INFORMATICS INSTITUTE OF TECHNOLOGY

INFORMATICS INSTITUTE OF TECHNOLOGY

In Collaboration with

UNIVERSITY OF WESTMINSTER

A Novel Abstractive Text Summarization approach for Lecture Content

A Dissertation by

Mohamed Boosary Fathima Farheen

Supervised by

Mr. Prasan Yapa

Submitted in partial fulfillment of the requirements for the BEng (Hons) in Software Engineering degree at the University of Westminster.

July 2022

DECLARATION

I hereby declare that the work, which is being presented in this dissertation, entitled **"A Novel Abstractive Text Summarization approach for Lecture Content"** in fulfillment of the requirements for the degree of BEng (Hons) in Software Engineering is an authentic record of my own work carried out during the final year period under the supervision of Mr. Prasan Yapa. To the best of my knowledge, it does not contain any material published or written by another person, except for the information derived from the literature, which has been duly acknowledged in the text by providing proper citations and respective references. No part of this dissertation was previously presented for another degree or diploma at this or any other university/institution.

Student Full Name:- Mohamed Boosary Fathima Farheen

Student Registration No:- 2018323 / w1742063

Signature:  Data: 24th July 2022

ABSTRACT

In the current technological era, with the advancement of information and communication technology, textual information can be found in abundance in a variety of formats for different topics such as news articles, educational transcripts, medical documents, legal documents, etc making it difficult for user(s) to comprehend and extract the significant information with ease in a short period. Hence it needed a technique to be able to condense this many textual information into a summary that concisely includes the salient contents of the main source. The field of automatic text summarization (ATS) which has been in existence since the 1950's is anticipated to find solutions to this issue by automating the process of concise summary creation. However, the machine-generated summaries are still far from human-generated summaries as the machine-generated summaries mostly suffered from issues such as repetition of words, not being able to handle out of vocabulary words (OOV), less semantic, etc.

The research study focused on designing and developing a system to generate concise coherent summaries for lecture video content by utilizing lecture video transcripts for that the system employed an adversarial process for abstractive text summarization which adopts a Generative Adversarial Network (GANs) architecture that utilized a transformer-based pointer generator network for the generator component and CNN based text classifier for the discriminator to address the research gap that identified within the existing literature. The author was able to achieve satisfactory result when compared to other international researchers in abstractive text summarization domain within given the limited resources and time frame.

Keywords: Natural Language Processing, Automatic Text Summarization, Extractive Text Summarization, Abstractive Text Summarization, Pointer Generator Network, Transformers, Generative Adversarial Network.

ACKNOWLEDGEMENT

First and foremost, I would like to express my deep and sincere gratitude to my project supervisor Mr. Prasan Yapa for accepting me to do this research project under his supervision and for providing invaluable continuous support throughout the project timeline. He has taught me the methodology on how to carry out this type of research project and how to present the research work clearly as possible. It was a great privilege to work under his guidance. I am extremely grateful for what he has offered me.

Secondly, I would like to express my appreciation for our final year module leader Mr. Guhanathan Poravi and to the entire IIT final year project committee panel who worked throughout the final year project in order to provide us with all the needed necessary guidance in successfully complete the project.

Next, I would like to convey my appreciation for all the domain and technical experts who provided valuable feedback with their expert knowledge in order to make this research project a success.

I also like to take this opportunity to thank my colleagues who provide me with their valuable feedback and support whenever needed.

Last but not least I would like to thank my beloved parents and my sibling for understanding me and for being there for me by encouraging me throughout the completion of the project.

TABLE OF CONTENT

LIST OF FIGURES

LIST OF TABLES

LIST OF GLOSSARIES

Acronym	Long Form
NLP	Natural Language Processing
ATS	Automatic Text Summarization
CNN	Convolutional Neural Network
RNN	Recurrent Neural Network
LSTM	Long Short-Term Memory Network
Seq2Seq	Sequence to Sequence
OOV	Out of vocabulary
GANs	Generative Adversarial Networks
RL	Reinforcement Learning
FR	Functional Requirement
NFR	Non-Functional Requirement
LCS	longest common subsequent
BERT	Bidirectional Encoder Representations from Transformers
TF-IDF	Term Frequency - Inverse Document Frequency

CHAPTER 1 : INTRODUCTION

1.1 Chapter Overview

The introduction chapter delivers a systematic detailed overview of the research carried out. Initially, an in-depth introduction related to the research problem domain is provided to point out the area of research problem identified by the author. Followed by that a brief outline of the problem definition is provided. The aim and objectives of the research that is required to drive the project are presented. Then the novelty of the research is pointed out with relative to the research gaps identified through the existing works. Finally, the chapter ends with an explanation of the research challenges related to the area to point out the significance and complexity of the research study.

1.2 Problem Domain

1.2.1 Automatic Text Summarization

With the technological evolution, the use of web resources on the internet (such as websites, news, blogs, social media, etc.) in today's world has increased dramatically, and as a result, textual data on the web has grown exponentially. Because of this massive increment in textual data, it has become laborious for users to locate the relevant information from a large volume of data in a short period of time (El-Kassas et al., 2021; Widyassari et al., 2020). Because of this issue, it has emerged the need for a process to summarize these many data in an efficient way to deliver the essence of the given textual data concisely. Manually summarization for extraction of salient information from large corpora of data is inefficient and impractical, as it consumes a longer period for user(s) to comprehend the context of massive textual documents and to identify the important keywords, sentences, phrases in the document which will also cause information loss in the end summary (El-Kassas et al., 2021; Gambhir and Gupta, 2017).

As a solution for this dilemma, around 1950’s the process of automation of text summarization known as **automatic text summarization** emerged as an essential part of natural language processing (NLP) and has been used in diverse domains to create concise coherent summaries within a short period of time, that contains the salient information of the original document with the aim to reduce the redundant and duplicate content in the end summary (El-Kassas et al., 2021; Modi and Oza, 2018).

1.2.2 Automatic Text Summarization Approaches

According to the architectural difference of summary creation, automatic text summarization processes are in two main types, which are extractive and abstractive. **Extractive Text Summarization** approach produces the end summaries by joining the most relevant/important sentences extracted from the original document in which the end summary will be an extracted summary that contains sentences, phrases, words that are included in the source document (El-Kassas et al., 2021; Gambhir and Gupta, 2017). In contrast, **Abstractive Text Summarization** approach produces abstract summaries with novel phrases and words that do not contain in the original document. Because of this ability to include new words, phrases, sentences that do not contain in the original document, abstractive text summaries are defined as mostly resemble to human-generated summaries (El-Kassas et al., 2021; Gambhir and Gupta, 2017; Modi and Oza, 2018; Widyassari et al., 2020).

1.2.3 Abstractive Text Summarization

As mentioned in the preceding section abstractive text summarization is one of the automatic text summarization approaches that can be utilized in the creation of automatic abstract summaries. The summary generated by the abstractive summarization approach resembles closely the human-generated summary, because of its ability to produce novel words, phrases that do not contain in the source document. So, because of this generation of novel words, phrases in the end summary, abstractive summarization approach is said to be quite complex, as to produce quality abstract summary it needs to represent the original document in an intermediate text representation to comprehend the context of the document for generation of novel words that does not contain in the original document. However, due to the availability of lack of techniques in the early days for text representation to understand the context of the text document and generation of the novel word and phrases, the research works investigated in this field was lesser than that of the extractive approach (El-Kassas et al., 2021; Zhang et al., 2020). But with the recent development in deep NLP techniques, the abstractive summarization process has provided remarkable results on the production of quality summaries with abstractive methods providing importance for research significant in this area. Also, as extensive research has already been done in the extractive summarization field researchers are more focused on investigating newer techniques in producing quality summaries using abstractive methods (Zhang et al., 2020).

1.2.4 Automatic Text Summarization for Lecture Video Content

With the current technology development, it has been made easy for user(s) to learn educational content through lecture videos on online platforms such as MOOCs (Massive Open Online Course), YouTube, etc. Compared to other learning process, lecture videos have been proven to make the learning process efficient as it delivers rich content of a particular topic to the user in an understandable format (Andra and Usagawa, 2019; Miller, 2019). Most of the time the user(s) are in need of revising lecture video content material that they might have previously referred or in need to refer in a necessary timeline such as when preparing for exams, interviews, and so on, making it is tedious for user(s) to go through the whole video in capturing the main information of the lecture video. So having an automatic text summarizer for the generation of summary that contains the gist of the lecture video content can be power up the learning process of the user(s) effectively (Miller, 2019; Vali et al., 2021). Lecture video can be summarized using several techniques such as image process, video processing, and text processing. Summarization of video lecture using image processing and video processing focuses mostly on the visually presented lecture videos and not with audio-only lecture video content in creation of summaries. Text processing for summarization, on the other hand, can utilize audio-rich lecture videos in the creation of coherent summaries.

1.3 Problem Definition

In education, lecture videos benefit user(s) to easily acquire knowledge of terms, basic facts, and simple concepts for a specific subject. User(s) might refer to a specific lecture video that they might have already referred to when they are in need of preparation for any near exams to get refreshed up with the related topic. But with the time constraint, it would be difficult to locate the valuable information within a short time. So rather than going through the whole lecture again, it would be beneficial for the user(s) to have a short concise informative summary that contains the essence of the related lecture video content concisely. Almost all the existing summarization systems related to lectures use dated traditional natural language processing techniques which face generalization issues and as well as which will produce subpar summaries which is not much useful to get the gist of the lecture content (Miller, 2019) and also almost all the of the text summarizers related to lecture domain is built by utilizing extractive summarization techniques which is simple and robust but contains more redundant, meaningless content in the end summary that makes the summary subpar.

1.3.1 Problem Statement

Current lecture video content summarization systems do not provide informative abstract summaries which contain the factual data that can be used to get the essence of the lecture video content. Having a lecture summarizer that generates a concise, coherent summary which contains salient information of the lecture video, will benefit the user(s) to easily comprehend the lecture content within a shorter time period

1.4 Research Motivation

By Investigating the use of existing related works around automatic text summarization in the lecture domain it was observed the lack of lecture summarizers that use the abstractive text summarization approach to create concise abstract summaries which could provide end-user(s) to get an overall gist of the lecture content in a short time without going through the whole lecture video content. Motivated from that fact along with the aim of contributing a novel improvement for the GAN-based abstractive text summarization process has led to initiate this research project.

1.5 Existing Works

1.5.1 Automatic Summarization in Lecture Domain

The below table includes a list of existing works of automatic summarizations either be it text or other techniques that have been done around the lecture domain.

1) Citation: (Urala Kota et al., 2018)
Technology: • Use of video processing techniques for segmentation of lecture video in creation of lecture transcript. • Use of pyAudioAnalysis library for audio extraction to create the transcript of the segmented video parts. • Use of TF-IDF and Cosine Similarity matrix for text summarization.
Improvement: • Use of video processing and text processing for creation of efficient summaries for lecture content.
Limitations:

- The system will prone to topic overlapping issues and some important points of the lecture video might not be included in the final summary.
- Generate only extractive summaries with traditional NLP approaches such as the utilization of graph-based approach.

2) Citation: (Miller, 2019)

Technology:

- Utilize pre-trained transformer model BERT for text embedding.
- Use of K-means clustering for selection of relevant sentences from cluster embeddings for creation of final summary.

Improvement:

- Employ state-of-art transfer learning approach BERT for extractive text summarization of lecture transcript.

Limitations:

- When the lecture content consists of 100 or more sentences, it is difficult to compress and represent the entire lecture context in a short summary.
- Contains words such as “this”, “those”, “these” in the end summary which does not much elaborate on the main context of the original document.
- Generate only extract summaries.
- Handling spoken language over written language.

3) Citation: (Xu et al., 2019)

Technology:

- Use of RLSA (run-length smoothing algorithm) for identification of entities in lecture video slides.
- Use of WMD (Word Mover’s Distance) to map the identified entity in lecture video slide to the audio transcript.

Improvement:

- Creation of a new framework that produces lecture notes based on the comparison between the extracted visual entity and respective speech description.

Limitations:

- The system required to have lecture videos that contain visual content for creation of lecture summaries.

• Does not elaborate the generic idea of video content in an informative way, as it only creates the slide in the lecture video as the final summary (lecture note), • Utilize extractive video summarization approach for creation of end summary.
4) Citation: (Andra and Usagawa, 2019)
Technology: • Use of powerSeg method for segmentation of lecture transcript. • Use of attention-based Seq2Seq model that contains LSTM for both encoder and decoder component for summarization of segmented transcripts.
Improvement: • Enhance the coherency and the quality of the end summary by utilizing segmentation and linguistic-based mechanisms. • Use of attention mechanism for capture the important key information for the final summary. • Creation of abstract summaries.
Limitations: • Consume long period for model training. • Difficult in capturing the context of a lengthy document.
5) Citation: (Vali et al., 2021)
Technology: • Use of TF-IDF and Cosine Similarity matrix
Improvement: • Purposed a novel framework to generate summaries for lecture videos by utilizing the textual transcript created from the extracted audio of the video.
Limitations: • Use of traditional NLP techniques in creation of final extract summary. • Use of extractive summarization approach in creation of extract summaries

Table 1 : Existing works on automatic summarization in lecture domain

1.5.2 GANs for Abstractive Text Summarization

The below table includes a list of existing works done around abstractive text summarization mostly considering the state-of-art GANs architecture.

1) Citation: (Liu et al., 2017)

Technology:

- GANs architecture with reinforcement learning,
  - **Generator** → Seq2Seq model with an encoder that consists of bi-directional LSTM and decoder that consists of attention-based LSTM
  - **Discriminator** → CNN for text classification to identify real and fake summaries
- Pointer-generator network

Improvement:

- Use of policy gradient in reinforcement learning to avoid exposure bias and non-differentiable task metrics issue in seq2seq model.
- Generation of diverse summaries that are more abstractive and readable.

Limitations:

- Model does not perform well in long text summarization.

2) Citation: (Xu et al., 2018)

Technology:

- GANs architecture with reinforcement learning,
  - **Generator** → Attention-based seq2seq model consist of an encoder and decoder that contains GRU layers.
  - **Discriminator** → Triple RNNs for text classification.

Improvement:

- The model was designed with the purpose to overcome the issue of exposure bias faced by sequence-to-sequence models in the inference (testing) stage.
- Introducing a novel triple RNN approach for discriminator as an improvement for classification of text.

Limitations:

- Generated summaries consist of duplicate phrases.

3) Citation: (Zhuang and Zhang, 2019)

Technology:

- GANs architecture with reinforcement learning,
  - **Generator** → use of Seq2Seq model
  - **Two Discriminators** → Similarity discriminator - CNN as the classifier
    → Readability discriminator - CNN as the classifier

Improvement:

- Introduced a novel dual discriminator which contained a similarity discriminator and readability discriminator. Similarity discriminator is used to identify if the generated summary contains the relevant information as of the input original document (text) and readability discriminator teach generator to produce human-readable summaries.
- Generation of semantic abstract summaries which contain the salient information of the original document.

Limitations:

- Having a fixed length for output summary, the model finds it difficult in generating summaries that contain longer sequences.

4) Citation: (Rekabdar et al., 2019)

Technology:

- GANs architecture with reinforcement learning,
  - **Generator** → Attention-based seq2seq model with encoder consist of bidirectional LSTM and decoder with unidirectional LSTM.
  - **Discriminator** → CNN for text classification

Improvement:

- Produce relevant, less repetitive, and more human-friendly summaries.
- Introduced a novel time decay attention mechanism to minimize the issue of the repeating phrase in summaries.

Limitations:

- N/A

5) Citation: (Xu and Zhang, 2021)

Technology: • GANs architecture with reinforcement learning, - **Generator** → Seq2Seq model with an encoder that consists of single layer bi-directional LSTM and decoder consist of single layer LSTM - **Discriminator** → TextCNN for text classification. • Hybrid Pointer-generator network
Improvement: • Improved the coherence and readability of the end summary. • Reduction of duplication phrases in the final summary. • Proper handling of out-of-vocabulary (OOV) words in the end summary.
Limitations: • Dues to the natural limitation in the word vector that is used, the generated summary is more likely extractive as it contains more phrases that are included in the original document.

Table 2 : Existing works on Abstractive Text Summarization domain

1.6 Research Gap

- It was observed that existing automatic text summarization systems in the education domain of lecture video content are mostly done using extractive summarization process (Miller, 2019; Urala Kota et al., 2018; Vali et al., 2021), while abstractive text summarization is rarely explored and even the founded only abstractive lecture summarization system utilize traditional deep natural language technique of seq2seq based RNN in generation of end summaries (Andra and Usagawa, 2019). However, RNN based text summarization process mostly suffers from generation of duplicate, factually inaccurate summaries (Deaton et al., 2019).

- Even though the research related to automatic text summarization initiate around 1950's, still no system has been able to generate summaries that are near to human written summaries, which make more room for researchers to introduce and experiment new techniques in creation of quality summaries (Widyassari et al., 2019).

- The mostly used seq2seq models for abstractive text summarization suffer from an issue called exposure bias which causes a discrepancy between the summaries generated in the

training and testing period making the generated summaries less coherent and less readable. So, to address this issue researchers have introduced an architecture of reinforcement learning strategy based on Generative Adversarial Networks. It was observed that all the existing workaround related to GANs are done by introducing novel concepts for generator or discriminator component of the GAN architecture with the intention of improving the quality of the final summary (Liu et al., 2017; Rekabdar et al., 2019; Xu et al., 2018; Xu and Zhang, 2021; Zhuang and Zhang, 2019). However, with the recent introduction of transformers in NLP, it has provided an opportunity to experiment this technique with the generator component of the GAN architecture with the intention to improve the coherence and readability of the summaries.

By considering the project research gaps the contribution to the body of knowledge is derived and mentioned below.

1.7 Contribution to Body of Knowledge

1.7.1 Technological Contribution

Contribution of a novel enhancement to the existing GANs based abstractive text summarization model, by employing the transformer-based approach for the generator component of the GANs framework with the intention to produce high coherent, readable abstract summaries.

1.7.2 Domain Contribution

Creation of a lecture video content summarization system to produce quality summaries that can be utilized by the end-user(s) to get an essence of the lecture video content in a shorter period and to make beneficial for enhancement of the learning process.

1.8 Research Challenges

- The abstractive text summarization approach needs a massive volume of datasets to work with deep NLP algorithms for creation of perfect abstract summaries (El-Kassas et al., 2021).
- Up to this day, there is a lack of direct evaluation matrices to evaluate the factuality and coherence of the generated abstract summaries in the field of automatic text summarization rather than the use of manual evaluation (El-Kassas et al., 2021; Gambhir and Gupta, 2017).
- Abstractive Text Summarization needs extensive natural language processing to understand the context of document for creation of summaries that contains novel words,

phrase, so it considers to be complex (El-Kassas et al., 2021; Gambhir and Gupta, 2017), and also required high computation powers to work around for generation of abstract summary (Miller, 2019).

According to the research challenges pointed above it is obvious that the field of abstractive text summarization has its own complexity because of the process of producing high-quality summaries that are closer to human-generated summaries. So, this project research can be extended to the higher level of future research that can contribute to the creation of new domain-specific datasets and evaluation matrices that are lacking in the automatic text summarization field and also can contribute in provide new novel architectural solutions to generate high-quality summaries.

1.9 Research Questions

RQ1: What are the ways to summarize a lecture video content other than using computer visions and audio processing techniques?

RQ2: What is the best automatic text summarization approach in recent times?

RQ3: What are the emerging state-of-art deep natural language processing techniques that can be used for abstractive text summarization?

RQ4: What are the ways to evaluate and benchmark an abstractive text summarization model?

1.10 Research Aim

This research project aim is to investigate, design, develop and evaluate an end-to-end working lecture summarizer that will generate a considerable level of semantically and syntactically meaningful concise summary that contains the salient information related to lecture video in a concise manner that would benefit the end-user as well as, contribute a novel improved GAN based architecture for abstractive text summarization domain.

To further elaborate, the project aims to create an automated text summarizer for transcripts of lecture videos that will generate semantically meaningful text summaries for the end-user which address the identified research gap in the exiting lecture summarizers with the intention to improve the quality of the end summary. The initial prototype model will be thoroughly evaluated with the existing evaluation methods to validate if the identified novelty of using an improved GAN-based architecture for abstractive text summarization process generates quality

abstractive summaries that are factually consistent, less redundant, and are syntactically and semantically accurate.

1.11 Research Objectives

The following table contains necessary research objectives which describe the process that is needed to conduct the research project successfully.

Research Objectives	Explanation	Learning Outcome
Problem Identification	• Find a broader problem domain. • Collect all the valid and useful research papers related to the defined problem domain to conduct a literature review.	LO1
Literature Review	• To analyze and comprehend the broader context of the automatic text summarization process and its related approaches. • To analyze and comprehend the ATS process of abstractive text summarization, as well as the research significance of it in comparison to other ATS approaches, and the pros and cons related to this process. • To identify the domains that apply automatic abstractive text summarization and the available datasets used in those specific domains. • To identify research gaps by analyzing the limitations and future works in the existing system of abstractive text summarization. • To identify what are the deep NLP algorithms, methodologies used around abstractive text summarization, and what are the shortages of those applied methods.	LO1 LO4 LO5 LO6 LO8
Data Gathering and Requirement Analysis	• To gather both data and system requirements for designing an abstractive text summarization system.	LO2 LO3 LO4 LO5

	• To identify end-user requirements in order to define the system's intended behavior to improve end-user satisfaction. • To get institutions from the domain, technical experts, and supervisors regarding the purposed system and related requirements, to fine-tune the system by mitigating the risk.	LO6
Prototype Design	• Design a process to extract the audio content from the lecture video using existing APIs. • Design a process to create a text transcript from the extracted audio content. • Design a novel improved GANs based architecture for abstractive text summarization process. • Design an end-to-end GUI application that utilizes the proposed system.	LO5
Development	• Develop the audio extraction process. • Develop the text transcript creation process. • Develop the automatic lecture summarizer model. • Develop the end GUI application integrated with the summarizer model.	LO5 LO7
Testing and Evaluation	• To evaluate the performance of the automated abstractive lecture summarizer model by utilizing the identified evaluation matrices. • To create a test plan to perform functional and non-function tests on the end system to identify the system defects. • To perform functional and non-functional requirement validations for the end application.	LO5 LO8
Documentation	• Creation of comprehensive document related to the research project.	LO8

Table 3 : Research Objectives

1.12 Research Scope

The scope of a project defines the core functionalities that will be taken into consideration based on factors such as milestone, goal, feasibility, data limitations, computational cost, etc.

1.12.1 In Scope

The below functionalities will be considered in the research project,

- **Single Document Input** – Based on input size a summarization system can be either single document or multiple document (Modi and Oza, 2018). According to the project requirement, it will be only focus on single-document summarization.
- **Monolingual Text Summarization** – By considering available datasets for abstractive summarization and feasibility constraints, it is decided to create a summarization system that uses mono-lingual format where the input text document and the generated summary will be in the same language (which in this research project it will be focus only on the English language).
- **Generic Summary Output** – The generated summary in automatic text summarization can be of two types: query-based or generic-based (El-Kassas et al., 2021; Gambhir and Gupta, 2017). The system will design to produce a generic summary since the final expectation of this project is to create a summary that contains the general idea of the given content.
- **Use of Abstractive Text Summarization approach** – As the research project aims to generate abstract summaries which closely resemble human-generated summaries (El-Kassas et al., 2021; Gambhir and Gupta, 2017) it was chosen the abstractive text summarization approach.
- **The final system use only YouTube video links for summarization** – for the video summarization section of the project as depicted in the above prototype diagram only YouTube video links will be considered to make the project more feasible.

1.12.2 Out Scope

The out-scopes are derived by considering the in-scopes of the project. The below functionalities will not be considered in the research project,

- **Not considering multiple document Input** – As the input text document of the project is a single text transcript document, the multiple document summarization process will not be taken into count within this project scope.

- **Not considering multi-lingual, cross-lingual summarizations** – Because of the dataset availability and feasibility considerations, multi-lingual and cross-lingual text summarization will be not considered within this project scope.
- **Not considering query-based summary** – As the research project's final expectation is to creation of a generic summary, it will not be considered the generation of query-based summaries within this project scope.
- This research project does not consider visually presented lecture video contents (such as whiteboard, slide explanations, etc.) as the project's focus is to generate summaries by utilizing transcripts that will be created from the audio content of the lecture video.

1.12.3 Prototype Feature Diagram

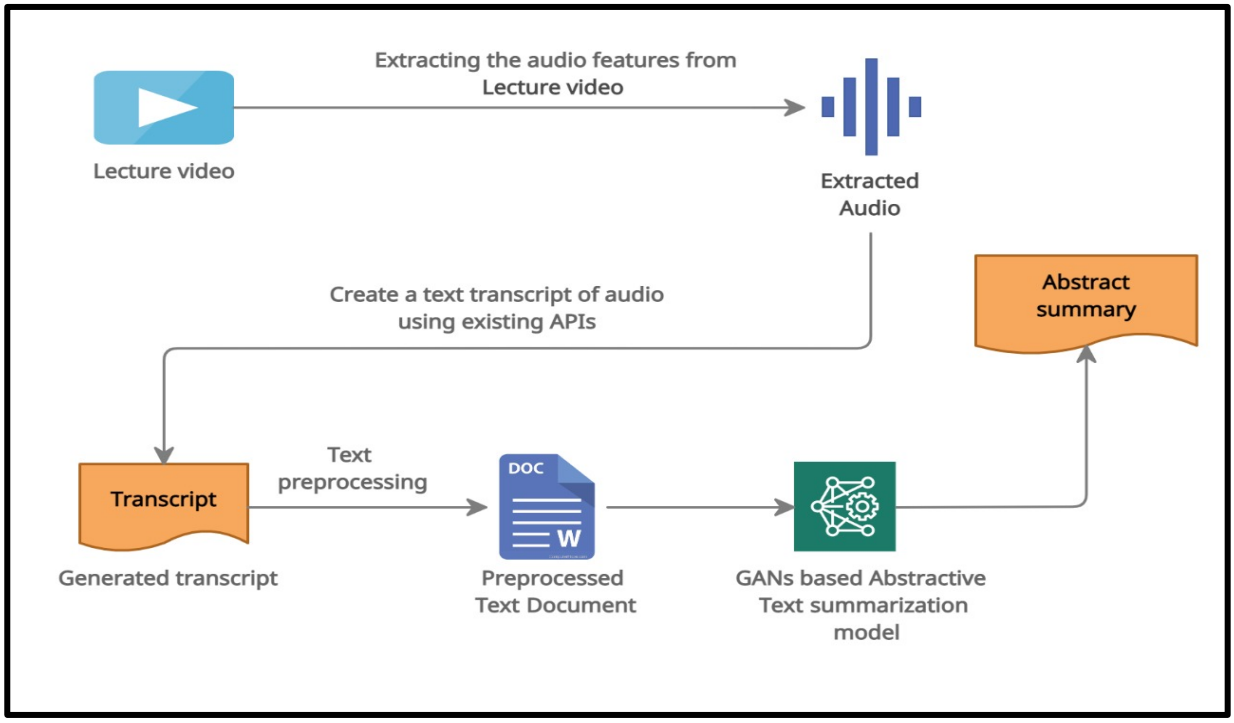


Figure 1 : Prototype Feature Diagram (self-composed)

1.13 Chapter Summary

The chapter pointed out the main research problem that the author elaborates on by providing a detailed explanation related to the research problem domain along with problem definition. After providing an explanation related to the problem domain the chapter moved on to provide details for the aim and objectives related to the research project which would be beneficial in driving the research project and followed by it the novelty of the research project and the research challenge was presented with relevant to identified facts from existing works to elaborate the research value of the project engaging.

CHAPTER 2 : LITERATURE REVIEW

2.1 Chapter Overview

Automatic Text Summarization is a broader and rising research area in the context of natural language processing that focuses on automating the process of short text summary creation. It's a broader and ongoing research field that can be investigated and experimented with newer techniques in the creation of quality summaries. To identify newer ways and insights of any research field it's necessary to conduct an in-depth review of previous literature. This chapter discusses in-depth the domain, technologies, challenges, evaluation methods, benchmarks, and datasets in the field of **Abstractive text summarization (an approach of the automatic text summarization process)** by critically evaluating the previous works which will assist in finding new gaps which are not explored yet.

