

**TRIBHUVAN UNIVERSITY**

**INSTITUTE OF ENGINEERING**

**THAPATHALI CAMPUS**

**Proposal**

**On**

**Presentify: Presentation Slide Generation using NLP**

**and Deep Learning**

**Submitted By:**

Atul Shreewastav (THA077BCT013)

Bidhan Acharya (THA077BCT015)

Nischal Paudel (THA077BCT028)

Yugratna Humagain (THA077BCT047)

**Submitted To:**

Department of Electronics and Computer Engineering

Thapathali Campus

Kathmandu, Nepal

February, 2024



**TRIBHUVAN UNIVERSITY**

**INSTITUTE OF ENGINEERING**

**THAPATHALI CAMPUS**

**Proposal**

**On**

**Presentify: Presentation Slide Generation using NLP**

**& Deep Learning**

**Submitted By:**

Atul Shreewastav (THA077BCT013)

Bidhan Acharya (THA077BCT015)

Nischal Paudel (THA077BCT028)

Yugratna Humagain (THA077BCT047)

**Submitted To:**

Department of Electronics and Computer Engineering

Thapathali Campus

Kathmandu, Nepal

In partial fulfillment for the award of the Bachelor’s Degree in

Computer Engineering

**Under the Supervision of**

Er. Saroj Sakya

February, 2024

# ACKNOWLEDGEMENT

We would like to deeply thank Department of Electronics and Computer Engineering of Institute of Engineering, Thapathali Campus for allowing us to broaden our knowledge and concepts on programming with the accomplishment of this project. We are also thankful towards our teacher and supervisor Er. Saroj Sakya sir and our senior Rajendra Baskota for providing us the ideas and resources necessary for this project. This project would not be initiated without the guidance and supervision of our respected supervisor.

Atul Shreewastav (THA077BCT013)

Bidhan Acharya (THA077BCT015)

Nischal Paudel (THA077BCT028)

Yugratna Humagain (THA077BCT047)

# ABSTRACT

This project employs a fine-tuned T5 transformer model trained on custom data to automatically extract key points from computer science research articles and transform these key points into presentation slides. The goal is to streamline the extraction of key insights from diverse articles and transform them in the form of a presentation slide. The T5 model is proficient in text-to-text transfer tasks and is fine-tuned and evaluated using domain-specific metrics. The resulting system integrates this capability and transforms a research article into coherent presentation slides. This project holds potential for revolutionizing how technical information is summarized and presented, fostering clearer communication in the research community.

*Keywords: Fine-Tuning, Presentation Generation, Research Paper Summarization, T5 Transformer*

TABLE OF CONTENT

[ACKNOWLEDGEMENT i](#_Toc158149819)

[ABSTRACT ii](#_Toc158149820)

[List of Figures vi](#_Toc158149821)

[List of Tables vii](#_Toc158149822)

[List of Abbreviations viii](#_Toc158149823)

[1. Introduction 1](#_Toc158149824)

[1.1 Background 1](#_Toc158149825)

[1.2 Motivation 1](#_Toc158149826)

[1.3 Problem Definition 1](#_Toc158149827)

[1.4 Objectives 2](#_Toc158149828)

[1.5 Scope and Application 2](#_Toc158149829)

[2. Literature Review 3](#_Toc158149830)

[2.1 Background and Related work 3](#_Toc158149832)

[3. Requirement analysis 4](#_Toc158149833)

[3.1 Functional and Non-Functional Requirements 4](#_Toc158149835)

[3.1.1 Functional Requirements: 4](#_Toc158149836)

[3.1.2 Non-Functional Requirements: 5](#_Toc158149837)

[3.2 Software Requirements 6](#_Toc158149838)

[3.3 Hardware Requirements 6](#_Toc158149839)

[3.4 Feasibility Analysis 7](#_Toc158149840)

[3.4.1 Economic Feasibility 7](#_Toc158149841)

[3.4.2 Technical Feasibility 7](#_Toc158149842)

[3.4.3 Operational Feasibility 7](#_Toc158149843)

[4. Dataset analysis 8](#_Toc158149844)

[4.1 Dataset Characteristics 8](#_Toc158149846)

[4.1.1 Size 8](#_Toc158149847)

[4.1.2 Features 8](#_Toc158149848)

[5. Methodology 9](#_Toc158149849)

[5.1 Data Collection 9](#_Toc158149851)

[5.2 Data Preprocessing 9](#_Toc158149852)

[5.3 Dataset Extraction 9](#_Toc158149853)

[5.4 Model Selection 10](#_Toc158149854)

[5.5 Content and Slide Generation 10](#_Toc158149855)

[5.6 System Architecture and description of Working Principle 10](#_Toc158149856)

[5.7 Natural Language Processing 12](#_Toc158149857)

[5.7.1 Natural Language Understanding 12](#_Toc158149858)

[5.7.2 Natural Language Generation 13](#_Toc158149859)

[5.7.3 Application of Natural Language Processing 13](#_Toc158149860)

[5.8 Summarization 14](#_Toc158149861)

[5.8.1 Domain-specific Summarization 14](#_Toc158149866)

[5.9 Transformer Model 15](#_Toc158149867)

[5.10 Model training 16](#_Toc158149868)

[6. Implementation design 17](#_Toc158149869)

[6.1 Text Extraction 17](#_Toc158149871)

[6.1.1 PDF Document 17](#_Toc158149872)

[6.1.2 PDF Link 17](#_Toc158149873)

[6.2 Gemini model 17](#_Toc158149874)

[6.3 Model 17](#_Toc158149875)

[6.3.1 Fine-Tuning 17](#_Toc158149876)

[6.3.2 Key insights Generation 18](#_Toc158149877)

[6.4 Slide Generation 18](#_Toc158149878)

[6.4.1 Section Definition 18](#_Toc158149879)

[6.4.2 Text-to-Slide Mapping 19](#_Toc158149880)

[7. Result and analysis 20](#_Toc158149881)

[7.1 Presentation Slide 20](#_Toc158149883)

[7.2 Performance analysis 21](#_Toc158149884)

[7.2.1 Loss Curves 21](#_Toc158149885)

[7.2.2 ROUGE Analysis 23](#_Toc158149886)

[8. Remaining task 24](#_Toc158149887)

[8.1 Frontend Development 24](#_Toc158149889)

[8.2 Customization by user for pptx formatting 24](#_Toc158149890)

[8.3 Performance Tuning 24](#_Toc158149891)

[9. appendices 25](#_Toc158149892)

# List of Figures

[Figure 3‑1: RAM and Disk Usage 7](#_Toc158148698)

[Figure 5‑1: Dataset Extraction 11](#_Toc158148699)

[Figure 5‑2: System Architecture of Presentify 12](#_Toc158148700)

[Figure 5‑3: Model Training 13](#_Toc158148701)

[Figure 5‑4: The Transformer - Model Architecture 17](#_Toc158148702)

[Figure 7‑1: Generated Slides 23](#_Toc158148703)

[Figure 7‑2: Epoch vs Loss Graph (in 6 epochs) 24](#_Toc158148704)

[Figure 7‑3: Epoch vs Loss Graph(in 7 epochs) 24](#_Toc158148705)

[Figure 7‑4: Epoch vs Loss Graph (in 8 epochs) 25](#_Toc158148706)

[Figure 7‑5: Epoch vs Rouge 26](#_Toc158148707)

# List of Tables

# List of Abbreviations

|  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| |  |  | | --- | --- | | AI | Artificial Intelligence | | AMD | Advanced Micro Devices | | API | Application Programming Interface | | GB | GigaByte | | GPU | Graphics Processing Unit | | HTML | Hypertext Markup Language | | ILP | Integer Linear Programming | | Intel | Integrated Electronics | | MSE | Mean Square Error | | NLG | Natural Language Generation | | NLP | Natural Language Processing | | NLTK | Natural Language Toolkit | | NLU | Natural Language Understanding | | NumPy | Numerical Python | | PC | Personal Computer | | PDF | Portable Document Format | | PPTX | PowerPoint | | RAM | Random Access Memory | | REGEX | Regular Expressions | | ROUGE | Recall-Oriented Understudy for Gisting Evaluation | | T5 | Text-To-Text Transfer Transformer | | TPU | Tensor Processing Units | | XML | Extensible Markup Language | |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |
|  |  |

# Introduction

To deal with the overwhelming volume of research articles in both academic and industrial domain, this project aims to utilize the transformative capabilities of natural language processing (NLP), through the fine-tuning of a T5 transformer model on a custom dataset drawn from the computer science domain research articles. It aims to use the model's ability in transforming complex textual data to extract key points and core concepts from these articles for the generation of presentation slides. By adapting the pre-trained T5 model parameters to better align with the computer science literature, we aim to restructure the process of knowledge extraction and presentation preparation from research articles.

## Background

In contemporary professional and educational settings, the creation of presentation slides plays a pivotal role in conveying information effectively. However, the manual generation of these slides can be time-consuming and resource-intensive. The advent of Natural Language Processing (NLP) technologies presents an opportunity to streamline this process by automating the extraction and organization of relevant content. This project addresses the need for a more efficient and user-friendly approach to presentation slide creation through the integration of NLP.

