A PRELIMINARY REPORT ON

VIDEO SUMMARIZATION

SUBMITTED TO THE SAVITRIBAI PHULE PUNE UNIVERSITY, PUNE
IN THE PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE

OF

BACHELOR OF ENGINEERING
(COMPUTER ENGINEERING)

SUBMITTED BY

ASEEM KANNAL	Exam No : 71700718K
ATHARVA BHAGWAT	Exam No : 71700740F
SAMEER KOLHAR	Exam No : 71701180B
PURVA SHETH	Exam No : 71701133L

DEPARTMENT OF COMPUTER ENGINEERING

PUNE INSTITUTE OF COMPUTER TECHNOLOGY

DHANKAWADI, PUNE - 411043

SAVITRIBAI PHULE PUNE UNIVERSITY

2019-2020

CERTIFICATE

This is to certify that the project entitled

VIDEO SUMMARIZATION

Submitted by

Aseem Kannal	Exam No : 71700718K
Atharva Bhagwat	Exam No : 71700740F
Sameer Kolhar	Exam No : 71701180B
Purva Sheth	Exam No : 71701133L

are bonafide students of this institute and the work has been carried out by them under the supervision of **Prof. A. A. Chandorkar**, approved for the fulfillment of the requirements of Savitribai Phule Pune University, for the award of the degree of **Bachelor of Engineering** (Computer Engineering).

Prof. A. A. Chandorkar
Guide,
Dept. of Computer Engg.

Prof. M. S. Takalikar
Head,
Dept. of Computer Engg.

Dr. P. T. Kulkarni
Principal,
Pune Institute of Computer Technology

Place: Pune

Date:

Acknowledgement

It gives us great pleasure in presenting the preliminary project report on **'Video Summarization'**.

We would like to take this opportunity to thank our guide **Prof. A. A. Chandorkar** for giving us all the help and guidance we needed. We are really grateful to them for their kind support. Their valuable suggestions were very helpful.

We are also grateful to **Prof. M. S. Takalikar**, Head of Computer Engineering Department, PICT for her indispensable support and suggestions.

In the end, our special thanks to the college for providing various resources such as laboratory with all needed software platforms, continuous internet connection, for our project.

Aseem Kannal
Atharva Bhagwat
Sameer Kolhar
Purva Sheth
(B.E. Computer Engineering)

ABSTRACT

With growing number of users accessing the internet, the amount of data being created has been increasing exponentially over the years. Today's user while having quick access to large repositories of data, lacks time to do so. Users prefer to access data in summarized format as it is less time consuming. We aim to provide the same functionality in videos by providing their summaries. We propose a Deep Learning approach using Convolutional Neural Network (CNN) and Recurrent Neural Network (RNN) that learn the video representations and sorts out frames that are given more importance than the others. Later combining these frames to form a summarized video which will be the optimal representation of the original video.

Contents

List of Abbreviations

CNN	Convolutional Neural Network
DSN	Deep Summarization Network
DTR-GAN	Dilated Temporal Relational Generative Adversarial Network
GAN	Generative Adversarial Network
GANs	Generative Adversarial Networks
LSTM	Long-Short Term Memory
RNN	Recurrent Neural Network

List of Figures

CHAPTER 1

INTRODUCTION

1.1 Overview

Video summarization is the task of extracting key frames from a video. The importance of a frame can be defined by how well it represents the context of the video. The aim is to extract such frames to form a substantial summary of the video.

1.2 Motivation

From news to study notes; today's user wants to gain information faster and quicker than ever before. Bite sized content has multiple benefits for it's consumer and the biggest being that it is easy to assimilate. Raw information is processed and compressed to shorten it. The resulting content would still retain the importance. We aim to do the same for videos. The aim is to create a shorter version of the video that users can view and still understand the context of the video.

1.3 Problem Definition and Objectives

The problem to be solved via this project is of video summarization. Given a video by the user, the aim is to extract the important frames and create a smaller video highlighting the important frames. Since the proposed model is unsupervised the videos will be independent of the domain. Given a video with n frames $\{f_1, f_2, \ldots, f_n\}$, the objective is to recognize important frames and extracting them using a vector $Y = \{y_1, y_2, \ldots, y_n\}$, $y_i = [0, 1)$, where Y contains the importance score for each frame.

1.4 Project Scope and Limitations

Via this project, we propose an Artificial Intelligence model based on Deep Learning techniques to summarize a user's input video. The model is trained in an unsupervised fashion, which makes our proposed model domain independent.

CHAPTER 2

LITERATURE SURVEY

2.1 Unsupervised Video Summarization with Adversarial LSTM Networks

This paper addresses the problem of selecting sparse subset of video frames that optimally represent the input video. It proposes a unsupervised approach using generative adversarial framework consisting of the summarizer and discriminator for learning. The summarizer is a LSTM network aimed at selection of video frames. The discriminator is another LSTM network aimed at distinguishing between the original video and its reconstruction from the summarizer. [1]

2.2 Deep Reinforcement Learning for Unsupervised Video Summarization with Diversity-Representativeness Reward

Published at the 32ND AAAI Conference on Artificial Intelligence, 2017. This paper uses a sequential decision making process to develop a Deep Summarization Network (DSN). For each video frame DSN predicts a probability which indicates how likely a frame is selected, and then takes actions based on the probability distributions to select frames, forming video summaries. [2]

2.3 Cycle-SUM: Cycle-consistent Adversarial LSTM Networks for Unsupervised Video Summarization

This paper proposes a model termed Cycle-SUM, which consists of a frame selector and a cycle consistent learning-based evaluator. The selector is a bidirectional LSTM network that learns the video representations that embed the long-range relationships among the video frames. The evaluator consists of two Generative Adversarial Networks (GANs), in which the forward GAN is learned to reconstruct original video from summary video, while the backward GAN learns to invert the processing. [3]

2.4 Dilated Temporal Relational Adversarial Network for Generic Video Summarization

This paper focuses on Dilated Temporal Relational Generative Adversarial Network (DTR-GAN) to achieve frame-level video summarization. Given a video, it selects the set of key frames which contain the most meaningful and compact information. [4]

2.5 Quasi Real-Time Summarization for Consumer Videos

Published at IEEE Conference on Computer Vision and Pattern Recognition, 2014. This paper focuses on a method which learns a dictionary from given video using group sparse coding, and updates it on-the-fly. A summary video is generated by combining segments that cannot be sparsely reconstructed using learned dictionary. [5]

2.6 Video Summarization: A Machine Learning Based Approach

This paper proposes a 2 pronged process for the task of video summarization. The first is the process of segmentation of the video into shots. This is performed using a neuro-fuzzy network. The second step includes independent component analysis which is strengthened by a content based filtering scheme. [6]

2.7 Unsupervised Video Summarization via Attention-Driven Adversarial Learning

This paper presents a new video summarization approach that integrates an attention mechanism to identify the significant parts of the video and is trained unsupervisingly via generative adversarial learning. The attention mechanism is integrated using an attention layer within the variational auto-encoder and by replacing the VAE with a deterministic attention auto-encoder. [7]

2.8 TransNet: A deep network for fast detection of common shot transitions

Shot boundary detection is an important first step in many video processing applications. This paper presents a simple modular CNN architecture. The network employs dilated convolutions and operates just on small resized frames. [8]

CHAPTER 3

SOFTWARE REQUIREMENTS SPECIFICATION

3.1 Assumptions and Constraints

3.1.1 Assumptions

The following assumptions are made based on the project scope:

- Videos to be summarized refer only to the visual aspect, we are not considering the audio associated with the videos.
- User is supported by an internet connection to upload and view the videos.
- Videos are not of length greater than 20 minutes.

3.1.2 Constraints

The following constraints are imposed upon the project:

- Amazon services available in the free tier are not powerful.
- Videos must be lesser than 20 minutes in length.
- High context video content like sports and movies are not supported by the current model.

3.2 Functional Requirements

- Users should be able to upload videos.
- API to support user actions like upload, view and download videos using the Web User Interface (UI).
- Summarized videos should be generated after a video is uploaded.

3.3 Non-Functional Requirements

3.3.1 Usability

- The web UI will be intuitive and easy to use.

3.3.2 Reliability

- The system must roll back any updates to the database in case of failure.

3.3.3 Performance

- The latency between the API request and the generated summary will be minimized.

3.3.4 Security

- Neither the video nor the generated summary will be stored in the platform's database without the user's consent.

3.4 System Requirements

3.4.1 Database Requirements

- mongo DB - v4.2

3.4.2 Software Requirements

- React Web Framework
- Redux JavaScript (JS) Library
- Pusher APIs
- AWS APIs
- Compute Unified Device Architecture (CUDA) Toolkit 9.0 or higher

3.4.3 Hardware Requirements

- Graphics Processing Unit (GPU): Nvidia GTX 1080 or higher (8 GB)
- Amazon Simple Storage Service (S3) instance
- Amazon Elastic Compute Cloud (EC2) instance
- ageMaker instance

3.5 SDLC Model to be applied

- We applied agile software development techniques such as pair programming during the development of this project.
- Agile techniques will allowed us to rapidly build prototypes while testing the software.