2.2 Concept Map

The identified research elements (which are Problem-solving techniques, Domains, Evaluation methods, Tools, etc.) by reviewing the existing works related to the field of research are illustrated in a concept map in **Appendix B.**

2.3 Problem Domain

2.3.1 Automatic Text Summarization aka ATS?

Natural Language Processing which is a subfield of Artificial Intelligence that empowers machines to understand human language in both written and spoken, is widely used across a diverse amount of applications such as Machine Translation, Speech Recognition, Chatbot, etc (Nadkarni et al., 2011). Automatic Text Summarization is one of many applications of natural language processing that has gained recent popularity and interest of researchers with the increment of the textual data in today's world.

Due to the development of technology the use of the internet has grown higher, which has caused the exponential growth in textual data. Because of this rapid increment of data, it has been laborious for users to comprehend and locates the salient information from a vast amount of textual documents within a short time period. Because of these issues, the need for an efficient solution for summarization of large volumes of textual data has emerged. Manual summarization is impractical and inefficient in these formats as it consumes a lot of time and is very tedious. So as a solution for this dilemma in the year of 1950's the automation of the text summarization

process was introduced which is commonly called Automatic Text summarization aka ATS (Bharti et al., 2017; El-Kassas et al., 2021; Gambhir and Gupta, 2017; Widyassari et al., 2019).

The main aim of the Automatic Text Summarization process is to generate a short concise summary that contains the main idea of the source document which will help the user(s) to save time and effort in comprehending the idea of the specific textual document (El-Kassas et al., 2021; Gambhir and Gupta, 2017). With the mass volume of available textual information in e-resource format having robust automatic text summarization systems can ease the process of identifying the relevant information of a source document without going through the whole document which will be prone to issue like redundancy and relevant information loss. When building up an automatic text summarization system it need to consider four main elements which are **coverage of information, information significance, redundancy in information and cohesion in text** in order to produce quality end summary (Gambhir and Gupta, 2017). Automatic Text Summarization contains two approaches which are Extractive Text Summarization and Abstractive Text Summarization where the former is researched extensively, and latter is not well researched due to its complexity. Also, according to the number of input source documents used for end summary creation, the Automatic Text Summarization can be classified as Single or Multiple Document Summarization (El-Kassas et al., 2021; Widyassari et al., 2020). More about these ATS classifiers will be discussed in the latter sections.

2.3.1.1 General Architecture of an Automatic Text Summarization System

The architecture of the Automatic Text Summarization System in general consists of following elements as depicted in the **Figure 2** which are,

1. **Pre-Processing:** The initial steps to prepare the textual data of a source document in a structured representation using NLP preprocessing techniques used in the field of automatic text summarization, such as stop word removal, lemmatization, stemming, and so on, which are depicted in **Figure 2**. (El-Kassas et al., 2021; Gambhir and Gupta, 2017; Widyassari et al., 2020).

2. **Processing:** Creation of the final concise summary by utilizing one of the automatic text summarization approaches either extractive or abstractive (will be discussed more on the latter section) (El-Kassas et al., 2021).

3. **Post-Processing:** The process of refining and enhancing the quality of the generated end summary by re-solving some post issues such as anaphora resolution and reordering the sentences selected for generation of the end summary (El-Kassas et al., 2021) .

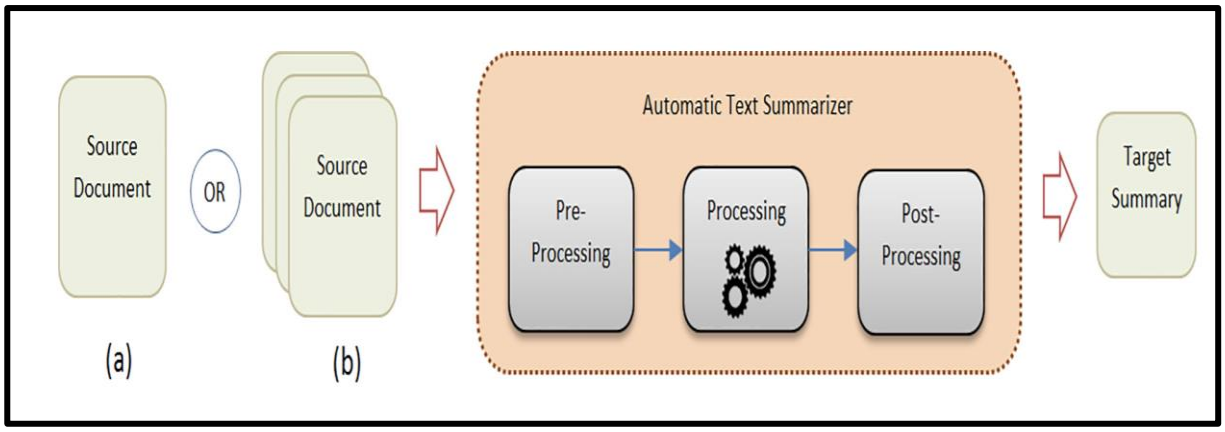


Figure 2 : Architecture of Automatic Text Summarization Process (El-Kassas et al., 2021)

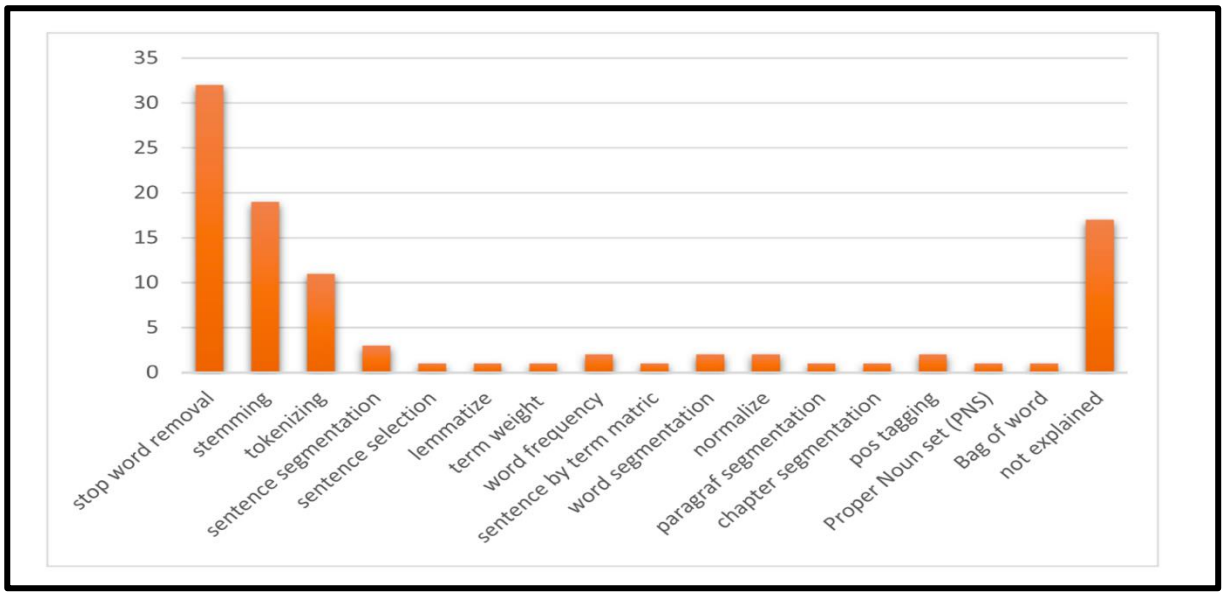


Figure 3 : Distribution of Preprocessing Techniques (Widyassari et al., 2019)

All the automatic text summarization systems up to now are built on top of this main general architecture for creation for robust summarizer for produce coherent summaries.

2.3.1.2 Various Classifications of Automatic Text Summarization system

Automatic Text Summarization systems can be classified into various parts based on several factors that employed when creating of a summarization system as illustrated in the **Figure 4.**

Based on Input Size:

Automatic Text Summarization systems can be divided into two types based on the input size of the source documents used in creation of summaries, as either **Single Document Text**

Summarization (SDS) or **Multiple Document Text Summarization (MDS**). Single Document Text Summarization utilize only one source document to extract the salient information in the creation of the end summary, whereas Multiple Document Text Summarization employs two or more documents to produce the end summary by extracting the relevant information from all the inputted source documents with the aim of reducing the repetition content in the end summary (Bharti et al., 2017; El-Kassas et al., 2021; Gambhir and Gupta, 2017; Modi and Oza, 2018; Widyassari et al., 2019, 2019). The process of producing summaries using MSD is much more complex than SDS because it involves mining multiple documents to find the important sentences that convey the essence of all the documents in the end summary and also prone to issues such as redundancy, coverage, temporal relatedness, compression ratio, and so on.(Dernoncourt et al., n.d.; El-Kassas et al., 2021; Gambhir and Gupta, 2017; Widyassari et al., 2019)

Based on Nature of the Output Summary:

Automatic Text Summarization can be further classified based on the type of Output Summary expected, which can be either **Query-based** (also known as query-focused, topic-focused, or user-focused) and **Generic-based** (Bharti et al., 2017; El-Kassas et al., 2021; Gambhir and Gupta, 2017) . The goal of Query-based text summarization is to extract the most relevant sentences that contains the salient information of the original text document that best answers the given search query to generate the final summary (example google search), whereas Generic summarizers generates the end summary that contains the overall general idea of the entire document in a concise manner.(Dernoncourt et al., n.d.; El-Kassas et al., 2021; Gambhir and Gupta, 2017)

Based on Summarization Algorithm:

ATS can be again classified based on the learning strategies used such as **supervised**, **unsupervised** or **semi-supervised** (El-Kassas et al., 2021).

Based on Summary Content:

The expected final summary content falls into two categories: **Indicative** or **Informative**. **Indicative summary** indicates the general idea of the source document in a concise manner, allowing the user(s) to decide whether to read the document. In contrast, an **Informative summary** cover the main contents/ideas of the source document in concise manner by excluding the irrelevant details to get the gist of the whole document in short time period (El-Kassas et al., 2021; Gambhir and Gupta, 2017; Widyassari et al., 2019).

Based on Summarization Domain:

The Automatic Text Summarizer can be **General** or **Domain Specifi**c. General summarizers, also known as domain-independent summarizers, used source documents from multiple domains to create the summarizer, whereas domain-specific summarizers utilize document specific to certain domain in creation of domain specific summaries (ex-New, Education, Medical Document, etc.) (El-Kassas et al., 2021).

Based on Summarization Approach:

Summarization approaches are mainly in two types which are **extractive text summarization** and **abstractive text summarization**. The extractive text summarization process extracts the significant sentences that contain salient information of the original document and concatenate the selected sentences in order to create short concise summaries, whereas the abstractive text summarization process representation the original source document in an intermediate representation to comprehend the context of the source document in order to create summaries which contain phrase and words that are out of the original document (Bharti et al., 2017; Dernoncourt et al., n.d.; El-Kassas et al., 2021; Gambhir and Gupta, 2017; Widyassari et al., 2020, 2019).

Based on Summary language:

According to the language of the source document and the language contains in the generated end summary, it can be classified automatic text summarization systems as mono-lingual, multi-lingual, and cross-lingual summarization systems. In a Mono-lingual summarization system, the produced summaries will be of the same language as the given source document's language (Ex- The Source document is in English language and the generate the end summary will be the same English language). In a Multi-lingual summarization system, the original text document can be in different languages and the generated summaries will also contain the languages used in the original documents. In a Cross-lingual summarization system, the source document will be in one language and the generated summary will be in another language (Ex- The source document in English language and the generated summary is in the French language) (El-Kassas et al., 2021; Gambhir and Gupta, 2017; Modi and Oza, 2018).

Based on Summary Type:

Based on purpose for which the ATS system used, the length of the end summaries can be differ. So according to that summary can be in different types which are Headline, **Sentence-Level**, **Highlights**, or **Full Summary** (Dernoncourt et al., n.d.; El-Kassas et al., 2021). Headline

Summarization produce a headline summary of the original document which is shorter than a sentence. Sentence-Level summarization produce a single sentence abstractive summary given the source document (Dernoncourt et al., n.d.; El-Kassas et al., 2021). Highlighted summarizations produce concise summaries that includes the salient information of an original document in the telegraphic form which is normally in the form of bullet points (El-Kassas et al., 2021). Full summarization typically generated summaries based on the required summary length or by using compression ratio technique (El-Kassas et al., 2021).

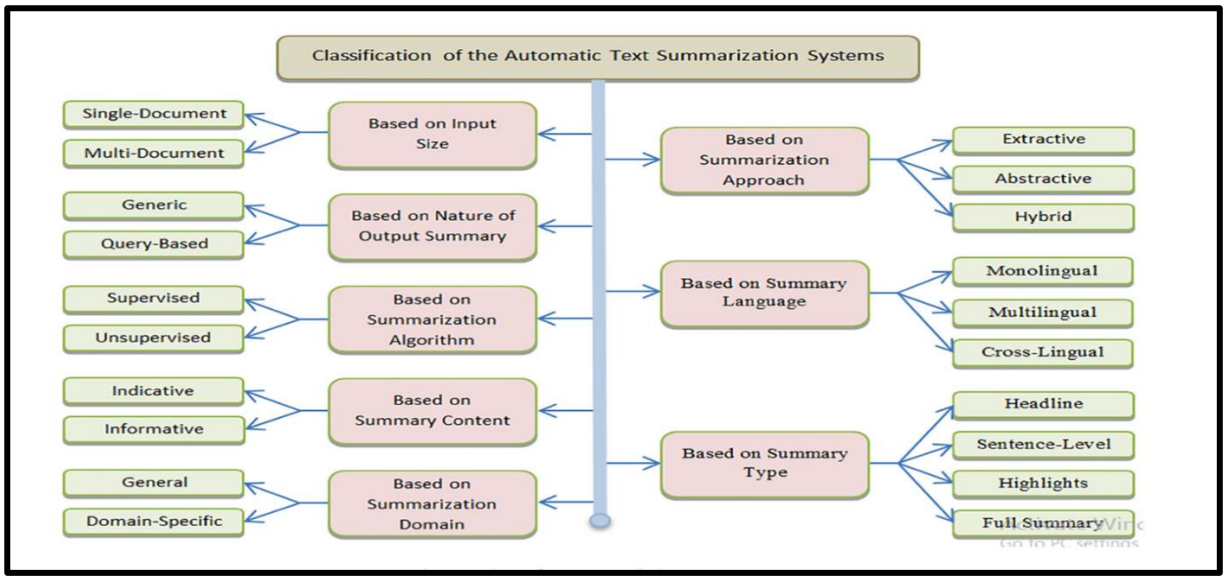


Figure 4 : Classification of Automatic Text Summarization System (El-Kassas et al., 2021)

2.3.1.3 Approaches of Automatic Text Summarization

In the above **Section 2.2.1.2** under the classification category of Automatic text summarization approaches, it was briefly explained about the two approaches of ATS which are extractive and abstractive. In this section will focus on more on it. There are two main research fields of automatic text summarization which are extractive text summarization and abstractive text summarization. Any of the Automatic Text Summarization systems that are developed or that will be developed, will employ either one of the two approaches in creation of concise coherent summaries.

2.3.1.3.1 Extractive Text Summarization

Extractive Text Summarization process use an extractive mechanism to grab the important sentences and phrases that contain the salient information of the sources document, where these extracted significant sentences will be joined together in creation of the **extractive summaries** which contains the same terminologies (Sentence or Phrase) in the original document (Bharti et

al., 2017; El-Kassas et al., 2021; Gambhir and Gupta, 2017; Joshi et al., 2017; Widyassari et al., 2020; Zhang et al., 2020). If explaining this process with an analogy it is similar to how we write summaries by only highlighting the main point on the original document (source document) and then concatenate those highlighted sentence in a proper order without paraphrasing the selected sentence. Extensive Research are already done in this area because of its simplicity and robustness of the generated summaries (El-Kassas et al., 2021; Gambhir and Gupta, 2017). Below diagram shows the general architecture of an extractive text summarization system.

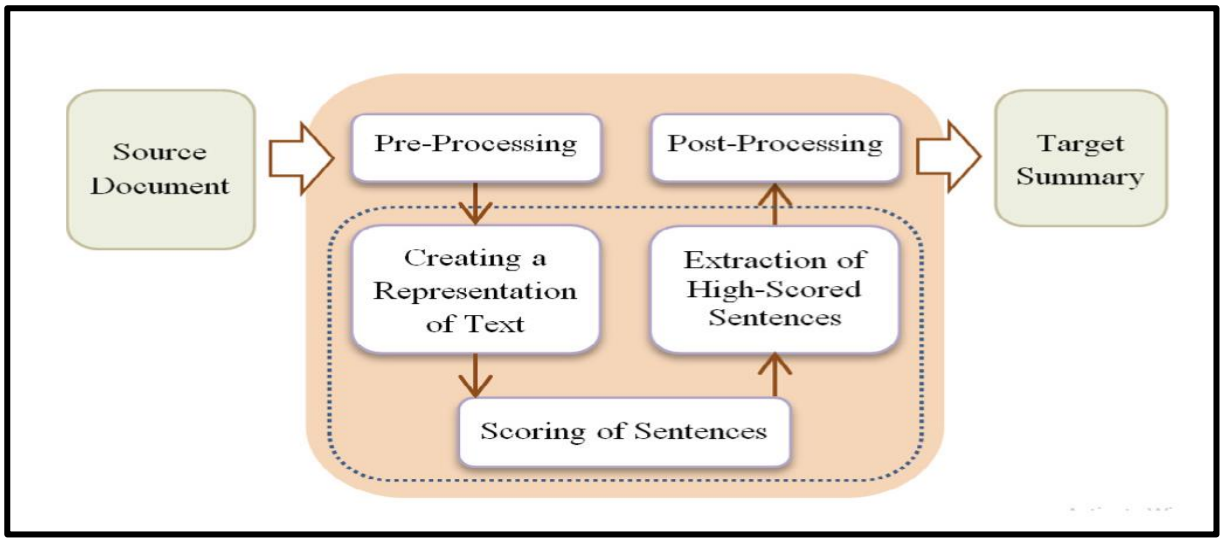


Figure 5 : Extractive Text Summarization Architecture (El-Kassas et al., 2021)

2.3.1.3.2 Abstractive Text Summarization

In contrast to extractive text summarization, Abstractive Text Summarization process is more closely resemble Human summarization process because of its ability to generate summaries which contain phrase and words that are out of the original text document. In this process the source document will be represent in an intermediate form to comprehend the context of the original document to generate **abstract summaries** which contain sentences, phrases, words that are not included in the original document. Abstractive Text Summarization process is much more complex as it need extensive NLP to understand the context of the text document to produce quality abstractive summaries. Because of this complexity most of the Research have been focused on using extractive text summarization technique in automation of summaries creation process (Bharti et al., 2017; El-Kassas et al., 2021; Gambhir and Gupta, 2017; Joshi et al., 2017; Widyassari et al., 2020; Zhang et al., 2020).

Figure 6 illustrate the general architecture of an abstractive text summarization system that intake an input source document that can be either single or multiple document(s) that will be

pre-process using text pre-processing technique such as stop word removal, lemmatization, stemming, tokenization (More of the text pre-processing techniques are depicted in **Figure 3**). Then with the use NLP technique it will create an intermediate representation of the pre-processed document that will be in the form of a word graph, rich semantic graph, word embedding etc, that will be processed using NLP or Deep NLP techniques to create the final abstractive summary. After the summary creation, the final summary will be further processed to refine and enhance the quality of the end summary (El-Kassas et al., 2021; Gambhir and Gupta, 2017).

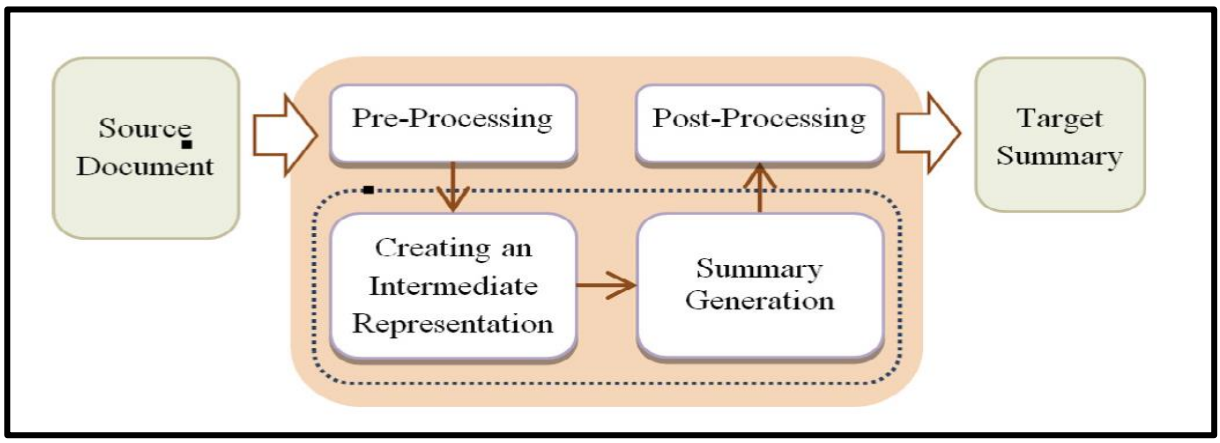


Figure 6 : Abstractive Text Summarization Architecture (El-Kassas et al., 2021)

2.3.1.4 Pros and Cons of Automatic Text Summarization Approaches

The section list down the disadvantages and advantages of mentioned approaches of automatic text summarization approaches.

2.3.1.4.1 Pros and Cons of Extractive Text Summarization

Advantages	Disadvantages
Simple in creation and does not need Extensive Natural Language Processing techniques for creation of the summaries (El-Kassas et al., 2021; Gambhir and Gupta, 2017).	The extractive approach is diametrically opposed and does not resemble human written summarization process (El-Kassas et al., 2021).
Generate robust summaries which are syntactically and semantically accurate as it utilizes the extracted sentences, phrases, words from the same original document in	Generated summaries may contain redundant sentences and might not include conflicting information correctly (El-Kassas et al., 2021).

creation of summaries (Xu and Zhang, 2021).	
The produce summary will be most of time contain the fracual information about the original document.	Because of the incorrect link between the extracted sentences the final summary can be less coherent which make it difficult to understand.

Table 4 : Pros and Cons of extractive text summarization

2.3.1.4.2 Pros and Cons of Abstractive Text Summarization

Advantages	Disadvantages
Generated abstract summaries closely resemble human written summaries (Gambhir and Gupta, 2017; Modi and Oza, 2018)	Abstractive process is much more challenging and complex as it required extensive natural language to represent the source document in an intermediate representation (semantic representation of the context) to understand the context (semantic meaning) of the document in order to paraphrase the information to a short summary (El-Kassas et al., 2021; Kieuvongngam et al., 2020, p. 2)
Capability to use words, phrases that does not consist in the original document, for creation of the abstract summary (Kieuvongngam et al., 2020, p. 2).	Required large records of dataset in order to generate quality abstractive summaries.
Generated abstract summaries are syntactically and semantically accurate which make the summaries more understandable (Modi and Oza, 2018).	Need to have a considerable amount of computation power to perform intensive NLP processing tasks.
Use of complex techniques such as paraphrasing, Generalization and real world knowledge in creation of high-quality summaries (Xu and Zhang, 2021).	

Table 5 : Pros and Cons of abstractive text summarization

2.3.1.5 Challenges in Automatic Text Summarization

Every field has its own challenges due to various reasons, so as for the automatic text summarization systems. Below are some challenges poses in automatic text summarization systems,

- Locating the most salient information of a source document that need to be include in the end summary.
- Difficult in Long Text Summarization.
- Difficult in Multiple Document Summarization.
- Generation of quality abstractive summaries.
- Need of evaluation metrics to evaluate machine generated summaries without the need to comparison with the human generated summary.
- Generated summaries may contain redundant, repetition phrases (Gambhir and Gupta, 2017).

Even though it has been introduced several newer techniques in generation of reasonable quality summaries in automated text summarization field, it is still not able to achieve to generate summaries that has the quality of the human written summaries (El-Kassas et al., 2021) which make opportunities for researchers to find new insight in improve the quality of the summaries.

2.3.2 Abstractive Text Summarization for Education domain

As stated in the preceding sections, due to the ease of creation of text summaries using the extractive approach, majority of text summarization research has been focused on producing extract summaries that contain the same contents in the original document, that do not resemble human-written summaries (El-Kassas et al., 2021). Abstractive summarization, on the other hand has received less attention due to its complexity but with new developments of state-of-the-art techniques in deep NLP in current timeline it has open more opportunities to researchers to do more experiment and provide new techniques in the creation of quality abstractive summaries that can deliver the main information of the original document in a concise manner making it easier by user(s) to comprehend by maintaining the semantical and syntactical correctness. Keeping this in mind, the author has chosen the abstractive text summarization process in finding a novel approach to generate coherent summaries with the goal of reducing repetition and redundancy phrases in the end summary.

Lecture videos in educational domain provides all the necessary learning context for the user(s) to understand a particular subject. In a situation where the user(s) want to the get the basic

essence of lecture videos in short time, they must go through the whole video again to understands a specific concept which consume much time. So having an automatic summarizer which can deliver the gist of the lecture video in a concise manner would benefits the user(s) in saving time as well as to be effective in learning process. By considering this author has chosen to apply the selected abstractive text summarization process to the lecture domain with the aim to generate coherent summaries that gives the salient information of the lecture video in a shorter manner.

2.4 Existing systems

2.4.1 Automatic Summarization in Lecture

Lecture is one of the many domains that automatic summarization is applied and can be applied to get benefit for enhance the learning process. Past works related to automatic summarization in lecture domain utilize techniques such as text processing, video processing, image processing or combination of these techniques to create summaries. Below contains several previous works that are done related to automatic summarization in lecture domain.

(Urala Kota et al., 2018) proposed an automatic summarization framework for lecture videos by creating a summary of lecture content that contains keyframes of handwritten text on the whiteboard. In this approach as the 1st step, all the handwritten text is detected and extracted using TextBoxes. TextBoxes is a neural network composed of 28 CNN and SSD architecture. After extracting the keyframes Temporal Analysis Algorithm is adopted to deal with the noise of the frames that occurs due to the motion of the lecturer. Then conflict minimization approach is used to deal with overlapping frames. This framework was evaluated by using the number of keyframes extracted as the summary and recall and precision of all whiteboard content in the set of test lecture videos. This framework only can use in a lecture video with a whiteboard and most vital information is missing in summarization as researchers don't consider about audio content of the lecture in the summarization process.

(Miller, 2019) proposed a lecture summarization service that was designed as a python RESTful service. The proposed lecture summarization service has adopted a BERT pre-trained model for textual embedding for lecture video transcript and the K-Means clustering approach has been adopted to find the most suitable sentences that are nearest to the centroid for extract and to create the best summary. This approach shows low performance in large lectures with long transcripts and is also very challenging when it comes to representing a whole lecture with a small ratio of sentences.

(Xu et al., 2019) proposed a system called LECTURE2NOTE that can create a lecture note by using slide-based lecture videos. In this approach as the 1st step, the visual entities are detected and extracted from the presentation slides. When extracting visual entities in the beginning system binarize the RGB slide images to clearly distinguish the contour of each region and perform morphological operations on each component. Then, in the horizontal direction, the run-length smoothing algorithm (RLSA) is used to combine the split components from the previous step and construct complete bounding boxes of connected regions. Third, overlapped and incorrectly combined regions are deleted to achieve region mergence. Then the visual entity recognition is performed by using a classifier to classify each visual entity by using English letter ratio, Aspect ratio, Special symbol count, character baseline fluctuation as the visual features. After extracting and detecting visual entities system is combine those visual entities with corresponding speech to text as a descriptive note. To do this task researchers proposed an algorithm for constructing the semantic relationship between speech texts and visual entities, which associates the visual entity with its corresponding descriptive speech texts and addresses situations that time-align technology cannot handle. This is approach gives a good system to create a lecture note but it does not summarize the descriptive note in the lecture so most of the time more unnecessary and redundant information can be present in the note.

