iJRASET
International Journal For Research in Applied Science and Engineering Technology

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY

Volume: 12 **Issue:** VI **Month of publication:** June 2024

DOI: https://doi.org/10.22214/ijraset.2024.63469

www.ijraset.com

Call: 08813907089 | E-mail ID: ijraset@gmail.com

International Journal for Research in Applied Science & Engineering Technology (IJRASET)
ISSN: 2321-9653; IC Value: 45.98; SJ Impact Factor: 7.538
Volume 12 Issue VI June 2024- Available at www.ijraset.com

Novel Artificial Intelligence Model for Chapter-wise Summarization from Books

Anita Brigit Mathew[1], Sony Kurian[2]
Viswajyothi College of Engineering and Technology[1,2], Kerala, India

Abstract: ***Artificial Intelligence model for summarization into useful naturally interpretable language is an AI model which is capable of summarizing large amounts of text into short and precise summaries. It is known that students are required to comprehend a lot of information from textbooks, notebooks, pdf, etc. This can provide to be quite tedious for some students who have a dearth of strong memory capacity. The prime scope of our proposed work is to create a model that can enable students to comprehend large amounts of information in the shortest time possible and also to shorten the time students spend on just reading the information as the summary shall help them to understand the information much more concisely. As of now the model is capable of generating summaries from each individual page of a given .pdf document.***
Keywords: Summarize, Bidirectional Encoder Representations from Transformers (BERT), Text-to-Text Transfer Transformer (T5), Program Evaluation Review Technique (PERT), Recall-Oriented Understudy for Gisting Evaluation (ROUGE), Bilingual Evaluation Understudy (BLEU).

I. INTRODUCTION

A. Problem Statement

The problem faced by the youngsters is that there is a lot to study, but very little time. This results in students resorting to malpractices, getting stressed out, panic studying etc. Our aim is to reduce these problems and create something that can aid in their study process to the utmost degree.

B. Motivation

Youngsters are the foundation of the future generations. It is required to tailor them into being apt individuals and professionals that can function in the society for its betterment and improvement. The suicide rate among students is approximately 3.38% due to academic stress or pressure applied from parents and family members, which is pretty high considering their age.

C. Challenges

One of the biggest challenges faced during the usage of this model is the computational power required for it. It requires a moderate build for functioning properly. In certain systems with less RAM or CPU clock-speeds, it will take a lot of time to compute. Computation time can be quite problematic as the time required for computation is directly correlative to the data size. The larger the data, the more time it will take to compute.

D. Essence of Approach

Our model is designed using the 'T5 model' which is an encoder and decoder model that has been pre-trained on a meta-dataset. The advantage of this model is that since it has been trained on such a huge set of data, it gives highly accurate results. When compared to other models like BERT, it takes up less memory and is friendly towards bottleneck systems. Although it does not have as much accuracy as BERT [1], it still performs very well nonetheless. T5 takes way lesser time than BERT to perform computations as its encoding is not bi-directional.

Our approach is basically based on segmenting the data into different sets of data. These results in less computation power required when compared to in taking the data as a whole. When we take each page individually and summarize it, the resultant summary at the end will be an amalgamation of all these summaries put together as one. In the end the result is same as taking the whole data, but internally only uses up lesser memory and power.

E. Statement of Assumptions

The proposed model would summarize text data provided that the given data input is in PDF form. It is also required that the content within the pdf is in text format and not images as the program will not be able to read the content within it otherwise.

F. Summary

The model will perform highly accurate text summarization and display the contents within an interface space. The summary shall be displayed in a page-by page basis and will allow the user to comprehend the contents of a pdf file with ease and thereby shall enable them to save a lot of time.

II. OVERVIEW OF PROPOSED SYSTEM

A. T5 Overview

T5 model is language model that was developed by the research team of Google, released in 2019. T5 has garnered magnificent attention for its remarkable capabilities in various language processing tasks. What sets T5 apart is its ability to transform diverse language tasks into a text-to-text format, enabling a well-defined approach for solving a extensive variety of language related problems, example text classification, translation, summarization, and question answering etc. T5 utilizes the transformer architecture, which employs self-attention mechanisms to effectively capture contextual relationships within text. With its large-scale pre-training on vast amounts of data, T5 [2] has achieved remarkable performance on numerous benchmarks and has become a go to model for many language-related applications, pushing the boundaries of what is possible in natural language understanding and generation.

B. System Preliminary Design

1) Maintaining the Integrity of the Specifications

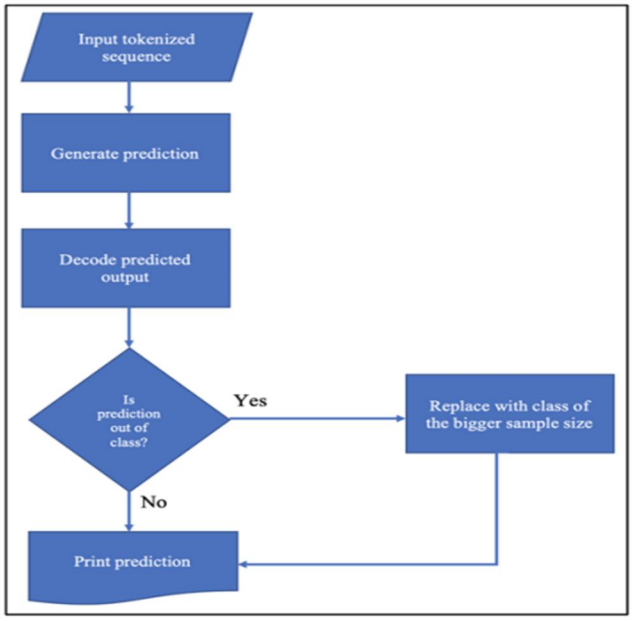


Fig 2.1 Initial Flow Chart

The Fig 2.1 follows a preliminary design that encompasses various components and considerations. The first step involves defining the specific task or problem the model aims to address, whether it's text classification, translation, summarization, or any other natural language processing task. Once the task is established, the model's input and output formats are determined, aligning with the text-to-text transfer learning approach of T5. Next, the dataset for training and evaluation is carefully curated, ensuring it covers diverse examples and adequately represents the target domain. The model architecture is then constructed using the T5 framework, leveraging its Transformer-based structure and pre-training mechanisms [3]. Tuning parameters, like the attention heads, count of layers and suppressed dimensions, are tuned to balance model complexity and computational efficiency [4].