CHAPTER 4

SYSTEM DESIGN

4.1 System Architecture

We will be producing a modern web application using React.JS and Redux. React will interact with the Document Object Model (DOM) on the browser and also display the UI components. Redux will be used as the single source of truth for our application. React will interact with Redux via *actions*. Redux on updating the state communicates the changes to the state via *reducers*.

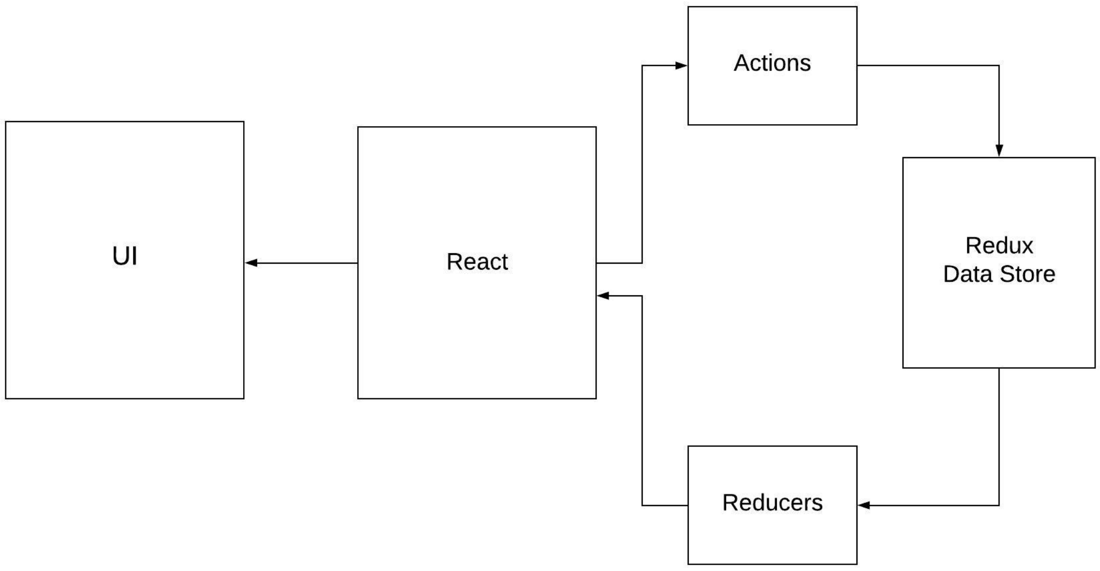


Figure 4.1: System Architecture (Frontend)

The client-side of the application will be supported by an equally robust server-side architecture. Upon sending a request to the via the Hypertext Transfer Protocol (HTTP) Application Programming Interface (API), the request first reaches the API gateway. User requests are filtered and validated before reaching server. The server, on receipt of the request and the video file from the user, stores the video file on Amazon S3. The Video Summaraization model then obtains this video, processes it, produces the resultant video and stores the result at S3. This video is then served to the client.

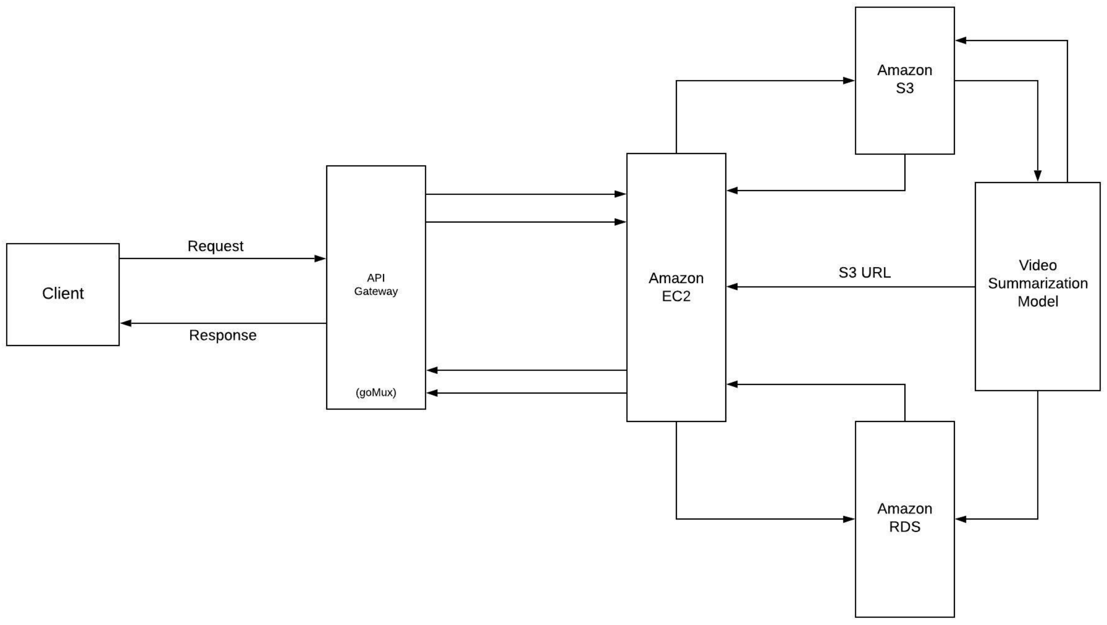


Figure 4.2: System Architecture (Backend)

4.2 Data Flow Diagrams

4.2.1 DFD Level 0

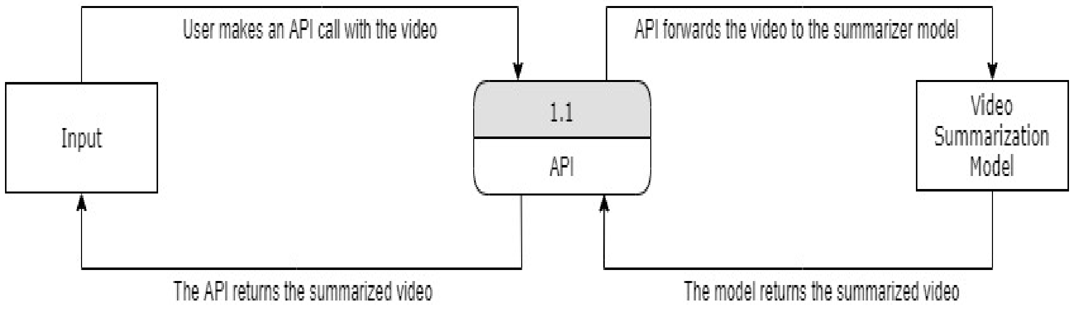

User makes an API call with the video
API forwards the video to the summarizer model
Input
1.1
API
Video Summarization Model
The API returns the summarized video
The model returns the summarized video

Figure 4.3: DFD Level 0

4.2.2 DFD Level 1

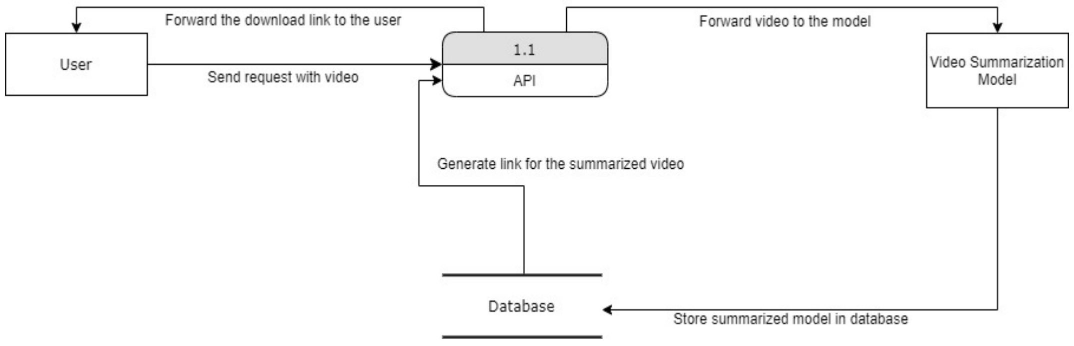

Forward the download link to the user
Forward video to the model
User
Send request with video
1.1
API
Video Summarization Model
Generate link for the summarized video
Database
Store summarized model in database

