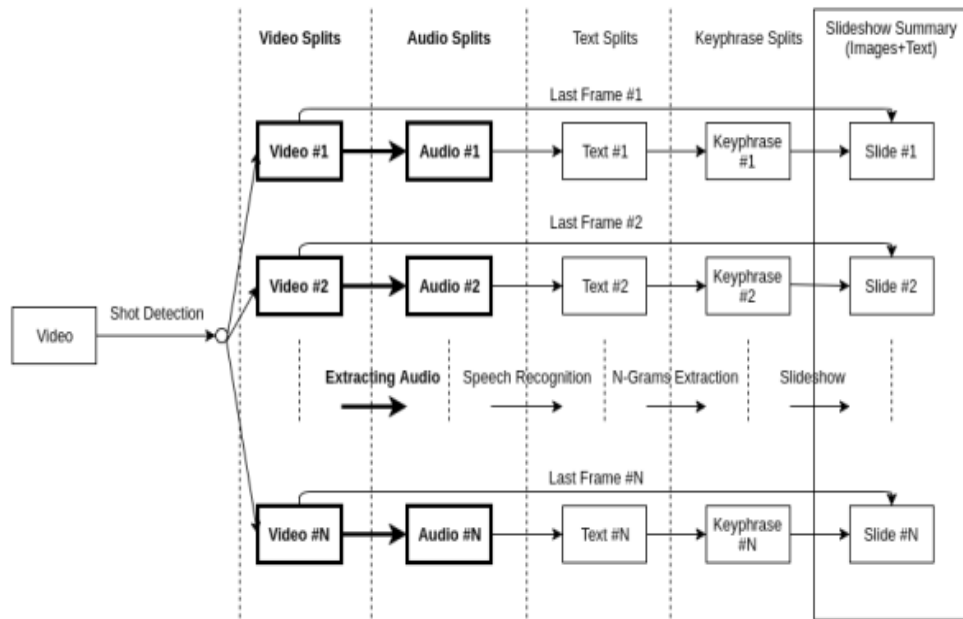


Fig. 3

5.4.4 Speech to Text

We now extract audios from each video split. This is necessary to find out what the teacher is saying during the video lecture. To separate audio from the video, we use online tools.

We use the Google Speech API for the speech to text conversion.

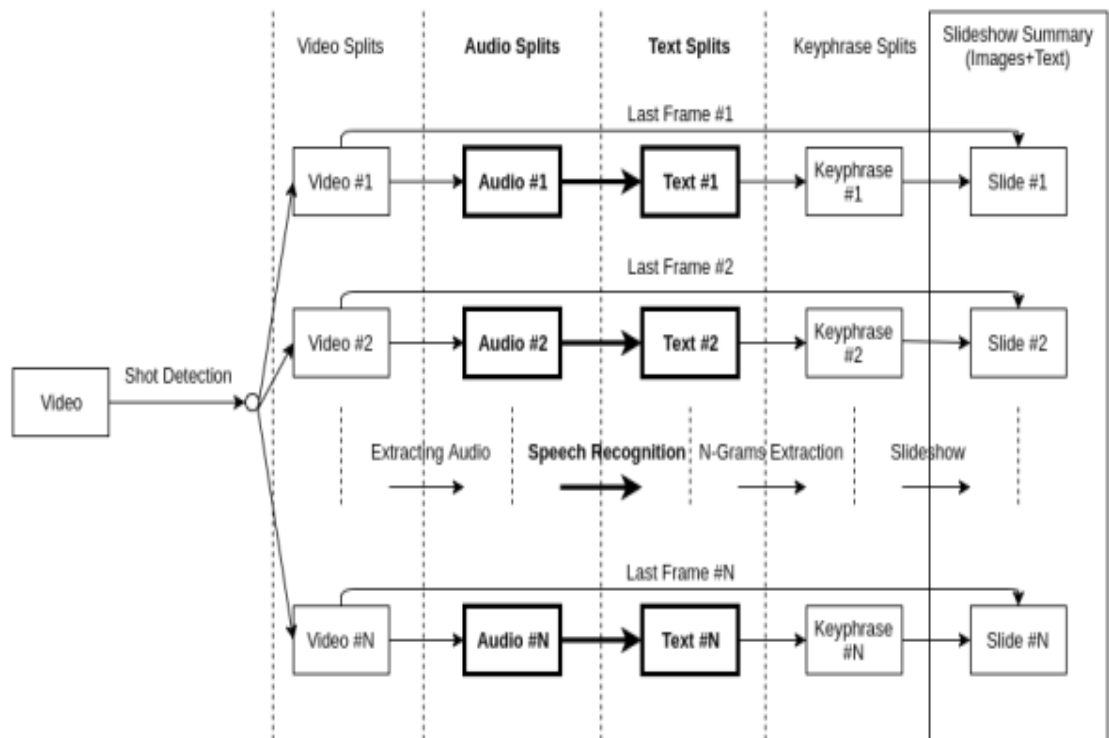


Fig. 4

5.4.5 Text to Keyphrases

It is important to convert the audio to text, so that we can extract the important keywords spoken for each of the sub-topics. For that we follow these steps:

1. Convert JSON output to raw text.
2. Remove special characters, punctuations etc.
3. Remove stopwords.
4. Generate all possible 1-Grams and 2-Grams, and record the frequency of each of them.