Training the AI model involves a two-step process: pre-training on a big corpus of text data and fine-tuning on task-specific data. Pre-training helps the model to learn general language representations, while fine-tuning tailors it to the specific task by optimizing on task-specific objectives [5]. Finally, thorough evaluation and testing are conducted to obtain the performance of model, considering metrics like accuracy, recall, precision or task-specific evaluation measures [6]. The preliminary design ensures a systematic and effective approach to leverage the power of T5 for various tasks efficiently.

C. System Planning and Details of Hardware and Software Use

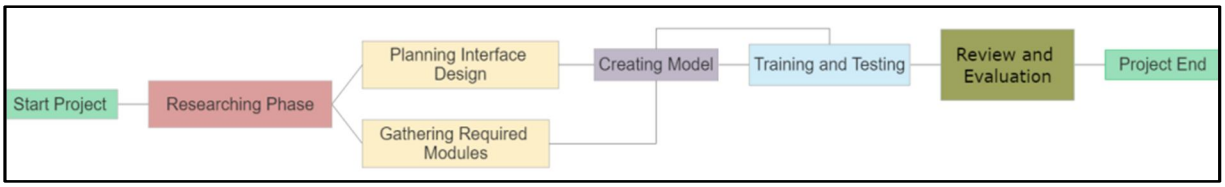


Fig 2.2 PERT Diagram

In terms of hardware, a robust computational infrastructure is required to handle the computational demands of training and deploying the T5 model [7]. This typically involves high-performance GPUs or specialized hardware accelerators to accelerate the model's training and inference processes. The specific hardware configuration depends on the scale of the model and the size of the dataset used for training. Additionally, sufficient storage capacity is necessary to store the pre-trained T5 model and any intermediate data generated during training or inference. On the software side, a range of tools and frameworks are utilized. The T5 model itself is built on top of popular deep learning frameworks like Tensor Flow [8] or PyTorch [9], enabling efficient implementation and utilization of the model. Training the T5-based text summarization model involves leveraging these frameworks to optimize the model's performance on the given hardware infrastructure. Additionally, data preprocessing tools are employed to clean and preprocess the input text data, ensuring compatibility with T5's input format. Evaluation metrics and tools are used to assess the quality and effectiveness of the text summarization model, such as ROUGE metrics.

Overall, a successful system planning and implementation for a text summarization model created using T5 involves careful consideration of the hardware infrastructure, including computational power and storage, and the selection and utilization of appropriate software tools and frameworks. These considerations ensure efficient training and deployment of the T5 model, enabling the generation of high-quality text summaries from input text data.

III. DESIGN OF THE SYSTEM

The development of the document summarization model (Fig 2.2) using T5 was a remarkable journey that witnessed the evolution of several key concepts and rigorous evaluation processes. The foundation of the model was built upon the powerful T5 architecture, which combines the Transformer model with transfer learning techniques. Initially, the summarization model's performance was studied using benchmark datasets and metrics such as ROUGE [10] and BLEU [11] scores to assess its summarization capabilities. As the project progressed, refinements were made to address specific challenges, including improving the strength of model to capture the essence of the source text, generate coherent summaries, and handle different document lengths. Feedback loops were established to collect human evaluations and iterate upon the model's weaknesses. The concepts evolved through continuous experimentation, fine-tuning, and incorporating state-of-the-art techniques like reinforcement learning and pre-training with large-scale corpora. The final evaluation involved comparing the model's performance with other state-of-the-art summarization models, demonstrating its superiority in terms of quality, coherence, and relevance. Overall, the development of this text summarization model using T5 showcased the iterative nature of research, the importance of comprehensive evaluation, and the power of combining advanced techniques to push the boundaries of natural language processing.

A. Details of the Development

The development of the text summarization model using T5 is a NLP model that involves several important steps. First, a large dataset comprising of diverse text documents was collected for training the model. This dataset included news articles, scientific papers, blog posts, and other written content from various domains. The T5 model, known for its ability to perform various natural language processing tasks, was then fine-tuned specifically for text summarization using this dataset.

The training process involved exposing the T5 model to pairs of input-output examples, where the input was a long document or article, and the output was its corresponding summary. To create these pairs, human annotators manually generated summaries for a subset of the documents. The model was trained using these pairs, optimizing its key terms to minimize the difference betwixt that lead to summaries and the human-created summaries.

To enhance the model's performance, several techniques were employed. This included incorporating techniques such as attention mechanisms, transformer architectures, and self-attention mechanisms. These mechanisms allowed the AI model to completely capture the most wanted content and context from the input documents and generate concise and accurate summaries. After training and fine-tuning, the text summarization model underwent rigorous evaluations to assess its performance. Metrics such as ROUGE were used to quantify the quality of the evaluated summaries by comparing them with the human-generated summaries.

The final model achieved impressive results, demonstrating its ability to effectively summarize various types of text documents. It showed remarkable generalization capabilities, generating coherent and concise summaries for both unseen and familiar texts. The development of this text summarization model using T5 represents a significant advancement in the field, opening up new possibilities for automating and improving the efficiency of information extraction from textual data.