(Andra and Usagawa, 2019) proposed an automatic summarization framework for lecture videos by summarizing the lecture transcript to capture the most significant information in the lecture. In this approach, the lecture transcripts are segmented and summarized using the attention-based RNN method. In the pre-processing stage Noise removal, Stopword removal, Lemmatization, and Stemming are incorporated to remove and clean unnecessary symbols and dialogs in the transcripts video. After pre-processing stage, segmentation is adopted to improve the coherency of topics in each segment. In the segmentation process, the segments are grouped based on similarity and coherence of spoken topic, this process is designed using the PowerSeg method. RNN model is based on the Seq2Seq approach with an attention mechanism to create more coherent and good summaries. Although this method shows some promising results this method suffers from the certain limitation that are specifically related to RNN methods.

2.4.2 GAN based Abstractive Text Summarization

(Xu and Zhang, 2021) proposed an adversarial framework for abstractive text summarizing process that composed with generative and discriminative components that jointly trained. The generative component is composed of a Seq2Seq model with an encoder that consists of bi-directional LSTM and decoder that consists of attention-based LSTM that incorporates a pointer

generator network to reduce redundant words in generated summaries and to handle out of vocabulary words in the summarization process. Generative components take original text as the input and generate a summary. The discriminative component is composed of a CNN model that works as a binary text classifier that aims to identify whether generated text summary is more abstract or not. The proposed framework is adopted the reinforcement learning approach to optimize the generation of summaries in the generative component to produce a highly rewarded summary. Both the generative and discriminative component trained to optimize as a min-max two-player game. Although this method shows some promising results this method suffers from certain limitation that are specifically related to LSTM methods.

(Xu et al., 2018) proposed an adversarial framework for abstractive text summarizing process that composed with generative component and discriminative component that jointly trained. The generative component is composed of double attention-based Seq2Seq model consist of an encoder and decoder that contains GRU layers. To find the most relevant parts of the original text, the researchers have used a novel attention mechanism called IARNN-WORD this mechanism helps to solve the attention offset issue in the traditional attention mechanism. The researchers have considered the text summarization process as a text compression problem rather than text alignment and they also consider that rich feature of the original texts is beneficial to discriminator, therefore they have adopted Triple RNN architecture to discriminative model. In triple RNN architecture two RNN models are separately used to capture the contextual information of source (original text) and target (summary), while the other RNN model is utilized as a binary text classifier that aims to identify whether generated text summary is machine generated or real. The proposed framework is adopted an RL strategy called policy gradient to guarantee the co-training of generator and discriminator and provide a highly rewarded summary. Although this method shows some promising results the generated summaries contain duplicate phrases and does not able to handle out of vocabulary words in the summarization process.

(Zhuang and Zhang, 2019) proposed an adversarial framework for the abstractive text summarizing process that is composed of a generative component and two discriminative component. The generative model is composed of a hybrid pointer-generator network that consists of an attention-based Seq2Seq model consisting of an encoder and decoder that contains LSTM layers. The authors have performed pre-training for the generator to get it to recognize the input texts and generate something meaningful. They have utilized the related human-written summary as the ground truth to train the generator by minimizing the cross-entropy loss given

the input text content. In this adversarial framework, the authors have proposed two types of discriminative models as similarity discriminator and readability discriminator. In similarity discriminator the framework tries to classify the level of similarity between the original reference summary and the generated summary, authors have proposed four classes of similarities as a similar class, incomplete class, redundant class, and irrelevant class. Similar class means the generated summary can concisely express the main concept of the source (original text). Incomplete class means the generated summary omits part of the source (original text) crucial material. Redundant class means the generated summary is redundant since the brief text provides too much inessential information. Irrelevant class means the is generated summary is not representing the meaning of the source (original text) at all. The similarity discriminator is composed of CNN based classifier. In readability discriminator composed of CNN based binary text classifier that trained to detect whether the generated summary is human-readable or not for that author has trained that model with positive labelled human summaries and negative labelled generated summaries. The proposed framework is adopted an RL strategy called policy gradient to guarantee the co-training of generator and discriminator and provide a highly rewarded summary. Although this method shows promising results this method is computationally expensive.

2.4.3 Benchmarking of Abstractive Text Summarization Systems

(Xu and Zhang, 2021) in their work, they have compared their strategy to three others: the abstractive model (ABS), pointer-generator coverage networks (PGC), and the abstractive deep reinforced model (DeepRL) (ML+RL version) by using ROUGE-1, ROUGE-2, ROUGE-L, and Human scores, they have used CNN/Daily Mail dataset as their benchmark dataset. Similarly **(Zhuang and Zhang, 2019)** have used CNN/Daily Mail dataset as their benchmark dataset and they have compared their framework performance against others models such as Pointer-Generator , Deep RL, GAN, and WGAN by using ROUGE-1, ROUGE-2 and ROUGE-L scores. CNN/Daily Mail is a text summarization dataset. According to their scripts, the corpus has 286,817 training pairings, 13,368 validation pairs, and 11,487 test pairs. On average, the source texts in the training set comprise 766 words and 29.74 sentences, whereas the summaries have 53 words and 3.72 sentences.

2.5 Technologies

In the domain of abstractive text summarization, there were several technological approaches used from the early stage up until the modern period. All those technologies utilize in the abstractive text summarization domain main fall under 3 categories which are Structure-based,

Semantic-based, and Deep-learning based approaches. Out of these approaches with the recent development of neural architectures in the natural language processing domain, all recent research related to abstractive text summarization model creation moved from traditional structured and semantic based approaches to deep learning-based approaches as it was found that deep learning-based models perform far better in creation of abstract summaries because of it ability performance well with large corpora.

2.5.1 Abstractive Text Summarization Technologies

Below the author has pointed out almost all the identified deep learning-based NLP technologies utilized in the field of automatic abstractive text summarization.

2.5.1.1 Deep Learning Based

In almost all of the literature related to deep learning based abstractive text summarization models are developed with respect to below mentioned one or more combinations of deep NLP techniques.

2.5.1.1.1 Bidirectional RNNs

Bidirectional RNNs are the earliest deep NLP technology utilized in the creation of abstract summaries. As the vanilla RNNs with LSTM cell was able only able to capture the context relation of a given textual corpus with previous words it was identified the context of text properly not captured so it was introduced Bidirectional RNNs with LSTM which was performed far most than vanilla RNNs as it was able to capture the textual context of giving text phrase in both directions in order to get make most intelligence text understanding. But with the extreme complexity of the creation of abstract summaries in the ATS domain this deep NLP technology alone did not produce any good results dues to reasons such as long sentence summarization issues, repetitions in the end summary, and large training time etc (Deaton et al., 2019).

2.5.1.1.2 Sequence to Sequence Model (Seq2Seq)

To address the issue in RNNs and to introduce a novel deep NLP model to work better with sequence-to-sequence problems it was introduced a new neural architecture named Seq2Seq which consists of an encoder and a decoder component that contain Bidirectional RNNs cells (Sutskever et al., n.d.). This deep NLP technology is the most used and state-of-the-art traditional approach in most of the sequence generation related models. This technique intakes a sequence data such as a text phrase and creates a context vector that captures the context of the input

sequence that will be passed to the decoder layer in order to produce a sequence that correspondent with the trained decoder context.

But even though the Seq2Seq model performed fairly well in the fields of generation of abstractive summary it still suffered from several issues such as the context vector that is used for the decoder part does not attend to specific relative words that could contribute in the generation of the decoder results as the context vector only contain capture whole input sequence context in forward and backward direction, high training time as Seq2Seq model utilized mostly RNNs in its encoder and decoder components, long sentence summarization issue, unable to handle out of vocabulary issues, etc. As this model alone had several downfalls with the time researchers utilized new techniques on top of Seq2Seq model such as applying of the attention mechanism (Deaton et al., 2019; Mohammad Masum et al., 2019, 2019; Nallapati et al., 2016; Syed et al., 2021).

2.5.1.1.3 GANs

It was observed in several literatures researchers have adapted the GANs-based framework in the creation of abstract summaries which have performed state-of-the-art results. Most of the research that utilized GANs approach pointed out the issues common issues in Seq2Seq models as an address factor for using GANs in abstractive text summarization. It performed fairly well in sequence generation because of its adversarial manner of training included in the architecture. As the GANs framework consists of two main components which are the generator and discriminator part the research studies have experimented on the generator part with different configurations of technologies such as introducing vanilla Seq2Seq model, Seq2Seq with attention, Seq2Seq with 2D attention, etc which directed to the creation of fairly good abstract summaries (Scialom et al., 2020; Xu et al., 2018; Xu and Zhang, 2021; Zhuang and Zhang, 2019). Also, the important factor which was noticed was that none of the research still did not experiment with this framework with the latest state-of-the-art neural architecture in NLP which is the transformer model which provided a significant gap in the research area.

2.5.1.1.4 Transformers

Transformer which is the recent new rise in deep NLP models also have been to some extent utilized in the generation of abstractive summaries but not most because of its recent identification in NLP domain. This technology outperformed most of the existed deep NLP approaches in sequence generation tasks because of the powerful architectural design to capture the context of a sequence more precisely. But in the context of abstractive text summary

generation, it is still not mostly utilized much in model creations which could pave the path for future researchers to do more experiments with this technology.

2.6 Evaluation Methods used in Abstractive ATS systems

When observing the existing literatures in the automatic abstractive text summarization field it was observed that there are only a limited amount of evaluation processes that could be done for the analysis of the quality of the end summary. One such method ROUGE score which is used to get the n-gram overlapping between the machine-generated summaries and golden truth summaries. Other than that the only other way around to evaluate the abstract summaries is by using human evaluations (El-Kassas et al., 2021; Kryściński et al., 2019; See et al., 2017; Xu and Zhang, 2021).

2.7 Chapter Summary

Automatic Text Summarization is a fascinating research area that can be applied to a variety of scenarios. This chapter begins with an introduction to the main problem domain which is automatic text summarization and then proceeds on reviewing the reasons for the rise of automating the text summarization process, its architecture, various classifications, its main approaches, pros and cons of those approaches, challenges its faces and then move on providing a review of the significance of automatic summarization in both abstractive and lecture domain. Further it was properly pointed out the deep NLP technologies that have been used up until now in the research domain along with a review of the existing evaluation techniques.

CHAPTER 3 : METHODOLOGY

3.1 Chapter Overview

This chapter presents all the considered methodological aspects relevant to the conducting research project which are research, development, and project management methodologies. Along with the methodologies the chapter also presents the considered resource requirements and risk mitigations relevant to the project.

3.2 Research Methodology

Research Methodology	Description
Philosophy	The author has chosen **Pragmatism** as the research philosophy out of other existing philosophies, as the project employs a **qualitative approach** for obtaining primary data for designing the model and also will utilize both **quantitative** and **qualitative approaches** for evaluation of the designed model.
Approach	The author has chosen **the deductive** approach over the inductive approach because the research target on testing and proofing a hypothesis for an existing theory. Which is improving the quality of the abstract summaries by utilizing a novel enhanced GANs architecture in abstractive text summarization.
Strategy	Among different research strategies **experiments, surveys, and interviews** are chosen. **Experiments** will be used as a quantitative approach in identifying the best techniques for the system and **surveys, interviews** will be used as qualitative approaches for to gather feedbacks for system evaluation and refinement.
Choice	**Mixed method** is chosen as the research choice because the research strategies of the project will employ **both quantitative and qualitative** approaches.
Time horizon	The **cross-sectional** time horizon was chosen because the data for the study will be collected over a single time frame.

Procedures and Techniques	Procedures such as literature reviews, experiments, surveys, interviews, discussions, observations, reports will be employed for the collection of data, analysis, evaluations, and validations of different aspects of the system.

Table 6 : Research Methodology Process

3.3 Development Methodology

3.3.1 Software Development Lifecycle Methodology (SDLC)

Prototype methodology is chosen as the software development methodology among many other methodologies such as Agile, Scrum, DevOps, etc. The reason for the selection of prototype methodology is that it follows an iterative trial-and-error process to refine the system until a successful prototype is achieved.

3.3.2 Design Methodology

Structured Systems Analysis and Design Method (SSADM) was chosen as the design paradigm of the research project as the designing of the core model of the system follows a procedural approach that contains specific tasks for each defined stages in the process. So, because of this reason, SSADM methodology was selected as the most suitable design paradigm

3.3.3 Requirement Elicitation Methodology

Literature reviews, Observations, Experiments, and **Surveys** will be employed as requirement elicitation methodologies.

3.3.4 Evaluation Methodology

Evaluation is a necessary part of a research project to validate if a system performing as expected. In the project, it will be utilized both **qualitative** and **quantitative** ways to evaluate the purposed summarizer model,

- **ROUGE Score** (**Recall-Oriented Understudy for Gisting Evaluation) -** Mostly used evaluation matrix in the field of abstractive text summarization. It is used to evaluate the similarity of the machine-generated summary and the golden truth summary by counting the overlapping n-grams using ROUGE-1, ROGUE-2, ROUGE-L, etc (Xu and Zhang, 2021).
- **Semantic Text Similarity Score** –This scoring model maps batches of sentence pairs to real-valued scores in the range [0,5] and 0 means poor semantically similar sentence and

5 means a highly semantically similar sentence. This score also considered to utilize to compare the semantic similarity between original summary and predicted summary.

- **Human Evaluation –** As the ROUGE score matrix only considers the n-gram overlapping between the machine-generated and ground truth summaries, it does not measure the quality of the summary based on the factual consistency, grammatical and syntactical correctness. So, human evaluation will be considered to qualitatively evaluate the machine-generated summary.
- **Benchmark** – Benchmark is a necessary process to evaluate a system in comparison to state-of-the-art systems to measure the system performance. As this project dealt with a specific dataset that is not employed by state-of-the-art approaches, it is expected to perform a baseline benchmarking using WikiHow.

3.4 Project Management Methodology

PRINCE2 is chosen as the project management methodology as this research project follows up a structural management process from project initiation to finalization in a controlled environment in order to achieve the expected milestone in each procedural step.

3.4.1 Gantt Chart

The Gantt chart relevant to project is provided in the **Appendix A.**

3.4.2 List of Deliverables

The below table contains all the formative and summative deliverables along with respective submission dates that need to be provided throughout the project timeline.

Deliverable	Date	Type of submission
Draft Project Proposal (Draft PP)	23rd Sep 2021	Formative Assessment (Report)
Literature Review	18th – 21st Oct 2021	Formative Assessment (Report)
Submission of Final Project Proposal and the ethical form	1st – 4th Nov 2021	Summative Assessment
Software Requirement Specification	22nd – 25th Nov 2021	Formative Assessment (Report)
Proof of Concept Version 1	6th – 17th Dec 2021	Formative Assessment

Interim Progress Report (IPR)	24th – 27th Jan 2022	Formative Assessment (Report)
Presentation of the Software Demo	21st Feb - 4th March 2022	Formative Assessment
Test and Evaluation Report	14th – 17th Mar 2022	Formative Assessment (Report)
Draft Project Reports	28th – 31st Mar 2022	Formative Assessment (Report)
Final Project Report	25th – 28th Apr 2022	Summative Assessment (Report)
Viva Voce Examination	16th – 30th May 2022	Summative Assessment (Report)

Table 7 : List of Deliverables

3.4.3 Resources Requirements

3.4.3.1 Hardware Requirements

- **Core i7 Process** – To perform extensive ML, NLP tasks.
- **Nvidia GTX 1050 GPU** – To accelerate model training.
- **16GB RAM** – To load a massive volume of data and to train deep NLP Models.
- **Hard Drive/SSD** – To persist data in local machine.
- **External Hard Drive** – To backup related project data externally.

3.4.3.2 Software Requirements

Software	Alternative software	Descriptions
Operating Systems	**Windows 10 / macOS / Linux distributions**	To handle high computation processes and to manage resources.
Word processing software	**MS Office / Google Doc**	For documentation.

Data science model building programming language	**R / Python**	Use for building data science models with the help of APIs provided.
Model building IDEs	**PyCharm / Kaggle IDE / Jupiter Notebook / Google Collabs**	Use as IDEs for developing Machine Learning related models easily.
Reference Management	**Zotero / Mendeley**	To store, manage and cite research papers.
Cloud data backups	**Google Drive / One Drive**	To persist and backup data in a cloud environment.
Deep learning libraries	**TensorFlow / Keras / PyTorch**	Libraries that assist in building deep learning models.
Natural language processing libraries	**NLTK / Spacy / Genism / Hugging Face**	Packages utilize in building of NLP models.
Data visualization libraries	**Matplotlib / Seaborn**	Use for visualization of data to identify correlations.
Machine learning libraries	**Scikit Learn / Pandas / NumPy**	Use as utility libraries in building machine learning models.
Version control systems	**GitHub / GitLab / Bitbucket**	Use as version control systems to manage project workload.
IDEs	**IntelliJ / VS code**	Use as development environments for building software applications.
Frontend frameworks	**React / Angular**	For developing frontend applications
Backend frameworks	**Flask / Django**	Frameworks utilize in creation of backend applications

Table 8 : Software Requirements

3.4.3.3 Skill Requirements

- Knowledge of statistics is required.
- Both theoretical and practical skills in Artificial Intelligence, Machine Learning, Deep Learning, Natural Language Processing are required.
- Critical thinking and evaluation skills are required to critically evaluate the existing system to find new insights.
- Need to have good knowledge of the research process that is necessary to conduct properly a research project.

3.4.3.4 Data Requirements

- **Online Dataset site that can be used in locating relevant data,**
  - Kaggle dataset repository, GitHub, Google public dataset, Paper with code, UCI machine learning dataset repository, etc.
- **Datasets for the NLP Project –** WikiHow dataset, VT-Sum dataset, X-Sum dataset
  - **WikiHow dataset** – A dataset consists of 230,000 data pairs of article and summary construct from an online knowledge base (Koupaee and Wang, 2018).
  - **VT-Sum dataset –** A Benchmark dataset that contains 125,000 pairs of lecture segmentation transcript-summary pairs (Lv et al., 2021).
  - **X-Sum dataset** – A dataset that uses for the evaluation of abstractive text summarization systems consists of 226,711 article and summary pairs (Narayan et al., 2018).

3.4.4 Risk and Mitigation

Risk Item	Severity	Frequency	Mitigation Plan
Extensive domain knowledge. As the project involves with abstract summarization process, a thorough understanding of theoretical aspects of the domain of natural language processing is required.	Medium	Low	Should properly allocate a sufficient timeline plan to do the research process and learning process parallelly to conduct an in-depth analysis of the reviews and to learn necessarily required concepts related to the domain.

Limited hardware resources It is required to have a good hardware configuration to be able to work with a highly intensive computational process.	High	Medium	As the project dealt with intensive use of deep NLP techniques it is required to have a good hardware configuration to the fast up the process of creation of models. As a solution, it can be use cloud-based resources which provide open-source facilities to work around hardware configuration issues if needed.
Requirement Changes It is needed to handle the requirement changes that may occur during the development process with each iteration properly.	High	High	The use of a proper iterative software development methodology will be beneficial to overcome the issue of rapid requirement changes because of its ability to adapt to the change of iterative requirements.
Limited Dataset As the project dealt with language summarization and text processing it is required to have a considerably massive volume of data to start on work with.	High	Low	Before initiating the project at the beginning when conducting research and review related to the problem that addressed, it is necessary to collect the identified dataset around the domain and select one that is most suitable for the research project and if the dataset is not available in

			sufficient amount, if possible, use techniques that can provide good results with lesser data around that specific domain.

Table 9 : Risk Management

3.5 Chapter Summary

Initially, the research methodology process of the conducting project was described under the main sub-parts which are philosophy, approach, strategy, choice, time horizon, and techniques. Secondly, the development methodology was explained by providing the selected software development methodology, elicitation methodology, and identified evaluation methodologies. Following the development methodology, the selected project management methodology was presented to explain how the management of the research project will be considered throughout the project timeline. After that, the work breakdown plan of the project was depicted in a grant chart by considering the respective deliverables of the research. Finally, the identified risk factor related to the project and its mitigation process was presented.

CHAPTER 4 : SOFTWARE REQUIREMENT SPECIFICATION

4.1 Chapter Overview

This chapter delivers in detail of how requirements are gathered for the research project and how the gathered requirement was analyzed in order to identify new findings and insight that benefit for conducting the research project. Initially, a rich picture is provided to illustrate the environment and system associated with the project. Then the stakeholder onion model diagram is provided to illustrate all the identified stakeholders related to the system followed by an explanation of the stakeholder viewpoints. Then the selected requirement elicitation methodologies and analysis of the gathered requirements through those methodologies are presented. Also, a context diagram, a use case diagram, and use case descriptions respective to the research project given. Lastly, all the identified functional and non-functional requirements related to the system are presented.

4.2 Rich Picture Diagram

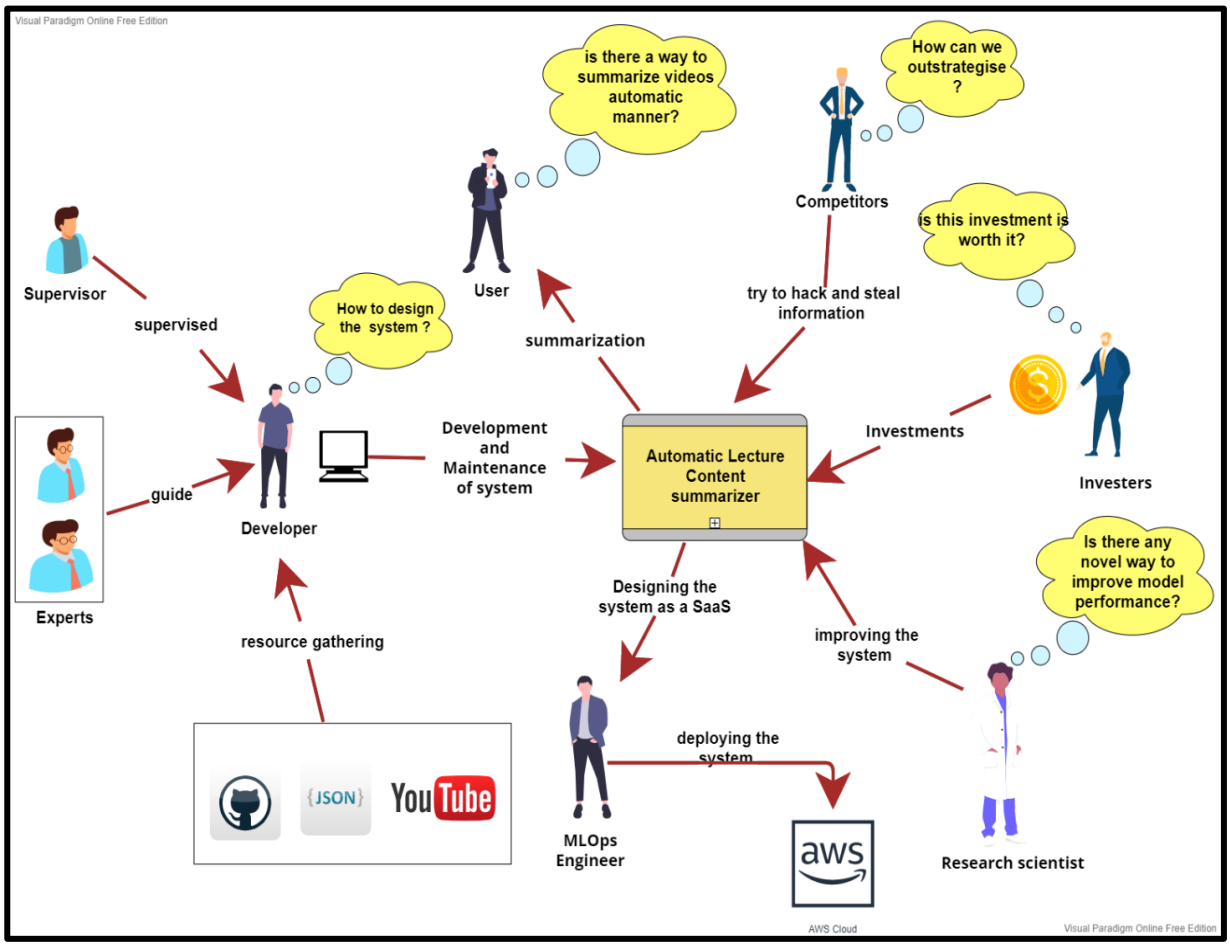


Figure 7 : Rich Picture of the System (Self Composed)

4.3 Stakeholder Analysis

All the recognized stockholders of the system and their associated environment with relevant relationships are depicted using an onion model along with an explanation of the role-play of each identified stakeholder in the system.

4.3.1 Stakeholder Onion Model

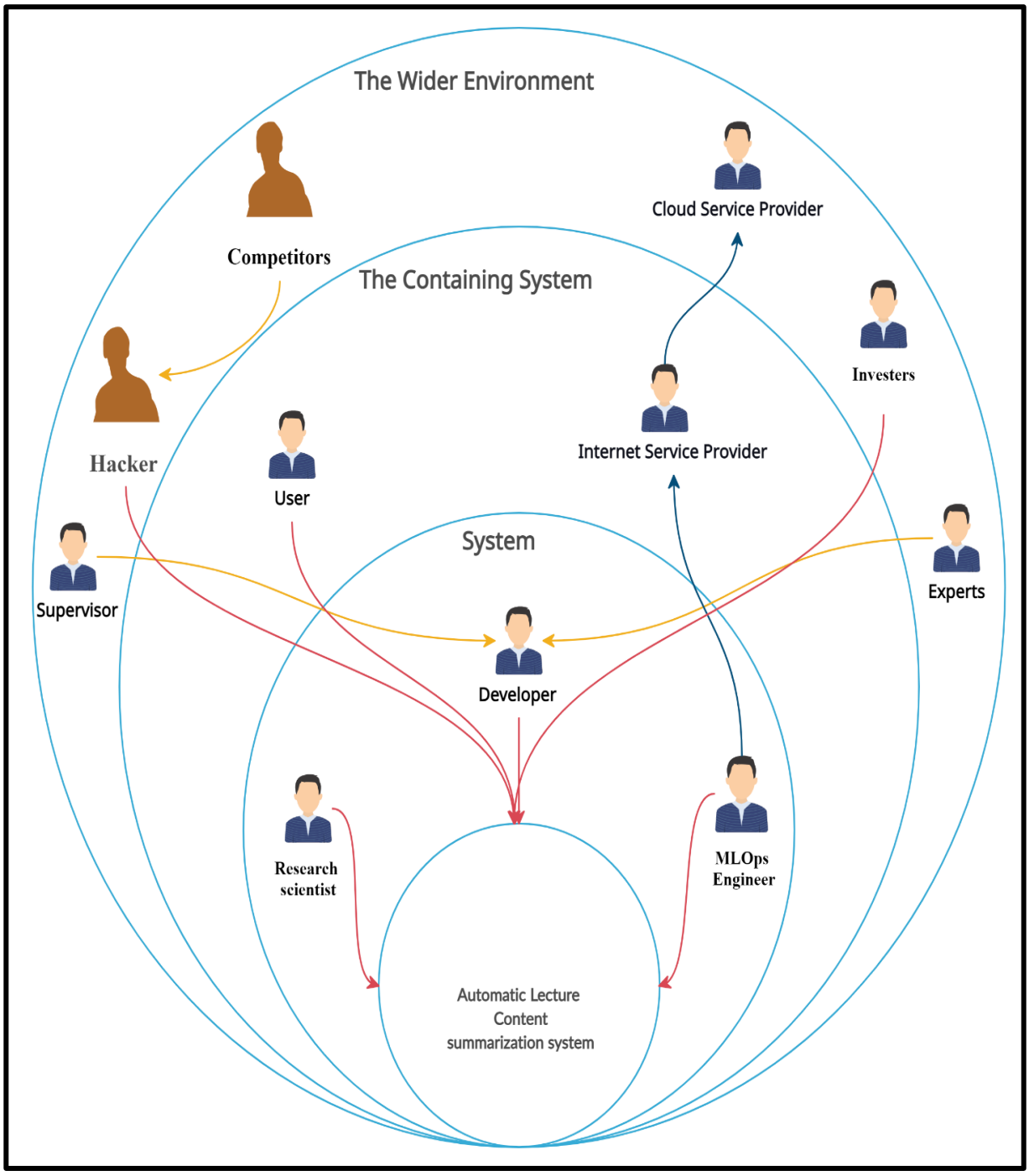


Figure 8 : Stakeholder Onion Model of the System (Self Composed)

4.3.2 Stakeholder Viewpoints

Stakeholder	Role	Description
The System Stakeholders		
Developer	System Operators	Responsible for developing and maintaining the system.
Research Scientist / Data Scientists		Responsible for identifying different novel approaches and to design and analyze the performance of the identified approaches through experiments in order to select the best methodology to develop the data science model.
MLOps Engineer		Responsible for deploying of machine learning model in the production environment and testing of the deployed model to ensure if it is working correctly and optimize code for low latency.
The Containing System Stakeholders		
User	Functional Beneficiary	Use of the summarization system for the creation of informative summaries for video content.
Internet Service Provider	Financial Beneficiary	Responsible for facilitating, domain registration, web hosting, and browser services to deploy the system and at the same time will earn financial benefits through these provided services.
The Wider Environment Stakeholders		
Experts	Advisory	Provide valuable feedbacks and insights to improve the system.