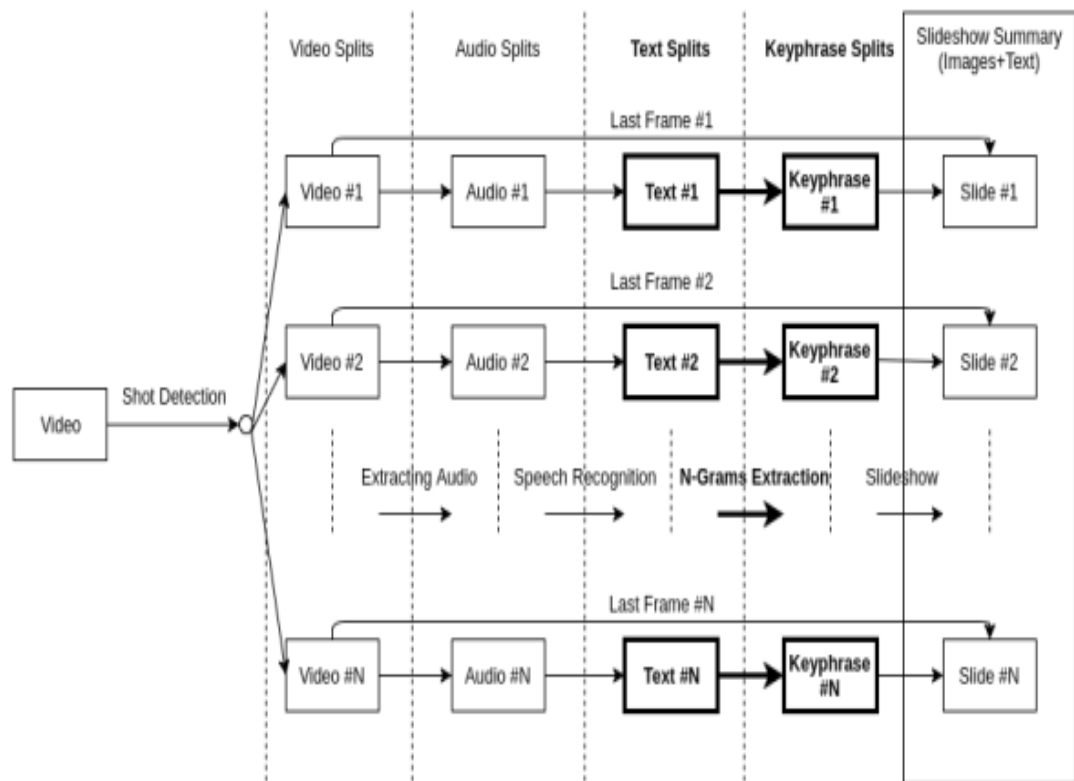


Fig. 5

5.4.6 Slideshow summary

The slides for the lecture will be an ordered sequence of images extracted from each of the video splits along with the keyphrases extracted from the text corresponding to the audio of each of the video splits.