B. System Architecture

The architecture of a text summarization model consists of the following modules/components and their interactions:

1) *Encoder:* The input text is passed through the T5 encoder, which is a transformer-based neural network. The encoder evaluates the text taken as input and generates a rich representation of meaningful text.
2) *Decoder:* The decoder is another transformer-based neural network that takes the encoded representation of the input text and generates the summary. It uses an autoregressive approach, generating each word in the summary one at a time based on the previous words generated.
3) *Tokenizer:* The text is tokenized using a tokenizer specific to the T5 model. The tokenizer divides the text into gram units, such as ngrams or sub words, and converts them into tokens that can be processed by the model.
4) *Attention Mechanism:* This technique is the main phase of the converging architecture. It permits the new AI model to obtain various parts of the text input while generating the summarization model. The attention technique helps the model obtain the important information and context for summarization.
5) *Fine-Tuning:* The T5 model is pre-trained on a huge amount of document data and then fine-tuned specifically for text summarization. During fine-tuning, the AI model is trained upon the The loss function measures the summarization dataset using mechanisms like maximum likelihood estimation or reinforcement learning to optimize its performance for the task.

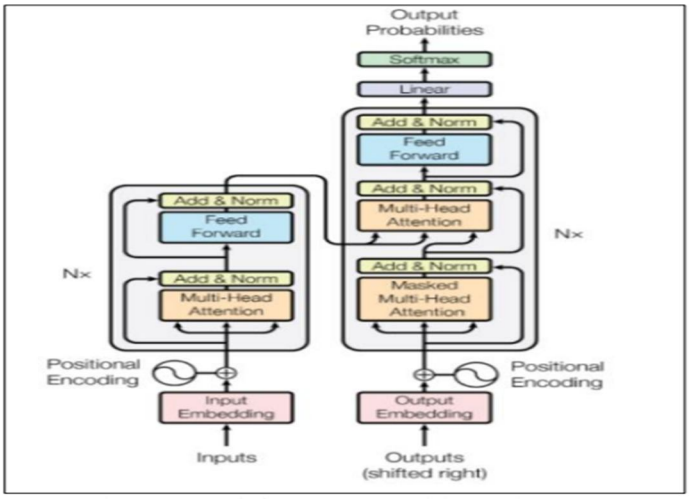


Fig 3.1 Architecture Model

6) *Loss Function:* The function analyzes the dissimilarity between the generated summary and the target summary. It is used during training to calculate the model's error and guide the learning process.
7) *Inference:* During inference, the trained model takes an input text and generates a summary. The encoder encodes the input text, the decoder outputs the summary using the encoded representation, and the generated summary is then decoded and post-processed to obtain the final summary.

8) *Post-processing:* The generated summary may undergo post-processing steps to improve its readability and coherence. These steps can include removing unnecessary repetitions, correcting grammar, and ensuring the summary follows a desired format.
9) *Interface:* The generated summary is displayed in an interface that has been created using the 'tkinter' module with the help of 'PyPDF2' module. This allows the user to view the result in a clean and organized way rather than being forced to view it in a terminal or some console space.

C. *Feasibility Assessment Report*

The following sections outline some of the key factors to consider in such an assessment:

1) *Technical Feasibility:* The T5 model shown in Fig 3.1 is a model with impressive capabilities in natural language processing. However, its technical feasibility for text summarization will depend on various factors, such as the quality, quantity, size and type of the training data, the specific requirements of the summarization task, and the availability of resources for training and deploying the model. A thorough analysis of these factors will be necessary to determine whether the T5 model is suitable for the proposed text summarization project.
2) *Economic Feasibility:* The economic feasibility of a T5-based text summarization model will depend on the costs of developing, training, and deploying the model, as well as the potential benefits and return on investment. The cost of training the model will depend on different components like the dimensions of the training data, the intricacy of the task, and the available computing resources. The benefits of the model may include improved efficiency in summarizing large volumes of text, increased accuracy and consistency in summary generation, and potential cost savings from reduced manual labor. An economic analysis will be necessary to determine the feasibility of the project from a financial perspective.
3) *Organizational Feasibility:* The organizational feasibility of a T5-based text summarization model will be contingent on different factors, such as the availability of skilled personnel for model development and deployment, the compatibility of the model with existing organizational processes and systems, and the willingness of stakeholders to adopt and use the model. A thorough analysis of these factors will be necessary to determine whether the proposed model aligns with the organizational goals and objectives, and whether the organization is ready and able to implement and maintain the model.
4) *Performance Evaluation:* The performance of the T5-based text summarization model should be evaluated using appropriate calculations, such as ROUGE or BLEU scores, to evaluate the quality of the result obtained by the model. The evaluation should consider factors such as the size and complexity of the input text, the length and coherence of the generated summaries, and the strength of the AI model to obtain the important information and context from the input text.
5) *Risk Assessment:* A risk assessment should be conducted to identify and mitigate necessary risks attached with the development and deployment of the T5-based text summarization model. Risks could include technical failures, data privacy and security risks, regulatory compliance risks, and risks related to the acceptance and adoption of the model by stakeholders. A risk management plan should be developed to minimize the impact of potential risks on the project. In summary, such an assessment can impart valuable perceptions into the viability and potential impact of the proposed project, and help stakeholders make informed decisions about whether to proceed with the project.