## Motivation

The motivation behind this project stems from the recognition of the challenges individuals face in generating presentation slides efficiently. With the proliferation of information in diverse formats, including PDF documents, there is a growing demand for tools that can seamlessly convert this content into visually engaging slides. The motivation is grounded in the belief that leveraging NLP technologies can significantly enhance the speed and accuracy of slide creation, thereby empowering users to focus more on the content itself rather than the manual aspects of formatting and organization.

## Problem Definition

The core problem this project aims to address is the labor-intensive nature of creating presentation slides, particularly when dealing with diverse data sources such as PDF documents. Manual extraction and structuring of content are prone to errors and can impede productivity. The project seeks to mitigate these challenges by automating the process using NLP algorithms, thereby reducing the time and effort required for slide creation.

## Objectives

The main objectives of our project are listed below:

* To develop an interface to facilitate the automated generation of presentation slides from user-inputted PDF documents using T5 transformer model.

## Scope and Application

The objective of this project is to optimize and improve the process of developing presentation slides specifically tailored for academic research papers within the domain of computer science. The core functionality will revolve around users submitting research papers in PDF format, with the system automatically generating concise and informative presentation slides based on the extracted content from these documents. This application will leverage advanced natural language processing (NLP) techniques to carefully analyze and condense the essential information contained in the PDFs, identifying crucial points, thematic elements, and key concepts. Furthermore, the system will intelligently organize the extracted content into a coherent slide format, ensuring a systematic flow of information. The overarching aim is to automate the labor-intensive task of manually constructing presentations, offering users a more efficient and user-friendly alternative within the context of academic research in computer science.

The application's application lies in various professional and educational settings where creating presentations is a frequent requirement. Professionals, students, and educators who regularly deal with PDF documents can benefit from this tool by saving time and effort in the presentation creation process. It caters to individuals who may not have the expertise or time to manually sift through lengthy documents to distill the key points for a presentation. Overall, the project aims to be a valuable asset in enhancing productivity and facilitating effective communication through automated presentation slide generation from PDF content.

# Literature Review



## Background and Related work

Yue Hu and Xiaojun Wan proposed “PPSGen: Learning-Based Presentation Slides Generation for Academic Papers” in 2015. [1]They suggest an innovative system called PPSGen to investigate the daunting task of automatically generating presentation slides from the academic papers. These slides can be used by the presenters to prepare official slides in less time. Regression methods are applied to determine the value of sentences in academic papers. Highly organized slides with good or solid structure are devised using the integer linear programming method by choosing key phrases and sentences. The enhanced slides are created from the submitted PPSGen system.

In 2016, Ektaa Meshram and D. A. Phalke suggested a "Technique for Generating Automatic Slides on the basis of Paper Structure Analysis". Most areas in all fields use slide presentations in an easy and aesthetically pleasing format to collaborate information with all concerned parties. [2]

In 2017, Athar Sefid et al proposed an approach to generating slides for scientific papers using deep neural networks. The authors propose a method that leverages sentence embeddings and context to score sentences for their inclusion in the slides. They achieve this through a two-step process: first, training a model to score sentences, and then using Integer Linear Programming (ILP) to select the most salient ones. Finally, they extract noun phrases from the chosen sentences to create bullet points for the slides. [3]

In 2020, Kevin Shaj et al proposed “Learning Based Slide Generator” which provides a good overview of the current state of the art in learning-based slide generation focusing on learning-based methods using natural language processing and deep learning. It evaluates extractive and abstractive approaches for summarization, noting the advantages of abstractive methods in fluency and coherence [4]. However, there are a few areas where the paper can be improved. The paper doesn’t specify a particular domain which may result in inaccuracy and relevance. It also fails to discuss the fact that these types of systems are still in early stages of development and can sometimes generate misleading slides.

In 2022, Tsu-Jui Fu et al proposed “DOC2PPT: Automatic Presentation Slides Generation from Scientific Documents” which tackle generating slides from scientific papers, but differ in methods from “Automatic Slide Generation for Scientific Papers”. While the previous article prioritizes factual accuracy through sentence selection, this paper focuses on paraphrasing and layout within a hierarchical model. This might sacrifice fidelity, limit flexibility compared to sentence-level approaches, and struggle with specific formatting needs. However, a deeper dive and evaluation on the same dataset is needed for a more conclusive comparison. [5]

Tools such as Microsoft PowerPoint, Open Office, and Apple Pages are used to conventionally prepare slide presentations. This resulted in an exhaustive process of preparation with a chance for failure. The biggest challenge would be missing out important information from research journals and conference papers. Addressing these 6 issues, our project aims to generate slides with more accuracy and relevance. Our project also focuses on making the content visually appealing.

# Requirement analysis



## Functional and Non-Functional Requirements

### Functional Requirements:

1. **User Input:**

* The system should allow users to input data in the form of PDF documents or arXiv links.

1. **Content Extraction:**

* Implement algorithms to intelligently extract relevant information from the provided input.

1. **Model Integration:**

* The system should integrate the fine-tuned T5 transformer model for key point extraction.
* Implement text extraction and preprocessing mechanisms compatible with T5 requirements.

1. **Summarization Process:**

* Automatically generate concise summaries of computer science research articles using the T5 model.

1. **Automated Slide Generation:**

* Develop mechanisms for automatically generating presentation slides based on the extracted content.

1. **Content Organization:**

* Organize the extracted content into coherent and visually appealing slides.

1. **Input Validation:**

* Enforce comprehensive input validation to address potential errors, particularly in the context of file uploads, ensuring the system can adeptly handle a diverse range of input data by overcoming limitations related to file types, sizes, and formats.

1. **Customization Options:**

* Provide options for users to customize the appearance and layout of generated slides.

### Non-Functional Requirements:

1. **Performance:**

* The system should perform content extraction and slide generation efficiently, even with large and complex input data.

1. **Scalability:**

* Design the system to be scalable, allowing it to handle an increasing number of users and larger datasets.

1. **Security:**

* Implement security measures to ensure the confidentiality and integrity of user data, especially when handling sensitive information.

1. **Reliability:**

* Ensure the system's reliability by minimizing downtime and errors during the content extraction and slide generation processes.

1. **Usability:**

* The user interface should be user-friendly, requiring minimal training for users to navigate and utilize the application effectively.

1. **Compatibility:**

* Ensure compatibility with popular web browsers to enhance the accessibility of the application.

1. **Maintainability:**

* Design the system with modular and well-documented code to facilitate ease of maintenance and future updates.

1. **Compliance:**

* Ensure compliance with relevant data protection regulations and standards to safeguard user privacy.

1. **Error Handling:**

* Implement robust error-handling mechanisms to address potential issues during content extraction and slide generation.

1. **Backup and Recovery:**

* Implement regular backup procedures and a robust recovery mechanism to prevent data loss and ensure system stability.

## Software Requirements

1. **Python:** Python is a high level, general purpose, interpreted, dynamic programming language. Python supports multiple programming paradigms, including object-oriented, imperative and functional programming or procedural styles.
2. **NumPy:** NumPy is a Python library for numerical computing, offering powerful data structures and tools for working with arrays, matrices, and mathematical functions. Its efficient operations make it essential for scientific computing, data analysis, and machine learning in Python.
3. **Pandas:** A Python library specializing in data manipulation and analysis, providing high-performance, easy-to-use data structures and tools for working with structured data like tables and time series.
4. **BeautifulSoup:** Beautiful Soup is a Python package for parsing HTML and XML documents (including having malformed markup, i.e., non-closed tags, so named after tag soup). It creates a parse tree for parsed pages that can be used to extract data from HTML, which is useful for web scraping. Beautiful Soup.
5. **Google Colab:** Colab is a hosted Jupyter Notebook service that requires no setup to use and provides free access to computing resources, including GPUs and TPUs. Colab is especially well suited to machine learning, data science, and education.
6. **FastAPI:** FastAPI is a modern, fast, web framework for building APIs with Python. It's designed for high performance and productivity, using Python type hints for automatic data validation and documentation generation.

## Hardware Requirements

Any normal PC is preferable for training the model using Google Colab but for training the model on a local system:

* Multi-core CPU (Intel Xeon or AMD Ryzen) to handle concurrent requests and computations.
* RAM (16GB or more) is required.

High Performance GPU to expedite model training and inference.

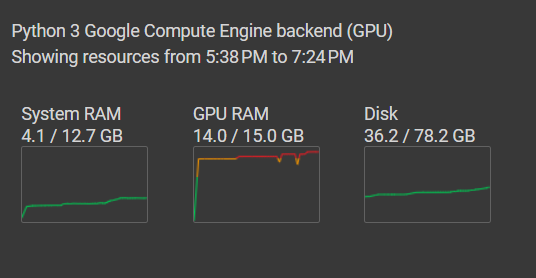


Figure 3‑1: RAM and Disk Usage

## Feasibility Analysis

### Economic Feasibility

The project is cost-effective in the long run despite initial development expenses. Users save time and resources compared to manual slide creation, leading to a positive return on investment. Potential for widespread adoption in professional and educational sectors further supports economic viability.

### Technical Feasibility

The project is technically feasible with well-established technologies like Transformers that use deep learning frameworks and NLP algorithms. Integration of PDF parsing and content extraction tools can be done using existing python libraries and APIs.

### Operational Feasibility

The project seamlessly integrates into existing workflows with an intuitive user interface. Minimal training is required due to the automated nature of the system, making it accessible to a wide range of users. Scalability ensures the project can handle varying workloads and accommodate a growing user base. Regular updates and maintenance are crucial to address evolving user needs and ensure continued operational efficiency.