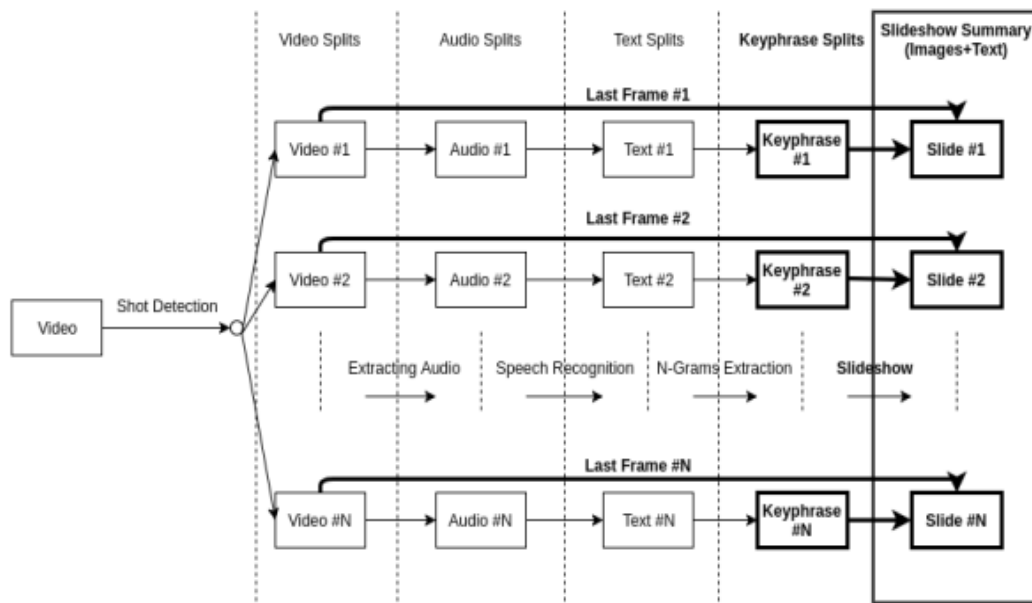


Fig. 6

5.5. Design Description

5.5.1. Master Class diagram

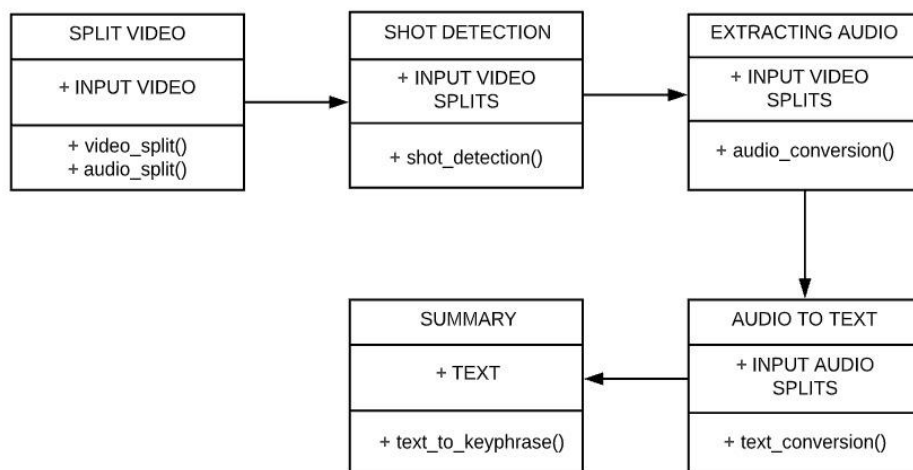


Fig. 7

5.7 User Interface Diagrams

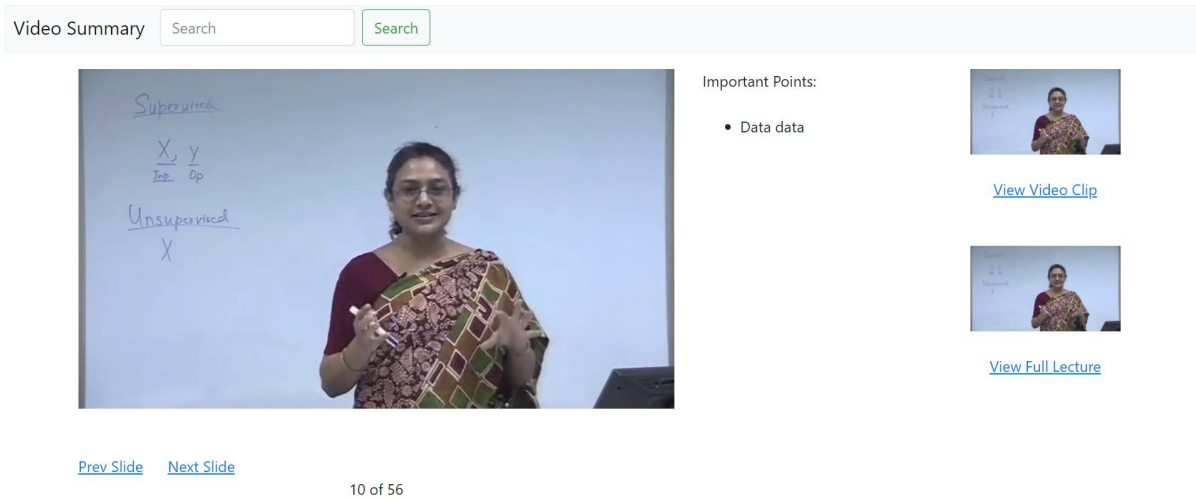


Fig. 8

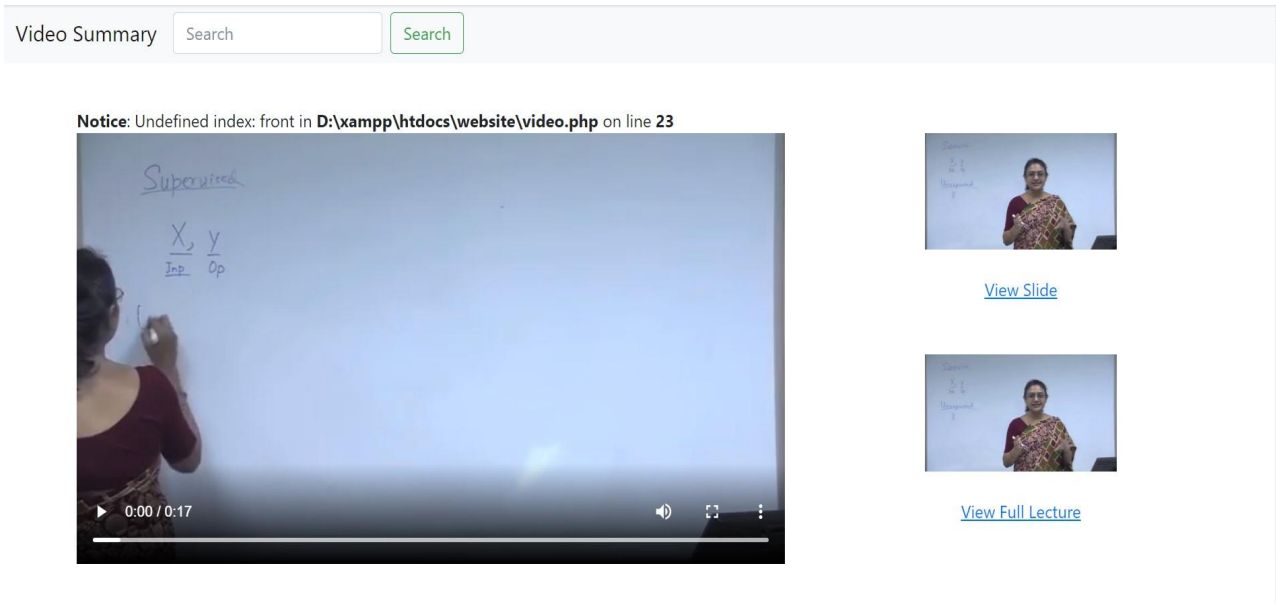


Fig. 9

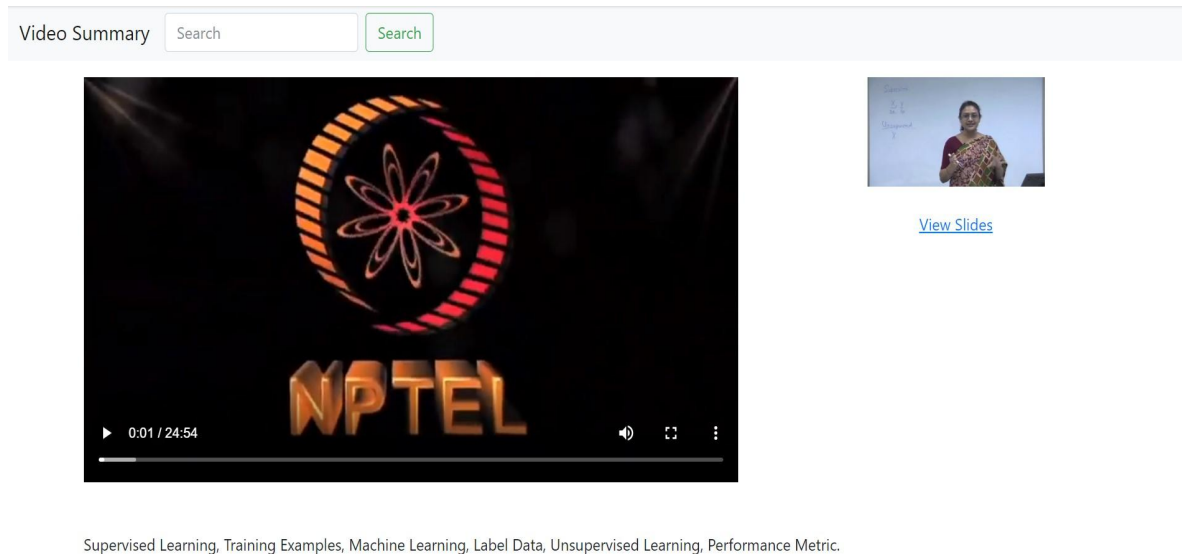


Fig. 10

6. Implementation and Pseudocode

6.1 Splitting the video using opencv and ffmpeg

```
PS C:\Users\Vrushabh\Desktop\Phase 2 Review 3\Capstone Phase 2> scenedetect -i video.mp4 detect-content split-video
[PySceneDetect] PySceneDetect v0.5.6
[PySceneDetect] Loaded 1 video, framerate: 29.871 FPS, resolution: 640 x 360
[PySceneDetect] Downscale factor set to 3, effective resolution: 213 x 120
[PySceneDetect] Ffmpeg codec args set: -c:v libx264 -preset veryfast -crf 22 -c:a aac
[PySceneDetect] Video output file name format: $VIDEO_NAME-Scene-$SCENE_NUMBER
[PySceneDetect] Detecting scenes...
100%|██████████| 44630/44630 [00:51<00:00, 868.31frames/s]
[PySceneDetect] Processed 44630 frames in 51.4 seconds (average 867.91 FPS).
[PySceneDetect] Detected 55 scenes, average shot length 27.2 seconds.
[PySceneDetect] Comma-separated timecode list:
00:00:00.569,00:00:07.030,00:00:31.167,00:00:37.996,00:00:40.406,00:00:48.039,00:01:40.363,00:01:44.113,00:02:01.889,00:02:08.584,0
0:03:11.855,00:03:47.541,00:03:48.646,00:03:53.701,00:04:00.798,00:04:09.937,00:04:11.343,00:04:17.336,00:04:45.925,00:05:42.333,00:0