Supervisor		Provide guidance and supervision for the research project in order to complete the project successfully.
Investors	Financial Beneficiary	Provide financial support for the development of the product with the intention to benefit profit from the product.
Cloud Service Provider	Financial Beneficiary	Responsible for facilitating cloud services to deploy and maintain the product in the cloud.
Competitor	Negative Stakeholder	A direct competitor who creates similar systems with similar functionalities.
Hacker		Hackers will try to harm the product by breaching into the system in order to steal or manipulate the information.

Table 10 : Stakeholder Viewpoints

4.4 Selection of Requirement Elicitation Method

Requirement elicitation is a significant process in any research project which can be employed in order to gather features and opinions for the development and improvement of the project from the identified stakeholders of the system. According to the nature of the conducting research project, several requirement elicitation methods are chosen which are **Brainstorm, Literature reviews, Survey Questionnaires, and Prototyping** are some of the methods that were explored and analyzed to gather new findings.

Method 1 : Brainstorming
Brainstorming is a main primary requirement elicitation process that any of the researchers or project developers will be engaged in before initiating any project. This methodology can be conducted individually as well as collaboratively within a group in order to collect new research ideas and find solutions for the problem identified by thinking more freely, without

the fear of judgment. Due to the above reason as the primary research gathering methodology Brainstorming was chosen.
Method 2 : Literature Review
The most popular and much necessary requirement gathering process in any research base project is to conduct a thorough literature review of the existing systems related to the identified problem in order to find new research gaps in existing works that have not been explored yet. Not only that but also this methodology will be beneficial in order to identify the research challenges faced by existing works in order to get an idea of the feasibility of the researching problem domain and also identification of the solution methodologies that were used by previous work that can be useful to employ in building a robust solution for the identified research problem. Hence because of this in the context of this project, it was chosen and used Literature Review in the area of Automatic Text Summarization and Abstractive Text Summarization to build a solid foundation for the conducting research.
Method 3 : Survey Questionnaire
Questionnaires which is a survey methodology was chosen as another important requirement elicitation methodology to gather findings by intaking opinions from the general audience related to the research projects. This will be helpful for the author of the project to understand the opinion of the audience towards the developing system and what feature they expected to have in the developing system which would benefit the author in order to make some important decisions related to the project.
Method 4 : Prototyping
Prototyping can be used as a requirement elicitation methodology in the phase of development of the system as it will be beneficial in identifying errors through trials which could be utilized recursively to improve and rectify changes in the design and development phase of the project. As the conducting project follows prototype software development methodology, this method was chosen as a requirement gathering methodology.

Table 11 : Selection of requirement elicitation methods

4.5 Discussion of Results

4.5.1 Findings from Literature Reviews

Findings
• **Identification of the research novelty in the context of domain and technology** It was identified that the existing workaround related to automatic text summarization in the lecture domain is mostly done by utilizing extractive text summarization approach which produces extractive summaries that create summaries by concatenating the extracted salient information from the source document while the abstractive text summarization approach was used lesser for the creation of abstract summaries. Also, it was observed most of the automatic text summarization systems related to the lecture domain use traditional NLP approaches which cause in creation of subpar summaries (Miller, 2019). • **Identification of relevant datasets** It was difficult in finding more domain-specific datasets related to abstractive text summarization as it requires a huge amount of data samples for a summarization system to generate better summaries. So, by conducting a thorough research was able to track down several datasets which closely can be utilized in building an automatic abstractive lecture summarization system. Some of the identified datasets are, - Wikihow dataset (Koupaee and Wang, 2018) - VT-SSum dataset (Lv et al., 2021) • **Identification of Research Challenges** As mentioned under the research challenge section in the introduction chapter it was able to identify the complexity, feasibility, and significant level of the engaging research project in the context of abstractive automatic text summarization. • **Identification of techniques and methodologies used in the context of abstractive text summarization** By a thorough literature survey, it was observed that most of the existing workaround in abstractive text summarization has utilized Seq2Seq models with different configurations to create quality summaries. While lesser experiments are done with the recent trending transformer technology which provides space for researchers to introduce new solutions.

Table 12 : Findings through Literature Review

4.5.2 Findings from Brainstorming

Findings
Identification of the initial idea/problem which is the possibility of creation of concise coherent summaries that are closer to human-generated summaries and how it can be applied to the domain of education for the enhancement of effective learning leads to aiding in coming up with the research project idea of abstractive text summarization for lecture domain.

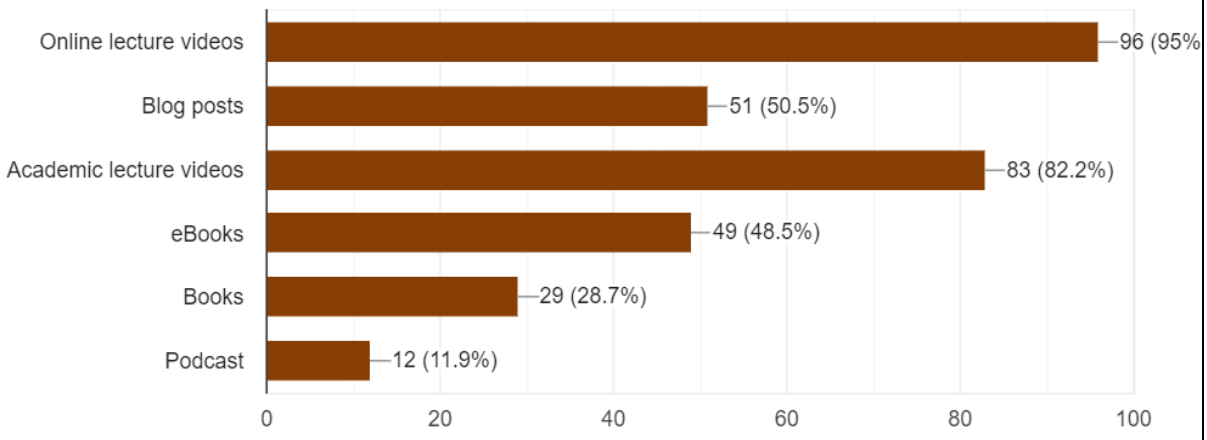
Table 13: Findings through Brainstorming

4.5.3 Findings from Survey Questionnaire

The findings taken by analyzing **101 responses** that were received from the distributed questionnaire were represented below. Screenshots related to the shared distributed questionnaire form are presented in **Appendix C**.

Question	Which type of educational content are you usually exposed to?
Aim of the Question	To identify major educational material that user(s) are mostly exposed to.

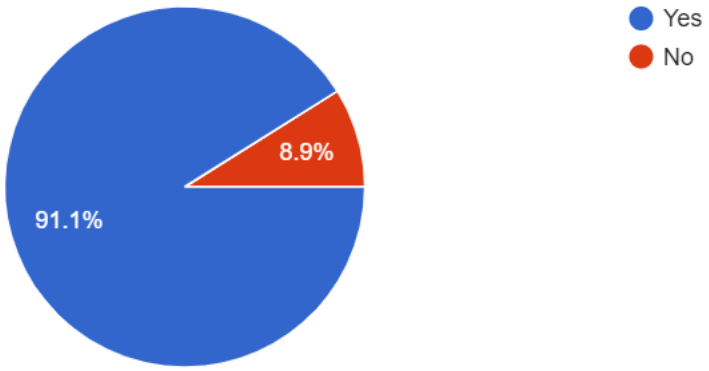
Findings



From the provided educational sources, it was observed that 96% of participants have chosen online lecture videos as the most exposed educational source and 83% of participants have been exposed to academic lecture videos while around or less than 50% of participants are exposed to other educational sources. So it can be concluded that the majority of the audiences who have undertaken the survey are exposed to the use of video-oriented educational sources. So as this project expectation is to utilize video transcripts as the input from videos for the summary generation it can expect that the majority of the audience response for the rest of the survey will tally with the project purpose.

Question	Do you feel any need of an Automatic Lecture Content Summarizer System?
Aim of the Question	To identify the opinion of the audience about the need of having an Automatic lecture content summarizer

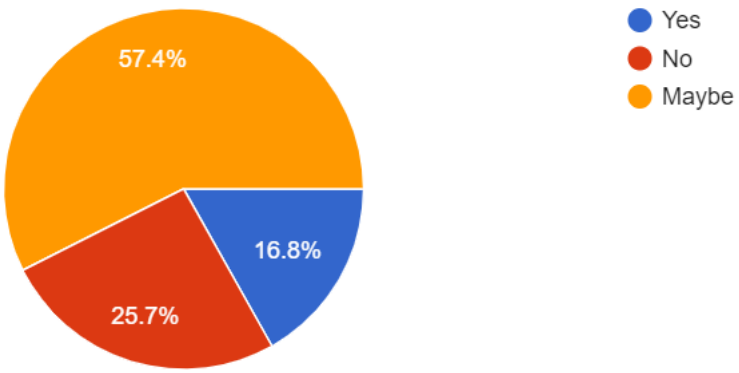
Findings



Around 91% of participants have felt that having an automatic text summarization will be helpful for them. This concludes that the currently engaging research project on automatic lecture content text summarization will be useful for the majority of user(s) who expect to have an automated summary creation tool.

Question	Are you satisfied with manual creation of summary by referring a lecture video ?
Aim of the Question	To identify the satisfaction level of the audience towards manual creation of summary

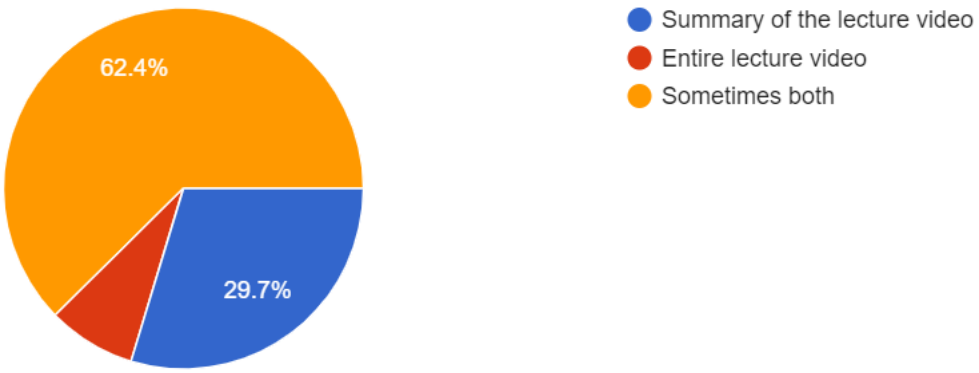
Findings



57% of participants think that manual summarization for lecture videos may be helpful for them, and around 26% of participants are not satisfied with manual summary creation for lecture videos while lesser participants provide their interest in manual summarization. According to the above observations, it can decide that the majority of the audience is skeptical about the process of the manual summary creation. So it can be concluded that having an automatic text summarization system that generates concise coherent summaries will be a good alternative for the manual summary generation process.

Question	Do you prefer to view a summary of a lecture video or the entire lecture video?
Aim of the Question	To identify if the participants are more intended towards watching a whole lecture video or a summary of the lecture video.

Findings

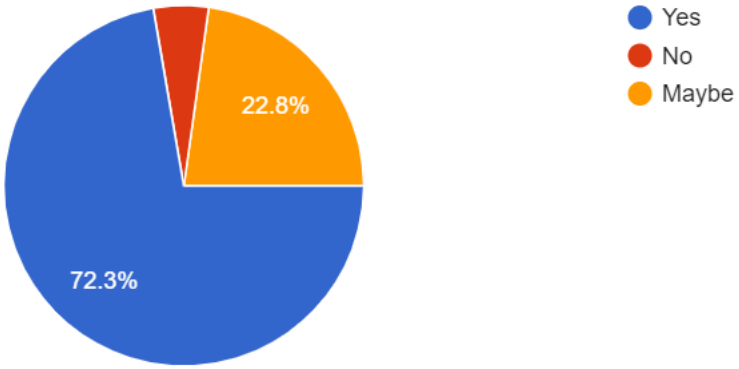


It is observed that around 62% of participants prefer to use both summaries of the lecture video and to refer to the whole lecture video content while around 30% of participants prefer to use a summary of a lecture video rather than watching the whole video. very lesser, around 8% of audiences are preferred to only watch lecture videos. So it can be confirmed that the majority of the audience will prefer having an informative summary related to a lecture video.

Question	Do you think that having an Automatic Lecture Content Summarization System would be beneficial for effective learning?

Aim of the Question	To identify the opinion of the participants about the effectiveness of having an automatic text summarization system for the learning process.

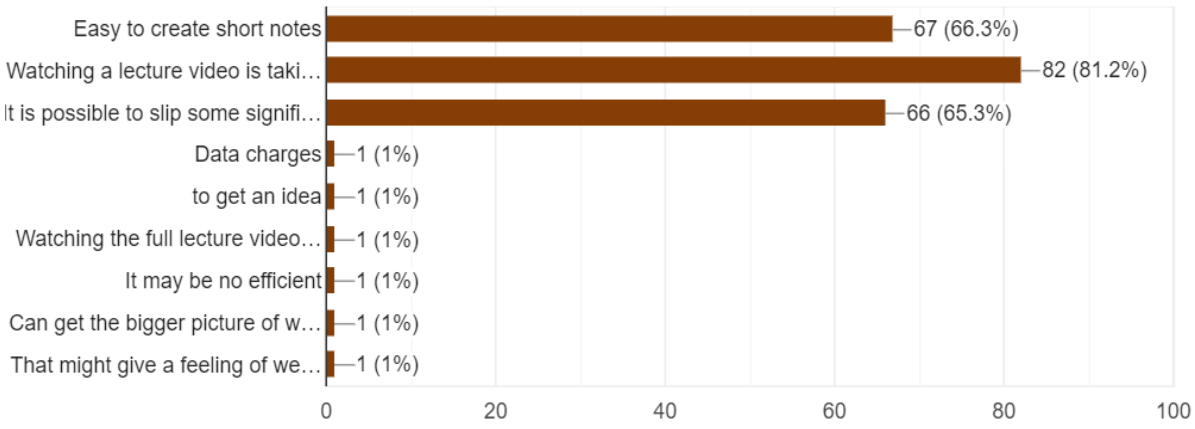
Findings



Around 72% of participants think that having an automatic text summarization will benefit for effective learning. This concludes that most of the audience is interested in having an automatic text summarization system to enhance their educational activities.

Question	What are the reasons for you to choose an Automatic Lecture Content Summarization System?
Aim of the Question	Identification of the reasons behind the user(s) to choose an automatic text summarization system

Findings

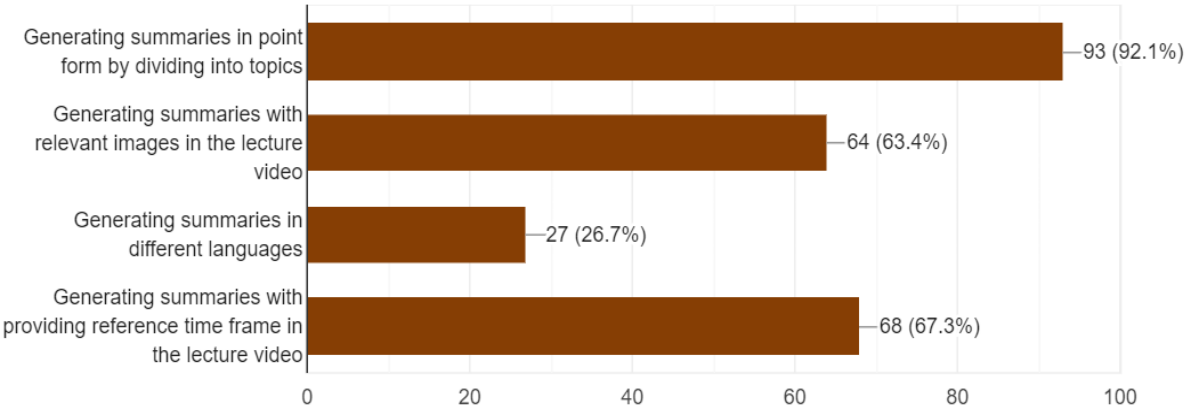


Almost around 82% of participants state that the purpose for them to reach for an automatic text summarization system is due to more consumption of their time when referring to a

lecture video. While around 67% of participants state that it will provide an easy way to create short notes and might contain significant information that they can refer later without going through the lecture video again. These observed statistics confirm all the reasons that pointed out to initiate this research project in automatic text summarization.

Question	What type of functions do you prefer to have in a Automatic Lecture Content Summarization System?
Aim of the Question	To identify audience perspectives on functionalities that they expect to have in an automatic text summarization system

Findings



93% of the participant willing to have an automatic text summarization system that is able to generate summaries in point form for different topics, while around rest of the others prefer to have other options mentioned above. As this project does not follow-on providing point base summary creation for different topics this mostly demanded feature can be included as a future requirement for the automatic text summarization system.

Table 14 : Findings through Survey Questionnaires

4.5.4 Finding from Prototyping

Findings
When initial conducting prototyping with several algorithms to identify the best techniques, it was observed that the transformer model is more efficient than the traditional LSTM (Long Short-Term Memory Network) based Seq2Seq models for automatic abstractive text summarization tasks.

Table 15 : Findings through Prototyping

4.6 Summary of Findings

Id	Findings	Brainstorming	Literature Reviews	Survey Questionnaire	Prototyping
1	Initiating the initial project idea which is automatic text summarization for educational video content.	✓			
2	Identification of the research gaps and the novelty of the research project in the context of both the domain and technology.	✓	✓		
3	Identification of relevant datasets related to abstractive text summarization in the educational domain. (Wikihow, VT-SSum)		✓		
4	Identification of research challenges related to abstractive text summarization.		✓		✓
5	Identification of most commonly used techniques in abstractive text summarization field in order to identify the technological gap that can be purpose as solution novelty. (Ex: Seq2Seq was identified as one of the most of the commonly used techniques in the context of automatic abstractive text summarization)		✓		
6	The system must design in a way that it intakes educational videos as the input source for the generation of textual summary.	✓	✓	✓	
7	The automatic text summarization system should produce concise coherent informative summaries		✓	✓	
8	Identifying that utilization of transformer model for automatic abstractive text summarization is more effective than utilization of traditional Seq2Seq model.				✓

9	Identification of several out-scope requirements that can be employed for the model improvement in the future. Such as - Point-based summary generation - Video Annotated Summary generation - Generation of summary in different languages etc.		✓	✓	
10	Identifying that utilization of transformer model for automatic abstractive text summarization is more effective than utilization of traditional Seq2Seq model.				✓

Table 16 : Summary of Findings

4.7 Context Diagram

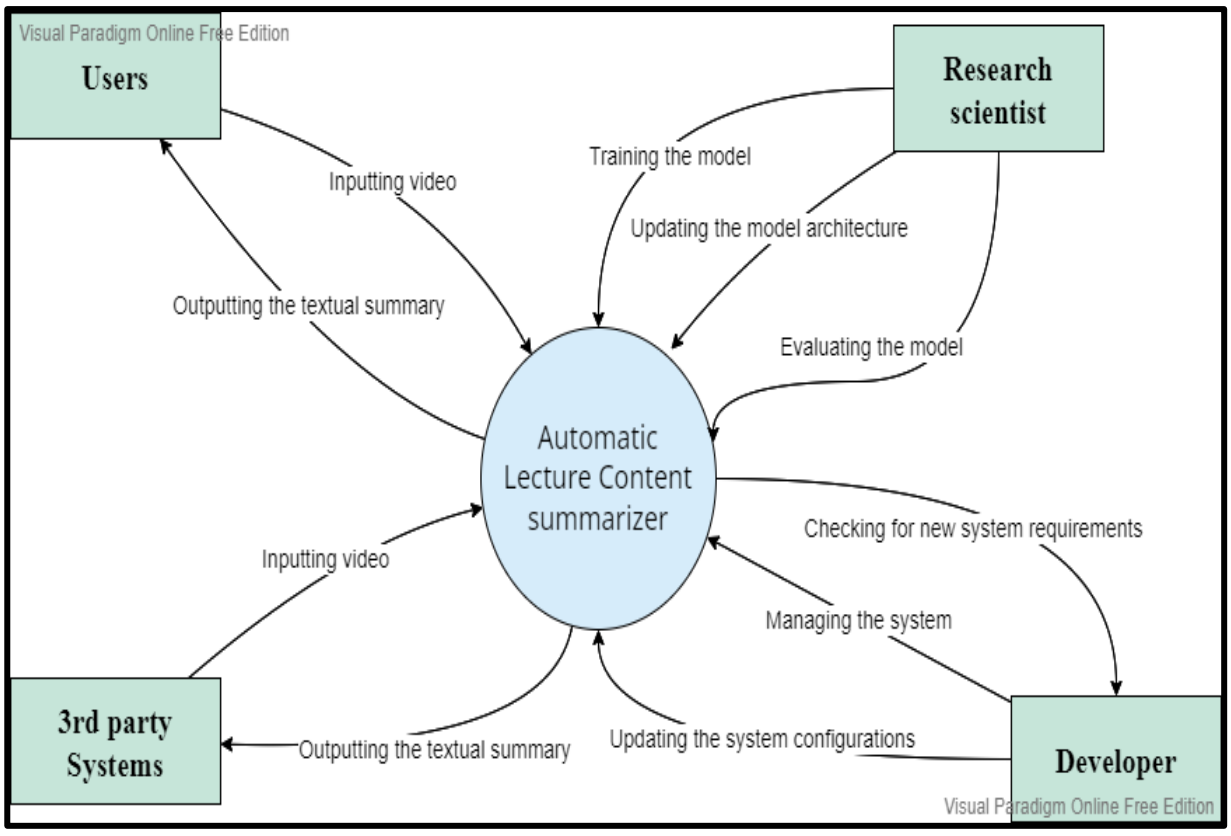


Figure 9 : Context Diagram (Self Composed)

The above context diagram depicts the flow of information between the system and external components. **Users** represent the general user(s) who interact with the system by inputting a lecture video which will be processed by the system and output an informative abstract summary. **Third part Systems** like online education platforms and other businesses in need of

summarization tools can also use the system as a supporting plugin by engaging the same interaction with the system as the general user(s) in order to produce an abstract summary for inputted video. **Research Scientists** also known as data scientists will be responsible for maintaining the data science core model of the system by upgrading and rectifying identified issues. The **developer** is responsible for managing the system by resolving identified issues in both the frontend and backend of the end application.

4.8 Use Case Diagram

The Use Case Diagram of the system is depicted below.

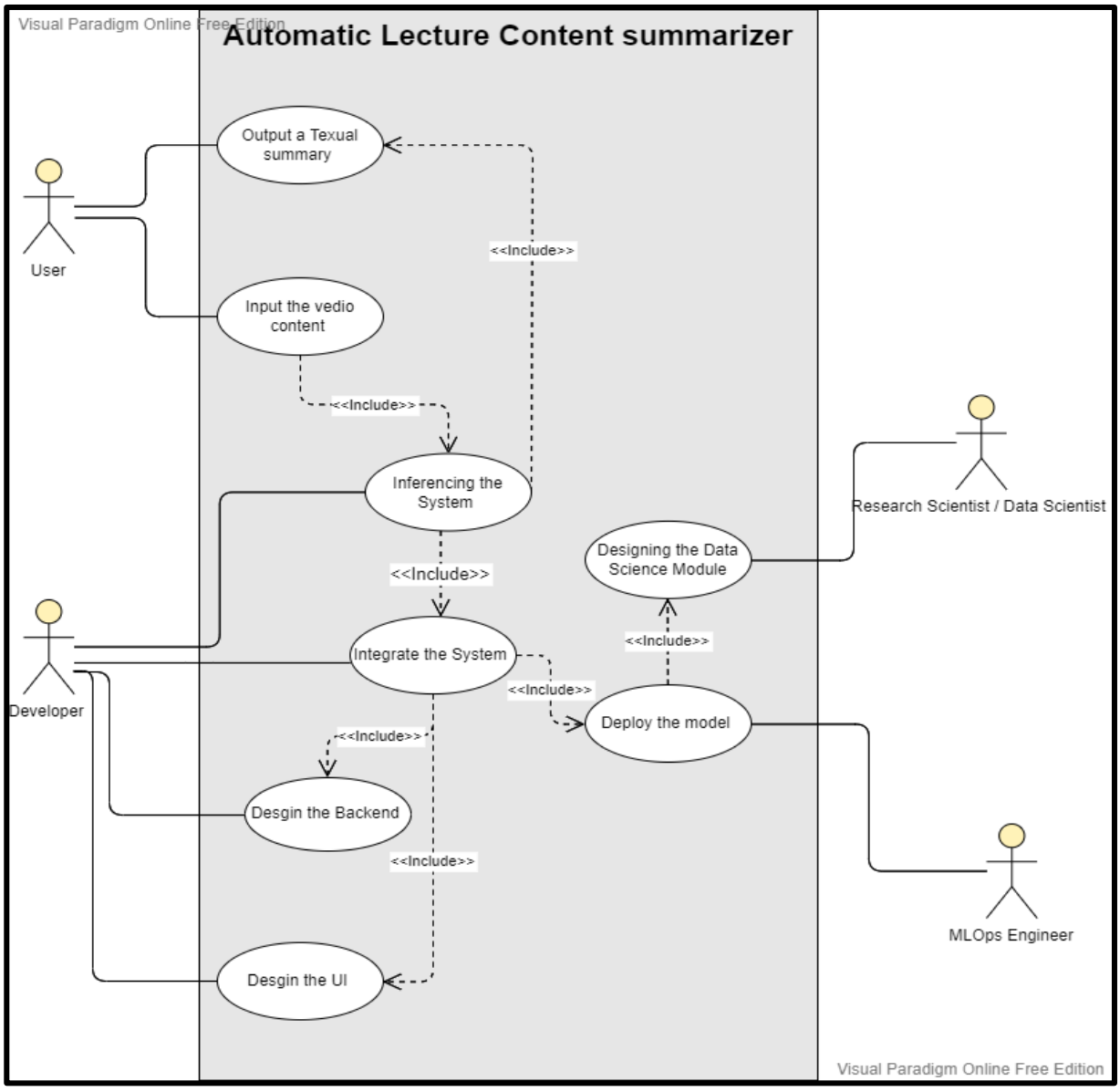


Figure 10 : Use Case Diagram (Self Composed)

4.9 Use Case Description

Below contains the Use Case Description tables for respective identified main Use Cases through the Use Case diagram.

Use Case Name	Input the Video Content
Description	Users can input video that needs to be summarized
Participating Actors	User
Pre-conditions	- Video must be in valid data format. - Maximum time duration of the inputting video should be 15 mins.
Extended use cases	N/A
Included use cases	Inferencing the system
Main flow	1. The user should input the video into the system 2. The audio clip of the video should be extracted for the creation of the video textual transcript. 3. Then the system should produce the abstract summary for created transcript and display it to the user.
Alternative flow	N/A
Exceptional flows	N/A
Post conditions	It should generate a summary that contains the context of the inputted video.

Table 17 : "Input the Video Content" Use Case Diagram

Use Case Name	Designing the Data Science Model
Description	Developing, training, testing, and evaluating data science model
Participating Actors	Research Scientist / Data Scientist, ML Ops Engineer
Pre-conditions	- Should contain the appropriated dataset related to the system - Should contain the architecture solution design for the data science model
Extended use cases	N/A
Included use cases	N/A
Main flow	1. Loading the dataset 2. Cleaning the loaded dataset 3. Splitting the pre-processed dataset to train and test sets

	4. Train the model 5. Test model 6. Creation of the best optimal model
Alternative flow	N/A
Exceptional flows	N/A
Post conditions	Creation of the main optimal data science model

Table 18 : " Designing the Data Science Model" Use Case Diagram

Use Case Name	Inferencing the system
Description	Using the system to get the final result
Participating Actors	Developer, User
Pre-conditions	- The data science system should be developed and deployed
Extended use cases	N/A
Included use cases	Integrate the system, Output a textual summary
Main flow	1. Integrated the system (Frontend, Backend, Data Science model) 2. Produce the result for inputted video.
Alternative flow	N/A
Exceptional flows	N/A
Post conditions	Output the generated textual summary

Table 19 : "Inferencing the system" Use Case Diagram

4.10 Functional Requirements (NR) with Prioritization

The identified Functional and Non-Functional requirement of the system is prioritized using the MoSCoW Prioritization technique according to the significance of the identified functionalities.