# Dataset analysis

The dataset utilized in this study consists solely of research articles sourced from the computer science domain via arXiv.org. This focused selection ensures relevance and alignment with the research topic, providing a targeted dataset for analysis and exploration.



## Dataset Characteristics

### Size

The dataset consists of a total of 15,655 rows, divided into three parts: training (10,958 rows), testing (3,131 rows), and validation (1,566 rows).

### Features

Each row of the dataset contains two main features:

Text: This column contains the content extracted from research articles. Each row represents a specific section of the article, including introduction, literature review, methodology, results, or conclusion.

Summary: This column contains the summarized version of each section (introduction, literature review, methodology, results, conclusion). The summary provides a condensed representation of the corresponding text.



# Methodology

This report outlines the progress and objectives of our project, focusing on the application of the T5 transformer model for summarizing computer science domain research articles and subsequently converting the summarized content into presentation slides.



## Data Collection

Collect a large corpus of research articles from the computer science domain, including various sections such as introduction, literature review, methodology, results, and conclusion.

## Data Preprocessing

For text cleaning using regular expressions (regex) and NLTK (Natural Language Toolkit), the initial step involves applying regex patterns to remove unwanted characters, symbols, and formatting artifacts from the raw text extracted from PDFs. This includes removing special characters, punctuation marks, extra whitespace, and non-alphanumeric characters. Following this, NLTK is used for word tokenization and custom. Stopword removal is then performed to filter out words that do not contribute significantly to the text's meaning. This combined approach of regex-based text cleaning and NLTK-based preprocessing helps prepare the text data for further segmentation, summarization, and analysis in the research article summarization project.

## Dataset Extraction

The dataset preparation process commenced with web scraping to retrieve links pointing to PDF files containing research articles from ArXiv.org within the computer science domain. Subsequently, text extraction from these PDF files was conducted using specialized parsing tools (PyMuPDF), facilitating the conversion of PDF documents into text format. Gemini Language Model was used for generation of required fields for dataset creation, including sections such as introduction, literature review, methodology, results, and conclusion, along with their corresponding summaries, based on the extracted text content. Finally, the dataset was populated by organizing the structured data format, incorporating both the raw text content and its corresponding summaries.

The purposed system architecture for dataset extraction is shown in the following block diagram.

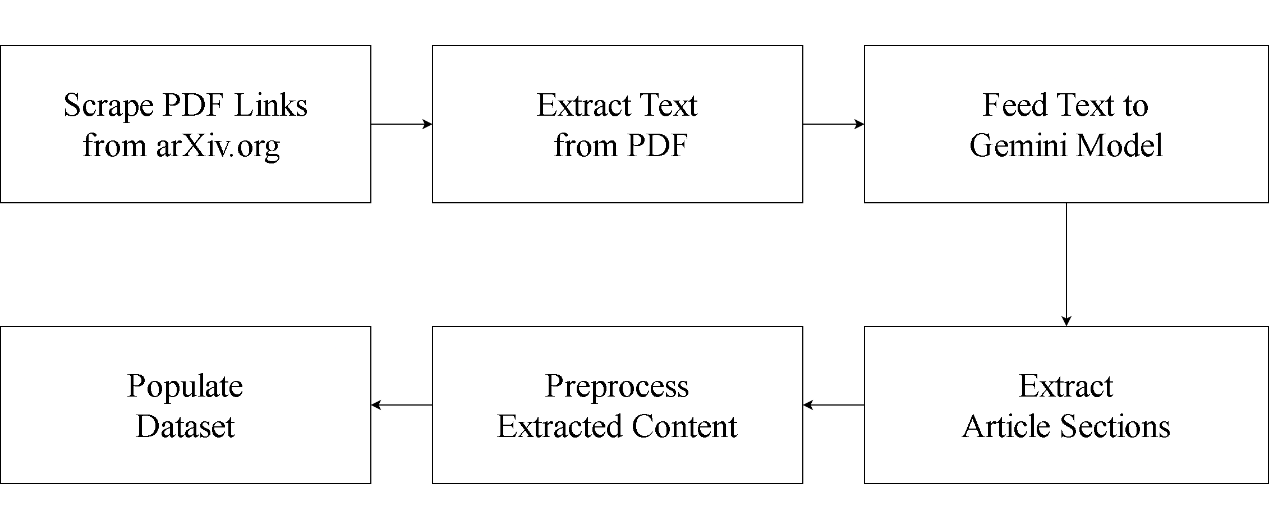


Figure 5‑1: Dataset Extraction

## Model Selection

Select the T5 transformer model configured for key point extraction. Customize the model input and output layers to accommodate the technical nature of computer science content.

## Content and Slide Generation

Trained the T5 model to extract informative key points of research articles. Developed a mechanism to convert the key points into presentation slides, ensuring clarity and coherence.

## System Architecture and description of Working Principle

The system architecture of Presentify comprises several key components designed to streamline the process of summarizing research articles and generating presentation slides. Initially, the system accepts input in either PDF format or via an Arxiv link, providing flexibility in data sources. Subsequently, text extraction is performed using PyMuPDF for PDF files and BeautifulSoup4 for Arxiv links, enabling the conversion of PDF content into text format.

The extracted text data is fed into Gemini, a language model, with prompts to extract specific topics such as Introduction, Literature Review, Methodology, Results, and Conclusion.

The next step involves leveraging a fine-tuned T5 model, which sequentially processes the content for each topic. The fine-tuned T5 model effectively summarizes the given content, distilling it into concise and coherent summaries.

Once the content has been summarized, it is passed to the presentation module, where it undergoes transformation into a presentation-ready format. This module formats the summarized content into the presentation slides, ensuring clarity and coherence in the presentation of key points and core concepts.

Finally, the user is presented with a standardly formatted presentation slide, providing an organized and neat overview of the research article's content. The output slide serves as an effective tool for communicating the main ideas and findings of the article in a professional and visually engaging manner.