5:45.446,00:05:57.532,00:06:22.673,00:06:38.172,00:06:47.546,00:06:56.216,00:07:04.954,00:07:28.521,00:07:51.419,00:07:52.926,00:09:0
4.399,00:09:12.232,00:10:59.860,00:11:58.009,00:12:04.135,00:12:22.882,00:12:27.569,00:12:31.352,00:13:58.324,00:14:02.978,00:16:43.6
99,00:17:41.581,00:20:00.375,00:21:07.128,00:21:13.756,00:21:14.560,00:22:06.114,00:22:27.572,00:22:29.682,00:24:24.406,00:24:27.218,
00:24:42.752,00:24:43.957,00:24:53.732
[PySceneDetect] Splitting input video using ffmpeg, output path template:
```

Fig. 11

```
[PySceneDetect] Comma-separated timecode list:
00:00:00.569,00:00:07.030,00:00:31.167,00:00:37.996,00:00:40.406,00:00:48.039,00:01:40.363,00:01:44.113,00:02:01.889,00:02:08.584,0
0:03:11.855,00:03:47.541,00:03:48.646,00:03:53.701,00:04:00.798,00:04:09.937,00:04:11.343,00:04:17.336,00:04:45.925,00:05:42.333,00:0
5:45.446,00:05:57.532,00:06:22.673,00:06:38.172,00:06:47.546,00:06:56.216,00:07:04.954,00:07:28.521,00:07:51.419,00:07:52.926,00:09:0
4.399,00:09:12.232,00:10:59.860,00:11:58.009,00:12:04.135,00:12:22.882,00:12:27.569,00:12:31.352,00:13:58.324,00:14:02.978,00:16:43.6
99,00:17:41.581,00:20:00.375,00:21:07.128,00:21:13.756,00:21:14.560,00:22:06.114,00:22:27.572,00:22:29.682,00:24:24.406,00:24:27.218,
00:24:42.752,00:24:43.957,00:24:53.732
PS C:\Users\Vrushabh\Desktop\Phase 2 Review 3\Capstone Phase 2>
```

Fig. 12

The output of the above code is as follows:

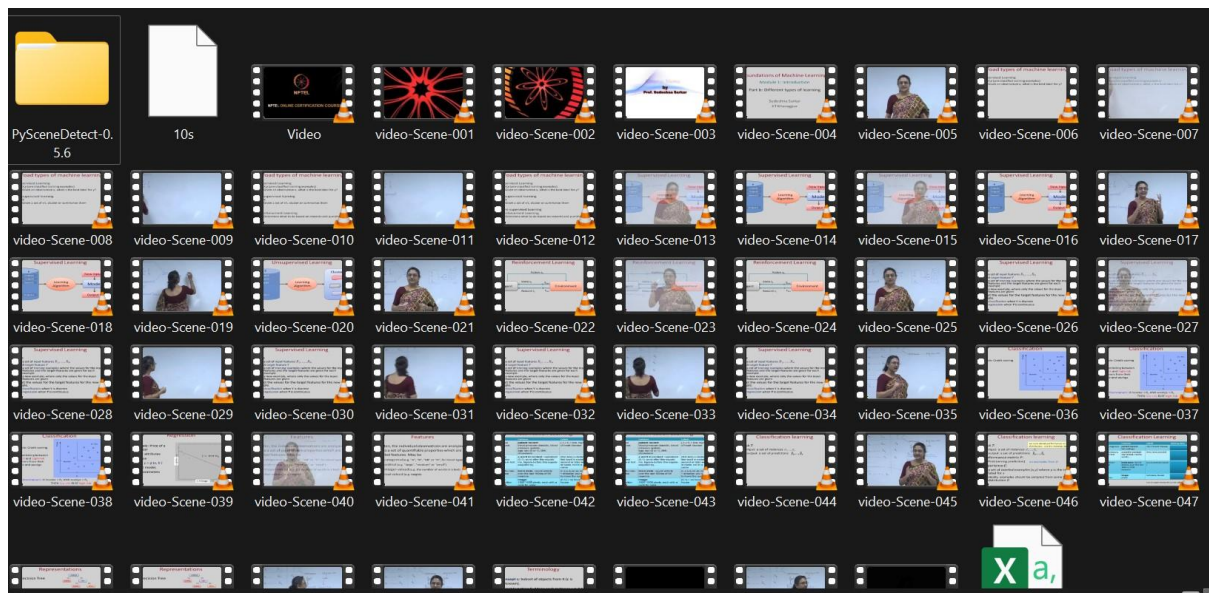


Fig. 13

6.2 Identifying last frame of the video

For extracting the last frame in the video, we count the total number of frames in the video, loop over all the frames until we reach the required frame using OpenCV.

```

1 import cv2
2
3 for i in range(1,57):
4     if i >= 10:
5         s = "0" + str(i)
6     else:
7         s = "00" + str(i)
8     vidcap = cv2.VideoCapture("TEMPV/SPLIT-" + s + ".mkv")
9     framecount = vidcap.get(cv2.cv.CV_CAP_PROP_FRAME_COUNT)
10    count = 0
11    success,image = vidcap.read()
12    success = True
13    while count < framecount-50:
14        success,image = vidcap.read()
15        count += 1
16    cv2.imwrite("TEMPI/FRAME-" + s + ".jpg", image)

```

Fig. 14

6.3 Speech to text conversion

We use the Google Speech API for the speech to text conversion:

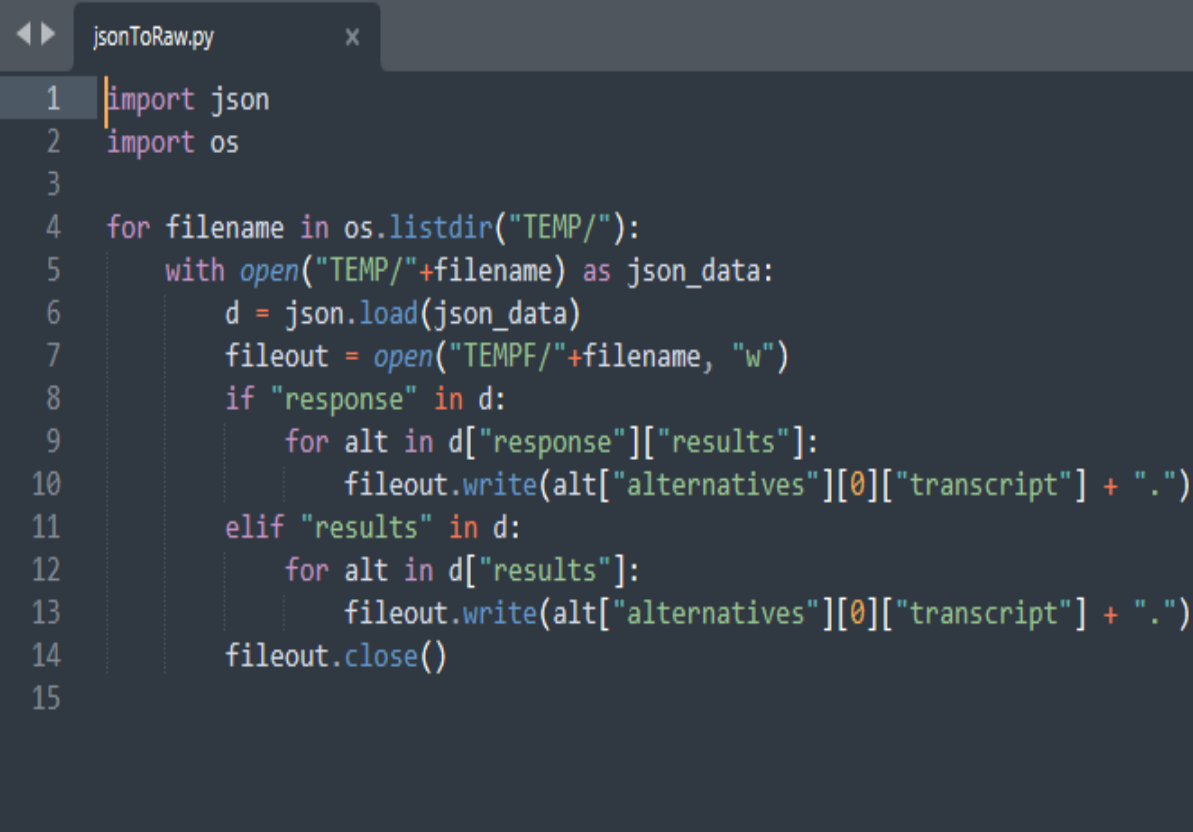
```

curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-001.flac"}}' > TEMP/OUT-001.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-002.flac"}}' > TEMP/OUT-002.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-003.flac"}}' > TEMP/OUT-003.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-004.flac"}}' > TEMP/OUT-004.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-005.flac"}}' > TEMP/OUT-005.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-006.flac"}}' > TEMP/OUT-006.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-007.flac"}}' > TEMP/OUT-007.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-008.flac"}}' > TEMP/OUT-008.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-009.flac"}}' > TEMP/OUT-009.txt
curl -s -H 'Content-Type: application/json' -H 'Authorization: Bearer `gcloud auth print-access-token` https://speech.googleapis.com/v1/speech:recognize -d '{"config":
  {"encoding": "FLAC", "sampleRateHertz": 44100, "languageCode": "en-US", "enableWordTimeOffsets": false}, "audio": {"uri": "gs://bamboo-foundation-6245/SPLIT-010.flac"}}' > TEMP/OUT-010.txt