Priority Level	**Description**
Must have (M)	The essential main requirement that the system should definitely contain.
Should have (S)	Important requirements that are not vital for the system but that can add significant value for the system.
Could have (C)	The requirements that can be optionally included in the system. Nice to have requirements that impact less for the system.
Will not have (W)	The requirement that the system will not contain and that are not prioritized for the current time frame.

Table 20 : "MoSCoW" Prioritization Levels

NR Id	Requirement Description	Priority Level
FR1	User(s) should be able to upload videos into the system through the YouTube video link directly for the creation of informative summaries.	Must have
FR2	The maximum time duration of the inputted video must be around 15 min.	Must have
FR3	The Input video URL should be validated on the frontend side to check if it is the expected YouTube video link.	Must have
FR4	Extraction of audio clip from the input video for the creation video transcript document.	Must have
FR5	The extracted audio clip should be converted into a textual transcript document.	Must have
FR6	Pre-processing of the created textual transcript by utilizing appropriate natural language processing techniques.	Must have
FR7	Preparing the train and test dataset for training the model	Must have
FR8	Creation of many to many transformer-based model	Must have
FR9	Adaptation of pointer generator network to the created transformer base model.	Must have
FR10	Implementation of GANs framework architecture to improve the model performance	Must have
FR11	Use of quality evaluation matric related to abstractive summaries to evaluate the model performance	Must have
FR12	Performing benchmarking on the implemented system for evaluation of the performance of the abstractive text summarization model.	Could have
FR13	Generation of an abstract informative summary as the output from the developed NLP model.	Must have

Table 21 : Functional Requirements

4.11 Non-Functional Requirements (NFR)

NFR Id	Requirement Description	Requirement type	Priority Level
NFR1	The system must be able to generate a summary for the inputted video within 10 mins.	Performance	Could have
NFR2	The system should contain a user-friendly UI interface	Usability	Must have
NFR3	The system should generate human-understandable abstract summaries at least for a considerable level.	Quality	Must have
NFR4	The system should not save videos or the generated transcript without user consent.	Security	Could have
NFR5	The system should be able to extensible with newer requirements.	Scalability	Should have
NFR6	The system must be deployed to be able to access with a wider range of user(s)	Compatibility	Could have

Table 22 : Non-Functional Requirements

4.12 Chapter Summary

The chapter provides a detailed explanation of the process of requirement gathering, identification of stakeholders, and functionalities of the research project by pointing out how the gathered and identified information would help for the development and enhancement of the system. It started with an illustration of a rich picture diagram of the system which helped in identifying the stakeholders of the system which was then depicted through another diagram of the onion model in order to provide a high-level visual representation of how the recognized stakeholders interacted with the system and followed by that providing an explanation for the role-play of each recognized stakeholder. The chapter then moved on to explaining the chosen requirement gathering methodologies for the system and reasoning about the selected methodologies and other necessary artifacts are provided.

CHAPTER 5 : SOCIAL, LEGAL, ETHICAL & PROFESSIONAL ISSUES

5.1 Chapter Overview

This chapter focus on presenting the identified social, legal, ethical, and professional issue in the research project with relevance to BCS code of conduct and the process for mitigating those identified issues.

5.2 SLEP Issues and Mitigations

Social	Legal
• In the survey, which was done in the requirement gathering section of the research project, none of the participants information was gathered. Also, the participants of the survey were properly informed through the project description section of the survey that their information will be kept anonymous and will only be used for academic purposes, and by accepting to fill the survey they allow the questionnaire author to gather project requirements. • The response received for the survey questionnaire was never added to the project to respect and protect the confidentiality of the participant's opinions. • All the interviewers that were used in the evaluation section of the research project were made aware that their information will not be disclosed without their consent and if they allow for disclosure of their information, it	• All the selected development technologies related to the research project are licensed under open-source licenses. • The dataset that was selected and utilized to develop the system is an openly available dataset for the research community in the automatic abstractive text summarization domain and the relevant publication related to the dataset was properly cited in the thesis. • All the participants and respondents feedbacks are kept confidential and used only for academic purposes.

will be only used for the research purpose of the project and also along with that it was informed that their feedbacks related to project will be disclosed for academic purposes.	
Ethical	**Professional**
• All the participants who are used for surveys and interviews are made aware of the research project in order to make them understand of what type of research they are contributing to and how it would their contributions will be utilized in the project. • The privacy of all participants of the research project was well respected and without their consent, no information was disclosed in the project. • Main consideration is given to plagiarism in the research project. All the outsource publication which is utilized in the project is properly cited with the relevant reference providing credibility to the authors of the publications.	• In the implementation phase of the system industrial standard and best practices for development was followed through in creation of a proper prototype. • No manipulation was done to the original collected requirements and feedback relevant to the project. • All the final results obtained for the research project were disclosed with proper reasons and without no manipulation to misinterpret the success of the project. • The dataset utilized for the project is selected by considering thoroughly the close relevance with the project domain and it was not manipulated with fabricated data to create fake results.

Table 23 : Identified SLEP issues and Mitigations

5.3 Chapter Summary

The chapter outlined all the considered SLEP issues related to the project with relevance to the BSC code of conduct in order to prove the competency, integrity, and ethical clearance of the project.

CHAPTER 6 : SYSTEM ARCHITECTURE & DESIGN

6.1 Chapter Overview

The chapter focuses on providing a detailed explanation of the design aspects considered throughout the research project, which will aid in the system's development. The chapter begins by presenting the design objectives/goals that the project's author considered to be present in the system. Following the specification of the design goals, the chapter will illustrate the system architecture diagram to represent the overall system functionality, as well as an in-depth explanation of the depicted architectural diagram. Finally, the chapter will present the project's chosen design paradigm as well as the system's data flow diagram.

6.2 Design Goals

Design Goals	Description
Correctness	The system should be able to generate an informative abstractive concise summary that contains the gist of the inputted lecture video source while the generated summary should be able to comprehend by the general user(s) with ease which means the summary must be semantically meaningful for a considerable level. If the final system does not perform as mentioned, it is considered the system if a failure.
Performance	The process of summary generation should take less amount of time in order to make the system an efficient tool to be used by the end-user.
Scalability	The system should be designed in a way that it can adapt new features without any breakdown and lags, in order to ease the usability of the system for new researchers to conduct future add-ons to the product.
Reusability	The automatic text summarization part of the system will be developed in a way that it can be deployed by other external systems to use that summarization component for their purposes.
Usability	The system should be designed by considering the usability easiness for both general user(s) and developers. Where having proper GUI that follows standard UI principles will ease the usability aspect of the general user(s) while consideration of relevant code development best practices will enhance the system usability for the developer(s).

Table 24 : Design Goals of the System

6.3 System Architecture Design

6.3.1 Tiered Architecture

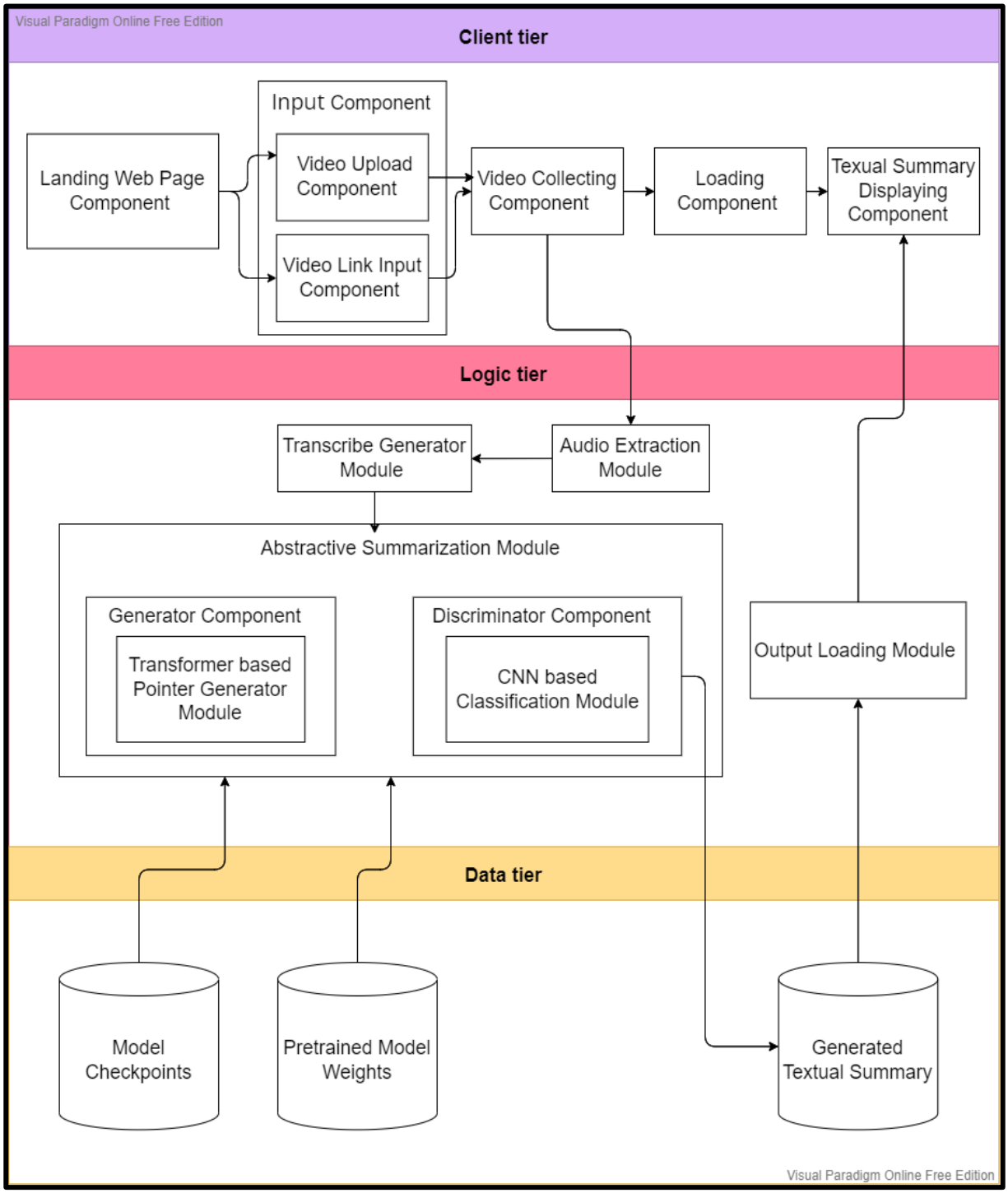


Figure 11 : Tired Architecture of the System (Self Composed)

The system's high-level architecture design for all the functionality that intake within the system is depicted through a tiered architectural diagram as above which consists of three main layers which are Client tier, Logic tier, and Data tier. The Client tier serves as the presentation layer of

the system which contains all the UI components that the system will adapt in later development. Its sole purpose is to display information and collection of information from the user. The Logical tier also known as the application tier layer serves to process the information gathered in the client tier sometimes in conjunction with data from the data tier by using business logic, or a specific set of business rules. The final layer which is the Data-tier layer serves the purpose of storing and managing the processed information in the end application.

Client Tier.

- **Landing Web page Component** – The initial UI interface that will be presented to the user(s).
- **Input Component** – The UI component that provides the ability to the user(s) to upload a valid video file for the system.
- **Video Collecting Component** – The Ui component that will display the upload video thumbnail.
- **Loading Component** – The UI component that will be presented until the end result from the system is available for the user(s).
- **Textual Summary Displaying Component** – The UI interface that displays the final output which is in this project the end summary.

Logic Tier

- **Audio Extractive Modul**e – Serves to extract the audio clip from the inputted video file in order to create the transcribe.
- **Transcribe Generation Module** – Serves to create the textual transcription document from the extracted audio clip of the inputted video.
- **Abstractive Text Summarization Module** – The main heart of the system that serves the process of the generation of abstractive textual summaries from the given video transcribes.
- **Output Loading Module** – The module that performs the loading of the generated end result to the client layer to be processed in the frontend to display for the user(s).

Data Tier

- **Model checkpoint –** Use to store the updated core model versions of the system.
- **Pre-trained Module weights –** Use to store the weight of the identified optimal model for inference.

- **Generated Textual Summary –** Use to store the summarized version of the transcribes in order to tackle the false tolerance of the system (Ex: In case of network breakdown)**.**

6.4 System Design

6.4.1 Choice of Design Paradigm

Structured Systems Analysis and Design Method (SSADM) was chosen as the design paradigm of the conducting project as the designing of the core model of the system follows a procedural approach that contains specific tasks for each defined stages in the process. So, because of this reason, SSADM methodology was selected as the most suitable design paradigm.

6.4.2 Data flow diagram

As **Structured Systems Analysis and Design Method (SSADM)** was chosen as the design paradigm for the conducting project the relevant one of the design diagrams that can be fallen under this paradigm is data flow diagram. So in this context, the data flow diagram is depicted to showcase the flow of data within the automatic abstractive text summarization system.

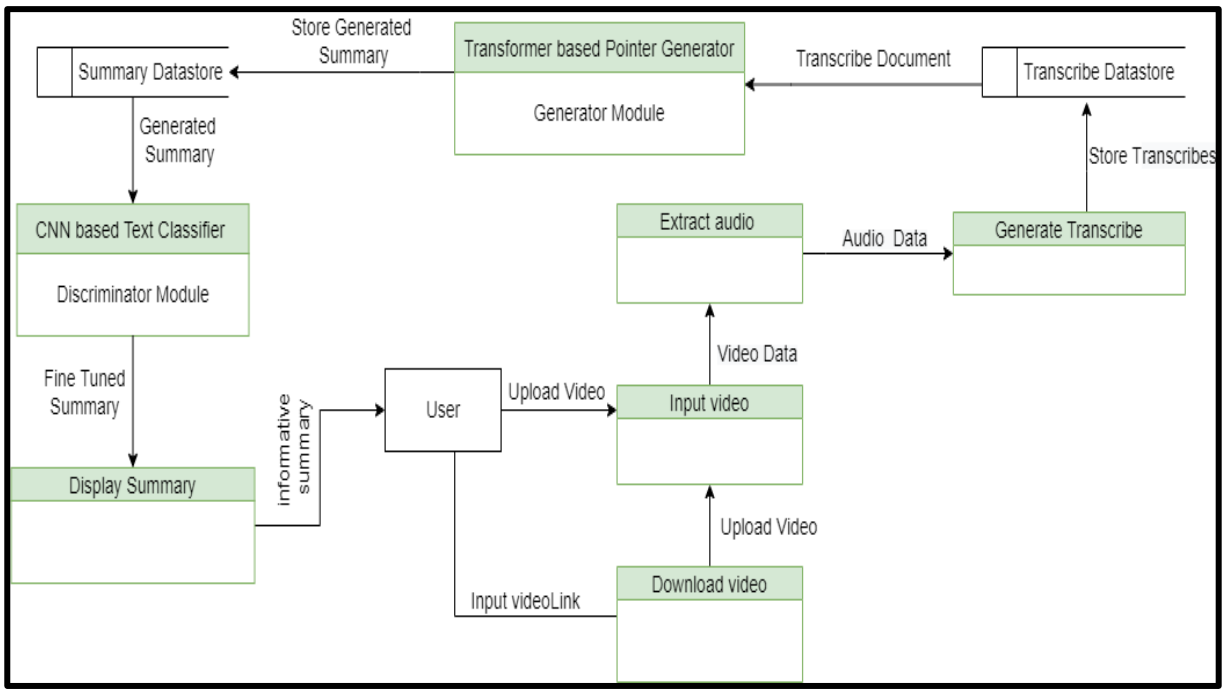


Figure 12 : Data Flow Diagram (Self Composed)

The above-illustrated Data Flow diagram of the automatic abstractive text summarization system showcase how the data flows through each process in the system.

6.4.3 UI Design

The UI for the abstractive text summarization system was designed as a web application with the intention of providing a simple and less complicated user interface to the end user(s). Along with the consideration of the user experience, the UI was designed by considering the project requirement scope. The below wireframes depict the landing page which consists of an input box that expects a YouTube video URL from user(s) for the generation of an abstract summary. The generated abstract summary results and the given YouTube video will be uploaded and displayed in the UI as depicted in the screenshots provided in the **Appendix D**.

6.4.4 System Process Flow Chart

The system process flow chart illustrated below displays how the data flow through the automatic abstractive text summarization system and how the decisions are taken in different stages in order to handle various events of the system.

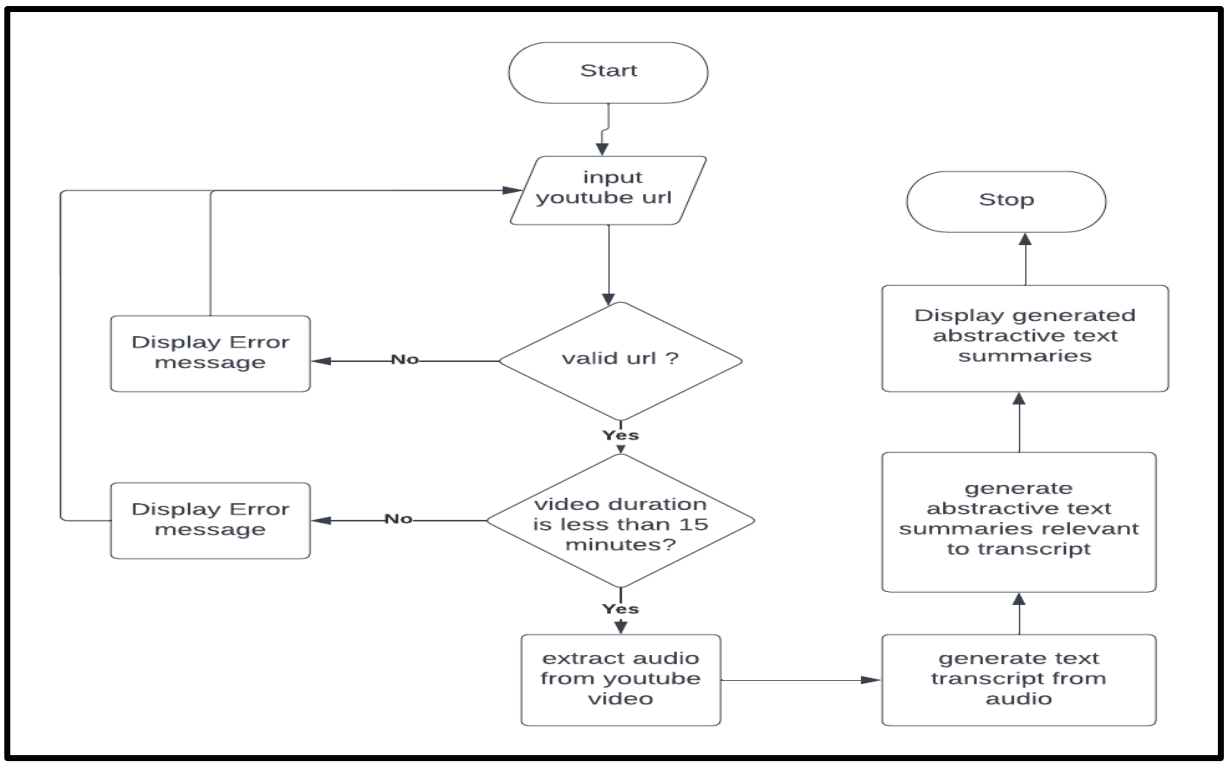


Figure 13 : Process Flow Diagram (Self Composed)

6.5 Chapter Summary

All the design considerations that are taken into count for designing the overall system of the conducting projecting were explained thoroughly throughout the chapter by providing the design goals, system high-level design, chosen design paradigm, and data flow diagram.

CHAPTER 7 : IMPLEMENTATION

7.1 Chapter Overview

The chapter presents all the necessary details that followed up in implementation of the system prototype for the conducting research project. It provides an explanation with valid reasons for the technological selection, data selection, programming language selection, Libraries utilized and IDE utilized. The initial prototype's core functionalities and APIs are explained with relevant code snippets to describe the overall implementation process intake in each phrase.

7.2 Technology Selection

7.2.1 Technology Stack

The overall technological stack that is chosen to utilize in various stages of the development process of the research project is given below.

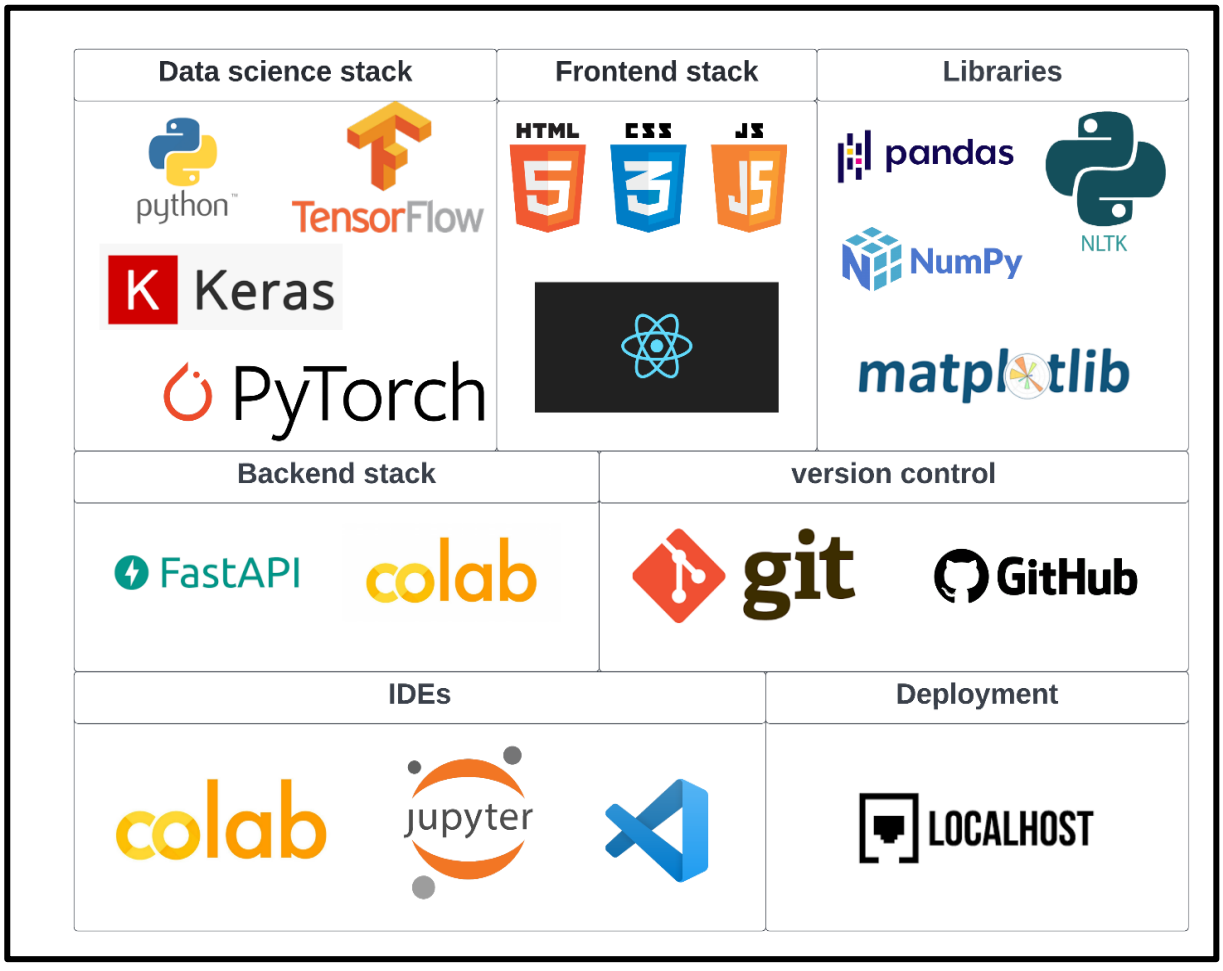


Figure 14 : Technology Stack (Self Composed)

7.2.2 Data Selection

As the research project is focused on generating an abstractive summary for educational video content it was needed of having a large-scale dataset that is suitable for addressing the identified research problem. Out of all the abstractive datasets presented currently in the research field, it was identified three matching datasets that fulfil data requirements to do the abstractive summarization process of the conducting project. Those data sets are,

WikiHow dataset – A dataset consists of 230,000 data pairs of article and summary construct from an online knowledge base (Koupaee and Wang, 2018). This was the dataset that utilize in the creation of the initial prototype model.

VT-SSum dataset – A Benchmark dataset that contains 125,000 pairs of lecture segmentation transcript-summary pairs (Lv et al., 2021).

X-Sum dataset – A dataset that uses for the evaluation of abstractive text summarization systems consists of 226,711 article and summary pairs (Narayan et al., 2018).

7.2.3 Selection of development framework

TensorFlow

TensorFlow is an open-source framework for building machine learning models from start to finish. TensorFlow was created by researchers and engineers on the Google Brain team, which is part of Google's Machine Intelligence Research division, to undertake machine learning and deep neural network research. The system is broad enough to be used in a variety of different fields. In this study, the TensorFlow framework was utilized to develop a transformer-based model.

7.2.4 Programming Language

Python was used as the main language in implementing the data science component of the research project as Python is known to be a popular Machine Learning language with support for various libraries. Python also supports multiple programming paradigms. It fully supports Object-Oriented Programming and structured programming as well as support for functional and aspect-oriented programming as well. Another key feature of Python is compatibility with major platforms and operating system architectures. Being an interpreted language, this allows Python to run the same code on any platform with no modifications to the source code. Other notable characteristics include large and robust libraries, support for a variety of open-source Python frameworks and tools, and test-driven development. By taking into consideration of the above

pointed out reasons it was used python rather than R language for the development of the initial prototype.

7.2.5 Libraries utilized

Keras

Keras is a Python-based deep learning API that runs on top of the TensorFlow machine learning framework. It was created with the goal of allowing for quick experimentation. This library was used when designing the layered components of the transformer-based model.

NumPy

NumPy is a Python library that adds support for huge, multi-dimensional arrays and matrices, as well as a large number of high-level mathematical functions to interact with these arrays. NumPy also includes a number of useful mathematical functions, such as random number generators, linear algebra procedures, and Fourier transformations. So, this library was employed to use multi-dimensional arrays.

Pandas

Pandas is a data manipulation and analysis software package for the Python programming language. It includes data structures and methods for manipulating numerical tables and time series, in particular. It's an open-source tool built on top of the Python programming language that's quick, powerful, versatile, and simple to use. This was used to manipulate the data and handle pre-processing tasks.

NLTK

The most commonly used library in the domain of natural language processing tasks and computational linguistics. This library was utilized for the pre-processing component of the data science model.

7.2.6 IDEs utilized

Jupiter Notebook

Jupiter Notebook which is an open-source tool was used as the IDE for the development of the initial data science prototype component as it provides the facility to work with the python programming language and it helps to utilize GPU and RAM to train the model in an optimal manner for a long period of time.

Google Colab

Google Colab aka Google Colaboratory is an open-source cloud host notebook provided by Google Research team that was specifically designed to run arbitrary python code. As it provides hosted facilities and is mostly used in domains of machine learning this editor was utilized in the creation of the system prototype.

7.2.7 Summary of Technology selection

Groups	Component	Tool
Frontend Component	Programming Language	JavaScript
	Frontend frameworks	React
	UI component library	React Ant Design
Data Science Component	Programming Language	Python
	Libraries	Scikit Learn, Pandas, NumPy, Matplotlib, TensorFlow, Keras, NLTK
	IDE's	Jupiter Notebook, Google Colab
Backend Component	Programming Language	Python
	Development Framework	Fast API, Colab Code
	IDE	Google Colab
Other tools	Version Control System	Git, GitHub

Table 25 : Summary of Technology selection

7.3 Implementation of Core Functionalities

This section provides an explanation for the implemented core functionalities in the final prototype with aid of relevant code snippets.