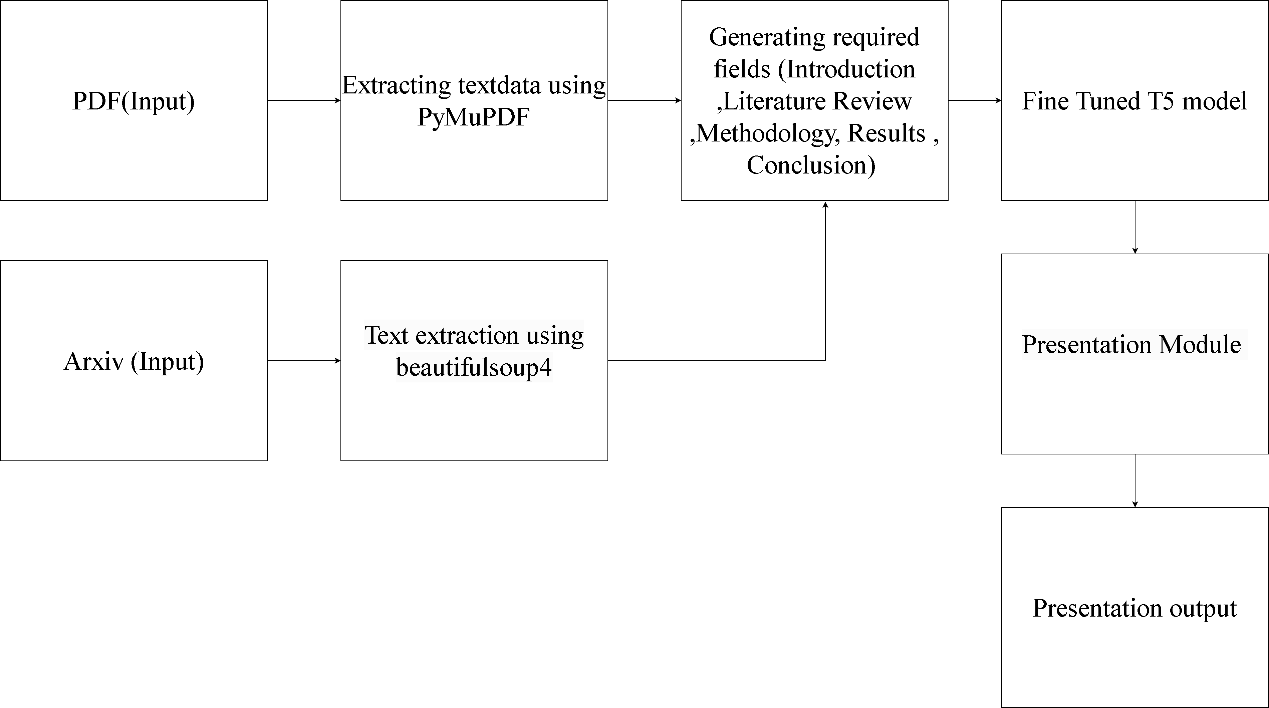


Figure 5‑2: System Architecture of Presentify

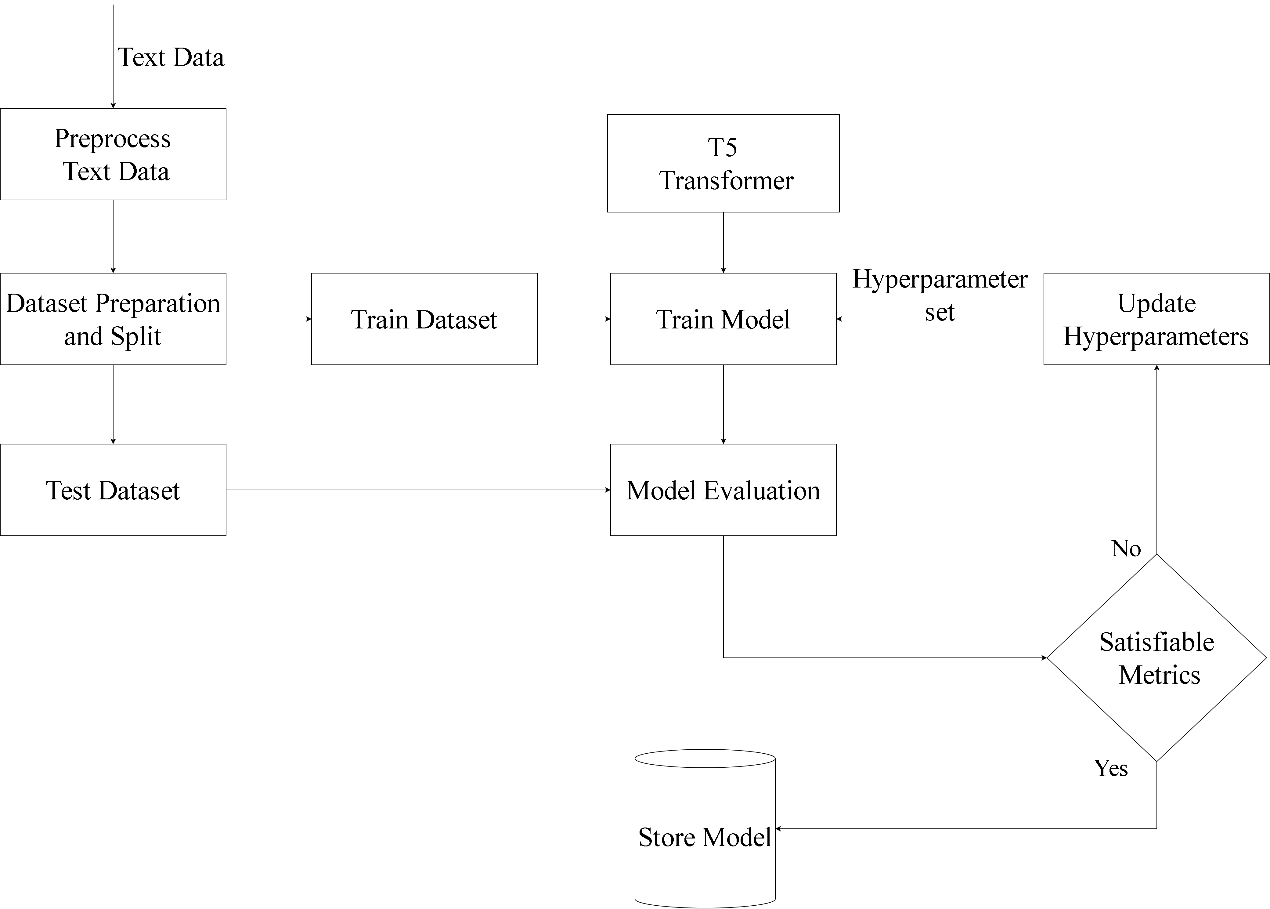


Figure 5‑3: Model Training

## Natural Language Processing

Natural Language Processing (NLP) sits at the intersection of Artificial Intelligence and Linguistics, with a focus on enabling computers to comprehend and respond to human language. The primary aim is to simplify user interactions with computers by allowing them to communicate in natural language. NLP is particularly beneficial for users who may not be familiar with machine-specific languages or lack the time to learn them. It views language as a system governed by rules and symbols, where symbols, manipulated by rules, convey or broadcast information. NLP encompasses two main components: Natural Language Understanding, which involves comprehending text, and Natural Language Generation, which focuses on creating text. The overarching goal is to bridge the gap between human communication and machine interaction.

### Natural Language Understanding

Natural Language Understanding (NLU) is the computer's capacity to comprehend human language, pivotal for applications like chatbots, voice assistants, and automated translation services. At its core, NLU involves parsing, a process that transforms natural language text into a structured format understandable by computers. For instance, when inputting content for a slide on the slide generator, NLU breaks down the sentences into parts of speech, such as nouns and verbs, to provide a structured representation. Beyond parsing, NLU encompasses tasks such as sentiment analysis, entity recognition, and semantic role labeling. In the case of the slide generator, NLU ensures that the generated text for each slide is not only grammatically correct but also contextually relevant, enhancing the overall quality and coherence of the presentation. NLU plays a crucial role in bridging the gap between human communication and machine understanding, facilitating various language-related applications.

### Natural Language Generation

Natural Language Generation (NLG) is the process of transforming machine-readable data into human-readable text. For instance, consider the application of an NLG system in a slide generator. Instead of generating random words, the NLG software ensures that the text for each slide is composed in a coherent and human-like manner. This meticulous process results in more engaging and contextually relevant presentations, showcasing the versatility of NLG in applications like content creation for slide decks. NLG plays a pivotal role in crafting contextually relevant and human-like text, enhancing user experiences in applications such as conversational agents and content generation.

### Application of Natural Language Processing

1. Text Categorization: Classifies large datasets into predefined categories, applicable in spam filters and trouble ticket categorization.
2. Spam Filtering: Employs filters like content filters, header filters, and rules-based filters to combat unwanted emails.
3. Information Extraction: Identifies key phrases in textual data, benefiting domain-specific search engines.
4. Summarization: Addresses information overload, providing valuable insights from large datasets.

## Summarization

In the domain of Natural Language Processing (NLP), text summarization has emerged as a pivotal focus in the present time. A substantial volume of textual content is routinely generated in digital format on the internet, encompassing news articles, product and service reviews, e-libraries, social media posts, personal and governmental blogs, websites, online tutorials, and e-publications, among others. Despite being dispersed and unprocessed, the text sourced from these outlets necessitates computational inspection for extracting valuable insights swiftly, efficiently, and on a scalable basis. The innovative methodologies within the domain of text summarization are designed to address and resolve this computational challenge.



### Domain-specific Summarization

Domain-specific summarization focuses on condensing information within a particular field or subject area, tailoring the summarization process to the unique characteristics and nuances of that domain. This specialized approach enhances the relevance and precision of the generated summaries. Similar to generic summarization, domain-specific summarization employs two primary strategies: extractive and abstractive.

Domain-specific summarization finds widespread application in various fields such as scientific research, technical literature, and industry-specific reports. By tailoring the summarization process to the unique features of a particular domain, this approach enhances the efficiency of information retrieval and supports a more comprehensive understanding of the subject matter. The choice between extractive and abstractive methods in domain-specific summarization depends on the specific characteristics of the documents and the desired goals of summarization within that particular domain.

## Transformer Model

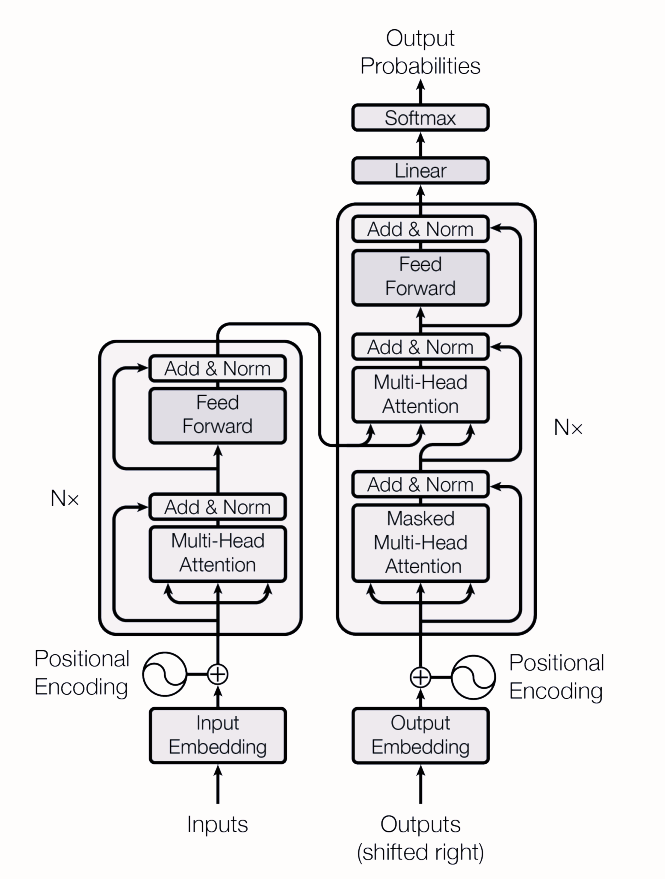


Figure 5‑4: The Transformer - Model Architecture

The Transformer architecture has significantly advanced artificial intelligence, particularly in natural language processing (NLP), by revolutionizing how models understand and process sequential data. [6] At its core, the Transformer architecture comprises encoder and decoder components. The encoder is responsible for analyzing input data and extracting meaningful representations through a series of layers. These layers employ a self-attention mechanism, which allows the model to weigh the importance of different words in a sentence simultaneously, capturing long-range dependencies efficiently. Following self-attention, the representations undergo processing through feedforward neural networks, enabling the model to capture complex patterns in the data.