Figure 4.4: DFD Level 1

4.3 UML Diagrams

4.3.1 Use Case Diagram

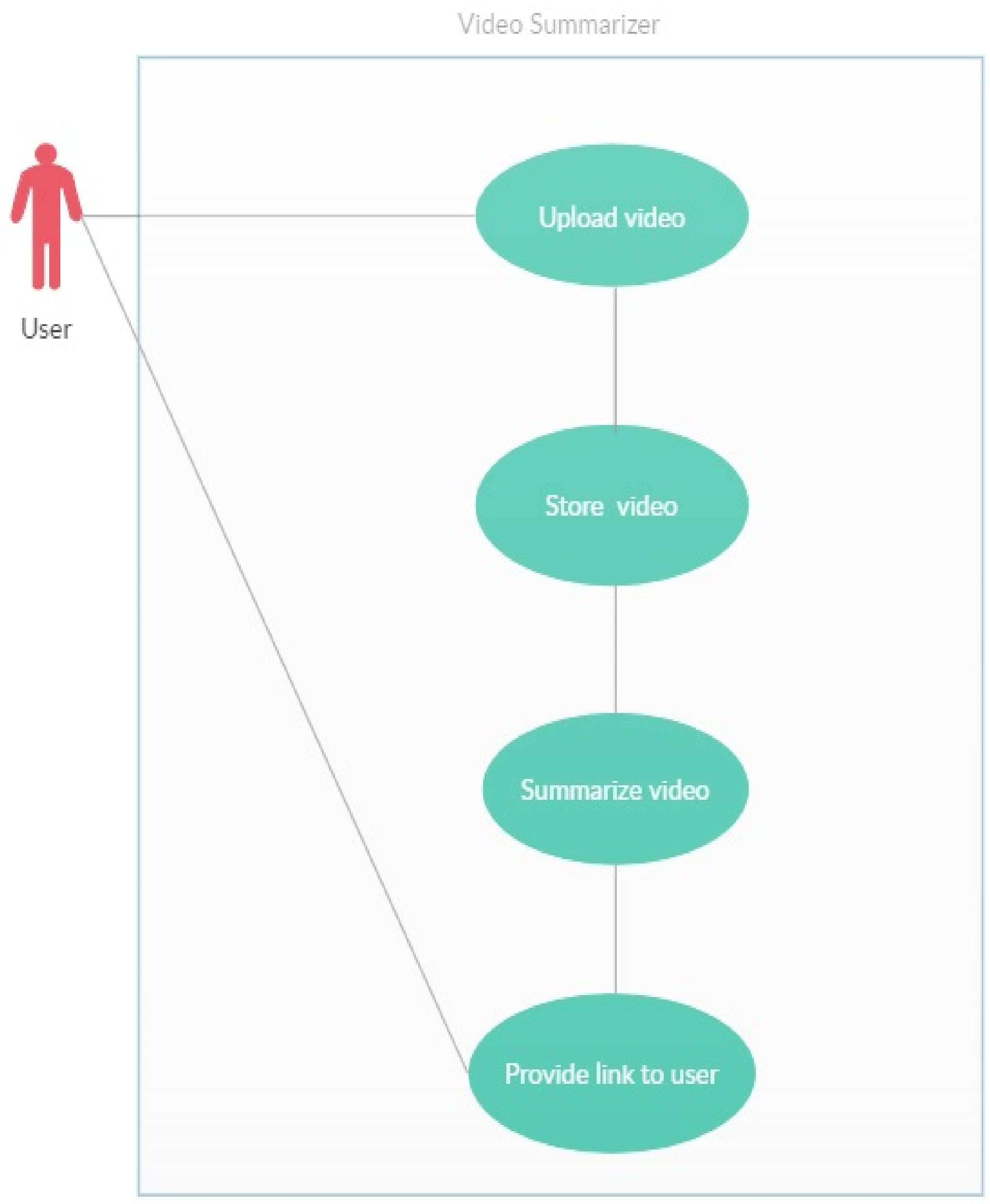


Figure 4.5: Use Case Diagram

4.3.2 Activity Diagram

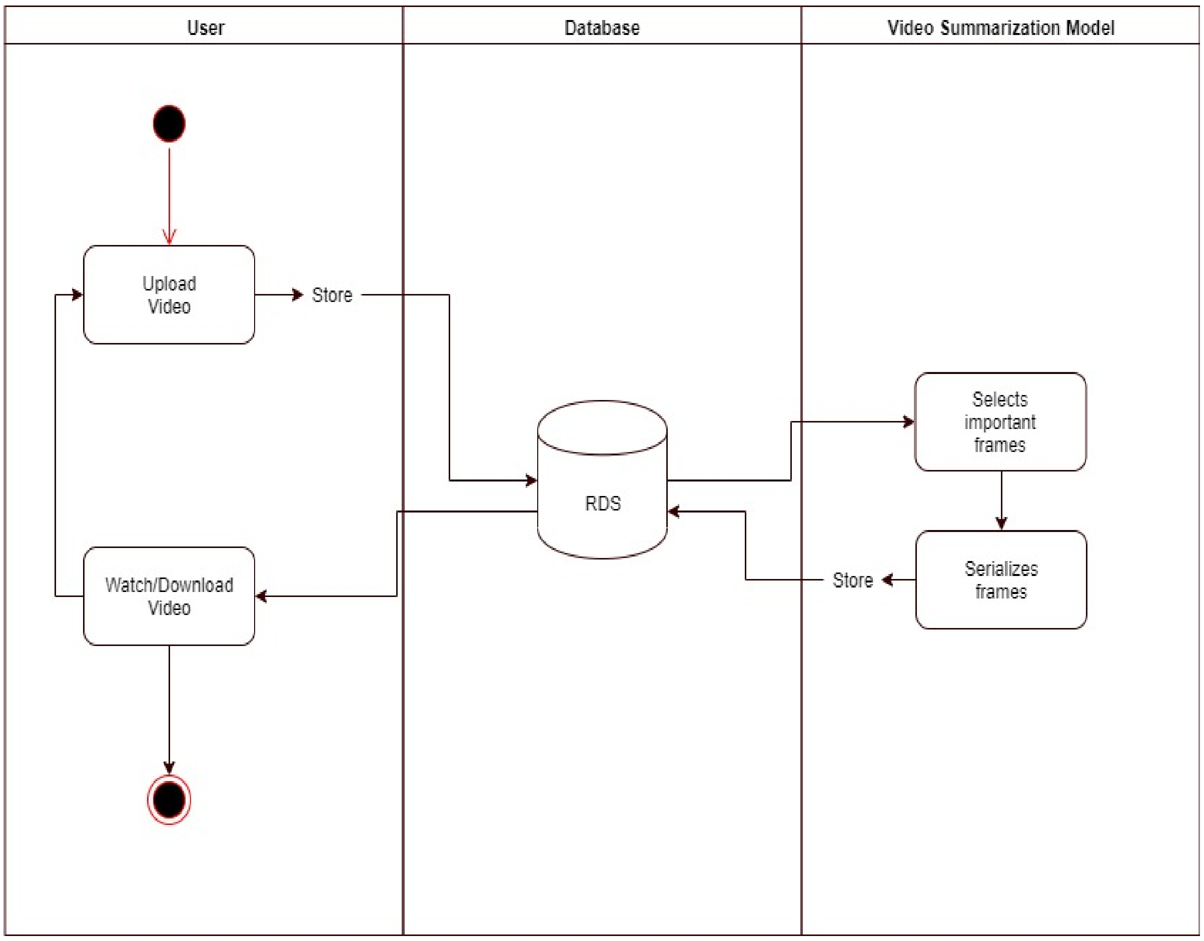


Figure 4.6: Activity Diagram

4.3.3 Sequence Diagram

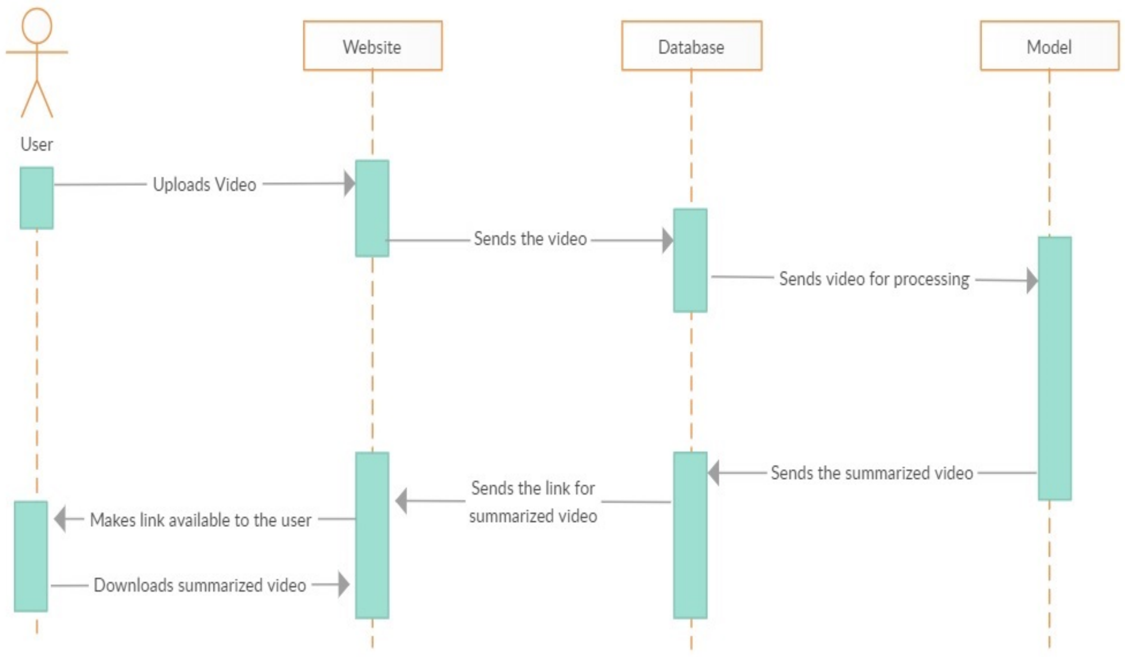

User
Website
Database
Model
Uploads Video
Sends the video
Sends video for processing
Sends the summarized video
Sends the link for summarized video
Makes link available to the user
Downloads summarized video