7.3.1 Pre-processing of Data

Before beginning the model-building process, it is critical to complete the basic pre-processing steps. Using unclean and sloppy text data is a potentially disastrous move. So, in this step, it will remove all of the unnecessary symbols, characters, and so on from the text by utilizing natural language processing techniques such as stop word removal, tokenization, contraction mapping etc.

```
In [57]: import nltk
         nltk.download("stopwords")

         stop_words = set(stopwords.words('english'))

         def text_cleaner(text):
             newString = text.lower()
             newString = BeautifulSoup(newString, "lxml").text
             newString = re.sub(r'\([^)]*\)', '', newString)
             newString = re.sub('"','', newString)
             newString = ' '.join([contraction_mapping[t] if t in contraction_mapping else t for t in newString.split(" ")])
             newString = re.sub(r"'s\b","",newString)
             newString = re.sub("[^a-zA-Z]", " ", newString)
             tokens = [w for w in newString.split() if not w in stop_words]
             long_words=[]
             for i in tokens:
                 if len(i)>=3:                  #removing short word
                     long_words.append(i)
             return (" ".join(long_words)).strip()

         [nltk_data] Downloading package stopwords to /root/nltk_data...
         [nltk_data]   Package stopwords is already up-to-date!
```

Figure 15 : Code Snippet for "Data Pre-processing - Stop Word Removal"

More code snippets related to the pre-processing of the data is provided in the **Appendix E1.**

7.3.2 Splitting the dataset

The below code snippet is related to the splitting of the pre-processed text data to train and test samples

```
[10]: from sklearn.model_selection import train_test_split
      document,document_test,summary,summary_test=train_test_split(wikiHowCleanedDF['cleaned_text'],wikiHowCleanedDF['cleaned_summary']
```

Figure 16 : Code Snippet for "Train & Test Data Split"

7.3.3 Creating dataset pipeline

Preparation of train dataset with tensor in order to train the transformer-based model.

Creating dataset pipeline

```
In [27]: inputs = tf.cast(inputs, dtype=tf.int32)
         targets = tf.cast(targets, dtype=tf.int32)

In [28]: BUFFER_SIZE = 20000
         BATCH_SIZE = 128

In [29]: dataset = tf.data.Dataset.from_tensor_slices((inputs, targets)).shuffle(BUFFER_SIZE).batch(BATCH_SIZE)

In [30]: dataset
Out[30]: <BatchDataset element_spec=(TensorSpec(shape=(None, 319), dtype=tf.int32, name=None), TensorSpec(shape=(None, 66), dtype=tf.int
         32, name=None))>

In [31]: for (batch, (inp, tar)) in enumerate(dataset):
             print(inp)
             print(tar)
             break
```

Figure 17 : Code Snippet for "Creation of dataset pipeline"

7.3.4 GANs framework

When designing the GANs architecture CyclicGAN approach was utilized to design the system.

7.3.4.1 Generator Component

The Generator Component of GANs framework consists of a transformer-based pointer generator model. All the utility functions code snippets and explanations that are relevant to the transformer such as positional encoding, masking, scaler dot product and multihead attention is provide in the **Appendix E2 .**

Encoder Component

Fundamental Unit of Transformer encoder

```
In [39]: class EncoderLayer(tf.keras.layers.Layer):
             def __init__(self, d_model, num_heads, dff, rate=0.1):
                 super(EncoderLayer, self).__init__()

                 self.mha = MultiHeadAttention(d_model, num_heads)
                 self.ffn = point_wise_feed_forward_network(d_model, dff)

                 self.layernorm1 = tf.keras.layers.LayerNormalization(epsilon=1e-6)
                 self.layernorm2 = tf.keras.layers.LayerNormalization(epsilon=1e-6)

                 self.dropout1 = tf.keras.layers.Dropout(rate)
                 self.dropout2 = tf.keras.layers.Dropout(rate)

             def call(self, x, training, mask):
                 attn_output, _ = self.mha(x, x, x, mask)
                 attn_output = self.dropout1(attn_output, training=training)
                 out1 = self.layernorm1(x + attn_output)

                 ffn_output = self.ffn(out1)
                 ffn_output = self.dropout2(ffn_output, training=training)
                 out2 = self.layernorm2(out1 + ffn_output)

                 return out2
```

Figure 18 : Code Snippet for "Encoder part of the transformer"

Decoder Component

Fundamental Unit of Transformer decoder

In [40]:
```
class DecoderLayer(tf.keras.layers.Layer):
    def __init__(self, d_model, num_heads, dff, rate=0.1):
        super(DecoderLayer, self).__init__()

        self.mha1 = MultiHeadAttention(d_model, num_heads)
        self.mha2 = MultiHeadAttention(d_model, num_heads)

        self.ffn = point_wise_feed_forward_network(d_model, dff)

        self.layernorm1 = tf.keras.layers.LayerNormalization(epsilon=1e-6)
        self.layernorm2 = tf.keras.layers.LayerNormalization(epsilon=1e-6)
        self.layernorm3 = tf.keras.layers.LayerNormalization(epsilon=1e-6)

        self.dropout1 = tf.keras.layers.Dropout(rate)
        self.dropout2 = tf.keras.layers.Dropout(rate)
        self.dropout3 = tf.keras.layers.Dropout(rate)


    def call(self, x, enc_output, training, look_ahead_mask, padding_mask):
        attn1, attn_weights_block1 = self.mha1(x, x, x, look_ahead_mask)
        attn1 = self.dropout1(attn1, training=training)
        out1 = self.layernorm1(attn1 + x)

        attn2, attn_weights_block2 = self.mha2(enc_output, enc_output, out1, padding_mask)
        attn2 = self.dropout2(attn2, training=training)
        out2 = self.layernorm2(attn2 + out1)

        ffn_output = self.ffn(out2)
        ffn_output = self.dropout3(ffn_output, training=training)
        out3 = self.layernorm3(ffn_output + out2)

        return out3, attn_weights_block1, attn_weights_block2
```

Figure 19 : Code Snippet for "Decoder part of the transformer"

Stacked encoder and decoder

Encoder consisting of multiple EncoderLayer(s)

In [41]:
```
class Encoder(tf.keras.layers.Layer):
    def __init__(self, num_layers, d_model, num_heads, dff, input_vocab_size, maximum_position_encoding, rate=0.1):
        super(Encoder, self).__init__()

        self.d_model = d_model
        self.num_layers = num_layers

        self.embedding = tf.keras.layers.Embedding(input_vocab_size, d_model)
        self.pos_encoding = positional_encoding(maximum_position_encoding, self.d_model)

        self.enc_layers = [EncoderLayer(d_model, num_heads, dff, rate) for _ in range(num_layers)]

        self.dropout = tf.keras.layers.Dropout(rate)

    def call(self, x, training, mask):
        seq_len = tf.shape(x)[1]

        x = self.embedding(x)
        x *= tf.math.sqrt(tf.cast(self.d_model, tf.float32))
        x += self.pos_encoding[:, :seq_len, :]

        x = self.dropout(x, training=training)

        for i in range(self.num_layers):
            x = self.enc_layers[i](x, training, mask)

        return x
```

Figure 20 : Code Snippet for "Stacked Encoder"

Decoder consisting of multiple DecoderLayer(s)

In [42]:
```
class Decoder(tf.keras.layers.Layer):
    def __init__(self, num_layers, d_model, num_heads, dff, target_vocab_size, maximum_position_encoding, rate=0.1):
        super(Decoder, self).__init__()

        self.d_model = d_model
        self.num_layers = num_layers

        self.embedding = tf.keras.layers.Embedding(target_vocab_size, d_model)
        self.pos_encoding = positional_encoding(maximum_position_encoding, d_model)

        self.dec_layers = [DecoderLayer(d_model, num_heads, dff, rate) for _ in range(num_layers)]
        self.dropout = tf.keras.layers.Dropout(rate)

    def call(self, x, enc_output, training, look_ahead_mask, padding_mask):
        seq_len = tf.shape(x)[1]
        attention_weights = {}

        x = self.embedding(x)
        x *= tf.math.sqrt(tf.cast(self.d_model, tf.float32))
        x += self.pos_encoding[:, :seq_len, :]

        x = self.dropout(x, training=training)

        for i in range(self.num_layers):
            x, block1, block2 = self.dec_layers[i](x, enc_output, training, look_ahead_mask, padding_mask)

            attention_weights['decoder_layer{}_block1'.format(i+1)] = block1
            attention_weights['decoder_layer{}_block2'.format(i+1)] = block2

        return x, attention_weights
```

Figure 21 : Code Snippet for "Stacked Decoder"

Combine Model

```
class Transformer(tf.keras.Model):
    def __init__(self, num_layers, d_model, num_heads, dff, vocab_size,batch_size, rate=0.1):
        super(Transformer, self).__init__()

        self.num_layers =num_layers
        self.vocab_size = vocab_size
        self.batch_size = batch_size
        self.model_depth = d_model
        self.num_heads = num_heads

        self.embedding = Embedding(vocab_size, d_model)
        self.encoder = Encoder(num_layers, d_model, num_heads, dff, vocab_size, rate)
        self.decoder = Decoder(num_layers, d_model, num_heads, dff, vocab_size, rate)
        self.final_layer = tf.keras.layers.Dense(vocab_size)


    def call(self, inp, extended_inp,max_oov_len, tar, training, enc_padding_mask, look_ahead_mask, dec_padding_mask):

        embed_x = self.embedding(inp)
        embed_dec = self.embedding(tar)

        enc_output = self.encoder(embed_x, training, enc_padding_mask)  # (batch_size, inp_seq_len, d_model)

        # dec_output.shape == (batch_size, tar_seq_len, d_model)
        dec_output, attention_weights, p_gens = self.decoder(embed_dec, enc_output, training, look_ahead_mask, dec_padding_mask)

        output = self.final_layer(dec_output)  # (batch_size, tar_seq_len, target_vocab_size)
        output = tf.nn.softmax(output) # (batch_size, tar_seq_len, vocab_size)
        #output = tf.concat([output, tf.zeros((tf.shape(output)[0], tf.shape(output)[1], max_oov_len))], axis=-1) # (batch_size, targ_seq_len, vocab_size+max_oov_len)

        attn_dists = attention_weights['decoder_layer{}_block2'.format(self.num_layers)] # (batch_size,num_heads, targ_seq_len, inp_seq_len)
        attn_dists = tf.reduce_sum(attn_dists, axis=1)/self.num_heads # (batch_size, targ_seq_len, inp_seq_len)

        final_dists =  _calc_final_dist( extended_inp, tf.unstack(output, axis=1) , tf.unstack(attn_dists, axis=1), tf.unstack(p_gens, axis=1), max_oov_len, self.vocab_size, self.batch_size)
        final_output =tf.stack(final_dists, axis=1)

        return final_output, dec_output, attention_weights
```

Figure 22 : Code Snippet for "Final Combined Transformer Model"

7.3.4.2 Discriminator Component

The Generator Component of GANs framework consists of a CNN based text classification model.

```
def make_discriminator_model():
  model = tf.keras.Sequential()
  model.add(layers.Embedding(decoder_vocab_size,
                    embedding_dim,
                    embeddings_initializer=tf.keras.initializers.Constant(embedding_matrix),
                    trainable=False))
  model.add(layers.Dropout(0.2))
  model.add(layers.Conv1D(filters=250,kernel_size=5,padding='valid',activation='elu'))
  model.add(layers.MaxPooling1D())
  model.add(layers.Conv1D(filters=250,kernel_size=3,padding='valid',activation='elu'))
  model.add(layers.GlobalMaxPooling1D())
  model.add(layers.Dense(units=250, activation='elu'))
  model.add(layers.Dropout(0.2))
  model.add(layers.Dense(1, activation='sigmoid'))

  return model
```

Figure 23 : Code Snippet for "Descriminator : CNN test classification"

7.3.5 Training the model

```
def train(dataset, epochs,report):

  for epoch in range(epochs):
      start = time.time()

      for (batch, (inp, tar)) in enumerate(dataset):
          train_step(inp, tar)

          # 151947 samples
          # we display 3 batch results -- 0th, middle and last one (approx)
          # 151947 / 128 (batch size) ~ 1,187; 1,187 / 2 = 593

          if batch % 593 == 0:
              print ('Epoch {} Batch {} generator Loss {:.4f} discriminator Loss {:.4f}'.format(epoch + 1, batch, generatorLoss.result(),discriminatorLoss.result()))

      if (epoch + 1) % 5 == 0:
          ckpt_save_path = ckpt_manager.save()
          print ('Saving checkpoint for epoch {} at {}'.format(epoch+1, ckpt_save_path))

      print ('Epoch {} generator Loss {:.4f} discriminator Loss {:.4f}'.format(epoch + 1, generatorLoss.result(),discriminatorLoss.result()))

      update_row = {
            'Epoch':epoch + 1 ,
            'Time': time.time() - start,
            'Generator_Loss':generatorLoss.result() ,
            'Discriminator_Loss':discriminatorLoss.result() ,}

      row_to_add = pd.Series(update_row,name= str(epoch))
      report = report.append(row_to_add)

      print ('Time taken for 1 epoch: {} secs\n'.format(time.time() - start))

  return report
```

Figure 24 : Code Snippet for "Training the model"

7.4 Implementation of APIs

Implemented Server API for the generation of the summary is depicted below.

```
@app.post("/summarizer/")
async def summarizer(url_res: Url):

  youtube_video_link = url_res.video_url

  if(check_vedio_duration(youtube_video_link)):

    audio_extraction(youtube_video_link)
    transcribe =generate_transcribe()
    summary = summarize(transcribe)
    item = {'transcribe':transcribe,'summary':summary}
    json_compatible_item_data = jsonable_encoder(item)

    return JSONResponse(content=json_compatible_item_data)

  else:
    error_msg = {'error':"Video duration is more than 15 minute"}
    return JSONResponse(content=jsonable_encoder(error_msg))
```

Figure 25 : Implementation of APIs

7.5 User Interface

The screenshots related UI interface of the end system is provided in the **Appendix F**.

7.6 Chapter Summary

The chapter concentrated on the implementation process of the system with respect to the defined design aspect. The chapter started by illustrating a diagram of the identified and utilized technology stack for each different phase of the development process along with justification for each selected and used technology stack. Then the core functionalities of the implemented prototype were explained with aid of the relevant code snippet along with the implemented APIs and finally, the chapter was terminated with some illustrations of the screenshot of the end prototype UI.

CHAPTER 8 : TESTING

8.1 Chapter Overview

This chapter presents all the details on the testings that were conducted on the final developed system to identify any defects that contain in the implemented functionalities and to evaluate the overall performance in order to verify the expected flow of the end system. Starting from defining the testing objectives the chapter moves on to explain how the model testing, benchmarking, model integration testing, functional testing, and non-functional testing were conducted on the system.

8.2 Objectives and Goals of Testing

The main objectives and goals of a testing process are to identify maximum defects in implemented functionalities of the end system to mitigate those issues early as possible in order to evaluate and create a final minimum viable end product that meets the expected flow. With that in mind, the author of the project followed the below-mentioned testing objectives in order to verify the developed final abstractive text summarization system.

- To identify maximum defects that contain in the implemented functionalities of the end system which were undetected in the implementation process.
- To verify whether all the major functional and non-functional requirements that were initially defined to be contained in the final system (based on the MoSCoW principle) are implemented and working as expected.
- To verify whether the final prototype fulfills the defined scope of the project.
- To verify whether the proposed solution architecture is properly implemented and works
  as expected.
- To verify whether the end summaries generated from the implemented abstractive text summarization system meet the project expectation.
- To verify whether the user interface provides a simple and better user experience.

8.3 Testing Criteria

The automatic abstractive text summarization system was tested by mainly considering the below-mentioned two testing criteria,

1. Functional Testing – Focus on testing the defined main functional requirements that work together in building the main flow of the end system.

2. Structural Testing – Focus on testing the defined non-functional requirements of the system in order to identify the quality of the system.8.4 Model Testing.

8.4 Model Testing

By considering the above model testing for the automatic abstractive text summarization model was done based on two categories as quantitative testing which was done by utilizing the rouge score metric and qualitative testing which was done by utilizing the semantic text similarity score metric.

8.4.1 Quantitative Testing

By investigating the existing abstractive summarization system as mentioned in the literature review section in chapter 2 and in under the evaluation methodologies section in chapter 3, it was observed the most used evaluation metric in the domain of abstractive text summarization is the ROUGE score. So it was used as the quantitative testing metric to evaluate the model.

8.4.1.1 Rouge Score

ROUGE aka Recall-Oriented Understudy for Gisting Evaluation is a popular metric that is mostly used in the domain of automatic abstractive text summarization, which aid in comparing the model-generated summary against the gold summary (human produce summary) to compute how many overlapping words or word sequences of gold summaries are captured in the machine-generated summary. The following formulas are used in computing ROUGE metric.

$$ROUGE(N)\ (Recall) = \frac{Number\ of\ overlapping\ n\ grams\ found\ in\ model\ and\ golden\ truth}{Numbrt\ of\ n\ grams\ found\ in\ the\ golden\ truth}$$

$$ROUGE(N)\ (Precision) = \frac{Number\ of\ n\ grams\ found\ in\ model\ and\ golden\ truth}{Number\ of\ n\ grams\ in\ model}$$

$$ROUGE(N)\ (F1\ Score) = 2 * \frac{ROUGE(N)\ (Precision) * ROUGE(N\ )(Recall)}{ROUGE(N)\ (Precision) + ROUGE(N)(Recall)}$$

The below table contain the ROUGE score results relevant to the F1 Score, Precision, and Recall criteria that were achieved in the developed abstractive text summarization model.

Criteria	ROUGE Score		
	1	2	L
F1 score	14.94	1.79	10.80

Precision	13.73	1.57	9.73
Recall	19.43	2.45	14.41

Table 26 : Rouge scores for the model

In language models**, unigrams** are single words in a corpus, while **bigrams** are two consecutive words in a corpus based on these ROUGE score metric is designed.

ROUGE-1 score utilizes unigrams when comparing model-generated summaries and golden truth summaries. Similarly, the **ROUGE-2** score utilizes bigrams when comparing model-generated and reference summaries, and the **ROUGE-L** score measures the longest common subsequent (LCS) between model-generated summaries and the golden truth summaries.

By observing the above-given ROUGE score results relevant to the developed model it can be concluded that the final abstractive text summarization model is able to generate more unigrams than bigrams and also it is able to generate an average amount of long subsequent.

The model is able to grab 14.41% of long common subsequent information extractively from the reference summary with 9.73 precision.

8.4.2 Qualitative Testing

The qualitative testing of the model was done by utilizing a metric named semantic text similarity score.

8.4.2.1 Semantic Text Similarity Score

Semantic Text Similarity Score is an NLP metric that is used to qualitatively evaluate the semantic similarity between two text corpora by producing a quantitative score between 0-5 which indicate the semantic correlation between the two corpora. A score of 0 indicates a very poor semantically correlated two text corpora while a score of 5 represents a high semantically correlated two text corpora. This metric is commonly used in paraphrase detection, semantic search, and question & answering fields. But this can be utilized in the field of abstractive text summarization in order to evaluate the semantic similarity between the machine generate summary and the golden truth summary.

Metric	Score
Semantic Text Similarity Score	1.87

Table 27 : Semantic Text Similarity Score

By observing the semantic text similarity score that received for the end model, which is 1.87, it can be concluded that the model able to produce a summary nearly 40% semantically similar to a human-generated abstractive summary.

To prove the above fact related to the semantic similarity score of the generated summaries of the model it is provided several examples of model generated summary and its corresponding golden truth summary (original summary).

	Golden Truth Summary	Model Generated Summary
1	purchase dog shampoo buy hair clippers get nail trimming tools purchase dog brush ardor comb prepare area praise dog brush dog bathe dog cut dog's nails discuss plan groomer take protective measures prevent dental issues	Select dog needs purchase high quality dog supplies make sure dog getting right dental groom dog check flea comb choose flea trim beard consider buying flea comb use flea care mats consult professional growth treatments hair snip ends check weekly gather planting area consider soaking
2	jailbreak iOS device install file Lydia jailbroken device download install mod loader iOS search mods Lydia install mod start Minecraft PE installing mod tap mod loader button Minecraft PE main menu tap mod list enable adjust play mod	Download install mods ios device download mod loader program open mods Lydia open mods Lydia tap download button upper right corner tap download button begin jailbreak file iOS device tap install rinse Minecraft tap download mod loader button upper right corner tap download button upper right corner Minecraft

Table 28 : Golden Truth vs Model Generated Summaries

8.5 Benchmarking

Wikihow is an abstractive text summarization benchmark dataset designed with 230,000 articles and summary pairs that were created by human authors by extracting and constructing summaries from an online knowledge base. It covers a different span of tutorial video articles so that makes a good and close match for this research project.

Abstractive Text Summarization Models	ROUGE Score		
	1	2	L
Seq-to-seq with attention (Chopra et al., 2016; Nallapati et al., 2016)	22.04	6.27	20.87
Pointer-generator (See et al., 2017)	27.30	9.10	25.65
Pointer-generator + coverage (See et al., 2017)	28.53	9.23	26.54
Research Model (GANs-based transformer + pointer generator without the coverage mechanism)	**14.94**	**1.79**	**10.80**

Table 29 : Benchmarking results of the system

The above table contains the F1 score of the ROUGE metric evaluated on several abstractive text summarization models which were tested on the mentioned WikiHoW benchmark dataset along with the F1score results of this research project for benchmarking.

When observing the above table results the proposed model has low results compared to existing abstractive models tested on the training dataset in comparison to ROUGE Score. However, the ROUGE scores don't provide an objective measurement of the abstractivness of the model but rather discuss the extractive ability of a model. GAN-based approach with pointer generator transformer model has more abstractiveness than the extractive ability. However, the author was able to produce a good model that was able to produce an average summary with limited computation resources, and limited deep learning architectural design knowledge within a short amount of period. So finally, it can be concluded that the end results of the model meet most of the research project expectations with the relevant scope.

8.6 Functional Testing

Functional testing of the system was conducted against the defined functional requirements in chapter 4. This was done as a black box testing process to identify if the implemented functionalities meet the expected requirements.

Test case	FR ID	Functional Description	Expected Result	Actual Result	Status (Passed/Failed)
1	FR 1	Entering a YouTube video link into the system	The video should be uploaded	The video gets uploaded	Passed

2	FR 2	Enters a URL of a YouTube video that is more than 15 min long	Displaying an error message	Displaying an error message	Passed
3	FR 3	Enters an invalid YouTube video URL	Displaying an error message	Displaying an error message	Passed
4	FR 4	Creates and integrates the audio extraction module into the system	The audio clip of the provided video is extracted	The audio clip of the provided video is extracted	Passed
5	FR 5	Creates and integrates the video transcript generation module into the system	Creation of a textual transcript from the extracted audio clip	Creation of a textual transcript from the extracted audio clip	Passed
6	FR 6	Pre-processing of the generated transcript	The transcript should be pre-processed	Transcript is pre-processed	Passed
7	FR 7	Dataset preparation for the model	Data should be split into train and test samples according to providing configuration	The dataset is split to train and test samples according to the provided configuration	Passed
8	FR 8	Designing and creation of transformer model	The base model should be the transformer model	The base model is a transformer model	Passed
9	FR 9	Integration pointer generator to the base model	The transformer model should have point	The transformer model has the	Passed

			generator functionality	point generator functionality	
10	FR 10	Integration of transformer + pointer generated with GANs to training	The model should be trained with GANs architecture	The model is trained with the GANs architecture	Passed
11	FR 11	Use of evaluation metrics to evaluate the model	The model should be evaluated with the appropriate evaluation metrics	The model is evaluated with the appropriate evaluation metrics	Passed
12	FR 12	Benchmarking the final model	The model should be benchmarked with the existing models of the relevant dataset utilized.	The model is benchmarked with the existing models of the relevant dataset utilized.	Passed
13	FR 13	The model should design to generate abstractive summaries	Should generate abstractive summaries	Model is able to generate abstractive summaries	Passed

Table 30 : Functional Testings

8.7 Module and Integration Testing

The solution architecture of the developed model follows a modular architectural pattern. So module and integration testing was performed to check whether the relevant modules of the overall system are properly integrated and work as expected according to specific functionality relevance to each module. The table related to the performed integration testing is provided in the **Appendix G**.

8.8 Non-Functional Testing

In accordance with the defined non-functional requirement in chapter 4, two non-functional requirements stood out as the most important requirements that should be contained in the system, which are usability and quality. So out of all the non-functional requirements, it was performed non-functional testing on these two selected important requirements.

8.8.1 Usability

To measure the usability of the GUI of the system it was conducted a usability testing in which it was evaluated the simplicity and user-friendliness of the UI for the user(s) by gathering feedbacks from several user(s). The received feedback concludes a positive impression on the user interface while some suggestions were provided for more improvement for user experience such as an example, it was mentioned by one evaluator to improve the response time of generating the summary by the system.

8.8.2 Quality

The author conducts a quantitative and qualitative evaluation of the end summary result in order to measure the quality of the results produced by the system. In accordance to the quantitative testing conducted in the prior model testing section, it was observed that the system produces human understandable abstract summaries to a considerable extent. Similarly, with respect to the qualitative analysis of the result, it was suggested by most of the evaluators that the summary is fairly understandable and could be improved semantically and grammatically. So it can be concluded that the system produces a fairly considerable summary as expected for a minimum viable product.

8.9 Limitation of the testing process

In the field of automatic abstractive text summarization field, there is a huge scarcity of evaluation metrics that could provide a good analysis on the machine-generated summaries. There are no evaluation metrics that could provide a proper measure of factual consistency, syntactical and semantical measure of a machine-generated abstract summary. Even the only and mostly used evaluation metric which is ROUGE score metric in the field of abstractive text summarization utilizes n-gram overlapping measures with respect to golden truth summaries which could only conclude the abstractiveness of the summary. Further human evaluation that could be done for testing and evaluating the machine-generated summaries could contain subjective opinions which is also not a more generalizable measure. Also, it was observed that

the chosen WikiHow dataset for the project also contains noises such as grammatical mistakes that could affect the performance of the model evaluation.

8.10 Chapter Summary

This chapter concluded all the testing approaches performed on the developed system to analyze any defects along with the performance of the system. The model testing was described by breaking into two sub-topics as quantitative testing and qualitative testing. After the model testing, it was included the benchmarking results of the final system and moved on to presenting results on functional, non-functional requirement testing, and module integration testing. Finally, the testing limitation of the research project was provided to present the blockers which limit the research testing.

CHAPTER 9 : EVALUATION

9.1 Chapter Overview

This chapter provides an exhaustive explanation of the final evaluation conducted to evaluate the overall research project. This included the feedbacks taken from various experts on different aspects of the conducted research and the end prototype along with the self-evaluation of the research author. Latterly the evaluation of the functional and non-functional requirements and the limitation of the evaluation process are presented.

9.2 Evaluation Methodology and Approaches

The evaluation process of the research was conducted by utilizing only a **qualitative approach** due to research was mainly focused on the abstractive text summarization domain as it produces abstractive summaries that can only be evaluated more precisely in a qualitative manner. The qualitative approach of the evaluation was conducted by utilizing surveys and interviews and sending a document that contains a comprehensive explanation of the research project and a demo video of the prototype.