On the other hand, the decoder utilizes the encoded representations from the encoder to generate the output sequence step by step. Similar to the encoder, the decoder employs self-attention mechanisms to focus on different parts of the input and previously generated output sequences. Additionally, it utilizes an encoder-decoder attention mechanism to capture relevant information from the encoder's output, ensuring coherence between the input and output sequences. To provide sequential information to the model, positional encodings are added to the input embeddings, allowing the model to differentiate between tokens based on their position in the sequence. This comprehensive approach, along with techniques like layer normalization, residual connections, and multi-head attention, enables the Transformer architecture to effectively model sequential data, making it a fundamental tool in modern NLP and other AI applications.

## Model training

Model training is a crucial step in the data science development process, where practitioners aim to find the optimal combination of weights and bias for a machine learning algorithm. The primary goal during model training is to minimize a loss function across the range of predictions. This process aims to create the most accurate mathematical representation of the relationship between data features and a target label in supervised learning or among the features themselves in unsupervised learning.

Loss functions play a vital role in model training as they determine how to optimize machine learning algorithms. Data science practitioners choose different types of loss functions based on the specific objective, type of data, and the algorithm being used. An example of a popular loss function is Mean Square Error (MSE).

# Implementation design



## Text Extraction

### PDF Document

When a PDF document is available, PyMuPDF is employed to extract the text data. PyMuPDF facilitates the extraction of textual content from PDF files, allowing for the retrieval of information contained within the provided PDF document.

### PDF Link

When the link to the PDF document is provided, the data extraction process involves web scraping using Beautiful Soup, a Python library for pulling data out of HTML and XML files. Beautiful Soup facilitates the extraction of text content from web pages by parsing the HTML structure of the provided link and extracting the relevant text data

## Gemini model

Once the data extraction process is completed, the extracted data is fed into the Gemini model. The Gemini model is then tasked with analyzing the entirety of the extracted text, from the beginning to the end, for each of the predefined sections: Introduction, Literature Review, Methodology, Results, and Conclusion. Utilizing its natural language processing capabilities, Gemini identifies and isolates the content corresponding to each section. This involves parsing through the extracted text and identifying the boundaries of each section, distinguishing the transition from one section to the next. By comprehensively analyzing the text data, Gemini effectively extracts and defines the content of each section, laying the groundwork for subsequent summarization and presentation slide generation tasks

## Model

### Fine-Tuning

The workflow involves an iterative process of fine-tuning the T5 model based on evaluation metrics. Initially, the model is trained on the provided dataset, and its performance is evaluated using metrics such as training loss, validation loss and ROUGE scores. Based on these evaluations, adjustments are made to the model's hyperparameters to refine its summarization accuracy further. This iterative refinement process aims to optimize the model's performance by fine-tuning its parameters in response to the observed evaluation metrics. Through successive iterations of training, evaluation, and parameter adjustment, the T5 model evolves to better capture the details of summarizing research articles, ultimately enhancing its effectiveness in generating accurate and concise key point.

### Key insights Generation

The text extracted by the Gemini model, which accurately identifies and segregates content corresponding to specific sections like Introduction, Literature Review, Methodology, Results, and Conclusion, undergoes summarization using a fine-tuned T5 model. This T5 model has been specifically adjusted to meet the summarization requirements of our project, aligning its parameters to optimize the summarization process for research articles. As the text from each section passes through the T5 model, it undergoes a process of condensation and simplification, resulting in concise and informative summaries that capture the essence of each section. This approach enables the creation of brief yet comprehensive summaries for each section, aiding in the comprehension of the key points and discoveries presented in the research article. Through the use of fine-tuned T5 summarization, our project enhances the summarization process, enabling users to efficiently extract important information from research articles spanning diverse sub-domains in the field of computer science.

## Slide Generation

### Section Definition

#### This PowerPoint presentation consists of six slides each serving a specific purpose to provide a comprehensive overview of the content derived from a research article.

The first slide serves as the title slide, presenting the title of the PDF document and the name of the author or authors. This slide sets the context for the presentation and introduces the audience to the topic of discussion.

Following the title slide, the subsequent five slides are dedicated to different sections of the research article. The second slide contains the Introduction section, which outlines the background, objectives, and scope of the study. It provides essential context for understanding the research problem.

The third slide is dedicated to the Literature Review, which summarizes existing literature and research relevant to the study. This section provides a critical analysis of previous studies and establishes the theoretical framework upon which the current research is built.

The fourth slide presents the Methodology section, detailing the research methods, techniques, and procedures employed in the study. It describes how the research was conducted, including data collection, analysis, and interpretation methods.

The fifth slide focuses on the Results section, which presents the findings and outcomes of the study.

The sixth slide contains the Conclusion section, which summarizes the main findings of the study and discusses their implications. It also highlights any limitations of the study and suggests areas for future research.

Overall, this PowerPoint presentation provides a structured and organized overview of the research article, guiding the audience through the various sections and key points of the study.

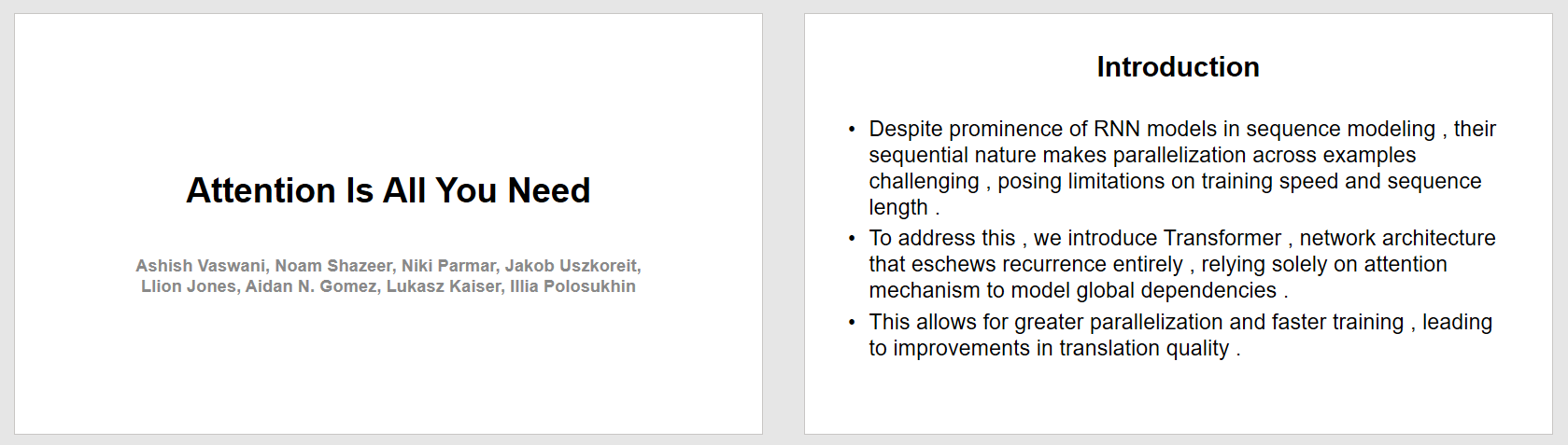
### Text-to-Slide Mapping

Once the text has been segmented and formatted into sections, the python-pptx library is utilized to create corresponding slides for each section. This involves generating a new slide for each section and populating it with the formatted text content. The python-pptx library provides functionality to customize the layout, styling, and formatting of the slides to ensure consistency and visual appeal. Different slide layouts is used for the title slide and content slides. Overall, the text-to-slide mapping process facilitates the transformation of the textual content extracted from the fine-tuned model into a structured presentation format, enabling effective communication of the research findings and insights to the audience.