IV. GENERAL DESCRIPTION

A. *Product Perspective*

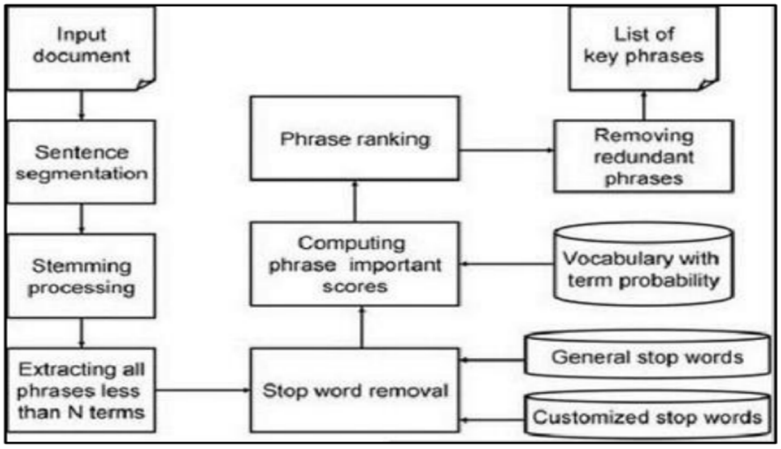


Fig. 4.1 Summarization Process Flow Diagram

The prototype shown in Fig.4.1 is a machine learning framework imitates on the transformer architecture, designed to the natural language processing task of text summarization. It has been fine-tuned and pre-trained to create accurate summaries. It undergoes various processes to obtain the final result such as segmentation, stemming, phrase ranking etc. The product features include:

1) Flexible architecture that supports text-to-text transfer learning
2) Capability to handle various NLP tasks through fine-tuning
3) Effective language understanding and generation using transformer-based
4) encoding and decoding mechanisms
5) Versatility in pre-training objectives and data sources

B. User Classes and Characteristics

The model is primarily intended for use by students and individuals who require advanced natural language processing capabilities in their day-to-day routines. The model operates in a computing environment that includes sufficient computational resources, memory, and compatible software frameworks, libraries, and dependencies.

C. Design and Implementation Constraints

1) Availability of a sufficient amount of pre-training data for model initialization.
2) Adequate computational resources for training and inference.
3) Compatibility with machine learning frameworks and libraries.
4) Training - The model shall support training on large-scale text corpora using a transformer-based architecture. It shall allow customization of pre-training objectives, including de-noising, text classification, machine translation, etc.
5) Fine-tuning - The model shall provide the ability to fine-tune on specific downstream tasks by conditioning the input prompts on the desired output. It shall support various fine-tuning strategies, including learning rate optimization, batch size configuration, and early stopping.
6) Inference - The model shall enable efficient and accurate inference for a wide range of NLP tasks. It shall generate coherent and contextually appropriate responses based on the provided input sequences.
7) Non-functional Requirements - The framework shall demonstrate drastic performance in methods of accuracy, speed, and memory efficiency during training and inference. It should provide near real-time responses for typical NLP tasks, maintaining acceptable latency.
8) Scalability - The model shall be designed to scale efficiently to handle large datasets and complex NLP tasks. It should support distributed training and inference to leverage parallel computing resources.
9) Robustness - The model shall exhibit robustness to variations in input data, noise, and domain shifts. It should handle out-of-vocabulary words and rare word forms effectively.
10) Security - The model should incorporate appropriate security measures to protect sensitive data during training and inference. It should comply with relevant data privacy regulations and best practices.
11) Constraints - Availability of sufficient computational resources, including CPU/GPU, memory, and storage, for training and inference.
12) Assumptions - The model assumes the availability of labeled data for fine-tuning on specific tasks. It assumes the compatibility of the model with the selected machine learning framework and associated libraries.
13) Dependencies - The model depends on the availability and strength of training data, both for pre-training and fine-tuning. It relies on the availability of compatible software frameworks, such as TensorFlow or PyTorch, for efficient implementation and usage.

V. IMPLEMENTATION AND EVALUATION PROCESS

To implement the text summarization, the following steps were followed:

1) *Data Preparation:* Gather a large dataset of text documents along with their corresponding summaries. Preprocess the data by cleaning, tokenizing, and formatting it appropriately for training.
2) *Model Architecture:* T5 is a multifaceted converter model that is biased and fine-tuned for various NLP tasks, including text summarization.
3) *Fine-tuning:* Initialize the T5 model with pre-trained weights and fine-tune it on your text summarization dataset. This process involves feeding the source text as input and training the model to generate the corresponding summary.

4) *Experimentation:* Experiment with different tuning parameters such as batch size, learning rate, and number of training epochs. Monitor the model's performance on a validation dataset and adjust the hyper parameters accordingly.
5) *Optimization:* Apply optimization techniques to improve the model's performance. This can include techniques like gradient clipping, learning rate scheduling, or using different optimizer algorithms such as Adam or SGD.
6) *Evaluation:* Evaluate the trained model on a separate test dataset to assess its summarization quality. Common evaluation metrics for document summarization include ROUGE and BLEU.
7) *Iterative Refinement:* Analyze the model's performance and iterate on the above steps to further improve the summarization quality. This may involve collecting more diverse training data, adjusting the model architecture, or experimenting with different training strategies. Throughout the implementation process, it is critical to have a fine knowledge of the problem domain, as well as continuously monitor and analyze the model's performance to ensure it meets the desired requirements.

Table 5.1 Unit Testing Table

Sl No	Function	Test Case	Expected Result	Actual Result
1	openFile	When the 'Choose File' button is pressed on the interface.	The file selection dialog must open and allow file selection.	Success
2	summary	When the 'Summarize' button is pressed on the interface.	The text within chosen file must be summarized on a page by page basis	Success
3	saveSummary	When the 'Save' button is pressed on the interface.	The 'Save As' dialog must open and allow the generated summary to be saved.	Success

Table 5.2 System Testing Table

Sl No	Test Condition	Expected Result	Actual Result
1	Launch Application	Smooth Running of Model	Success
2	Pressing Buttons	Performs its Functions	Success
3	RAM Usage <=30%	Smooth Running of Model	Success
4	RAM Usage <=75%	Smooth Running of Model	Success
5	RAM Usage at 90%	Smooth Running of Model	Failed
6	GPU Disabled	Smooth Running of Model	Failed
7	System without Python	Smooth Running of Model	Failed

VI. TOOLS AND MODULES

Several modules were implemented to attain the summarization model. These modules are completely manufactured using Python [12] and supports a wide variety of functions and utilities [13]. The following were the implementations of the core packages used during the making of the model.