Figure 4.7: Sequence Diagram

4.3.4 Class Diagram

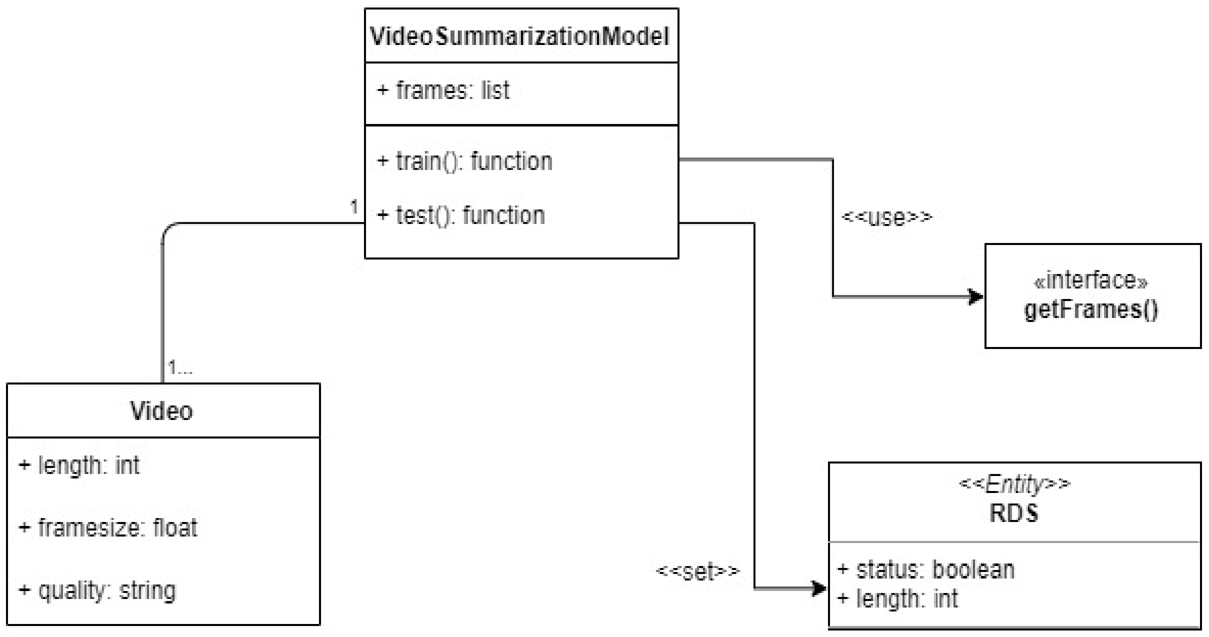


Figure 4.8: Class Diagram

4.3.5 Component Diagram

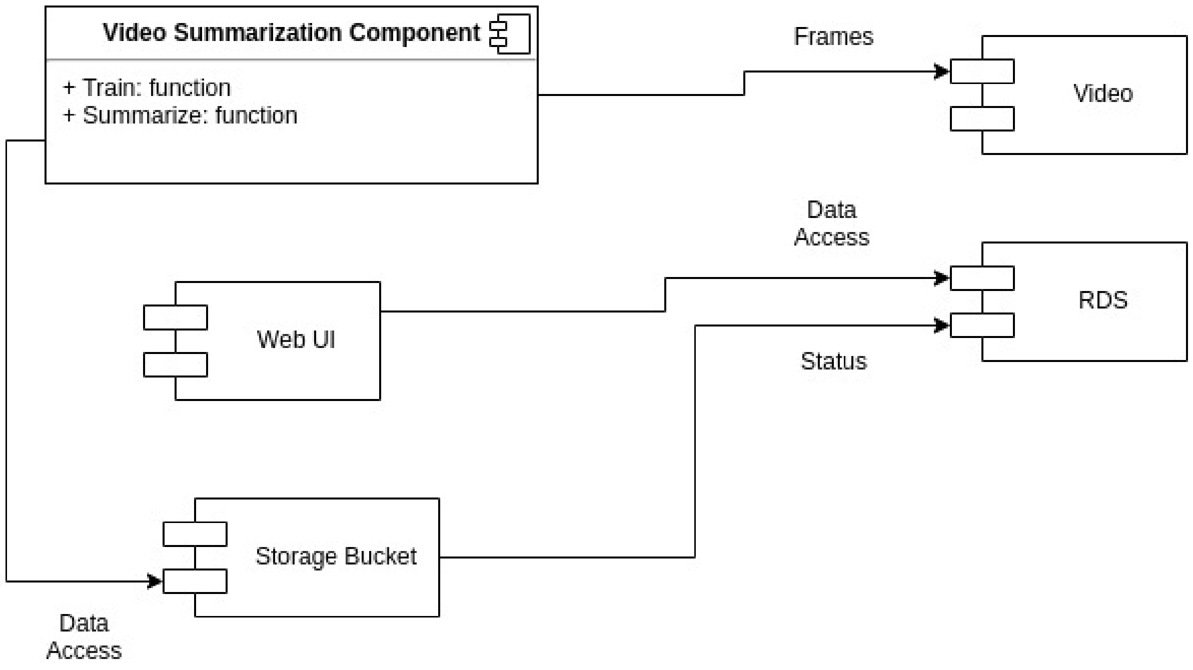


Figure 4.9: Component Diagram

CHAPTER 5

PROJECT PLAN

5.1 Project Estimates

5.1.1 Reconciled Estimates

- Cost: Rs.500 for Amazon services.
- Time: 10 months

5.1.2 Project Resources

- Amazon Sagemaker
- Amazon S3
- Amazon EC2

5.2 Risk Management

Risk Analysis and Management is a key project management practice to ensure that the least number of surprises occur while the project is underway. While we can never predict the future with certainty, we can apply a simple and streamlined risk management process to predict the uncertainties in the project and minimize the occurence or impact of these uncertainties. This improves the chance of successful project completion and reduces the consequences of those risks.

5.2.1 Rish Identification

- RISK ID 1
  - Risk Category: Technical
  - Risk Description: Schedule delay to be expected if the requirements are not clear.
- RISK ID 2
  - Risk Category: Technical

- Risk Description: Time, cost and scope deviation to be expected if requirements will not be final at project kick-off.

- RISK ID 3
  - Risk Category: Project Management
  - Risk Description: Delay to be expected if the software resources are not supplied by the customer.

5.2.2 Risk Analysis

- RISK ID 1 & 2:
  - Project Impact: Nothing can be performed if requirements are not clear yet.
  - Likelihood: Possibility of this risk is low.
  - Consequence: Extent of this risk's impact on the project is significant.
- RISK ID 3:
  - Project Impact: It is important to get software resources in order to start working on the project.
  - Likelihood: Possibility of this risk is low.
  - Consequence: Extent of this risk's impact on the project is severe.

5.3 Project Schedule

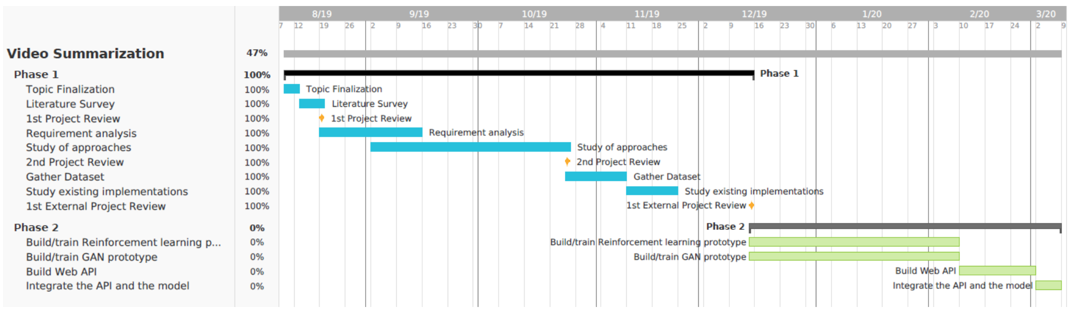


Figure 5.1: Gant Chart

1. June-August 2019: Getting familiar with the problem statement.

2. September 2019: Trying different approaches for video summarization.
3. October-November 2019: Finalizing system architecture, system design and software requirement specification.
4. December-January 2020: Selection of one algorithm based on various parameters.
5. February-April 2020: Creating UI for end users and testing.