# Result and analysis



## Presentation Slide



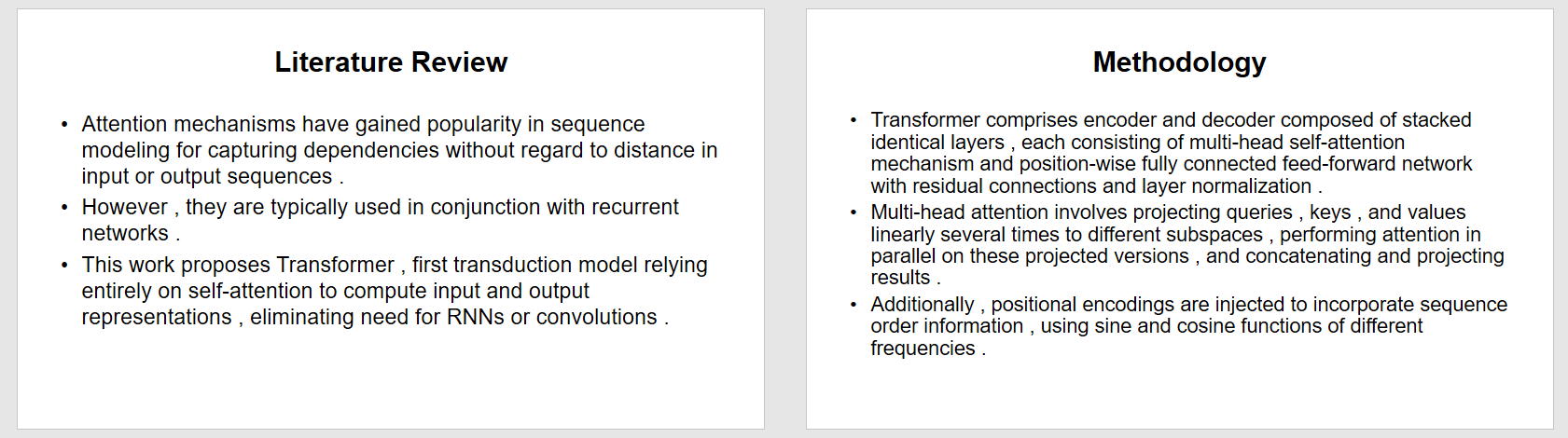
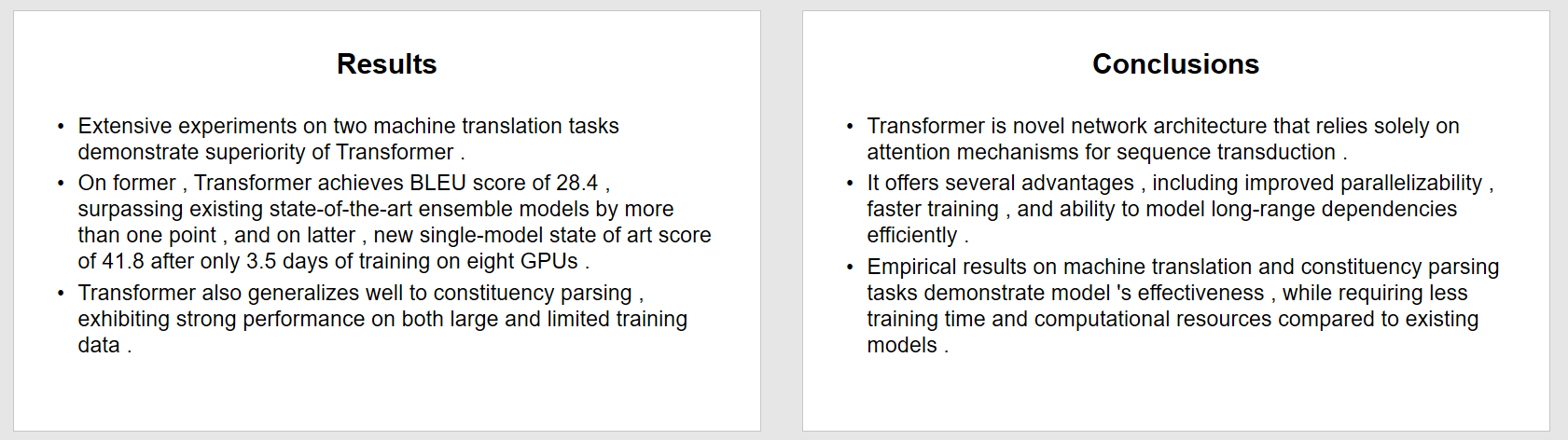


Figure 7‑1: Generated Slides



## Performance analysis

### Loss Curves

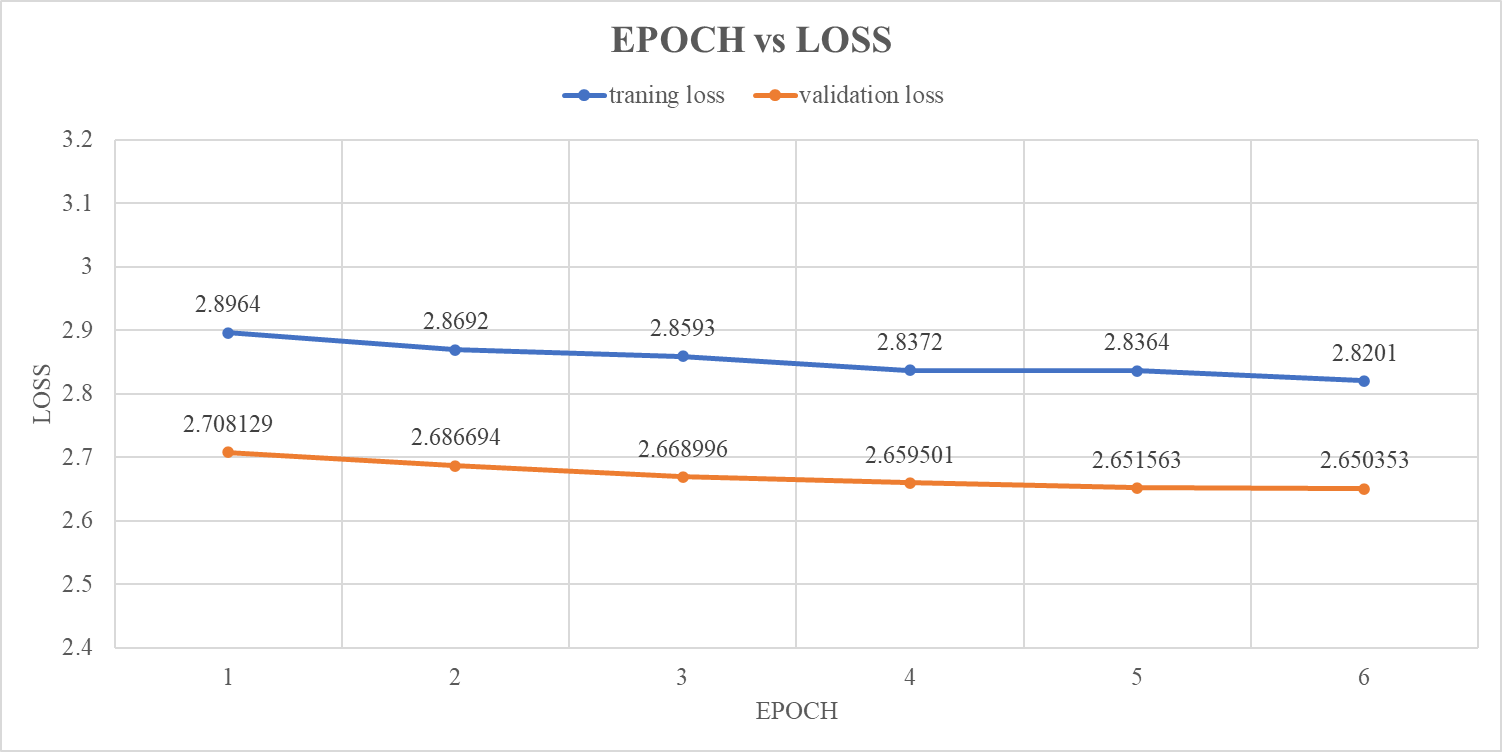


Figure 7‑2: Epoch vs Loss Graph (in 6 epochs)

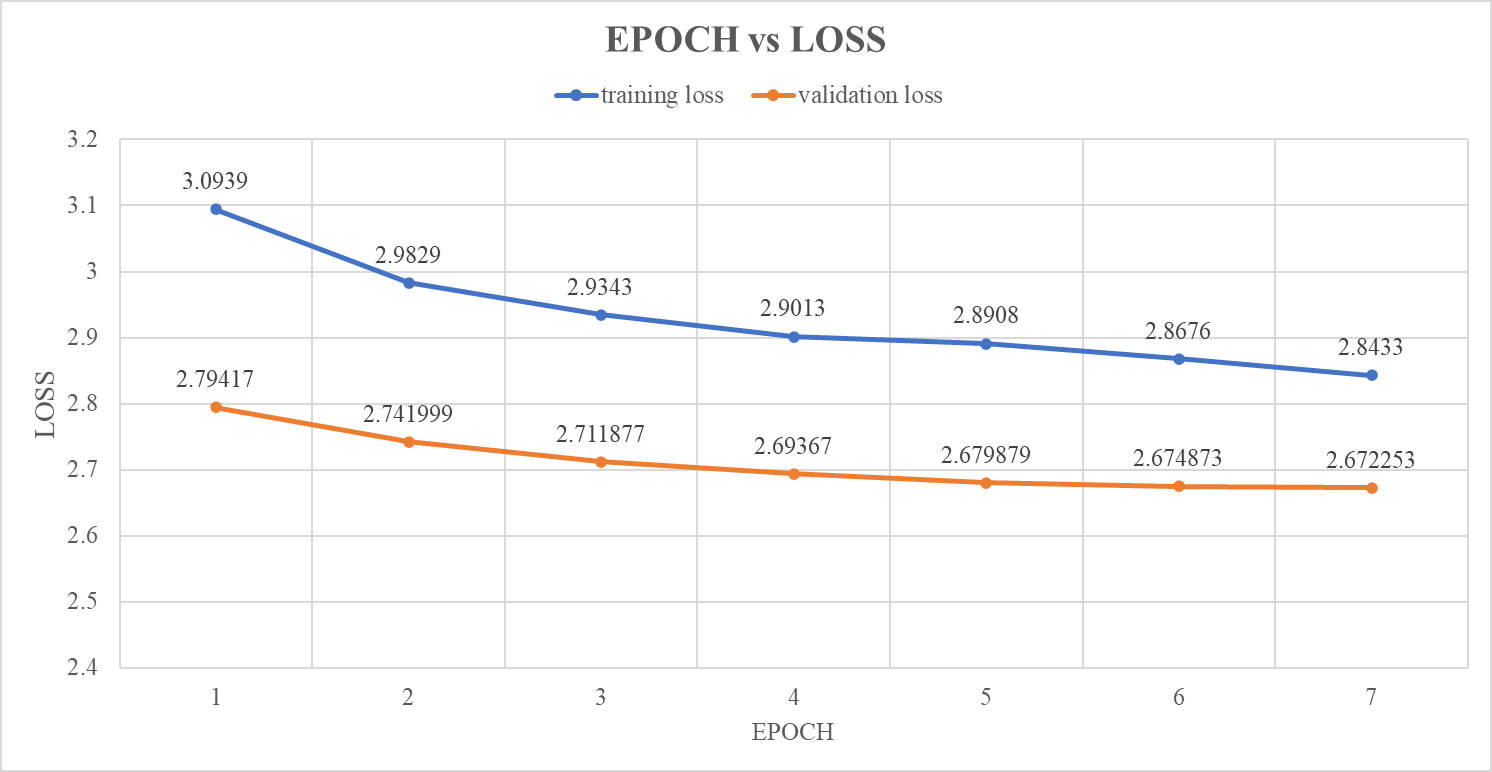


Figure 7‑3: Epoch vs Loss Graph(in 7 epochs)