9.3 Evaluation Criteria

Criteria	Evaluation Purpose
Selection of the project concept, its identified research gaps, and its novelty	To identify whether the research addresses a solid research gap with a good novel approach within the given time.
The scope and the depth of the research undertaken	To evaluate to identify whether this research has enough scope and depth for a single undergraduate to complete the given time period.
Proposed abstractive text summarization system architecture	To evaluate the research project correctly utilizing the state-of-the-art approaches that currently trending in the NLP domain.
End prototype UI design and its usability	To evaluate the user experience and the user opinion about the developed prototype.
System results	To evaluate the quality of the abstractive summaries, generate from the system
Overall research contribution	To evaluate whether this research has done a significant amount of workload and whether this research was able to

	provide a novel approach to abstractive summarize lecture content

Table 31 : Evaluation Criteria

9.4 Self-Evaluation

Criterial	Author's Self Evaluation
Selection of the project concept, its identified research gaps, and its novelty	This research was trying to address an issue faced by many students when trying to refer to a summary of a lecture video as they have to go through the whole video in capturing the main information, so the concept of the research is timely and in need of the students. As per the research gap, there was no research conducted on the generation of abstractive summaries by utilizing GANs + pointer generator-based transformers and the author was able to address this research gap with a solid novel approach by utilizing state-of-the-art approaches.
The scope and the depth of the research undertaken	The scope of this research project was mainly focused to design an abstractive text summarization model by utilizing state-of-the-art approaches like GANs and pointer generator-based transformers and designing the other components of the system with the help of 3rd party open source tools and frameworks and delivering a minimum viable product within 9 to 10 months period and when comparing to other state-of-the-art systems develop by Ph.D. experts this research project scope and dept is very solid.
Proposed abstractive text summarization system architecture	By using GANs architecture the model can produce more abstractive summaries and by utilizing a transformer-based model Autor tries to capture long text sequences when computing abstractive text summaries and by utilizing a pointer generator network author tries to generate summaries with more accurate information however due to model complexity increases by combining server complicated and state of the art approaches it generally requires a highly complicated dataset to train the model to get a good result.

End prototype UI design and its usability	The author has designed the final prototype in a minimalistic and simplistic way so the prototype could easily be used by any nontechnical user without any prior knowledge. The author also uses frontend validation and displays error messages in a cool manner. But there is some delay in API calls due to 3rd party library that use to generate audio-to-text transcribing taking some time to process.
System results	The abstractive summaries that generate from the system have some grammatical and semantical errors however the model provides a readable summary that can understand by a human user. The reason for this broken English summary is the model can produce more abstractive sentence species, but it could not able to fully grab the semantics and grammar of a sentence and this issue is commonly present in many state-of-the-art systems too.
Overall research contribution	The author has identified a solid research gap that exists within the abstractive text summarization domain and tried to address this research gap with a novel state-of-the-art approach with many limitations and research challenges that present when conducting this research. Finally, the author has been able to deliver a satisfactory system that meets almost all the research objectives with an excellent contribution.

Table 32 : Self-Evaluation

9.5 Selection of Evaluators

Evaluators for this specific research were selected based on three categories in accordance with their level of expertise. They are,

1. Domain experts who have an expert understanding of AI, NLP, and deep learning domains.
2. Technical experts who have research experience in the NLP domain.
3. Student researcher (also can be considered as the end user(s)) who has a keen interest in the research domain and also have hand on experience in machine learning-related projects.

ID	Position	Research Experience
Category 1 : Domain Experts		
EV1	Associate Tech Lead (NLP / Blockchain)	Has experience in designing chatbots and experience in designing Natural language understanding models
EV2	Associate AI/ML Engineer	Has a hand on experience in NLP and computer vision models design
EV3	AI/ML Engineer	Has research experience in both NLP and computer vision domain
Category 2: Technical Experts		
EV4	Software Engineer	Has research experience in the NLP domain
EV5	Software Engineer	Has research experience in the NLP domain
Category 3: Student Researcher		
EV6	Software Engineer and Currently following an MSc degree in computer science	Has search experience in deep learning algorithms

Table 33 : Selected Evaluators

9.6 Evaluation Results

Evaluation results gathered from the above-selected evaluators in accordance with the defined evaluation criteria of the research project are presented below.

9.6.1 Expert Opinions

Criteria 1: Selection of the project concept, its identified research gaps, and its novelty

Question: What is your opinion on this research project concept, its identified research gaps and its novelty?	
Evaluator	**Feedback**
EV1	The concept is really interesting, and it has a potential business value.
EV2	Research illustrates the application of GANs to summarize and improve the coherence and readability of those summarize, by employing a transformer-based approach as the generator, which is a novel approach and a very exciting concept.

EV3	The researcher has identified a good and timely relevant research concept that is important to more students who face trouble and who are lazy to refer to whole lecture videos. When considering the research gap researcher has conducted a good literature review and identified a good research gap also the researcher is also able to address that research gap with a novel concept by utilizing GANs with transformer-based pointer Generator networks.
EV4	In the era of distant learning, summarizing lecture content is a timely problem. The research student has accurately identified the research gaps that exist within the abstract text summarization approach and addressed the gap using state-of-the-art Generative Adversarial Networks (GAN) combined with pointer-generated transformer architecture.
EV5	This research has focused on a much-needed requirement for students. We have seen a great increase in online education in the recent past and hence the researcher has identified a problem relevant to the current context. Selecting abstractive text summarization for the research is a good choice since the domain of the project is educational. Hence, abstractive text summarization can give more meaningful summaries. Using a transformer-based approach is another good thing because the use of transformers have shown improved results in recent research.
EV6	This is a good idea and I think this will help numerous students to continue their studies through online learning.

Table 34 : Criteria 1 - Experts Opinion

Criteria 2: The scope and the depth of the research undertaken

Question: What is your opinion on the scope and the depth of the research undertaken?	
Evaluator	**Feedback**
EV1	The research project has a satisfactory scope and depth.
EV2	From the scope point, it is a good contribution. This level of implementation requires a solid understanding of deep learning concepts and implementations.
EV3	As this research project is conducted by a single researcher it is clear that the chosen research scope is a little bit overwhelming for a single person. But because the researcher was able to identify all the technical and domain requirements to design the scope and depth of research seem to be well defined and very satisfiable.

EV4	The researcher has conducted in-depth research into the research domain and delivered an outstanding solution utilizing available state-of-the-art technologies.
EV5	Considering the domain, the scope chosen seems sufficient. Since most of the lecture videos will cover a specific topic/subject, it will be sufficient to take a single document as input. Further restricting the scope to monolingual is fine since this will reduce the complexity and will let the research focus on the actual purpose of the research which is to output an informative summary of the lecture. It would be great if the query-based summarization feature was addressed since that would let students get summaries on specific areas of the lecture.
EV6	individual project with a fair amount of depth and a good amount of scope.

Table 35 : Criteria 2 - Experts Opinion

Criteria 3: Proposed abstractive text summarization system architecture

Question: What is your opinion on the proposed abstractive text summarization system architecture?	
Evaluator	**Feedback**
EV1	You have used a novel architecture for this kind of limitation to achieve more effective summaries.
EV2	GAN adaptively synthesizes training instances for querying to improve learning and it's good that you have used the Pointer-generator approach because it allows generating text even if the text does not exist in the context.
EV3	The proposed abstractive text summarization system architecture utilizes approaches like GANs, Transformers, and pointer generator networks so it is clear the proposed architecture is a great state-of-the-art architecture.
EV4	The ROGUE score achieved for the summarization model is significant with regard to the time and resources that were available to the researcher.
EV5	The proposed architecture seems to be good and has focused on the necessary techniques that would capture the context of the words which would result in better and more meaningful summaries. The transformer-based approach which uses an attention mechanism is a good choice since it will help in weighing the most significant words of the input data. This would in turn help in more relevant and informative summaries.

EV6	Architecture is seem well defined through proper requirement analysis.

Table 36 : Criteria 3 - Experts Opinion

Criteria 4: End prototype UI design and its usability

Question: What is your opinion about the end prototype UI design and its usability?	
Evaluator	**Feedback**
EV1	Overall UI design is ok because this is just a prototype. But it would be better if you can improve the response time.
EV2	Overall high-level architecture is straightforward. GUI prototype is relatively simple and easy to use which means it provides a great user experience.
EV3	UI seems to be minimalistic and pleasant and even an average user is also able to easily adapt to using the system. But It seems that the response time of summary generation is take a little bit of time due to the audio-to-text framework that is used. It could be improved by commercial license API . Anyway the prototype is excellent in its usability considering the fact it is a minimum viable product and a prototype.
EV4	Prototype UI design is minimalistic and extremely user-friendly even to an non-tech audience.
EV5	The UI looks good. It is convenient and easy to use. But since the processing time is a bit longer, it would be better if there is an indicator or progress bar that shows the current progress. That way would be more user-friendly rather than having a simple loading animation.
EV6	The prototype serves the purpose of the research project.

Table 37 : Criteria 4 - Experts Opinion

Criteria 5: System results

Question: What is your opinion about the system results (Evaluations Results and Generated Final Summary examples)?	
Evaluator	**Feedback**
EV1	Fairly ok. but the grammar of the result should be improved.
EV2	Seems like this tool is capable enough improve to producing content of Google-level quality.
EV3	Final generated summaries seem to be facing a deficiency in grammar and semantics. But it is acceptable as the researcher to try to address the research

	gap of an abstractive text summarization domain as currently available all the state-of-the-art models related to that domain too suffer from the same issue and the evaluation results are satisfiable considering all the research limitations such as domain knowledge limitations and hardware limitations and time limitations that have to face by the researcher.
EV4	Evaluation results are satisfactory when compared to other international researchers in the same domain. However, given the limited resources and time frame the researcher had, the achieved results are quite good.
EV5	The output summary gives a quite reasonable summary of the document. It gives a general idea of the document and has considerably addressed the identified problem. Additionally, since this is an abstractive summarization approach and there is not much-related research, the results achieved in this research can be considered an improvement in the abstractive summarization research domain.
EV6	Feel like can be improved. (Reason for me to say this is, that I felt that the generated summary contains broken English. The English can be improved semantically)

Table 38 : Criteria 5 - Experts Opinion

Criteria 6: Overall research contribution

Question: What is your opinion about the overall research contribution?	
Evaluator	**Feedback**
EV1	I agree that you have used state-of-the-art technologies and it has a steep learning curve. So, I am highly satisfied with your research contribution.
EV2	A good amount of contribution and this project exceeded my expectations.
EV3	The researcher has done a great contribution to the research community by testing a novel hypothesis using GANs + transformer base pointer generator model that can produce a considerably acceptable abstractive summary with all the existing limitations.
EV4	The researcher has conducted thorough research and delivered an excellent solution using state-of-the-art approaches available. I would say this research has contributed immensely to the specific domain that is addressed.
EV5	The researcher has identified a demanding and much-needed research problem that has a lot of practical use in the real world. The researcher has also selected

	the scope in such a way that the intent of the research, which is to summarize the lecture documents, can be focused more. In my opinion, selecting abstractive text summarization is a good choice, and also the architecture proposed in the research has made a good impact on the final results of the research. UI is convenient to use but could have focused a bit more on the user experience.
EV6	Fair amount of contribution for an individual project.

Table 39 : Criteria 6 - Experts Opinion

9.7Limitations of Evaluation

There was a limited amount of domain experts that the author was able to interview due to the prevailing economic crisis in Sri Lanka. The evaluation mostly consisted of mixed feedbacks as abstractive text summaries were evaluated more subjectively.

9.8Evaluation of Functional Requirements

The evaluation of the implemented functional requirement completion rate is 100% and its requirement analysis is provided in **Appendix H.**

9.9 Evaluation of Non-Functional Requirements

The evaluation of the implemented Non-functional requirement completion rate is 83.33% and its requirement analysis is provided in **Appendix I**.

9.10 Chapter Summary

The chapter covers all the evaluation feedback received from various experts along with the author's own self-evaluation of the developed system to evaluate the final quality of the end system.

CHAPTER 10 : CONCLUSION

10.1 Chapter Overview

This chapter presents the final conclusion of the overall work carried out in the entire research project. The chapter sums up by providing details on the achievements gained by the author on the initially defined aims and objective of the research project, the benefit of utilization of the existing knowledge gained from the course, use of existing skills, and newly learned skills which aid in the achievement of learning outcome of the conducted research project. Finally, the chapter presents the problems and challenges faced in the conducted research project along with the research limitations and possible future enhancements that can be done to existing work to encourage and pave the path for future researchers to expand the work done on abstractive text summarization in the creation of quality summaries.

10.2 Achievements of Research Aims & Objective

This research project aim is to investigate, design, develop and evaluate an end-to-end working lecture summarizer that will generate a considerable level of semantically and syntactically meaningful concise summary that contains the salient information related to lecture video in a concise manner that would benefit the end-user as well as, contribute a novel improved GAN based architecture for abstractive text summarization domain.

The defined aim of the research project was able to successfully achieved by designing, developing, and evaluating the end-to-end working summarizer that utilizes the developed novel GANs-based abstractive text summarization approach in lecture video that generates a considerable level of semantically and syntactically meaningful concise abstract summaries.

10.3 Utilization of Knowledge from the Course

Module	Description
Programming Principles 01 and 02, Object Oriented Programming	These modules provided the much need knowledge on coding experience with python language, with its various APIs and programming best practices that benefited the author in the development process of the research prototype.
Web Design and Development, Client Server Architecture,	These modules provided the fundamental understanding of a client and server

Server-side Web Development	architecture and the process of development of a proper web application using existing various technologies and frameworks which aid the author in designing the system architecture of the end application along with the creation of the frontend and the backend components.
Software Development Group Project	The learning gained through this module was the base to this main research project as the learnings and experiences that were gained from it aid the author in process of finding a proper research solution to the existing problem by carrying out an extensive literature review until the creation of end prototype.
Algorithms: Theory Design and Implementation	This module provided a proper knowledge in understanding complicated algorithms that contain data structure and other logic that could utilize for solving problems in a logical manner which was aided to the conducted research project.

Table 40: Utilization of knowledge from the Course

10.4 Use of Existing Skills

- The knowledge gathered by following online courses on MOOC platform played a big role in learning and sharpening the knowledge of the new and existing technologies that benefit to successful completion of the research.
- The experience of that gain in industrial placement such as learned new technologies (specifically react which utilize in the creation of the frontend of the research), creation of web APIs, features developments, identification of bugs, use of industrial programming best practices, and documentation of work aided the whole development process of this research project.

10.5 Use of New Skills

The following skills were gained by conducting this research project,

Deep Learning and NLP – In the initial phase of the research project author did not had much knowledge of deep learning and natural language processing domain under which the project falls. So the author utilizes the knowledge from online sources and lecture content conducted at the university to learn, gain and improve these skills.

Automatic Text Summarization models – The language models such as Seq2Seq with attention, GANs, and transformers utilized in the automatic text summarization domain were completely new for the author which was necessary to learn in order to understand literature reviews and to continue work on the project. So by following external artifacts the project author was able to gradually build a good understanding of the automatic text summarization models.

10.6 Achievements of Learning Outcomes

Descriptions on learning	**Learning Outcomes**
The first and most significant achievement that the author gained by engaging in this research project is learned about how to conduct a proper research in order to identify a critical problem in a selected existing domain and to come up with a proper solution by investigating any gaps or limitations by critically evaluating the existing works that published in the respective problem domain	LO1, LO2, LO4,
All the phrase of the research that needed to be done in the project was planned priorly and allocated a considerably proper timeline to complete each phase of work and the necessary requirements that need to be gathered were conducted at the beginning of the project	LO2, LO3
The SLEP issue and its mitigation process relevant to the project were identified along when building the solid research concept.	LO6
The automatic text summarization domain along with the deep NLP models utilized in this domain which is relevant to the project was completely new for the author which led the author in learning all these necessary skills along with the research.	LO1, LO5
After finalizing all research components, the project prototype was developed.	LO7
All the necessary research activities that were conducted and its results were properly documented with relevant explanations.	LO8

Table 41 : Achievement of Learning Outcomes

10.7 Problems and Challenges Faced

Problem / Challenge	Explanation
Hardware limitations	As this research utilizes GANs architecture with transformer + pointer generator network it had to be trained with high-performing computers. The author had to use a supercomputer with 16 GB RTX 3080 VGA to train the model. But when training author had to limit the max batch size to 16 batches due to training occupying nearly 90% of GPU memory. so the author could not able to test training with different batch sizes due to existing hardware limitations.
Long Duration of training the model	To train a 100 epoch of the model it took 24 hours so in that period that laptop has to feed continuous power and uninterrupted training time to train the model properly and if unexpected power cuts occur have to checkpoint the weights to save the training results properly.
Steep Learning Curve	This research project had a steep learning curve due to its utilization of a state-of-the-art approach to designing the model. The Author had to gain knowledge of NLP techniques, GANs, transformers, and pointer generator networks to design and implement the abstractive summarization model. Most of the existing abstractive summarization models were designed and implemented by Ph.D. researchers.

Table 42: Problems and Challenges Faced

10.8 Deviations

In the requirement gathering phase, the author planned to utilize a dataset that mainly focused on the lecture domain, but it was not used due to the dataset was prepared for extractive summarization purposes which are not suitable for designing an abstractive text summarization model.

10.9 Limitation of the Research

- Although WikiHow Dataset was a benchmark dataset that was used for designing abstractive text summarization models it had some errors in reference summaries that were not able to preprocess due to hardware and time limitation.

- For higher duration input video the system would take a long time period for the generation of summaries and the generated summaries for long duration video would be sub-par.
- The author could not utilize beam search to improve the generation of summaries due to the presence of hardware limitations.
- The author could not utilize reinforcement training techniques to improve the performance of the model due to the presence of hardware limitations.
- The abstractive summarization model could not be able to fine-tune due to the complexity of the model when conducting hyperparameter tuning.
- When compared to other experts who publish more research in the abstractive summarization domain the author had limited domain knowledge as most of these experts are PhD. holders.

10.10 Future Enhancement

- The Xsum abstractive text summarization dataset can be utilized for training the model to generate summaries.
- Can adopt a Transfer learning approach to train more data to fit model complexity with data complexity.
- Can improve the training process by utilizing reinforcement techniques.
- The transformer pointer generator model can improve by the use of a coverage mechanism.
- Different GANs Architectures can be tested to find the most suitable architecture for the abstractive text summarization process.
- Can use a different text classification model as a discriminator to improve the GANs Architecture.

10.11 Achievement of the contribution to the body of knowledge

This research project tries to address a common and timely issue that many students are facing when referring to video lectures and tutorials as there is no proper summary of the video content, so the author tries to solve this problem by proposing a system that utilizes an abstractive text summarization model to generate a summary from the transcript of the video lectures or tutorials. The author also identified a solid research gap that exists within the current literature of the abstractive text summarization domain and address that gap by introducing a novel GANs-based architecture that uses a transformer-based pointer generator network as the generator component and CNN model architecture as the discriminator model this proposed architectural solution is

a great contribution toward abstractive text summarization research domain. The author also develops a web application that deploys the developed model so the system can access by many students for educational purposes. This research also contributes to the abstractive text summarization domain by introducing the ability to generate abstractive summaries by combining several state-of-the-art concepts.

10.12 Concluding Remarks

Conducting this research was very challenging for an undergraduate like me as the domain of abstractive text summarization is more complex and it requires more domain-related knowledge in NLP techniques and algorithms also when addressing the research gap, I had to learn the concepts of GANs, concepts of Transformers and the concept of pointer generator networks. This research was conducted based on a solid literature review and executed with a clear plan. This project also received a good amount of appreciation and positive feedback from experts, and I am truly satisfied with the outcome of this research and my contribution to making it real.

REFERENCES

Andra, M.B., Usagawa, T., 2019. Automatic Lecture Video Content Summarizationwith Attention-based Recurrent Neural Network, in: 2019 International Conference of Artificial Intelligence and Information Technology (ICAIIT). IEEE, Yogyakarta, Indonesia, pp. 54–59. https://doi.org/10.1109/ICAIIT.2019.8834514

Bharti, S.K., Babu, K.S., Jena, S.K., 2017. Automatic Keyword Extraction for Text Summarization: A Survey 12.

Chopra, S., Auli, M., Rush, A.M., 2016. Abstractive Sentence Summarization with Attentive Recurrent Neural Networks, in: Proceedings of the 2016 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies. Association for Computational Linguistics, San Diego, California, pp. 93–98. https://doi.org/10.18653/v1/N16-1012

Deaton, J., Jacobs, A., Kenealy, K., 2019. Transformers and Pointer-Generator Networks for Abstractive Summarization 9.

Dernoncourt, F., Ghassemi, M., Chang, W., n.d. A Repository of Corpora for Summarization 7.

El-Kassas, W.S., Salama, C.R., Rafea, A.A., Mohamed, H.K., 2021. Automatic text summarization: A comprehensive survey. Expert Syst. Appl. 165, 113679. https://doi.org/10.1016/j.eswa.2020.113679

Gambhir, M., Gupta, V., 2017. Recent automatic text summarization techniques: a survey. Artif. Intell. Rev. 47, 1–66. https://doi.org/10.1007/s10462-016-9475-9

Joshi, A., Fidalgo, E., Alegre, E., de León, U., 2017. Deep Learning based Text Summarization: Approaches, Databases and Evaluation Measures 4.

Kieuvongngam, V., Tan, B., Niu, Y., 2020. Automatic Text Summarization of COVID-19 Medical Research Articles using BERT and GPT-2. ArXiv200601997 Cs.

Koupaee, M., Wang, W.Y., 2018. WikiHow: A Large Scale Text Summarization Dataset. ArXiv181009305 Cs.

Kryściński, W., McCann, B., Xiong, C., Socher, R., 2019. Evaluating the Factual Consistency of Abstractive Text Summarization. ArXiv191012840 Cs.

Liu, L., Lu, Y., Yang, M., Qu, Q., Zhu, J., Li, H., 2017. Generative Adversarial Network for Abstractive Text Summarization. ArXiv171109357 Cs.

Lv, T., Cui, L., Vasilijevic, M., Wei, F., 2021. VT-SSum: A Benchmark Dataset for Video Transcript Segmentation and Summarization. ArXiv210605606 Cs.

Miller, D., 2019. Leveraging BERT for Extractive Text Summarization on Lectures 7.

Modi, S., Oza, R., 2018. Review on Abstractive Text Summarization Techniques (ATST) for single and multi documents, in: 2018 International Conference on Computing, Power and Communication Technologies (GUCON). IEEE, Greater Noida, Uttar Pradesh, India, pp. 1173–1176. https://doi.org/10.1109/GUCON.2018.8674894

Mohammad Masum, A.K., Abujar, S., Islam Talukder, M.A., Azad Rabby, A.K.M.S., Hossain, S.A., 2019. Abstractive method of text summarization with sequence to sequence RNNs, in: 2019 10th International Conference on Computing, Communication and Networking Technologies (ICCCNT). IEEE, Kanpur, India, pp. 1–5. https://doi.org/10.1109/ICCCNT45670.2019.8944620

Nadkarni, P.M., Ohno-Machado, L., Chapman, W.W., 2011. Natural language processing: an introduction. J. Am. Med. Inform. Assoc. 18, 544–551. https://doi.org/10.1136/amiajnl-2011-000464

Nallapati, R., Zhou, B., santos, C.N. dos, Gulcehre, C., Xiang, B., 2016. Abstractive Text Summarization Using Sequence-to-Sequence RNNs and Beyond. ArXiv160206023 Cs.

Narayan, S., Cohen, S.B., Lapata, M., 2018. Don’t Give Me the Details, Just the Summary! Topic-Aware Convolutional Neural Networks for Extreme Summarization. ArXiv180808745 Cs.

Rekabdar, B., Mousas, C., Gupta, B., 2019. Generative Adversarial Network with Policy Gradient for Text Summarization, in: 2019 IEEE 13th International Conference on Semantic Computing (ICSC). IEEE, Newport Beach, CA, USA, pp. 204–207. https://doi.org/10.1109/ICOSC.2019.8665583

Scialom, T., Dray, P.-A., Lamprier, S., Piwowarski, B., Staiano, J., 2020. Discriminative Adversarial Search for Abstractive Summarization. ArXiv200210375 Cs.

See, A., Liu, P.J., Manning, C.D., 2017. Get To The Point: Summarization with Pointer-Generator Networks. ArXiv170404368 Cs.

Sutskever, I., Vinyals, O., Le, Q.V., n.d. Sequence to Sequence Learning with Neural Networks 9.

Syed, A.A., Gaol, F.L., Matsuo, T., 2021. A Survey of the State-of-the-Art Models in Neural Abstractive Text Summarization. IEEE Access 9, 13248–13265. https://doi.org/10.1109/ACCESS.2021.3052783

Urala Kota, B., Davila, K., Stone, A., Setlur, S., Govindaraju, V., 2018. Automated Detection of Handwritten Whiteboard Content in Lecture Videos for Summarization, in: 2018 16th International Conference on Frontiers in Handwriting Recognition (ICFHR). IEEE, Niagara Falls, NY, USA, pp. 19–24. https://doi.org/10.1109/ICFHR-2018.2018.00013

Vali, H., Rao, P.S., Smarth, S., Nanepag, S., Gurral, N., Nair, Dr.P., 2021. Automatic Lecture Summarizer. Int. J. Adv. Res. Sci. Commun. Technol. 185–189. https://doi.org/10.48175/IJARSCT-V4-I3-030

Widyassari, A.P., Affandy, A., Noersasongko, E., Fanani, A.Z., Syukur, A., Basuki, R.S., 2019. Literature Review of Automatic Text Summarization: Research Trend, Dataset and Method, in: 2019 International Conference on Information and Communications Technology (ICOIACT). IEEE, Yogyakarta, Indonesia, pp. 491–496. https://doi.org/10.1109/ICOIACT46704.2019.8938454

Widyassari, A.P., Rustad, S., Shidik, G.F., Noersasongko, E., Syukur, A., Affandy, A., Setiadi, D.R.I.M., 2020. Review of automatic text summarization techniques & methods. J. King Saud Univ. - Comput. Inf. Sci. S1319157820303712. https://doi.org/10.1016/j.jksuci.2020.05.006

Xu, C., Wang, R., Lin, S., Luo, X., Zhao, B., Shao, L., Hu, M., 2019. Lecture2Note: Automatic Generation of Lecture Notes from Slide-Based Educational Videos, in: 2019 IEEE International Conference on Multimedia and Expo (ICME). IEEE, Shanghai, China, pp. 898–903. https://doi.org/10.1109/ICME.2019.00159

Xu, H., Cao, Y., Jia, R., Liu, Y., Tan, J., 2018. Sequence Generative Adversarial Network for Long Text Summarization, in: 2018 IEEE 30th International Conference on Tools with Artificial Intelligence (ICTAI). IEEE, Volos, Greece, pp. 242–248. https://doi.org/10.1109/ICTAI.2018.00045

Xu, T., Zhang, C., 2021. Reinforced Generative Adversarial Network for Abstractive Text Summarization. ArXiv210515176 Cs.