```

Fig. 15

6.4 Convert Json to raw text



```
1 import json
2 import os
3
4 for filename in os.listdir("TEMP/"):
5     with open("TEMP/"+filename) as json_data:
6         d = json.load(json_data)
7         fileout = open("TEMPF/"+filename, "w")
8         if "response" in d:
9             for alt in d["response"]["results"]:
10                 fileout.write(alt["alternatives"][0]["transcript"] + ".")
11         elif "results" in d:
12             for alt in d["results"]:
13                 fileout.write(alt["alternatives"][0]["transcript"] + ".")
14         fileout.close()
15
```

Fig. 16

6.5 User Interface

6.5.1 - Landing Page

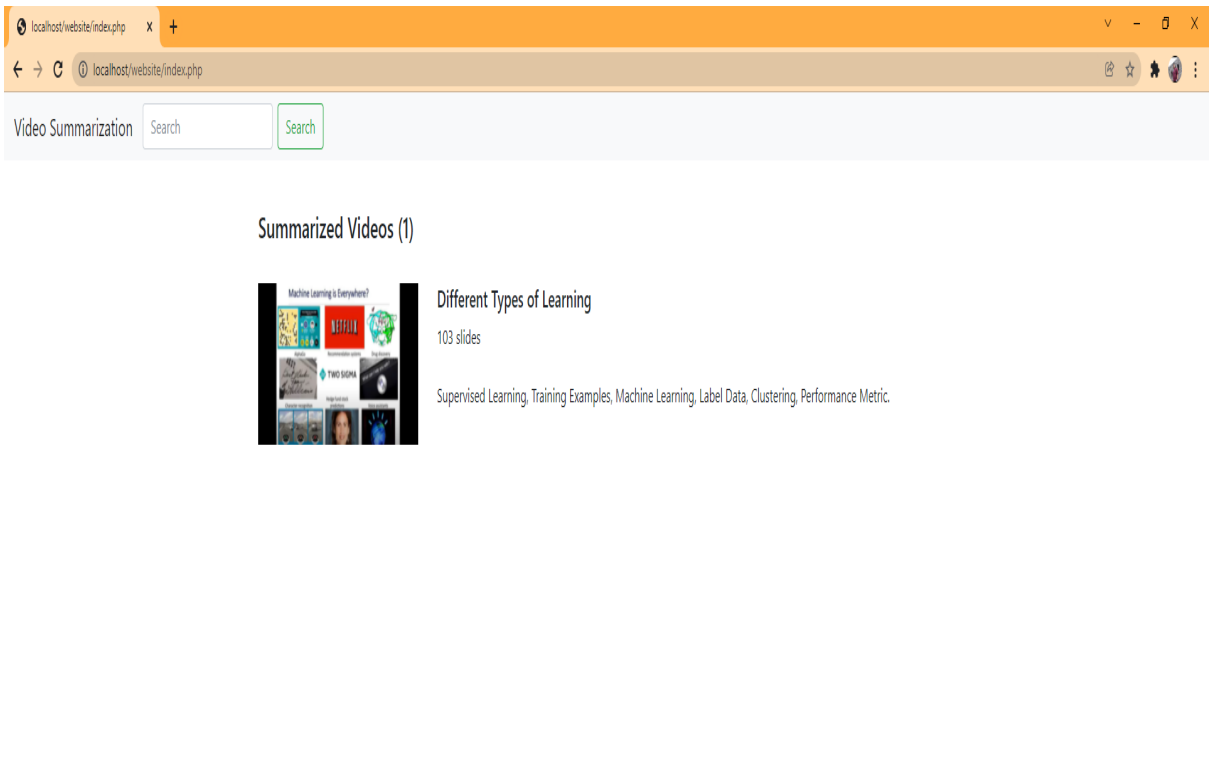


Fig. 17

6.5.2 - Slides

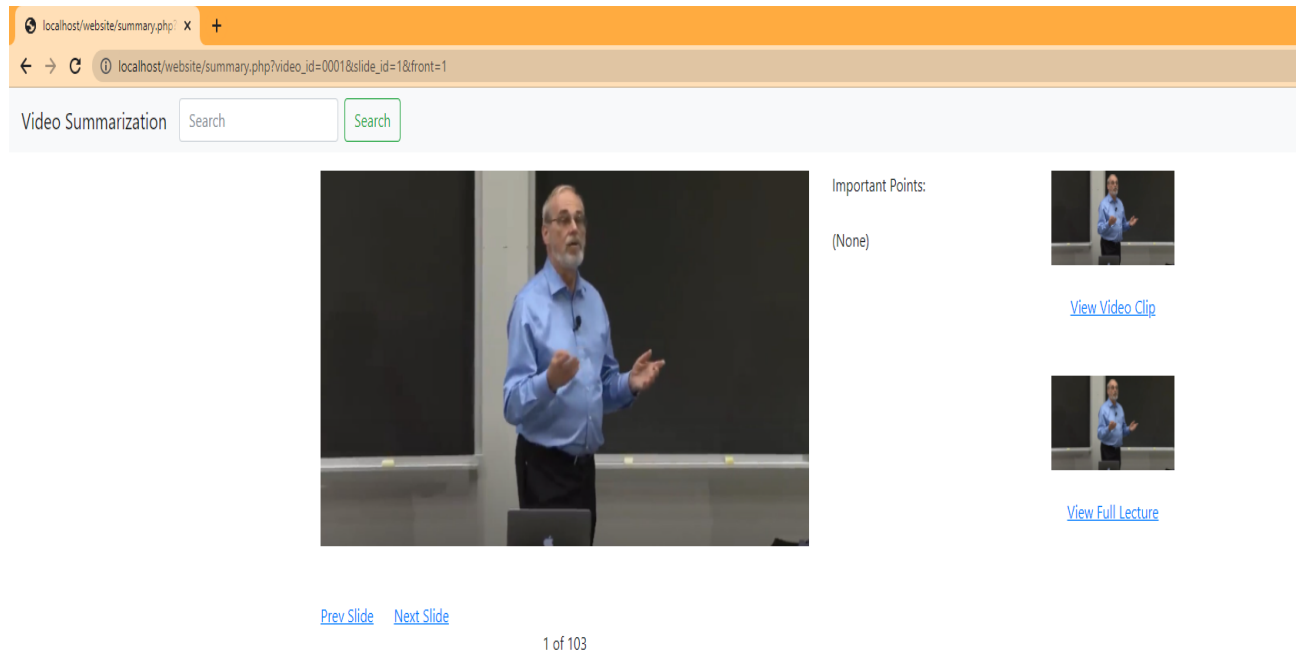
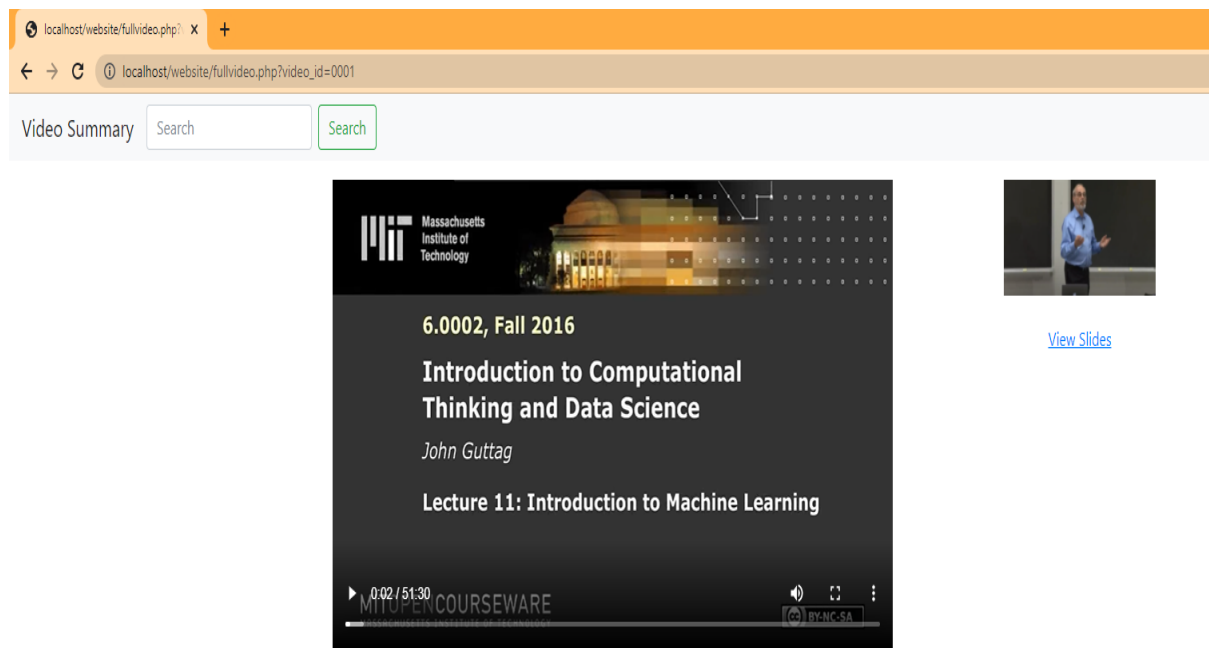


Fig. 18

6.5.3 - Viewing the Full Video



Supervised Learning, Training Examples, Machine Learning, Label Data, Unsupervised Learning, Performance Metric.

Fig. 19

6.5.4 - Video Clip

The screenshot shows a web browser window with the address bar displaying `localhost/website/video.php?video_id=0001&slide_id=6`. Below the browser, there is a search bar with the text "Video Summarization" and a "Search" button. The main content area displays a video player. The video frame shows a slide titled "The plan ahead" with the following text:

- Machine learning is a huge topic – with whole courses devoted to it
 - e.g., 6.008, 6.036, 6.860, 6.862, 6.867, and as central part of courses in natural language processing, computational biology, computer vision, robotics, other areas
- In 6.0002, we will
 - Provide an introduction to the basic ideas, including ways to measure distances between examples, and how to group examples based on distance to create models
 - Introduce classification methods, such as "k nearest neighbor" methods
 - Introduce clustering methods, such as "k-means"

Below the video frame, there is a progress bar showing 0:01 / 0:04. To the right of the video player, there is a thumbnail of the slide and a "View Slide" link. Below the thumbnail, there is a small video frame showing a person and a "View Full Lecture" link.