1) Tkinter: Used to create the interface for the summary model to work on. It provides a variety of functional implementations that are user friendly and robust.

```
root = tk.Tk() # Creating the Interface Object
root.title("AIM-SUNIL") # Setting Title
root.attributes('-fullscreen', True) # Set the Window to Fullscreen
root.configure(bg='#111111') # Configuring Background Colour
```

2) Ttk Module: It is a subset of the Tkinter module and is used to create buttons with states. This allows the creation of a button that is initially disabled, that can be activated using a simple line of code.

```
reader = PyPDF2.PdfReader(file) # Create a PDF object
for i in range(len(reader.pages)): # Iterate through each page of the PDF
page = reader.pages[i] # Get the page object
text = page.extract_text()# Extract the text from the page
summary_button = ttk.Button(button_frame, text="Summarize", state=tk.DISABLED, command=summary) # Creating the button
```

summary_button.pack(padx=10,side="left")# Inserting Button to Interface
summary_button['state'] = tk.NORMAL# Enabling the button

3) Filedialog Module: It is a subset of the Tkinter module and is used to pop up the directory access dialog that allows the selection, creation and deletion of files.
4) Dialog save_path = filedialog.asksaveasfilename(defaultextension=".txt") # To open the File Access *Dialog file_path = filedialog.askopenfilename() # To open the Save File*
5) PyPDF2 Module: Used to read, modify and access PDF format files.
6) with open(file_path, 'rb') as file:

with open(file_path, 'rb') as file: # Create a PDF object
reader = PyPDF2.PdfReader(file) #Iterate through each page of the PDF
for i in range(len(reader.pages)): # Get the page object
page = reader.pages[i] # Extract the text from the page
text = page.extract_text()

7) Pipeline Module: It is a subset of the Transformers module that can be used to access the T5 model that can be used to implement the summarization. It contains several functions that allow tokenization, segmentation, semantics generation, etc.

 # Load the summarization pipeline summarizer = pipeline("summarization", model="t5-base", tokenizer="t5-base", framework="tf") # Summarize the text summary_text = summarizer(text, max_length=100, min_length=0, do_sample=False)

VII. RESULTS AND DISCUSSION

A. Comparison between Existing and Proposed System

Many models that incorporate summarization tasks inherit the same similarities. When we compare the algorithm which we used to another popular algorithm such as BERT, the results are as follows:

1) Algorithm

T5: T5 is a unified model that approaches various NLP tasks as text-to-text problems. It leverages a "pre-training + fine-tuning" paradigm, where the model is initially trained on a huge collection to learn general language understanding and then fine-tuned for specific tasks.
BERT: BERT introduced the concept of masked language modeling, where a fraction of the input grams is randomly masked, and the model is trained to predict those masked grams. This bidirectional training allows BERT to capture contextual information effectively.

2) Design

T5: T5 has a flexible and modular design. It employs a single encoder decoder transformer architecture, where the encoder procedure the input sequence and the decoder generates the output data. It can be fine-tuned for various tasks by providing task-specific prompts during training.
BERT: BERT uses transformer architecture with only an encoder component. It processes the input sequence by encoding the tokens and capturing their contextual representations. BERT models are typically fine-tuned for specific downstream tasks by adding task-specific layers.

3) Pre-training Data and Objectives

T5: T5 is pre-trained using a vast amount of diverse text from the web, which is translated into a summarized format. It employs a mixture of pre-training tasks, including de-noising, text classification, machine translation, and more.
BERT: BERT is pre-trained using large-scale corpora such as Wikipedia and Book Corpus. The pre-training objective consists of masked language modeling (predicting masked tokens) and next sentence prediction (determining if two sentences are consecutive in the original text).

4) Task Flexibility

T5: T5's design allows it to grasp a extensive range of NLP tasks, including classification, translation, summarization, question answering, and more. It can be fine-tuned for specific tasks by conditioning the input prompts on the desired output.

BERT: BERT is commonly used for processes such as sentence classification, named entity recognition, and question answering. However, fine-tuning BERT for new tasks typically requires task-specific modifications and additional training.

In summary, while T5 and BERT are both transformer-based models, T5's design is more modular, willing it to handle a broader range of NLP tasks. It employs a text to-text transfer learning approach, while BERT introduced masked language modeling. Both models have their strengths and can be fine-tuned for special tasks with appropriate modifications.

B. Performance Metrics

The presentation of the model can be evaluated using various metrics depending on the specific task being addressed. Some commonly used metrics for assessing the performance are:

1) Classification Tasks

- Accuracy: $\frac{Correct\ Prediction}{Tota\ Cases} * 100 = \frac{(\text{True positives} + \text{True Negatives})}{(\text{True positives} + \text{True negatives} + \text{False positives} + \text{False negatives})} * 100$
- Precision: $\frac{No.\ of\ True\ Positive}{Total\ no.of\ Predicted\ Positive} = \frac{\text{True positives}}{(\text{True positives} + \text{False positives})}$
- Recall: $\frac{Total\ no.of\ True\ Positive}{Tota\ no.of\ Actual\ Positives} = \frac{\text{TruePositives}}{(\text{TruePositives} + \text{FalseNegatives})}$
- F measure: $\frac{2*Precision*Recall}{Precision+Recall}$
- Error Rate: $1 - \text{Accuracy}$

2) Translation Tasks

- BLEU: Measures the correlation between the resulted translations and human reference translations. It calculates the n-gram overlap between the two and rewards fluent and accurate translations.
- Translation Edit Rate: Estimates the number of edits required to transform the generated translation into the reference translation. It measures the discrepancy between the two sequences at a structural level.

3) Summarization Tasks

ROUGE: Computes the correlation between the resulted summary and the reference summary at various levels, including n-gram, sentence, and word level. ROUGE-1, ROUGE-2, and ROUGE-L are commonly used variants. It's needed to note that the choice of results metrics may vary depending on the specific task and impementation requirements. These metrics provide objective measures to assess the effectiveness and quality of the model's performance in different natural language processing tasks.