CHAPTER 6

PROJECT IMPLEMENTATION

6.1 Overview of Project Modules

6.1.1 Frontend

This module is client-facing component of the project. Users & Clients can authenticate themselves using the login system. Once the users are authenticated, they can upload files from their devices and view the summarized videos on the frontend platform.

6.1.2 Backend

This module is the interface for the model present in the SageMaker and the client-facing frontend module. This will authenticate users, upload the videos and frames to the S3 bucket and stream the summarized videos to the frontend module.

6.1.3 Shot Boundary Detection

This module is executed for pre-processing the uploaded video. This module will detect shot boundaries and create a metadata file for the video. This metadata file is used to pass the video shot by shot to the summarization module.

6.1.4 Video Summarization

This module performs the summarisation of the pre-prossed video given by the shot boundary detection module. It summarises the video using Cycle-GAN and also uses the knapsack algorithm to choose the important shots in the video.

6.2 Tools and Technologies Used

- IDE: Emacs, Visual Studio Code
- CLIs: npm, AWS command line interfaces.
- Python 3: Shot Boundary Detection module is developed in python.

- PyTorch: Video Summarization module is developed in pyTorch.
- OpenCv: Shot Boundary Detection module uses OpenCv to process frames.

6.3 Algorithms Details

6.3.1 Shot Boundary Detection

This algorithm is used during the pre-processing phase. This algorithm uses a deep learning approach using a CNN. It proposes a TransNet architecture. As an input, the network takes a sequence of N consecutive video frames and applies series of 3D convolutions returning a prediction for every frame in the input. Each prediction expresses how likely a given frame is a shot boundary.

The main building block of the model is designed as four 3D $3 \times 3 \times 3$ convolution operations. The convolutions employ different dilation rates for the time dimension and their outputs are concatenated in the channel dimension. This approach significantly reduces the number of trainable parameters compared to standard 3D convolutions with the same field of view. Multiple DDCNN cells on top of each other followed by spatial max pooling form a Stacked DDCNN block. The TransNet consists of multiple SDDCNN blocks, every next block operating on smaller spatial resolution but a greater channel dimension, further increasing the expressive power and the receptive field of the network.

Two fully connected layers refine the features extracted by the convolutional layers and predict a possible shot boundary for every frame representation independently. Rectified Linear Unit (ReLU) activation function is used in all layers with the only exception of the last fully connected layer with softmax output. Stride 1 and the 'same' padding is employed in all convolutional layers. [8]

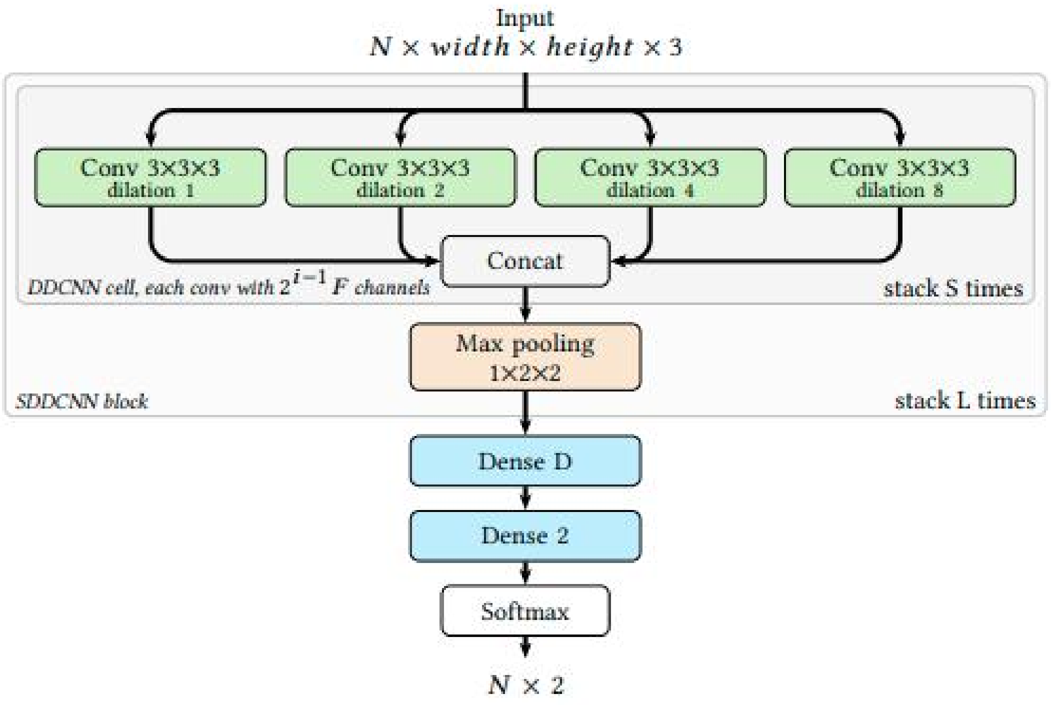


Figure 6.1: TransNet Architecture

6.3.2 0/1 Knapsack

We apply solution of the 0/1 Knapsack problem using dynamic programming for choosing frames in the summarized video. The length of summarized video is 15% of the original video. Hence, based on the scores of the frames we choose all the frames which add up to $0.15 * lengthOfOriginal$.

Problem: Given two n tuples $\{s_1, s_2, \ldots, s_n\}$ and $\{w_1, w_2, \ldots, w_n\}$, and $W > 0$, we wish to determine the subset of $S \subseteq \{1, 2, \ldots, n\}$ that maximizes $\sum s_i$ subject to $\sum w_i \leq W$, where $W = 0.15 * lengthOfOriginal$.

Solution:

```
Knapsack(v,w,n,W){
    for (w=0 to W) V[0,w]=0
    for (i=1 to n)
        for(w=0 to W)
            if(w[i]≤W)
                V[i,w]=max(V[i-1,w],v[i]+V[i-1,W-w[i]])
            else
                V[i,w]=V[i-1,w]
    return V[n,W]
}
```

6.3.3 Cycle-GAN

A model termed Cycle-SUM, which consists of a frame selector and a cycle consistent learning-based evaluator is used to generate summaries. The selector is a bidirectional LSTM network that learns the video representations that embed the long-range relationships among the video frames. The evaluator consists of two GANs, in which the forward GAN is learned to reconstruct original video from summary video, while the backward GAN learns to invert the processing.

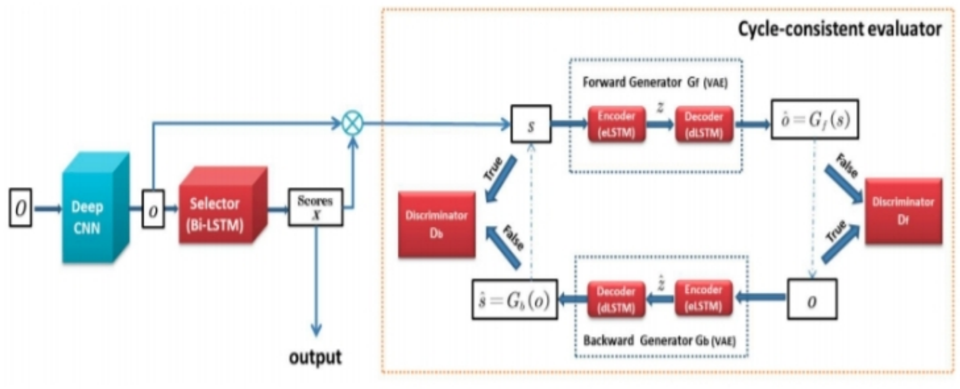


Figure 6.2: Cycle GAN

The cycle-consistent learning objective is used to maximize the mutual information between the summary video s and the original video o. Formally, the mutual information *I(o, s)* is defined as

$$I(o,\ s) \triangleq \sum_{o} p(o) D_{KL}(p(s|o)||p(s)),$$

where D_{KL} is the KL-divergence between two distributions. The video summarization model tries to produce s such that its conditional distribution *p(s—o)* gives the maximal mutual information with *p(s)*. However, though it is easy to obtain empirical distribution estimation of original video o, it is difficult to obtain ground truth distribution *p(s)* of corresponding s in an unsupervised learning scenario.
A cycle-consistent learning objective can be used to relieve such learning difficulty. We notice that,

$$I(o,\ s) = \frac{1}{2}\left[\sum_{o} p(o) D_{KL}(p(s|o)||p(s)) + \sum_{s} p(s) D_{KL}(p(o|s)||p(o))\right] \tag{6.1}$$

The above mutual information computation "anchors" at *p(o)* that can be faithfully estimated and thus eases the procedure of learning distribution of s even in an unsupervised learning setting.
To effectively model and optimize the above learning objective,the Fenchel conjugate to derive its bound that is easier to optimize is adopted. The Fenchel conjugate of a function f is defined as $f^*(t) \triangleq sup_{u \in domf}[ut - f(u)]$, or equivalently $f(u) = sup_{t \in domf^*}[ut - f^*(t)]$.