Fig. 20

6.5.5 - Search

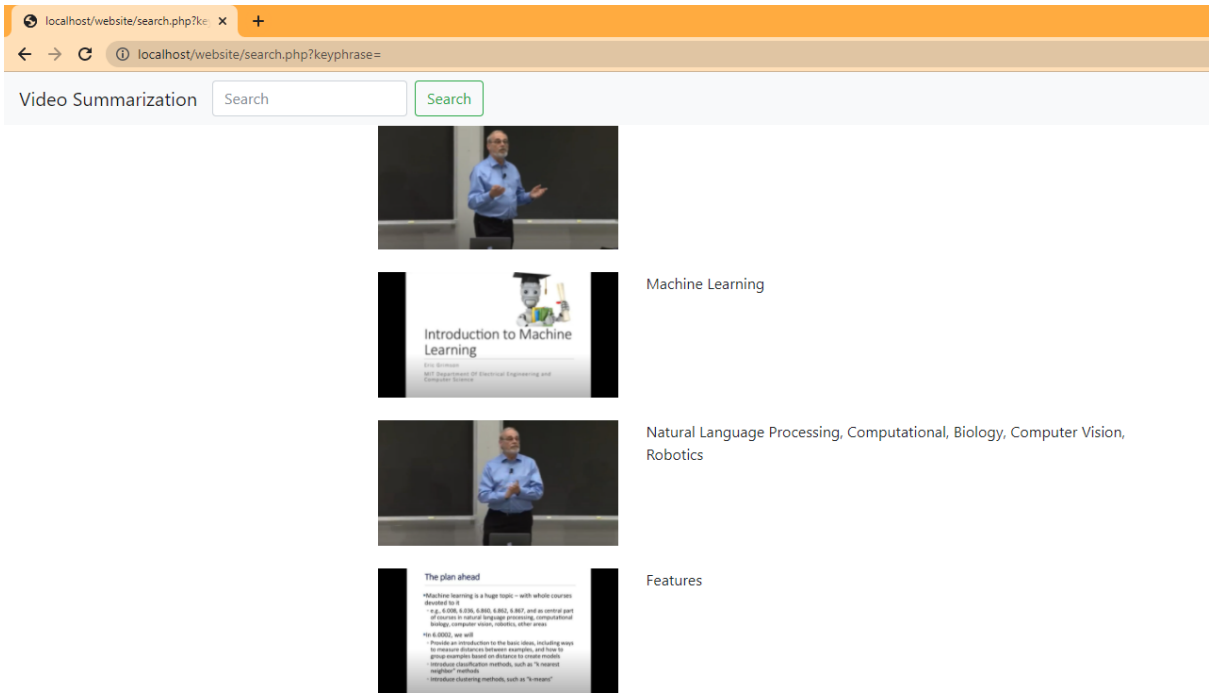


Fig. 21

7. Results And Discussion

7.1 Results

- We have a website that helps user to summarize their lecture videos in the form of slides
- We have also made sure to attach the clip associated with the slide so that if students want an explanation about the topic(s) on the slide they don't need to go through the entire lecture
- We also have a search bar that helps students do a topic based search so that they don't need to go through all the slides to find the topic they are searching for.

7.2 Discussion

- We need to increase the accuracy of each component as well as the overall accuracy.
- We need to develop methods to filter and cover up the errors getting carried forward from the previous component.
- We need to ensure that the users have a great experience, checking if we can smoothen any process that will help the users to have a better experience.

8. Conclusion

Some of the applications of video summarization are Video Database Management, Consumer Video Analysis, CCTV Surveillance Video Summary, Sports Video Highlights Generation, Lecture Video Summarization and many more. The application that we are focusing on is lecture video summarization.

Our conclusions:

- The accuracy of the components individually is good, however, there is a huge scope for improvement.
- Because the components are executed sequentially, the overall accuracy decreases as the errors in each component get carried forward and affect the components next in line.

Appendix A: Definitions, Acronyms and Abbreviations

CCTV : Closed Circuit Television.

FTP : File Transfer Protocol.

Frames : Snapshot of the video at a particular time.

User : Refers to the student/ teacher who is using the product.

Appendix B: References

1. Nosql database : <https://www.mongodb.com/nosql-explained>
2. K. Davila and R. Zanibbi, "Whiteboard Video Summarization via Spatio-Temporal Conflict Minimization," 2017 14th IAPR International Conference on Document Analysis and Recognition (ICDAR), Kyoto, Japan, 2017, pp. 355-362, doi: 10.1109/ICDAR.2017.66.
3. Ramamohan Kashyap Abhilash, Choudhary Anurag, Vaka Avinash and D. Uma, "Lecture Video Summarization Using Subtitles", 2020 2nd EAI International Conference on Big Data Innovation for Sustainable Cognitive Computing, EAI/Springer Innovations in Communication and Computing. https://doi.org/10.1007/978-3-030-47560-4_7
4. He, L., Sanocki, E., Gupta, A., and Grudin, J., "Auto-Summarization of audio-video presentations", 1999 In Proceedings of the ACM Multimedia Conference (ACMMM), Orlando, FL.
5. A. Vimalaksha, S. Vinay, A. Prekash and N. S. Kumar, "Automated Summarization of Lecture Videos," 2018 IEEE Tenth International Conference on Technology for Education (T4E), Chennai, India, 2018, pp. 126-129, doi: 10.1109/T4E.2018.00034

Appendix C: Record of Change History

#	Date	Document Version No.	Change Description	Reason for Change
1.	10-04-2021	1.0	Initial commit on the introduction and description of sections	Initial commit
2.	11-04-2021	1.1	User Interface Diagrams, Swimlane Diagram	Inclusion of diagrams
3.	15-04-2021	1.2	System Architecture and completion of the rest of the design.	Inclusion of diagrams
4.	10-12-2021	1.3	Final commit and fixes	Final commit

Table 1