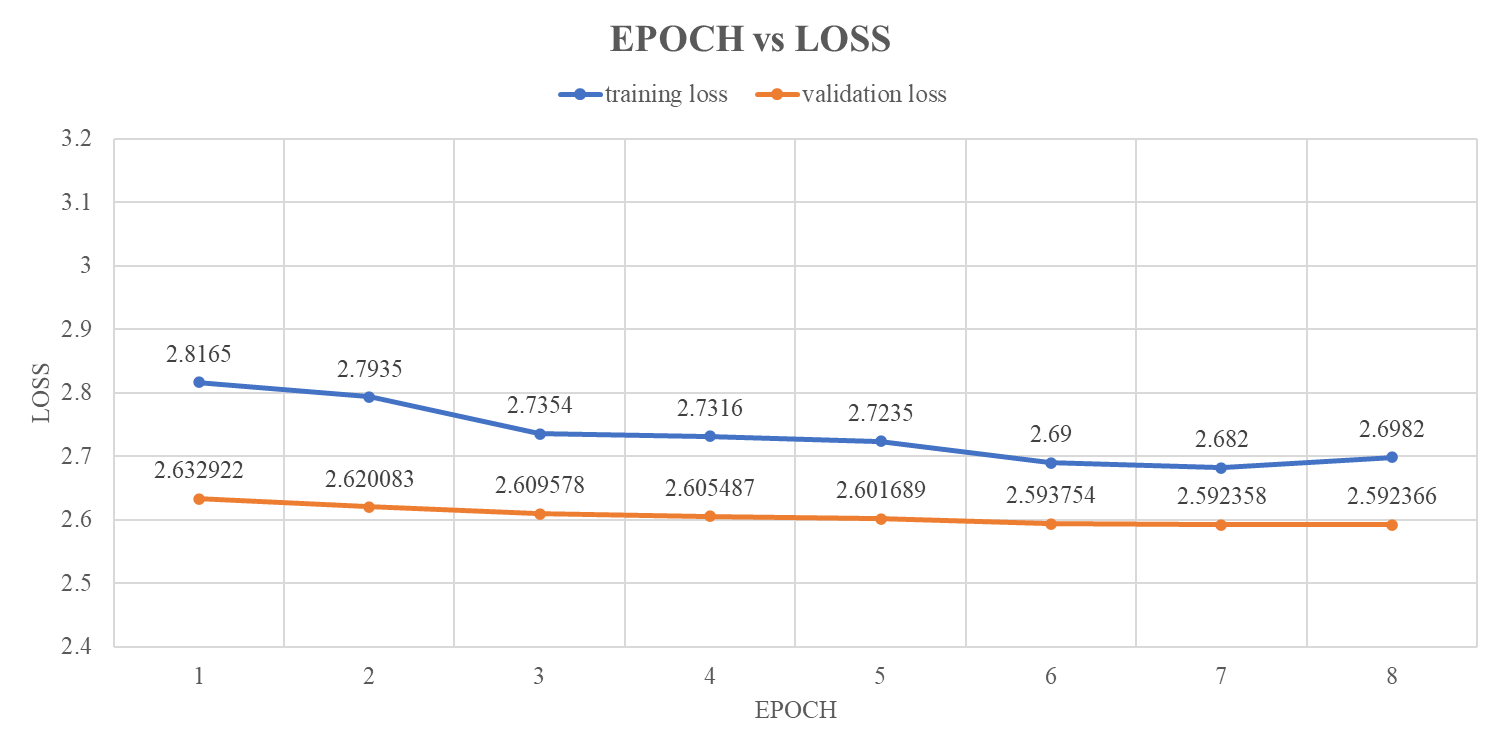


Figure 7‑4: Epoch vs Loss Graph (in 8 epochs)

From the above shown loss curves, we can see that the training loss and validation loss are slowly decreasing with the specific hyperparameters: learning rate 0.00005, batch size 8, weight decay 0.01. The training process involves 10 epochs and the models progress was saved every epoch, ensuring checkpoints for comparison. The most effective model was then selected from these checkpoints.

While training for 8th epoch the training loss decreases rapidly at first, but then it starts to level off and increase slightly at the end. This suggests that the model is starting to learn the noise in the training data. The validation loss, continues to decrease for a while, but then it starts to increase at the end. This is a strong sign of overfitting which is not desirable condition for our model.

### ROUGE Analysis

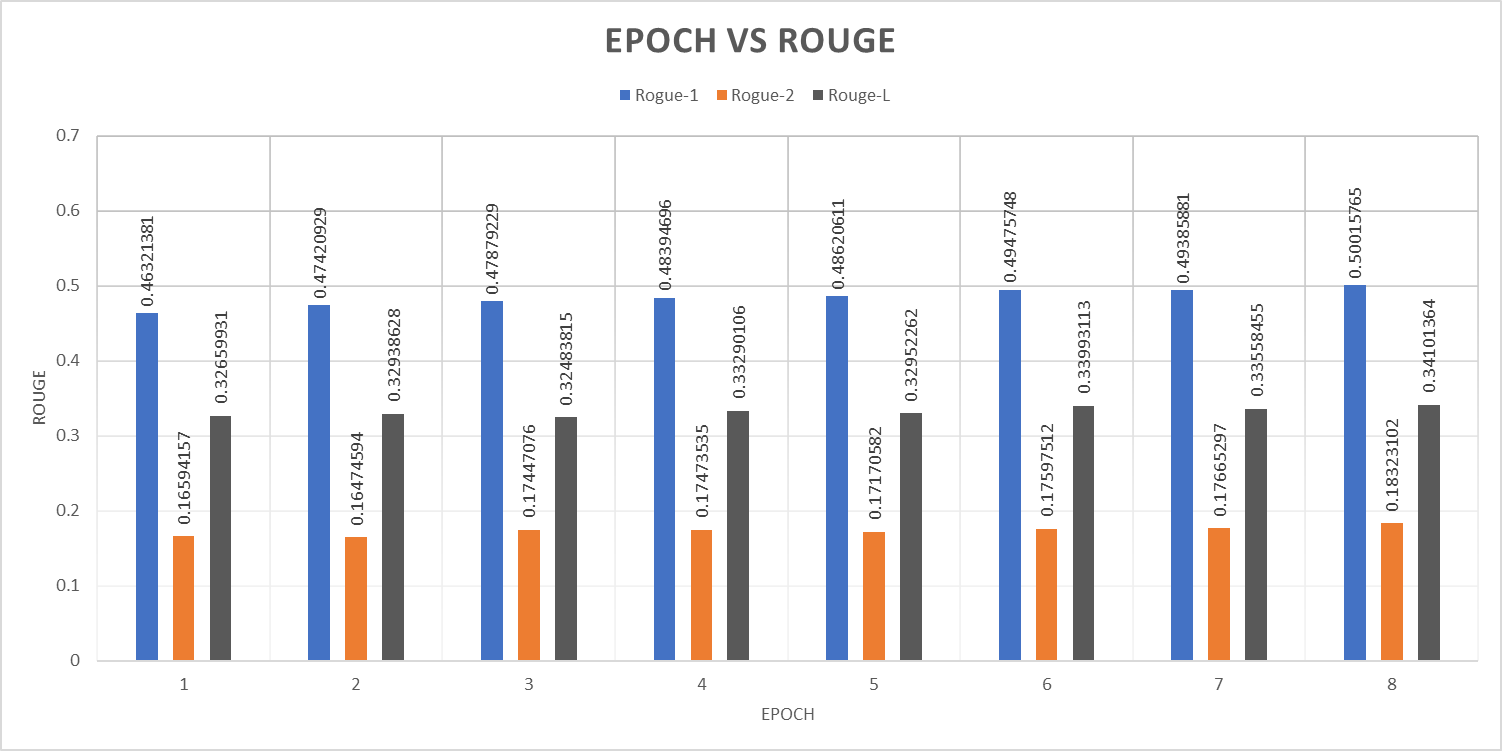


Figure 7‑5: Epoch vs Rouge

Upon assessing the ROUGE metrics, it becomes evident that the ROUGE-1, ROUGE-2, and ROUGE-L scores peak at the 8th epoch. However, despite achieving optimal performance at this stage, overfitting becomes apparent. Consequently, the model's performance at the 8th epoch is deemed unreliable and disregarded. Hence, 7 epochs of training were found to be most optimal.