Thus, defining $f(u) = logu$, we have the following upper bound for the KL-divergence between distributions p and q:

$$\begin{aligned} D_{KL}(p||q) &= -\sum_x q(x)\log\frac{p(x)}{p(x)} \\ &= -\sum_x q(x)\sup_t\left(t\frac{p(x)}{q(x)} - f^*(t)\right) \\ &= -\sum_x q(x)\sup_t\left(t\frac{p(x)}{q(x)} + 1 + \log(-t)\right) \\ &\leq -\sup_{T\in\tau}\left(\sum_x p(x)(x) + \sum_x q(x)\log(1-T(x))\right) \end{aligned}$$

where $t = T(x) - 1$ and τ is an arbitrary class of functions $T : X \to R$.The above inequality is due to the Jensen's inequality and functions is only a subset of all possible functions. Therefore, we have

$$\begin{aligned} &D_{KL}(p(s|o)||p(s)) \\ &\leq -\sup_{T\in\tau}\left(\sum p(s|o)(s) + \sum p(s)\log(1-T(s))\right) \\ &\sim -\sup_{T\in\tau}\frac{1}{|S|}\left(\sum_s^{p(s|o)}\log T(s) + \sum_s^{p(s|o)}\log(1-T(s))\right) \end{aligned}$$

Here S is the set of produced summary videos. We can use a generative model to estimate *p(s—o)*. We followed a generative-adversarial, which uses two neural networks, G_f and D_f , to implement sampling and data transformation. Here G_f is the forward generative model, taking the condition o as input and outputting a sample of summary sample *s*. D_f is the forward discriminator model. We learn the generative model G_f by finding a saddle-point of the above objective function, where we minimize w.r.t. G_f and maximize w.r.t. D_f :

$$\min_{G_f}\max_{D_f}\mathcal{L}(G_f, D_f) = \frac{1}{|S|}\left(\sum_s^{G_f(o)}\log D_f(s) + \sum_s^{p(o)}\log(1-D_f(s))\right) \tag{6.2}$$

The above objective is similar to the one of GANs, but the generative model is a conditioned one.

Similarly, we can obtain the learning objective to optimize the KL-divergence $D_{KL}(p(o|s)||p(o))$ by solving

$$\min_{G_b}\max_{D_b}\mathcal{L}(G_b, D_b) = \frac{1}{|O|}\left(\sum_s^{G_f(s)}\log D_b(o) + \sum_o^{p(o)}\log(1-D_b(o))\right) \tag{6.3}$$

Substituting 6.2 and 6.3 into 6.1 gives the following cycle learning objective to maximize the mutual information between the original and summary video:

$$\min_{G_f, G_b} \max_{D_f, D_b} \mathcal{L}(G_f, D_f, G_b, D_b) =$$

$$\left(\sum_{s}^{G_f(o)} \log D_f(s) + \sum_{s}^{p(s)} \log(1 - D_f(s)) + \sum_{s}^{G_b(s)} \log D_b(o) + \sum_{o}^{p(o)} \log(1 - D_b(o)) \right),$$

where the constant number of original frames is omitted. To relieve the difficulties brought by the unknown distribution *p(s)*, we use the following cycle-consistent constraint to further regularize the generative model and the cycle learning processing:

$$Gb(Gf(o)) \sim o,$$

and

$$Gf(Gb(s)) \sim s$$

The cycle learning with the above consistent constraint is called as the cycle-consistent learning.

CHAPTER 7
SOFTWARE TESTING

7.1 Overview of Project Testing

7.1.1 Unit Testing

It is the most popular from tests that can be used to assure the developer that a particular method would suffice the requirements of the product being delivered. Many utility methods and abstractions can be tested with the use of unit testing. Unit tests are also, often, simpler tests to write and to run.

7.1.2 Alpha Testing

The objective of this form of testing is to identify all possible issues or defects before releasing it into a production environment. Alpha testing is carried out at the end of the software development phase but before the Beta Testing. Still, minor design changes may be made as a result of such testing. Alpha Testing is conducted at the developer's site. Inhouse virtual user environment was created for this type of testing.

7.1.3 Acceptance Testing

An acceptance test is performed by the client and verifies whether the end to end the flow of the system is as per the business requirements or not and if it is as per the needs of the end user. Client accepts the software only when all the features and functionalities work as expected. It is the last phase of the testing, after which the software goes into production.

7.1.4 Ad-Hoc Testing

The name itself suggests that this testing is performed on an ad-hoc basis i.e. with no reference to the test case and also without any plan or documentation in place for such type of testing. The objective of this testing is to find the defects and break the application by executing any flow of the application or any random functionality. Ad-hoc testing is an informal way of finding defects and can be performed by anyone in the project. It is difficult to identify defects without a test case but sometimes it is possible that defects found during ad-hoc testing might not have been identified using existing test cases.

7.1.5 Accessibility Testing

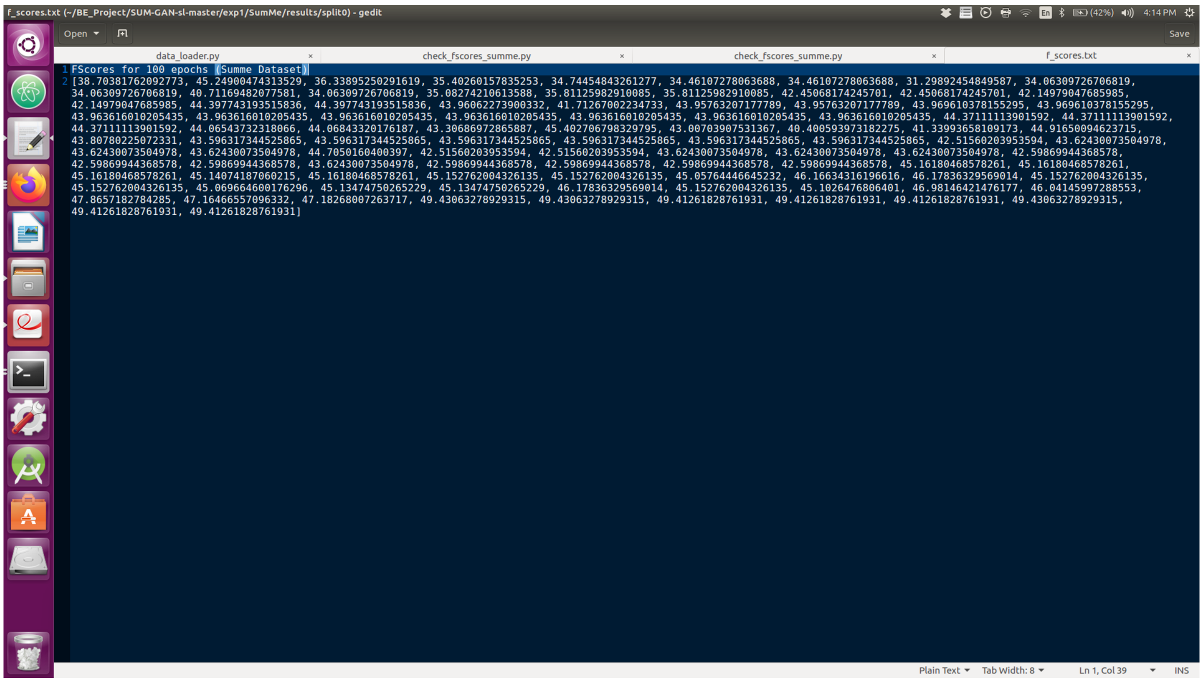
The aim of accessibility testing is to determine whether the software or application is accessible for disabled people or not. Here disability means deaf, color blind, mentally disabled, blind, old age and other disabled groups. Various checks are performed such as font size for visually disabled, color and contrast for color blindness etc.

7.1.6 Snapshot Testing

Developing UI's that are pixel perfect can achieved with the help snapshot tests. To make sure the UI is consistent throughout the app's usage, we require snapshot testing. We used Jest framework to test the UI on our frontend platform.