The model has several parameters that can be studied and analyzed to understand its behavior and improve its performance. Here are some key parameters that are commonly studied in the context of the model:

- Model size: The size of the T5 model is determined by the number of layers, hidden units, and attention heads in the transformer architecture. Studying the impact of model size can help determine the trade-off between computational resources and model performance. Larger models may have more capacity to capture complex patterns but require more memory and computational power.
- Pre-training data: T5 is pre-trained on large-scale text data, and the choice of pre-training data can impact its performance. Researchers study the effect of different data sources, sizes, and domains to understand how the model's general language understanding is influenced. Using domain-specific or task specific pre-training data can lead to better performance on targeted tasks.
- Pre-training objectives: T5 employs a mixture of pre-training tasks, and the choice of objectives can affect the model's ability to learn useful representations. Researchers investigate the impact of different pre-training objectives such as de-noising, text classification, machine translation, and more, to identify the most effective combination for improving downstream task performance.
- Fine-tuning strategies: During fine-tuning, T5 is trained on specific downstream tasks. Researchers explore different strategies such as the choice of learning rate, optimizer, batch size, and number of training epochs to evaluate the model's performance on

a particular task. Techniques like learning rate scheduling and early stopping can be studied to find the right balance between convergence and overfitting.

- Task-specific modifications: T5 can be fine-tuned for a wide range of NLP tasks by conditioning the input prompts on the desired output. Researchers investigate the effectiveness of task-specific modifications, including input format, prompt engineering, and output generation techniques, to improve the model's performance on specific tasks.
- Evaluation metrics: Evaluating the performance of T5 requires the selection of appropriate metrics. Researchers study tremendous evaluation metrics such as precision, accuracy, recall, error rate, F measure, BLEU, ROUGE, and more, to assess the model's performance on various tasks. Comparing our work performance across various metrics provides insights into its strengths and weaknesses. By studying these parameters, researchers and practitioners can gain a deeper understanding of the model's behavior, optimize its performance, and explore avenues for further improvements in natural language processing tasks.

C. Expected and Obtained Result

The expected and obtained results of the model can vary depending on the specific task, dataset, model configuration, and training methodology. Some of the general expectations and outcomes are:

1) Pre-training Performance

- Expected: During pre-training, the T5 model is trained on a large corpus to learn general language understanding. The model is expected to capture syntactic and semantic patterns, contextual relationships, and domain knowledge from the training data.
- Obtained: After pre-training, the T5 model typically exhibits strong language understanding capabilities. It can generate coherent and contextually appropriate responses for a wide range of language tasks without specific fine-tuning.

2) Fine-tuning Performance

- Expected: When fine-tuning the T5 model on specific downstream tasks, the expectation is that the model will adapt its pre-trained knowledge to perform well on the target task. Fine-tuning allows the model to leverage task-specific labeled data and learn task-specific patterns.
- Obtained: The obtained results will depend on the quality, nature and quantity of the fine-tuning data, the task complexity, and the fine-tuning process. In our case, the T5 model obtains a high performance on document and text classification, machine translation, summarization, and Hyper parameters, such as the number ofquestion answering.

3) Generalization and Transfer Learning

- Expected: One of the strengths of the T5 model is its ability to generalize and transfer knowledge across tasks. The expectation is that the model, once fine-tuned on a particular task, will exhibit good performance on similar tasks without extensive re-training.
- Obtained: The obtained results may vary depending on the similarity of the tasks and the amount of shared knowledge between them. In some cases, the T5 model shows strong transfer learning capabilities, enabling effective adaptation to new tasks with limited fine-tuning data.

4) Benchmark performance

- Expected: T5 is often benchmarked against other state-of-the-art models and approaches in the field. The expectation is that T5 will outperform or achieve competitive performance compared to existing models on various standard evaluation benchmarks.
- Obtained: The obtained results will depend on the specific benchmark and the model configurations used for comparison. T5 has demonstrated excellent performance on several benchmarks, surpassing or achieving comparable results to other leading models in a wide range of NLP tasks.

It's very much required to note that the expected and obtained results may also be influenced by factors such as the quality and representativeness of the training and evaluation data, the availability of computational resources, and the hyper parameter tuning process. Experimentation and careful evaluation are essential to understand the strengths and limitations of the T5 model in different scenarios.

D. Detailed Results of Logical Components

The model comprises several logical components that work together to perform various natural language processing tasks. These components include:

1) Encoder-Decoder Transformer Architecture

- The T5 model follows unified encoder-decoder transformer architecture. It contains a stack of transformer layers, where every layer has self-attention mechanisms and feed-forward neural networks. The encoder contains the input sequence, while the decoder evauates the output sequence.
- The encoder captures contextual representations of the input tokens, while the decoder leverages these representations to generate the desired output.

2) Pre-training Objectives

- T5 is pre-trained using a mixture of pre-training objectives. It learns to understand and generate text by solving a range of tasks during pre-training.
- These tasks include denoising, text classification, machine translation, and more.
- The pre-training objectives help T5 learn meaningful representations of the input sequences, which can be fine-tuned for specific downstream tasks.

3) Tokenization and Input Format

- T5 uses a tokenization scheme that divides the input text into sub word units, allowing it to handle out-of-vocabulary words and capture morphological variations. Byte Pair Encoding (BPE) is commonly used for tokenization in T5.
- T5 requires input data in a text-to-text format, where both the input and output sequences are provided as text strings. This format enables T5 to deal with a wide range of NLP tasks, including classification, translation, summarization, and more.