# Remaining task



## Frontend Development

One of the remaining tasks involves the development of the frontend interface where user can upload the PDF or arXiv link of research paper. This frontend will serve as the user-facing component of our application, providing an intuitive and user-friendly platform for users to interact with.

## Customization by user for pptx formatting

Another important task on our agenda is to enable users to customize the formatting of the generated PowerPoint presentations (PPTX). This customization feature empowers users to tailor the appearance and layout of their presentation slides according to their preferences and specific requirements. By allowing users to adjust fonts, colors and styles, we aim to provide flexibility and customization options that cater to diverse presentation needs and styles.

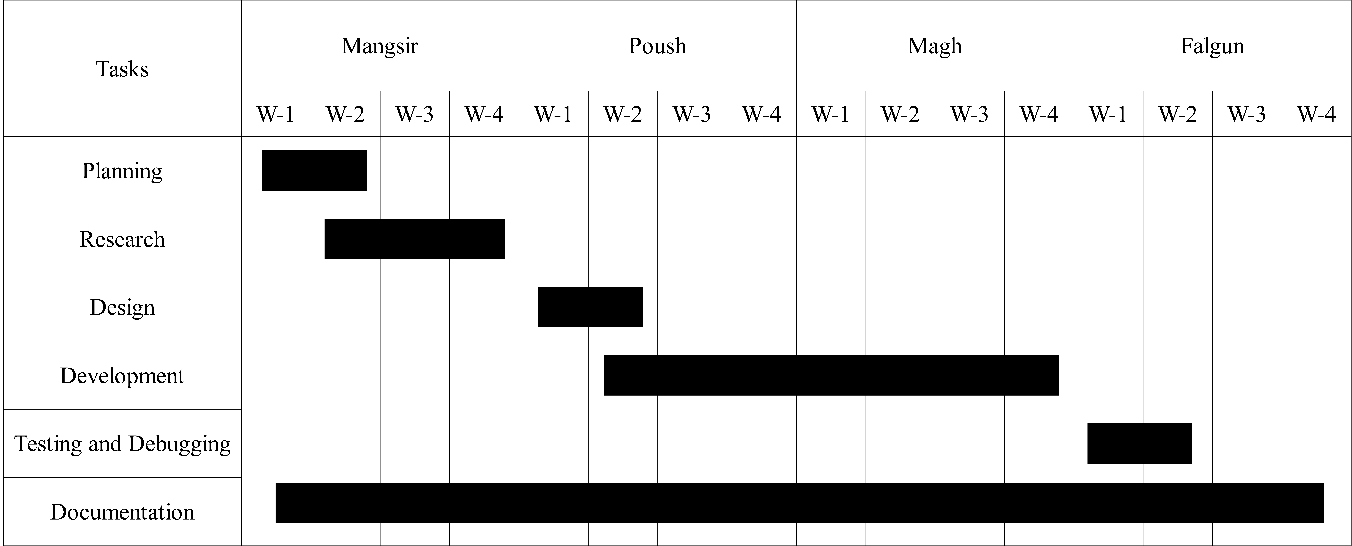
## Performance Tuning

Additionally, a key focus of our remaining tasks is to further enhance the accuracy of our summarization. This involves continuous refinement and optimization of the underlying machine learning models, fine-tuning parameters, and exploring advanced techniques to improve the quality and precision of the generated key insights. By prioritizing accuracy improvement efforts, we aim to deliver reliable and high-quality results that meet the expectations and needs of our users effectively.

# appendices

**Appendix A: PROJECT SCHEDULE**

Table 9‑1: Project Schedule



**Appendix B: PROJECT BUDGET**

Table 9‑2: Project Budget

|  |  |
| --- | --- |
| Task | Price |
| Tesla T4 GPU usage Cost | Rs.46.66/hour \*112.2hour = Rs. 5235 |
| Cloud Storage | Rs. 2.66 per GB per Month \*100GB\*2Months = Rs. 532 |
| Miscellaneous | Rs. 1000 |
| Total | Rs. 6767 |

**Appendix C: CODE SNIPPETS**

|  |
| --- |
| **For** i **in** range**(200):**  url**=**f'https://arxiv.org/search/advanced?advanced=&terms-0-operator=AND&terms-0-term=&terms-0-field=title&classification-computer\_science=y&classification-physics\_archives=all&classification-include\_cross\_list=include&date-filter\_by=all\_dates&date-year=&date-from\_date=&date-to\_date=&date-date\_type=submitted\_date&abstracts=show&size=50&order=-announced\_date\_first&start={str(num)}'  print**(**url**+**'\n\n\n\n\n'**)**  response **=** requests**.**get**(**url**)**  response **=** response**.**content  soup **=** BeautifulSoup**(**response**,**'html.parser'**)**  papers **=** soup**.**find\_all**(**'li'**,** class\_**=**'arxiv-result'**)**  **for** i **in** range**(**len**(**papers**)):**  title **=** papers**[**i**].**find**(**'p'**,**class\_**=**'title'**)**  authors **=** papers**[**i**].**find\_all**(**'p'**,**class\_**=**'authors'**)**  noob\_authors **=** **[]**  single **=** ''  **for** author **in** authors**:**  author\_names **=** author**.**find\_all**(**'a'**)**  **for** author\_name **in** author\_names**:**  noob\_authors**.**append**(**author\_name**.**text**)**  abstracts **=** papers**[**i**].**find**(**'span'**,**class\_**=**'abstract-full'**)**  single **=** ', '**.**join**(**noob\_authors**)**    pdf\_link **=** papers**[**i**].**find**(**'span'**)**  pdf\_link **=** pdf\_link**.**find**(**'a'**)**  article\_data **=** **{**'title'**:**''**,**'authors'**:**''**,**'abstract'**:**''**,**'pdf\_link'**:**''**,**'textdata'**:**''**}**  article\_data**[**'title'**]** **=** title**.**text**.**strip**()**  article\_data**[**'authors'**]** **=** single  article\_data**[**'abstract'**]** **=** abstracts**.**text  *# if article\_data['pdf\_link']:*  **try:**  article\_data**[**'pdf\_link'**]** **=** pdf\_link**[**'href'**]**  **except:**  article\_data**[**'pdf\_link'**]** **=** 'No link'    print**(**f"Article {count} extracted"**)**  count **+=1**  *# print(article\_data)*  all\_data**.**append**(**article\_data**)**    num**+=50** |

1. Link Extraction
2. Textdata Extraction

count **=** **0**

*# textdata = []*

**for** item **in** all\_data**:**

count**+=1**

*# if count == 100:*

*# break*

**try:**

pdf\_link **=** item**[**'pdf\_link'**]**

response **=** requests**.**get**(**pdf\_link**)**

response**.**raise\_for\_status**()**

pdf\_document **=** fitz**.**open**(**stream**=**response**.**content**,** filetype**=**"pdf"**)**

**except:**

**continue**

*# Initialize an empty string to store text*

text **=** ''

textdict **=** **{**'textData'**:**''**}**

*# Iterate over pages and extract text*

**try:**

**for** page\_num **in** range**(**pdf\_document**.**page\_count**):**

page **=** pdf\_document**[**page\_num**]**

text **+=** page**.**get\_text**()**

all\_data**[**count**-1][**'textdata'**]** **=** text

*# article\_data['textdata'][count-1]= text*

*# textdata.append(textdict)*

**print(**f"{count+2900}. Text Extracted from {item['title']}"**)**

*# print(text)*

*# Close the PDF file*

pdf\_document**.**close**()**

**except:**

**continue**

# References

|  |  |
| --- | --- |
| [1] | X. W. Yue Hu, "PPSGen: Learning-Based Presentation Slides Generation for Academic Papers," *IEEE,* vol. 27, no. 4, pp. 1085 - 1097, 2015. |
| [2] | D. A. P. Ektaa Meshram, "Technique for Generating Automatic Slides on the basis of Paper Structure Analysis," *International Journal of Innovative Research in Science,Engineering and Technology,* vol. 5, no. 6, pp. 10349-10356, 2016. |
| [3] | J. W. M. L. G. Athar Sefid, "Automatic Slide Generation for Scientific Papers," *CEUR Workshop Proceedings,* vol. 2526, pp. 23-30, 2017. |
| [4] | K. Shaj, " Learning Based Slide Generator," *International Journal of Engineering Research & Technology (IJERT),* vol. 9, no. 07, pp. 1076-1079, 2020. |
| [5] | W. Y. W. D. M. Y. S. Tsu-Jui Fu, "DOC2PPT: Automatic Presentation Slides Generation from Scientific Documents," *Proceedings of the AAAI Conference on Artificial Intelligence,* vol. 7, no. 36, pp. 14510-14518, 2022. |
| [6] | N. S. N. P. J. U. L. J. A. N. G. L. K. I. P. Ashish Vaswani, "Attention Is All You Need," *In Advances in Neural Information Processing Systems (NIPS),* pp. 5998-6008, 2017. |