CHAPTER 8
RESULTS

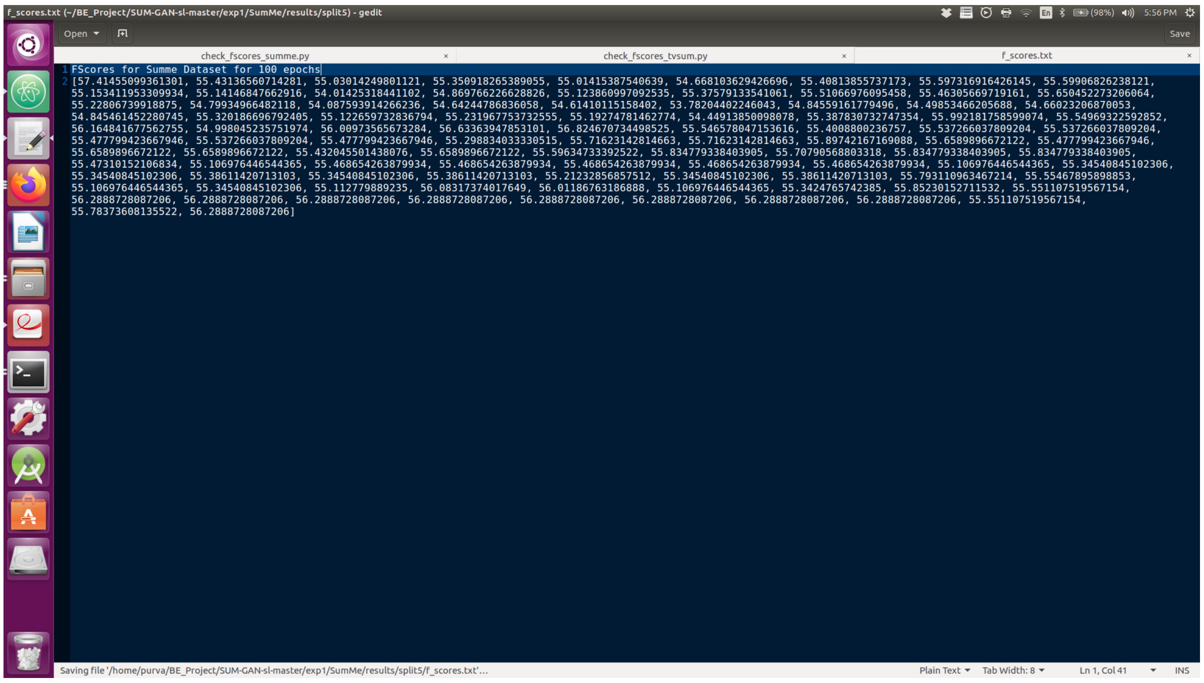
8.1 Unsupervised Video Summarization with Ad-versarial LSTM Networks



```
FScores for 100 epochs (Summe Dataset)
[38.70381762092773, 45.24900474313529, 36.33895250291619, 35.40260157835253, 34.74454843261277, 34.46107278063688, 34.46107278063688, 31.29892454849587, 34.06309726706819,
34.06309726706819, 40.71169482077581, 34.06309726706819, 35.08274210613588, 35.81125982910085, 35.81125982910085, 42.45068174245701, 42.45068174245701, 42.14979047685985,
42.14979047685985, 44.397743193515836, 44.397743193515836, 43.96062273900332, 41.71267002234733, 43.95763207177789, 43.95763207177789, 43.969610378155295, 43.969610378155295,
43.963616010205435, 43.963616010205435, 43.963616010205435, 43.963616010205435, 43.963616010205435, 43.963616010205435, 43.963616010205435, 44.37111113901592, 44.37111113901592,
44.37111113901592, 44.06543732318066, 44.06843320176187, 43.30686972865887, 45.402706798329795, 43.00703907531367, 40.400593973182275, 41.33993658109173, 44.91650094623715,
43.80780225072331, 43.596317344525865, 43.596317344525865, 43.596317344525865, 43.596317344525865, 43.596317344525865, 43.596317344525865, 42.51560203953594, 43.62430073504978,
43.62430073504978, 43.62430073504978, 44.7050160400397, 42.51560203953594, 42.51560203953594, 43.62430073504978, 43.62430073504978, 43.62430073504978, 42.59869944368578,
42.59869944368578, 42.59869944368578, 43.62430073504978, 42.59869944368578, 42.59869944368578, 42.59869944368578, 42.59869944368578, 45.16180468578261, 45.16180468578261,
45.16180468578261, 45.14074187060215, 45.16180468578261, 45.152762004326135, 45.152762004326135, 45.05764446645232, 46.16634316196616, 46.17836329569014, 45.152762004326135,
45.152762004326135, 45.069664600176296, 45.13474750265229, 45.13474750265229, 46.17836329569014, 45.152762004326135, 45.1026476806401, 46.98146421476177, 46.04145997288553,
47.8657182784285, 47.16466557096332, 47.18268007263717, 49.43063278929315, 49.43063278929315, 49.41261828761931, 49.41261828761931, 49.41261828761931, 49.43063278929315,
49.41261828761931, 49.41261828761931]
```

Figure 8.1: Screenshot of F1 scores for 100 epochs

8.2 Unsupervised Video Summarization via Attention-Driven Adversarial Learning



```
FScores for Summe Dataset for 100 epochs
[57.41455099361301, 55.43136560714281, 55.03014249801121, 55.350918265389055, 55.01415387540639, 54.668103629426696, 55.40813855737173, 55.597316916426145, 55.59906826238121,
55.153411953309934, 55.14146847662916, 54.01425318441102, 54.869766226628826, 55.123860997092535, 55.37579133541061, 55.51066976095458, 55.46305669719161, 55.650452273206064,
55.22806739918875, 54.79934966482118, 54.087593914266236, 54.64244786836058, 54.61410115158402, 53.78204402246043, 54.84559161779496, 54.49853466205688, 54.66023206870053,
54.845461452280745, 55.320186696792405, 55.122659732836794, 55.231967753732555, 55.19274781462774, 54.44913850098078, 55.387830732747354, 55.992181758599074, 55.54969322592852,
56.164841677562755, 54.998045235751974, 56.00973565673284, 56.63363947853101, 56.824670734498525, 55.546578047153616, 55.4008800236757, 55.537266037809204, 55.537266037809204,
55.477799423667946, 55.537266037809204, 55.477799423667946, 55.298834033330515, 55.71623142814663, 55.71623142814663, 55.89742167169088, 55.6589896672122, 55.477799423667946,
55.6589896672122, 55.6589896672122, 55.432045501438076, 55.6589896672122, 55.59634733392522, 55.834779338403905, 55.70790568803318, 55.834779338403905, 55.834779338403905,
55.47310152106834, 55.106976446544365, 55.468654263879934, 55.468654263879934, 55.468654263879934, 55.468654263879934, 55.468654263879934, 55.106976446544365, 55.34540845102306,
55.34540845102306, 55.38611420713103, 55.34540845102306, 55.38611420713103, 55.21232856857512, 55.34540845102306, 55.38611420713103, 55.793110963467214, 55.55467895898853,
55.106976446544365, 55.34540845102306, 55.112779889235, 56.08317374017649, 56.01186763186888, 55.106976446544365, 55.3424765742385, 55.85230152711532, 55.551107519567154,
56.2888728087206, 56.2888728087206, 56.2888728087206, 56.2888728087206, 56.2888728087206, 56.2888728087206, 56.2888728087206, 56.2888728087206, 55.551107519567154,
55.78373608135522, 56.2888728087206]
```

Saving file '/home/purva/BE_Project/SUM-GAN-sl-master/exp1/SumMe/results/split5/f_scores.txt'...

Figure 8.2: Screenshot of F1 scores for 100 epochs

8.3 Frontend Platform

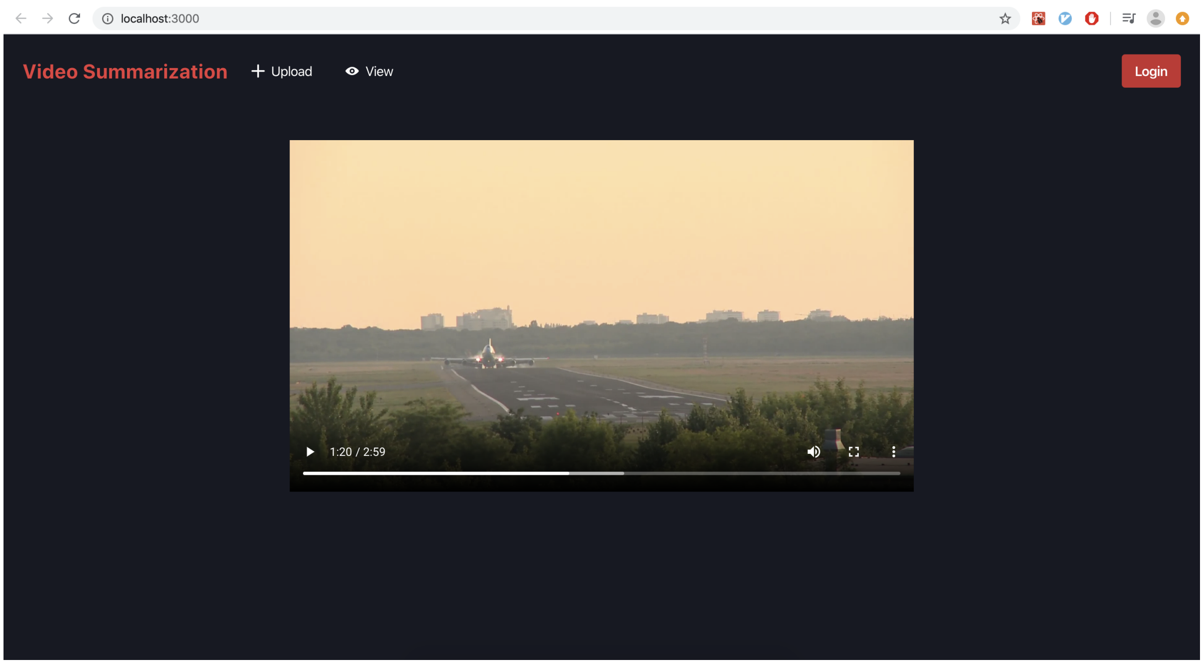


Figure 8.3: Screenshot of Frontend Platform showcasing a video

CHAPTER 9
CONCLUSION

9.1 Conclusion

We studied various algorithms which support the problem statement. These algorithms helped us to work on a wide range of methods and libraries to improve efficiency of the final algorithm. With this project we successfully built an application which will summarize uploaded videos and allow the user to download the summarised videos.