E. Discussion

The model's logical components contribute to its remarkable performance in various natural language processing tasks. The discussion of the key aspects of these components is as follows:

1) *Encoder-Decoder Architecture:* The encoder-decoder transformer architecture allows T5 to handle tasks that require generating an output sequence based on an input sequence. It enables T5 to perform tasks like translation and summarization effectively. The self-attention mechanism in transformers enables the model to capture dependencies between tokens, considering the context of the entire input sequence. This helps T5 understand the input more comprehensively.
2) *Pre-training Objectives:* T5's pre-training objectives expose the model to a wide range of linguistic tasks. By solving these tasks during pre-training, T5 learns to capture various language properties, including syntax, semantics, and context. The diverse set of pre-training objectives allows T5 to acquire a rich understanding of language, making it a versatile model for downstream tasks.
3) *Tokenization and Input Format:* T5's tokenization scheme, such as BPE, enables the model to handle different word forms and rare words efficiently. This enhances the model's ability to understand and generate text in a more fine-grained manner. The text-to-text input format provides a unified framework for training T5 on a wide range of NLP tasks. It simplifies the task-specific modifications required during fine-tuning, making it easier to adapt the model for various applications.

VIII. CONCLUSION & FUTURE WORK

A. Conclusion

In conclusion, the “Summarization based Artificial Intelligence Model”, uses the T5 model architecture that allows the summarization of text using several techniques such as segmentation, stemming, tokenization and phrase ranking. The model takes in a PDF file as an input and reads the text present in it, and then summarizes the content within it by first segmenting it into various different parts and then assigning weights to all these individual words. These weights can indicate the importance of the words and as to whether it is a required point to be added into the summary. Although this seems simplistic in concept, it is quite arduous to implement in practice. Not only does it take a lot of computational power, but as the size of the data to be summarized is increased, the time required for it to summarize is also increased due to the number of computations required to be done.

We have implemented an algorithm that could run in an average specification system, but provide highly accurate results. Thus, this model can be classified as a model that is the best-case scenario for most low-end specification systems. What we have learned from this project are various techniques and methods of text summarization that includes the importance of preprocessing the data to ensure highquality inputs, choosing diverse and representative training data, fine-tuning the model on the specific summarization task, optimizing hyperparameters for optimal performance, and evaluating the model using appropriate metrics. This project has also

helped us in our academics as it enhanced our knowledge about NLP and different paradigms associated with it. It also enhanced our logic implementations skills. An example of this could be the aforementioned weight assignment of words. An incorrect logical implementation of this assignment can jeopardize the entire summary's meaning and semantics.

B. Future Work

We wish to implement a feature to this model in the future such that it can generate questions from the generated summary, which in turn can allow students to have an idea as to what type of questions may arise from a particular topic. This will require some query generation algorithms to be implemented. Question generation can be quite arduous as each weight to be assigned to the words will be harder to calculate. It will require further more computational power to execute this as well. We believe that the best option to implement this would be to transition from the T5 architecture to the BERT architecture. This will allow Bi-directional encoding and further analysis as to how much value a word holds. This can make it easier to generate questions. But this will require a very large dataset to train upon, which can be very expensive. For future work, there are several areas to explore in text summarization:

1) Enhanced summarization techniques: We can work on developing advanced techniques to improve the quality and coherence of summaries generated by T5. This could involve incorporating external knowledge, leveraging reinforcement learning, or exploring other advanced neural architectures.

2) Domain-specific summarization: Our model can be further fine-tuned on specific domains or industries to create domain-specific summarization models. This would enable more accurate and relevant summaries in specialized fields such as medical research, legal documents, or financial reports.

3) Multimodal summarization: The model's capabilities can be extended to incorporate multimodal inputs, such as combining text with images, videos, or audio. This would open up opportunities for summarizing multimedia content, which is prevalent on social media platforms and news websites.

4) Incremental summarization: Instead of generating a summary from scratch, the model can be explored for incremental summarization, where the model updates an existing summary as new information is added. This would be useful for real-time summarization of news articles or dynamic documents.

5) Ethical considerations: As text summarization becomes more powerful, it is crucial to address ethical considerations such as bias, fairness, and transparency. Future work should focus on developing methods to mitigate bias, provide transparency in the summarization process, and ensure the fairness and accountability of automated summarization systems.

Author Contributions: Created a new summarization model for serving a helping hand for the youngsters to capture and analyze the concepts at a faster rate.

Funding: No funding involved in this work. This is a research work.

Data Availability: Any available pdf file format was taken for evaluation

Conflict of interest: No, conflicts of interest.

Human Participants and/or Animals: Only myself and co-author

Ethical approval: yes approved.

IX. ACKNOWLEDGMENT

This work has been carried out in spite of the difficulties faced. The original work was not funded by any agency. Access to the data is granted openly as it supports any pdf document. The present paper is the original done by us.

REFERENCES

[1] Raffel, Colin, Noam Shazeer, Adam Roberts, Katherine Lee, Sharan Narang, Michael Matena, Yanqi Zhou, Wei Li, and Peter J. Liu. "Exploring the limits of transfer learning with a unified text-to-text transformer." The Journal of Machine Learning Research 21, no. 1 (2020), ISSN: 5485-5551.

[2] Pang, Kuo, Shaoxiong Li, Yifan Lu, Ning Kang, Li Zou, and Mingyu Lu. "Association rule mining with fuzzy linguistic information based on attribute partial ordered structure," (2023).