Zhang, M., Zhou, G., Yu, W., Liu, W., 2020. A Survey of Automatic Text Summarization Technology Based on Deep Learning, in: 2020 International Conference on Artificial Intelligence and Computer Engineering (ICAICE). IEEE, Beijing, China, pp. 211–217. https://doi.org/10.1109/ICAICE51518.2020.00047

Zhuang, H., Zhang, W., 2019. Generating Semantically Similar and Human-Readable Summaries With Generative Adversarial Networks. IEEE Access 7, 169426–169433. https://doi.org/10.1109/ACCESS.2019.2955087

APPENDICES

Appendix A – Gantt Chart

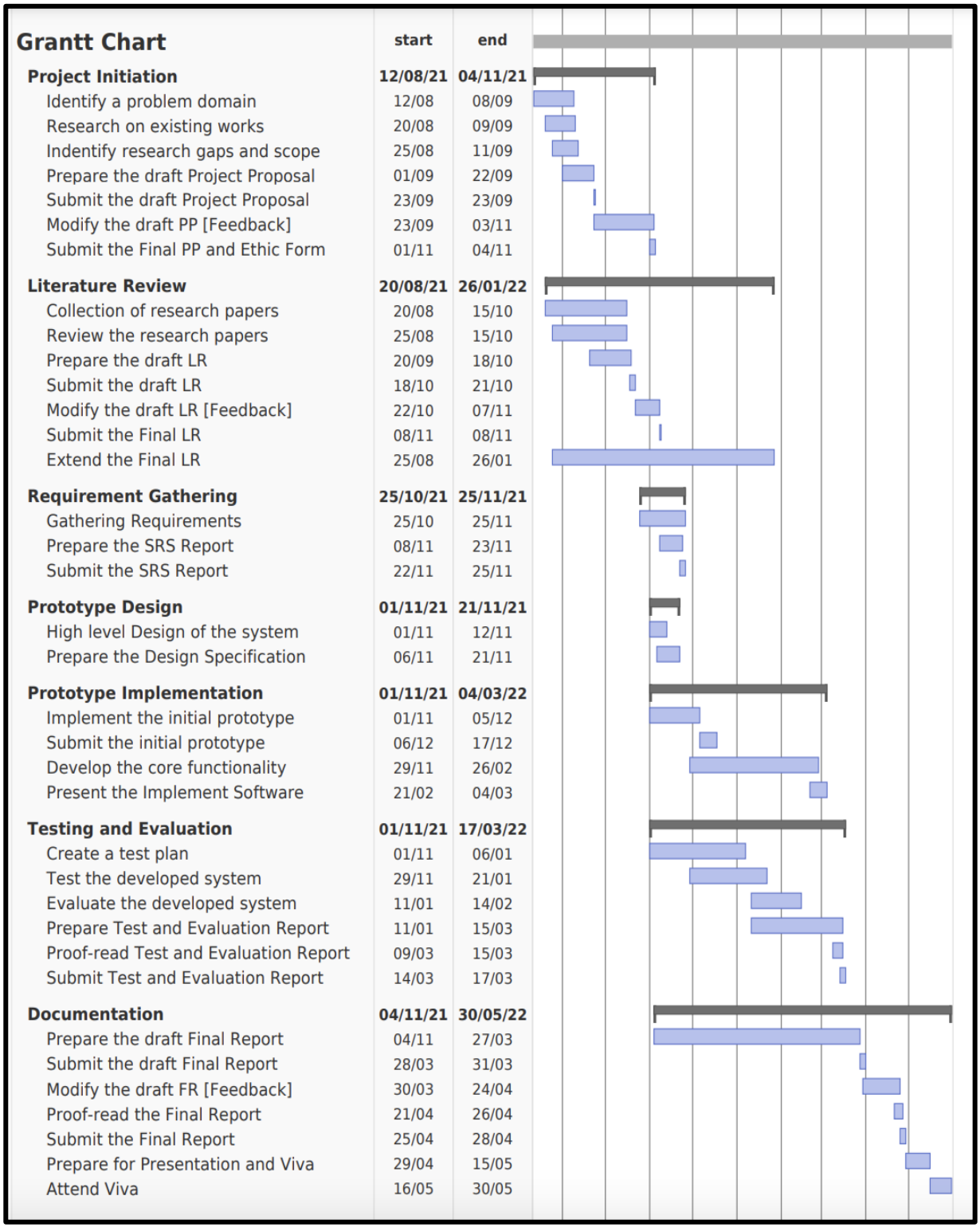


Figure 26 : Gantt Chart (self-composed)

Appendix B – Concept Map

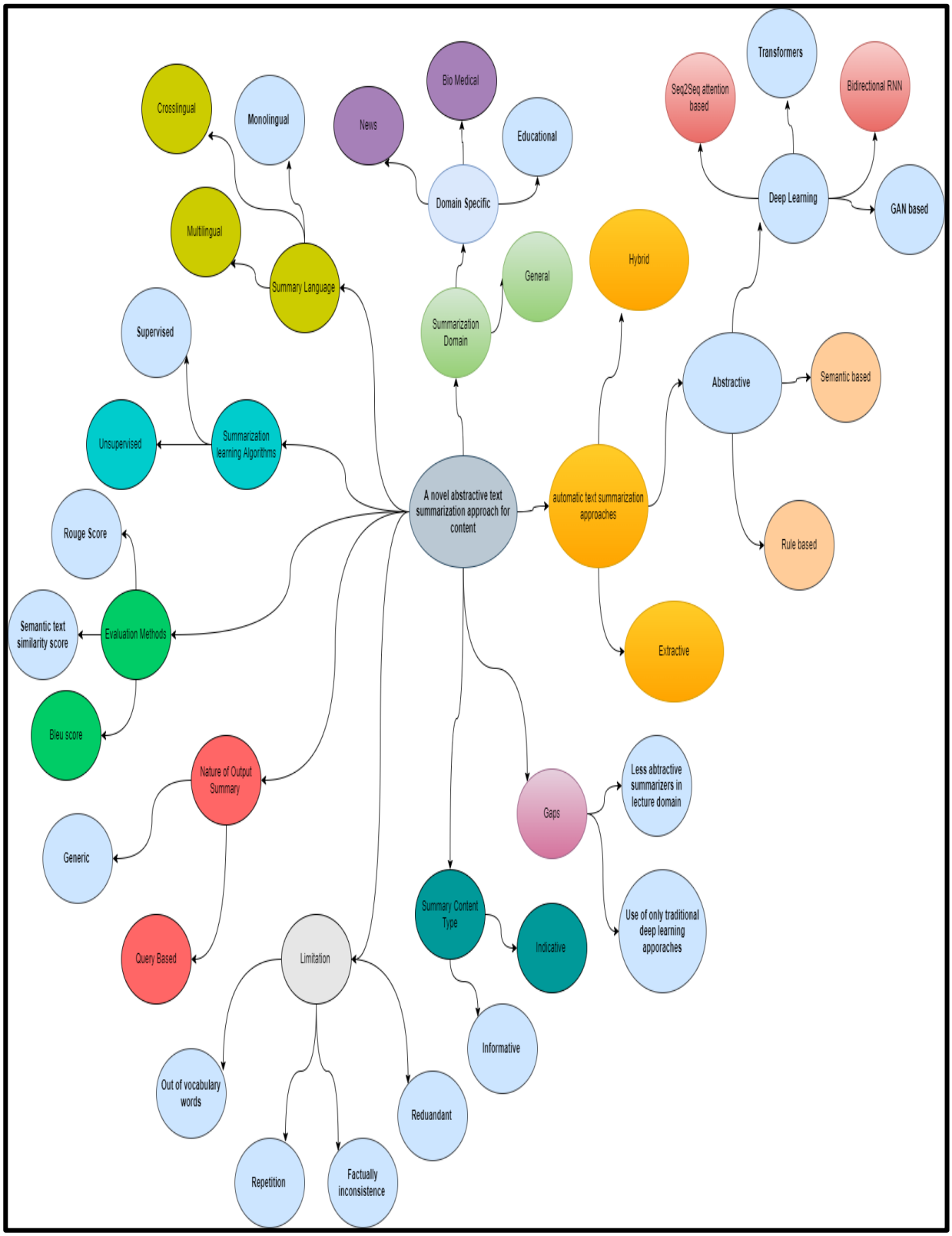


Figure 27 : Concept Map (Self Composed)

Appendix C – Requirement Engineering Survey

Automatic Lecture Video Content Summarization System

Hi,

I'm Farheen Boosary, a final year undergraduate student following the BEng (Hons) Software Engineering Degree at the Informatics Institute of Technology (IIT) affiliated with the University of Westminster, UK.

I am inviting you to complete the following survey to assist me to gather requirements for my Final Year Research Project.

The goal of this research is to develop an automated summarizer system that able to produce a concise and precise textual summary for lecture video content by utilizing a Deep Neural Approach.

Your responses are highly appreciated. This questionnaire's results will be used only for academic purposes and all the responses will be collected anonymously. Please share this survey with anyone who would be interested in this study.

Thank You!

farheen.2018323@iit.ac.lk (not shared) Switch account

* Required

4. Which type of educational content are you usually exposed to? *

- [ ] Online lecture videos
- [ ] Blog posts
- [ ] Academic lecture videos
- [ ] eBooks
- [ ] Books
- [ ] Podcast
- [ ] Other:

5. Do you feel any need of an Automatic Lecture Content Summarizer System ? *

- ◯ Yes
- ◯ No

6. Are you satisfied with manual creation of summary by referring a lecture video ? *

- ◯ Yes
- ◯ No
- ◯ Maybe

7. Do you prefer to view a summary of a lecture video or the entire lecture video? *

- ◯ Summary of the lecture video
- ◯ Entire lecture video
- ◯ Sometimes both

8. Do you think that having an Automatic Lecture Content Summarization System would be beneficial for effective learning? *

- ◯ Yes
- ◯ No
- ◯ Maybe

9. What are the reasons for you to choose an Automatic Lecture Content Summarization System? *

- [ ] Easy to create short notes
- [ ] Watching a lecture video is taking too much time
- [ ] It is possible to slip some significant points in the lecture video due to a lack of concentration
- [ ] Other:

10. Do you think that subject of the lecture video is important when generating a textual summary? *

- ◯ Yes
- ◯ No
- ◯ For some extend

11. What type of functions do you prefer to have in a Automatic Lecture Content Summarization System? *

- [ ] Generating summaries in point form by dividing into topics
- [ ] Generating summaries with relevant images in the lecture video
- [ ] Generating summaries in different languages
- [ ] Generating summaries with providing reference time frame in the lecture video
- [ ] Other:

12. Your opinions and suggestions on Automatic Lecture Content Summarization System. Please mention below if you have any. (Optional)

Your answer

Submit Clear form

Appendix D - UI wireframes

Figure 28 : Landing Page UI Wireframe (Self Composed)

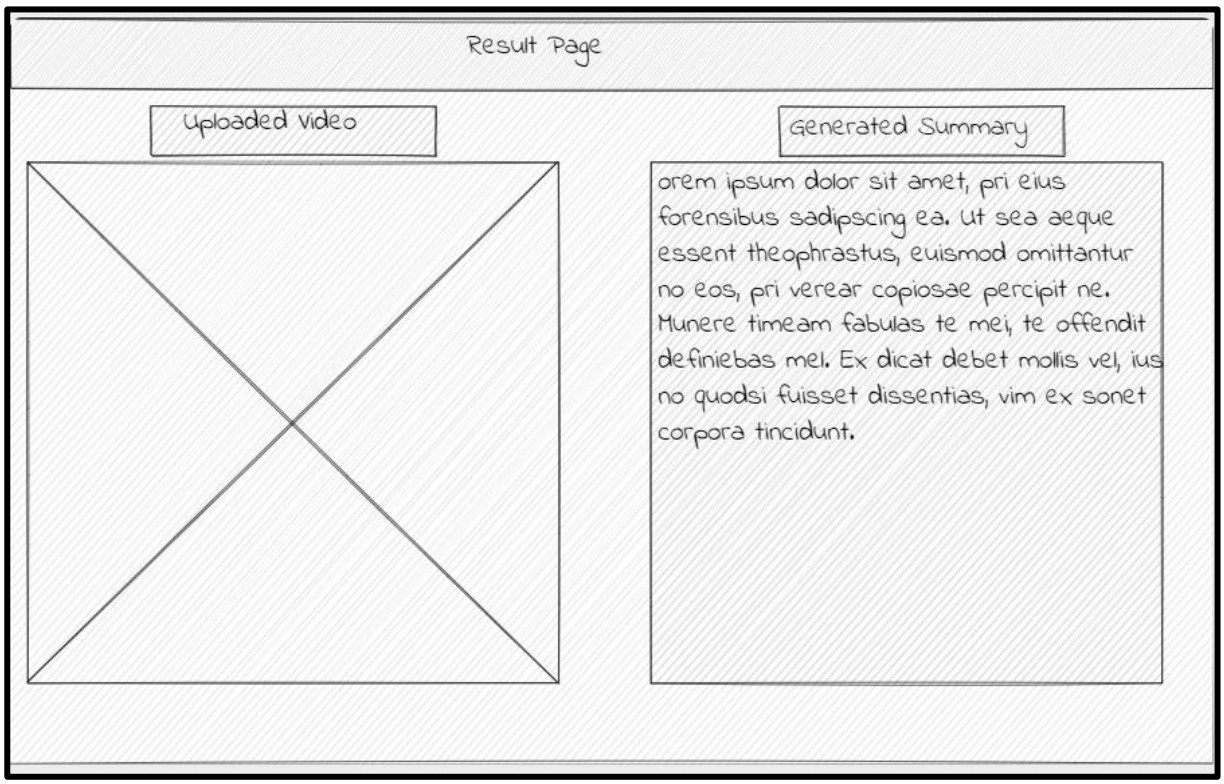


Figure 29 : Results Page UI Wireframe (Self Composed)

Appendix E – Core Functionality Screenshots

Appendix E1 - Pre-processing

Text Cleaning

```
In [55]: contraction_mapping = {"ain't": "is not", "aren't": "are not","can't": "cannot", "'cause": "because", "could've": "could have",
                                "didn't": "did not", "doesn't": "does not", "don't": "do not", "hadn't": "had not", "hasn't": "has not
                                "he'd": "he would","he'll": "he will", "he's": "he is", "how'd": "how did", "how'd'y": "how do you",
                                "I'd": "I would", "I'd've": "I would have", "I'll": "I will", "I'll've": "I will have","I'm": "I am",
                                "i'd've": "i would have", "i'll": "i will",  "i'll've": "i will have","i'm": "i am", "i've": "i have",
                                "it'd've": "it would have", "it'll": "it will", "it'll've": "it will have","it's": "it is", "let's":
                                "mayn't": "may not", "might've": "might have","mightn't": "might not","mightn't've": "might not have",
                                "mustn't": "must not", "mustn't've": "must not have", "needn't": "need not", "needn't've": "need not h
                                "oughtn't": "ought not", "oughtn't've": "ought not have", "shan't": "shall not", "sha'n't": "shall not
                                "she'd": "she would", "she'd've": "she would have", "she'll": "she will", "she'll've": "she will have
                                "should've": "should have", "shouldn't": "should not", "shouldn't've": "should not have", "so've": "so
                                "this's": "this is","that'd": "that would", "that'd've": "that would have", "that's": "that is", "ther
                                "there'd've": "there would have", "there's": "there is", "here's": "here is","they'd": "they would",
                                "they'll": "they will", "they'll've": "they will have", "they're": "they are", "they've": "they have",
                                "wasn't": "was not", "we'd": "we would", "we'd've": "we would have", "we'll": "we will", "we'll've":
                                "we've": "we have", "weren't": "were not", "what'll": "what will", "what'll've": "what will have", "wh
                                "what's": "what is", "what've": "what have", "when's": "when is", "when've": "when have", "where'd":
                                "where've": "where have", "who'll": "who will", "who'll've": "who will have", "who's": "who is", "who
                                "why's": "why is", "why've": "why have", "will've": "will have", "won't": "will not", "won't've": "wi
                                "would've": "would have", "wouldn't": "would not", "wouldn't've": "would not have", "y'all": "you all
                                "y'all'd": "you all would","y'all'd've": "you all would have","y'all're": "you all are","y'all've": "y
                                "you'd": "you would", "you'd've": "you would have", "you'll": "you will", "you'll've": "you will have
                                "you're": "you are", "you've": "you have"}
```

Figure 30 ; Code Snippet for "Data Pre-processing - Contraction Mapping"

```
In [1]: # since < and > from default tokens cannot be removed
        filters = '!"#$%&()*+,-./:;=?@[\\]^_`{|}~\t\n'
        oov_token = '<unk>'

In [ ]: document_tokenizer = tf.keras.preprocessing.text.Tokenizer(oov_token=oov_token)
        summary_tokenizer = tf.keras.preprocessing.text.Tokenizer(filters=filters, oov_token=oov_token)

In [70]: document_tokenizer.fit_on_texts(document)
         summary_tokenizer.fit_on_texts(summary)

In [71]: inputs = document_tokenizer.texts_to_sequences(document)
         targets = summary_tokenizer.texts_to_sequences(summary)
```

Figure 31 : Code Snippet for "Data Pre-processing - Tokenization"

Appendix E2 – All the core utility function of transformer model

Positional Encodings

The below code snippet contains the function responsible for obtaining the positional encoding for the input sequences. Positional Encoding usually induces a notion of the ordering among the input text sequence.

```
[32]: def get_angles(position, i, d_model):
          angle_rates = 1 / np.power(10000, (2 * (i // 2)) / np.float32(d_model))
          return position * angle_rates

[33]: def positional_encoding(position, d_model):
          angle_rads = get_angles(
              np.arange(position)[:, np.newaxis],
              np.arange(d_model)[np.newaxis, :],
              d_model
          )

          # apply sin to even indices in the array; 2i
          angle_rads[:, 0::2] = np.sin(angle_rads[:, 0::2])

          # apply cos to odd indices in the array; 2i+1
          angle_rads[:, 1::2] = np.cos(angle_rads[:, 1::2])

          pos_encoding = angle_rads[np.newaxis, ...]

          return tf.cast(pos_encoding, dtype=tf.float32)
```

Figure 32 : Code Snippet for "Positional Encoding"

Masking

The **padding mask** is responsible for ignoring the external padding added to sequences that are shorter than the max length. The **lookahead mask** for masking is responsible for ignoring the last words that occur after a given current word in the target sequence by contributing to the prediction of the current word.

Masking

- Padding mask for masking "pad" sequences
- Lookahead mask for masking future words from contributing in prediction of current words in self attention

```
In [34]: def create_padding_mask(seq):
             seq = tf.cast(tf.math.equal(seq, 0), tf.float32)
             return seq[:, tf.newaxis, tf.newaxis, :]

In [35]: def create_look_ahead_mask(size):
             mask = 1 - tf.linalg.band_part(tf.ones((size, size)), -1, 0)
             return mask
```

Figure 33 : Code Snippet for "Masking"

Scaler Dot Product

Scaled Dot Product

```
In [36]: def scaled_dot_product_attention(q, k, v, mask):
             matmul_qk = tf.matmul(q, k, transpose_b=True)

             dk = tf.cast(tf.shape(k)[-1], tf.float32)
             scaled_attention_logits = matmul_qk / tf.math.sqrt(dk)

             if mask is not None:
                 scaled_attention_logits += (mask * -1e9)

             attention_weights = tf.nn.softmax(scaled_attention_logits, axis=-1)

             output = tf.matmul(attention_weights, v)
             return output, attention_weights
```

Figure 34 : Code Snippet for "Scaler Dot Product"

Multi-head attention

Multi-Headed Attention

In [37]:
```
class MultiHeadAttention(tf.keras.layers.Layer):
    def __init__(self, d_model, num_heads):
        super(MultiHeadAttention, self).__init__()
        self.num_heads = num_heads
        self.d_model = d_model

        assert d_model % self.num_heads == 0

        self.depth = d_model // self.num_heads

        self.wq = tf.keras.layers.Dense(d_model)
        self.wk = tf.keras.layers.Dense(d_model)
        self.wv = tf.keras.layers.Dense(d_model)

        self.dense = tf.keras.layers.Dense(d_model)

    def split_heads(self, x, batch_size):
        x = tf.reshape(x, (batch_size, -1, self.num_heads, self.depth))
        return tf.transpose(x, perm=[0, 2, 1, 3])

    def call(self, v, k, q, mask):
        batch_size = tf.shape(q)[0]

        q = self.wq(q)
        k = self.wk(k)
        v = self.wv(v)

        q = self.split_heads(q, batch_size)
        k = self.split_heads(k, batch_size)
        v = self.split_heads(v, batch_size)

        scaled_attention, attention_weights = scaled_dot_product_attention(
            q, k, v, mask)

        scaled_attention = tf.transpose(scaled_attention, perm=[0, 2, 1, 3])

        concat_attention = tf.reshape(scaled_attention, (batch_size, -1, self.d_model))
        output = self.dense(concat_attention)

        return output, attention_weights
```

Figure 35 : Code Snippet for "Mutlti-Headed Attention"

Appendix F – UI Screenshot

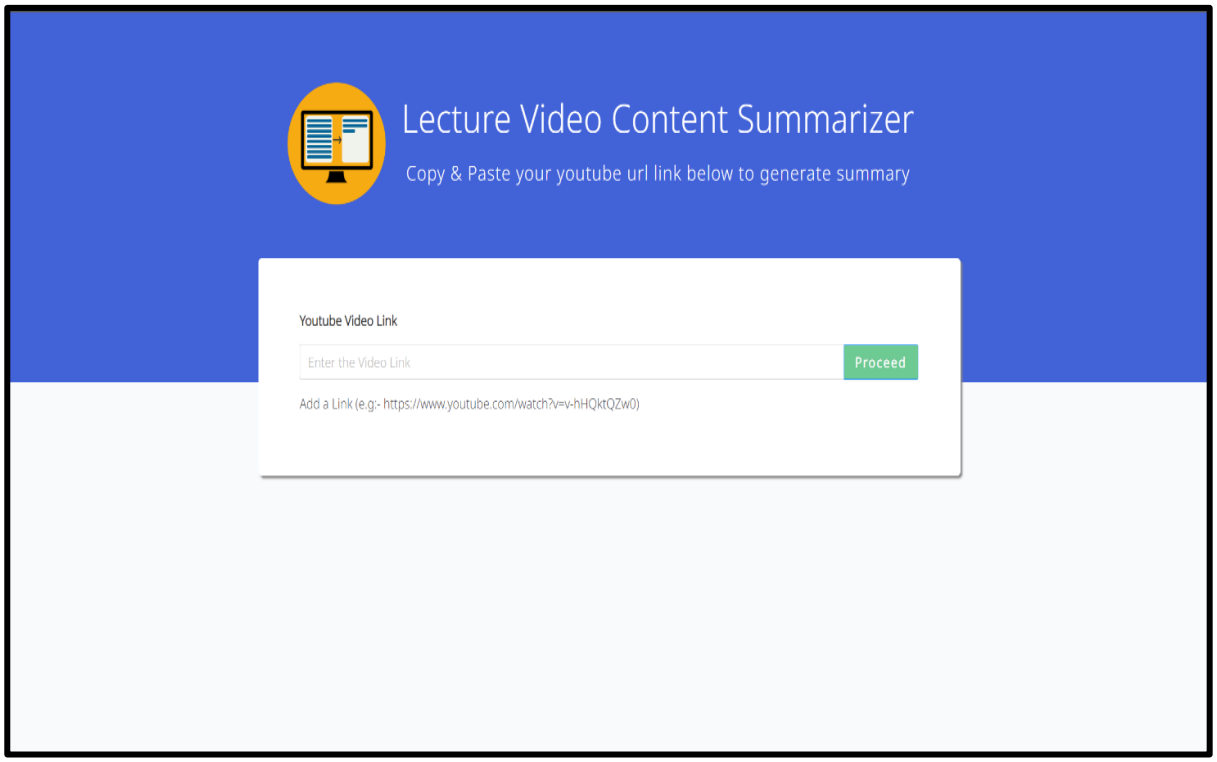


Figure 36 : UI home interface

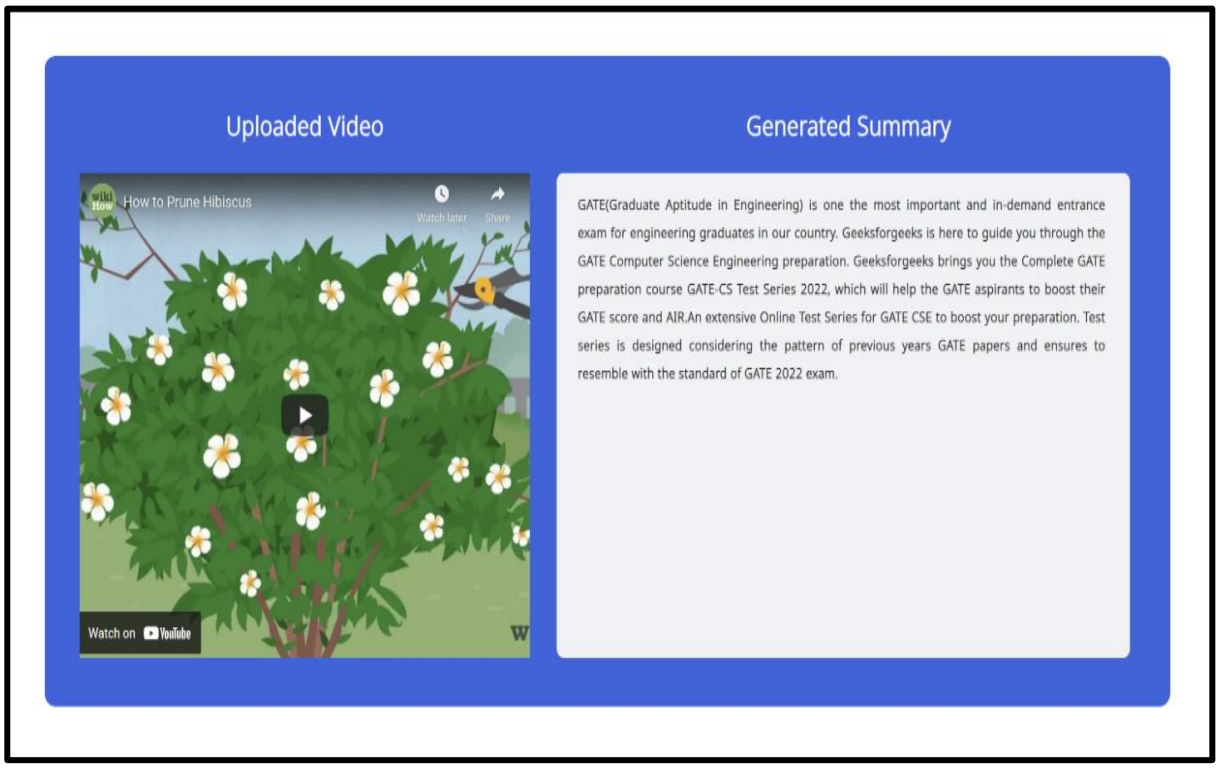


Figure 37 : UI home interface

Appendix G – Module and Integration Testing

Modules	Inputs	Expected Output	Actual Output	Status (Passed/ Failed)
Dataset module	Training Configuration, Dataset	Should Split train and test sample of the dataset	Split train and test sample of the dataset	Passed
Pre-processing module	Datasets	Should pre-process the data according to the provided NLP configuration	The dataset is pre-processed according to the given configurations	Passed
Audio extraction module	URL of the video clip	The audio clip of the inputted video URL should be extracted	The audio clip is extracted from the provided video URL	Passed
Transcription generation module	Extracted audio clip of the video	Should generate a transcript from the extracted audio clip	Generate a transcript from the extracted audio clip	Passed
GANs-based Abstractive Text Summarization module	Generated video transcript	Should generate final abstract summaries relevant to the given transcript	Generate abstract summaries relevant to the given transcript	Passed
Evaluation Module	Testing data, Generated summaries	Should provide the quantitative and qualitative results of the model.	Provides the quantitative and qualitative results of the model.	Passed
UI Module	Video URL	Should display the video along with the	The summary is displayed along with the	Passed

		generated summary in the UI.	provided video in the UI.	

Table 43 : : Module & Integration Testing

Appendix H – Evaluation of Functional Requirements

FR ID	Requirement Description	MoSCoW Priority Level	Evaluation
FR1	User(s) should be able to upload videos into the system through the YouTube video link directly for the creation of informative summaries.	M	**Implemented**
FR2	The maximum time duration of the inputted video must be around 15 min.	M	**Implemented**
FR3	The Input video URL should be validated on the frontend side to check if it is the expected YouTube video link.	M	**Implemented**
FR4	Extraction of audio clip from the input video for the creation video transcript document.	M	**Implemented**
FR5	The extracted audio clip should be converted into a textual transcript document.	M	**Implemented**
FR6	Pre-processing of the created textual transcript by utilizing appropriate natural language processing techniques.	M	**Implemented**
FR7	Preparing the train and test dataset for training the model	M	**Implemented**
FR8	Creation of many to many transformer-based model	M	**Implemented**
FR9	Adaptation of pointer generator network to the created transformer base model.	M	**Implemented**
FR10	Implementation of GANs framework architecture to improve the model performance	M	**Implemented**
FR11	Use of quality evaluation matric related to abstractive summaries to evaluate the model performance	M	**Implemented**

FR12	Performing benchmarking on the implemented system for evaluation of the performance of the abstractive text summarization model.	C	**Implemented**
FR13	Generation of an abstract informative summary as the output from the developed NLP model.	M	**Implemented**
Functional Requirement Completion Percentage = (13 /13) × 100 = 100%			

Table 44 : Evaluation of Functional Requirements

Appendix I – Evaluation of Non-Functional Requirements

NFR ID	Requirement	Priority Level	Evaluation
NFR1	The system must be able to generate a summary for the inputted video within 10 mins.	C	**Implemented**
NFR2	The system should contain a user-friendly UI interface	M	**Implemented**
NFR3	The system should generate human understandable abstract summaries at least for a considerable level.	M	**Implemented**
NFR4	The system should not save videos or the generated transcript without user consent.	C	**Implemented**
NFR5	The system should be able to extensible with newer requirements.	S	**Implemented**
NFR6	The system must be deployed to be able to access with a wider range of user(s)	C	**Not Implemented**
Non-Functional Requirement Completion Percentage = (5 / 6)× 100 = 83.33%			

Table 45 : Evaluation of Non-Requirement Functionalities