9.2 Future Work

- Stream processed video from within the platform.
- Use audio as an aspect to summarize the video providing the user a consistent experience with respect to the original content.
- Summarize the video according to time constraint provided by the user.

9.3 Applications

- System can be applied to streaming services.
- System can help busy users watch more amount of videos in limited time.

ANNEXURE A
FEASIBILITY STUDY

A.1 Problem Statement Feasibility

A.1.1 Technical Feasibility

- The web platform can run smoothly on Amazon Web Services.
- Training of the model can be done on the NVIDIA GTX 1080Ti GPUs.

A.1.2 Economic Feasibility

- The GPUs required are present in college, so no extra cost will be incurred there.
- Amazon Web Services offer free tokens up to a year which fits in our time frame of the project.

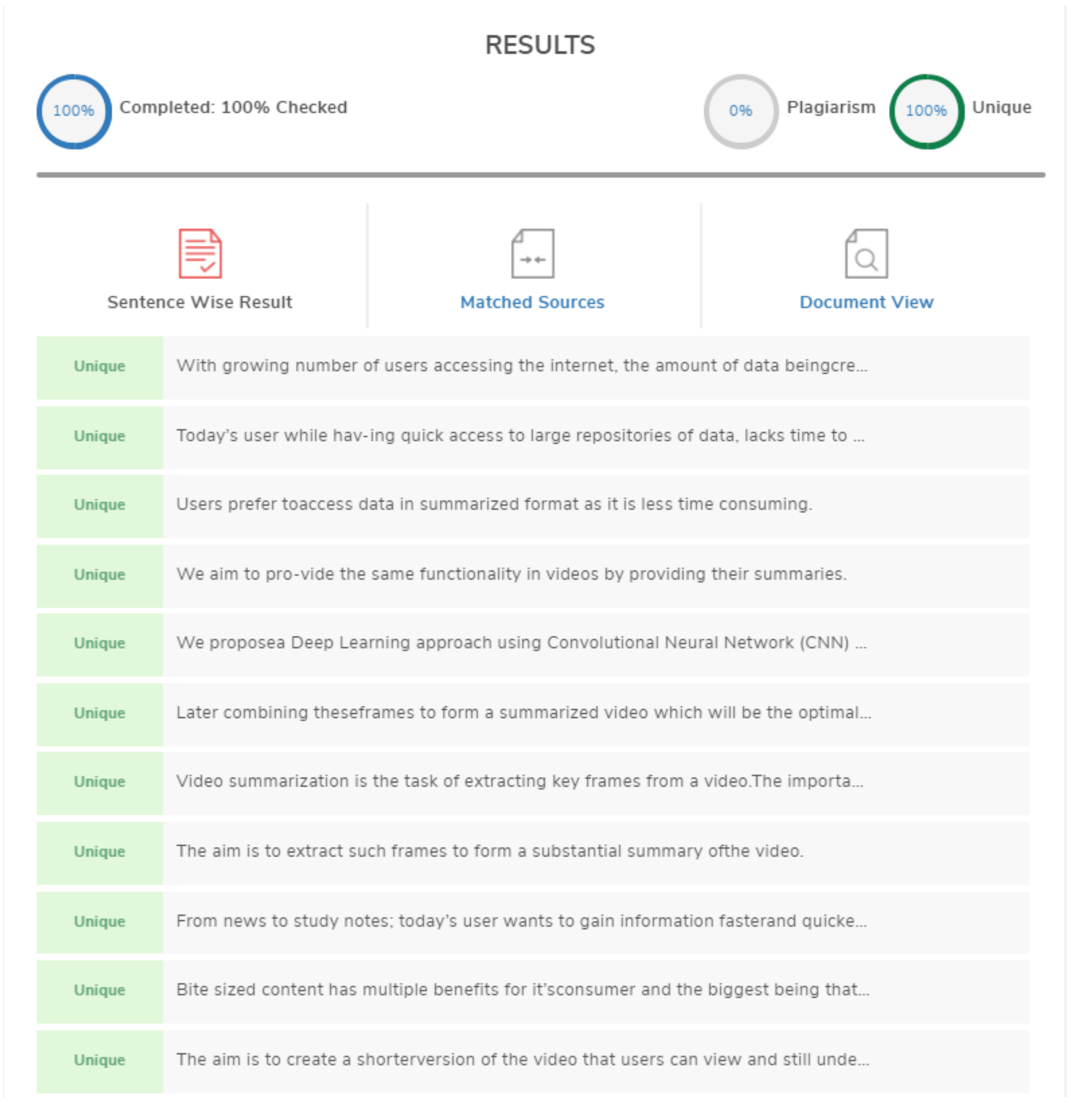
A.1.3 Legal Feasibility

- All the papers we're referring to are published on Arxiv, so no legal lapses there.
- All the code repositories we are referring to have an MIT Open Source Software license.

A.1.4 Scheduling Feasibility

- The entire project will take 10 months to be completed.

ANNEXURE B
PLAGIARISM REPORT



RESULTS

Unique	With growing number of users accessing the internet, the amount of data beingcre...
Unique	Today's user while hav-ing quick access to large repositories of data, lacks time to ...
Unique	Users prefer toaccess data in summarized format as it is less time consuming.
Unique	We aim to pro-vide the same functionality in videos by providing their summaries.
Unique	We proposea Deep Learning approach using Convolutional Neural Network (CNN) ...
Unique	Later combining theseframes to form a summarized video which will be the optimal...
Unique	Video summarization is the task of extracting key frames from a video.The importa...
Unique	The aim is to extract such frames to form a substantial summary ofthe video.
Unique	From news to study notes; today's user wants to gain information fasterand quicke...
Unique	Bite sized content has multiple benefits for it'sconsumer and the biggest being that...
Unique	The aim is to create a shorterversion of the video that users can view and still unde...

Figure B.1: Plagiarism Report

CHAPTER 10
REFERENCES

[1] B. Mahasseni, M. Lam, and S. Todorovic, “Unsupervised video summarization with adversarial lstm networks,” in *Proc. IEEE Conf. Comput. Vis. Pattern Recognit*, pp. 1–10, 2017.

[2] K. Zhou, Y. Qiao, and T. Xiang, “Deep reinforcement learning for unsupervised video summarization with diversity-representativeness reward,” p. 8, December 2017.

[3] L. Yuan, F. E. H. Tay, P. Li, L. Zhou, and J. Feng, “Cycle-sum: Cycle-consistent adversarial LSTM networks for unsupervised video summarization,” *CoRR*, vol. abs/1904.08265, 2019.

[4] Y. Zhang, M. Kampffmeyer, X. Liang, D. Zhang, M. Tan, and E. P. Xing, “Dilated temporal relational adversarial network for generic video summarization,” *Multimedia Tools and Applications*, vol. 78, pp. 35237–35261, Dec 2019.

[5] B. Zhao and E. P. Xing, “Quasi real-time summarization for consumer videos,” in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 2513–2520, June 2014.

[6] K. Bhattacharya, S. Chaudhury, and J. Basak, “Video summarization: A machine learning based approach.,” pp. 429–434, 01 2004.

[7] E. Apostolidis, E. Adamantidou, A. I. Metsai, V. Mezaris, and I. Patras, “Unsupervised video summarization via attention-driven adversarial learning,” in *MultiMedia Modeling* (Y. M. Ro, W.-H. Cheng, J. Kim, W.-T. Chu, P. Cui, J.-W. Choi, M.-C. Hu, and W. De Neve, eds.), (Cham), pp. 492–504, Springer International Publishing, 2020.

[8] T. Souček, J. Moravec, and J. Lokoč, “Transnet: A deep network for fast detection of common shot transitions,” *arXiv preprint arXiv:1906.03363*, 2019.