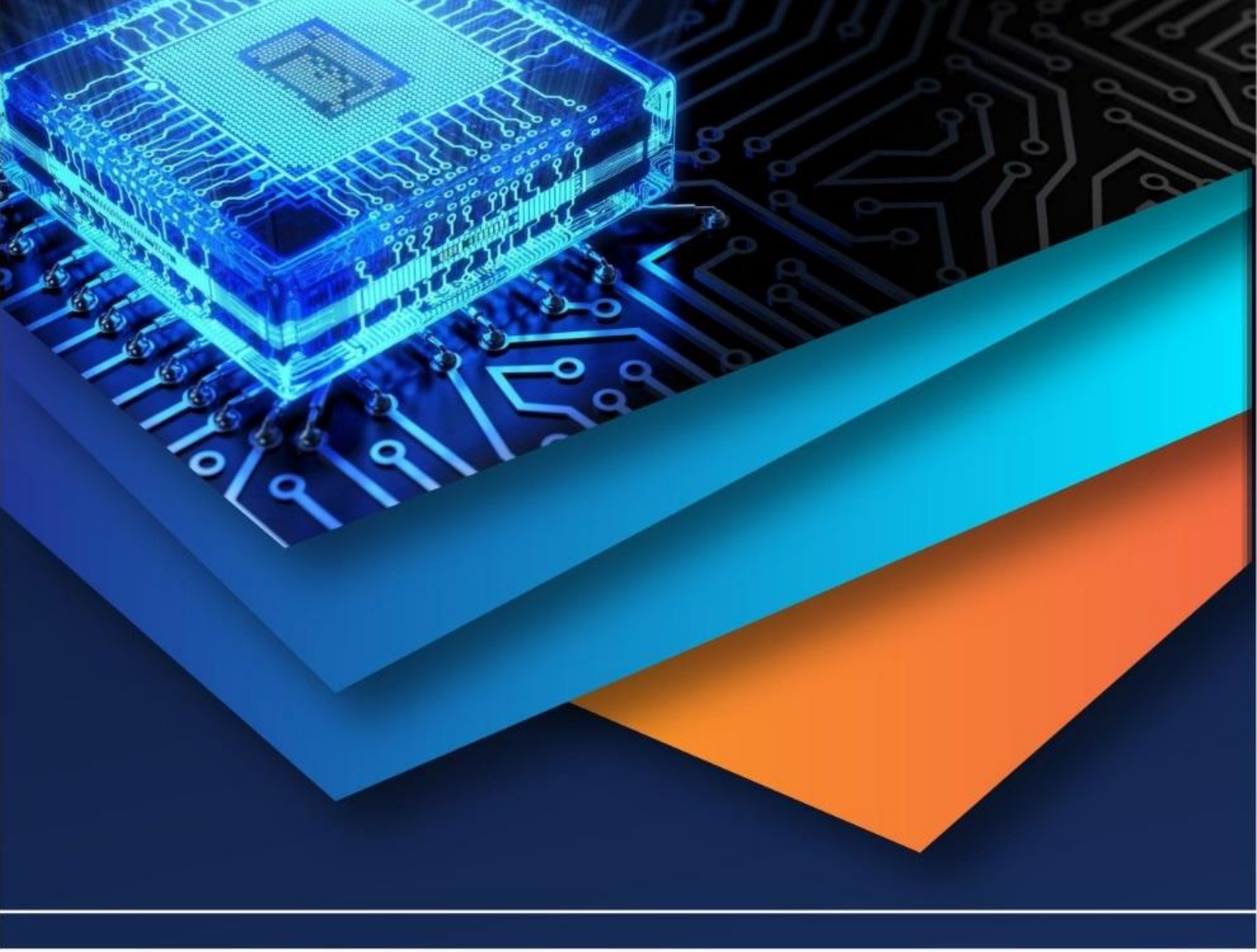
[3] Surana, Shraddha, Sugam Dembla, and Prateek Bihani. "Identifying contradictions in the legal proceedings using natural language models." SN Computer Science 3, no. 3 (2022), pp: 187-192.

[4] Kynabay, Bakdaulet, Aimoldir Aldabergen, and Azamat Zhamanov. "Automatic summarizing the news from inform. kz by using natural language processing tools," In 2021 IEEE International Conference on Smart Information Systems and Technologies (SIST), pp. 1-4. IEEE, 2021.

[5] Hong Yun, Zou, Yasser Alshehri, Noha Alnazzawi, Ijaz Ullah, Salma Noor, and Neelam Gohar. "A decision-support system for assessing the function of machine learning and artificial intelligence in music education for network games." Soft Computing 26, no. 20 (2022): 11063-11075.

[6] Ferrandin, Mauri, and Ricardo Cerri. "Multi-label classification via closed frequent labelsets and label taxonomies." Soft Computing (2023): 1-34.

[7] Gowthul Alam, M. M., and S. Baulkani. "Geometric structure information based multi-objective function to increase fuzzy clustering performance with artificial and real-life data." Soft Computing, 23, no. 4 (2019): 1079-1098.
[8] Masum, Zahra Hoseyni. "Mining stock category association on Tehran stock market." Soft Computing 23 (2019): 1165-1177.
[9] Li, Yinglong, Hong Chen, Mingqi Lv, and Yanjun Li. "Event-based k-nearest neighbors query processing over distributed sensory data using fuzzy sets." Soft Computing 23 (2019): 483-495.
[10] Wang, Fusheng, Hongyong Yang, and Yize Yang. "Swarming movement of dynamical multi-agent systems with sampling control and time delays." Soft Computing 23, no. 2 (2019): 707-714.
[11] Song, Yafei, Xiaodan Wang, Wen Quan, and Wenlong Huang. "A new approach to construct similarity measure for intuitionistic fuzzy sets." Soft Computing 23, no. 6 (2019): 1985-1998.
[12] Naresh, R., M. Sayeekumar, G. M. Karthick, and P. Supraja. "Attribute-based hierarchical file encryption for efficient retrieval of files by DV index tree from cloud using crossover genetic algorithm." Soft Computing 23 (2019): 2561-2574.
[13] Gholami, Jafar, Kareem Kamal A. Ghany, and Hossam M. Zawbaa. "A novel global harmony search algorithm for solving numerical optimizations." Soft Computing 25 (2021): 2837-2849.
[14] Asir, Thangaraj, K. Mano, and T. Tamizh Chelvam. "Correction to: Classification of non-local rings with genus two zero-divisor graphs." Soft Computing 25 (2021): 3355-3356.



10.22214/IJRASET

45.98

IMPACT FACTOR: 7.129

IMPACT FACTOR: 7.429

INTERNATIONAL JOURNAL FOR RESEARCH

IN APPLIED SCIENCE & ENGINEERING TECHNOLOGY