2024 APPLICATION FOR SUMMARIZING PDF FILE CONTENT Le Hong Anh

THE UNIVERSITY OF DANANG
DANANG UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

GRADUATION PROJECT THESIS

MAJOR: INFORMATION TECHNOLOGY

SPECIALTY: SOFTWARE ENGINEERING

PROJECT TITLE:

APPLICATION FOR SUMMARIZING PDF FILE CONTENT

Instructor:	**Ph.D. PHAM CONG THANG**
Student:	**LE HONG ANH**
Student ID:	**102200008**
Class:	**20T1**

Da Nang, 06/2024

THE UNIVERSITY OF DANANG
DANANG UNIVERSITY OF SCIENCE AND TECHNOLOGY
FACULTY OF INFORMATION TECHNOLOGY

GRADUATION PROJECT THESIS

MAJOR: INFORMATION TECHNOLOGY

SPECIALTY: SOFTWARE ENGINEERING

PROJECT TITLE:

APPLICATION FOR SUMMARIZING PDF FILE CONTENT

Instructor:	**Ph.D. PHAM CONG THANG**
Student:	**LE HONG ANH**
Student ID:	**102200008**
Class:	**20T1**

Da Nang, 06/2024

SUMMARY

Topic title: Application for summarizing PDF file content

Student name: Le Hong Anh

Student ID: 102200008 Class: 20T1

In today's rapidly advancing technological landscape, the surge in information availability demands that we absorb more knowledge daily through sources like articles and blogs. However, balancing our primary work responsibilities with the constant need to stay updated becomes increasingly challenging with the overwhelming amount of information requiring our attention each day.

An ideal solution to optimize this process is an application capable of automatically summarizing the content of PDF files. This system would allow users to quickly grasp essential information accurately, significantly saving time compared to reading entire documents.

The goal of this project is to develop a "PDF Content Summarization Application" leveraging machine learning models and natural language processing techniques. This application will automatically ingest content from PDF files and generate concise, informative summaries, enabling users to swiftly understand key points without the need to read the entire text. This not only enhances work efficiency but also supports more effective learning and research.

DA NANG UNIVERSITY
UNIVERSITY OF SICIENCE AND TECHNOLOGY
FALCUTY INFORMATION TECHNOLOGY

THE SOCIALIST REPUBLIC OF VIETNAM
Independence - Freedom - Happiness

GRADUATION PROJECT REQUIREMENTS

Student Name: Le Hong Anh Student ID: 102200008

Class: 20T1 Faculty: Information Technology Major: Software Engineering

1. *Topic title:* Application for summarizing PDF file content
2. *Project topic:* ☐*has signed intellectual property agreement for final result.*
3. *Initial figure and data:*

……………………………………..……………………………………………..……...
...………………………………………………………………………………………
…..…………………………….…..……………………….………………………

Content of the explanations and calculations:

…...……………………………………………………………………………………
…...……………………………………………………………………………………
…...……………………………………………………………………………………
…...……………………………………………………………………………………
…...……………………………………………………………………………………

4. *Drawings, charts (specify the types and sizes of drawings):*

…...……………………………………………………………………………………
…...……………………………………………………………………………………
…...……………………………………………………………………………………
…...……………………………………………………………………………………

5. *Name of instructor:*	*Content parts:*
Ph.D. Pham Cong Thang	

6. *Date of assignment:* *……../……./2024.*
7. *Date of completion:* *……../……./2024.*

Da Nang, date 20 month 6 year 2024

Head of Division…………………. **Instructor**

Ph.D. Pham Cong Thang

PREFACE

Primarily, I would like to express my deepest gratitude to the professors in the Department of Information Technology, as well as the faculty members at Da Nang University of Science and Technology. I am extremely grateful for their resolute and thorough teaching, as well as for imparting invaluable knowledge to all students in the Department of Information Technology during our time at the university. This knowledge has contributed to the completion of my graduation thesis.

With the desire to equip myself with solid knowledge to meet future job requirements, I understand the importance of mastering both theory and practice. To achieve this goal, I undertook this graduation thesis under the guidance of Professor Pham Cong Thang. Thanks to his patient guidance throughout the process, I have gained significant and valuable experience.

I would like to extend my heartfelt thanks to all the faculty members in the department, especially Professor Pham Cong Thang, for helping me successfully complete this graduation thesis. Throughout the process of implementation and study at the university, when applying theoretical knowledge to practical situations, I realized that there are still many shortcomings, especially in the application of learned knowledge to practice. Therefore, there are still many limitations in this report, and some aspects may not be fully perfected. I kindly ask for the professors' understanding and suggestions to help me improve the report to its fullest extent.

Da Nang, date 20 month 6 year 2024

Student Performed

Le Hong Anh

ASSURRANCE

I hereby affirm:

1. The graduation project report titled "Application for Summarizing PDF File Content" is my individual research and implementation conducted under the guidance of Professor Pham Cong Thang.

2. I have independently sought out information, conducted research, and synthesized knowledge to complete this project and report.

3. All information has been properly cited from the sources listed in the [References] section.

Da Nang, date 20 month 6 year 2024

Student Performed

Le Hong Anh

TABLE OF CONTENT

LIST OF PICTURES

LIST OF TABLES

LIST OF SYMBOLS, ACRONYM

Acronym	Explain
AI	Artificial intelligence
LLM	Large Language Model
ML	Machine Learning
DL	Deep Learning
NLP	Natural Language Processing
Js	JavaScript

INTRODUCTION

1. Analysis of the current situation:

Due to fast-paced innovations and technological advancement, every aspect of our lives has taken a step toward modernization. The ease with which we can obtain information from any random PDF document becomes much of a burden to most people. Of course, search tools and internet information retrieval services can vastly help us find the needed information in an instant, but extracting it from PDF files and reading it still consumes so much time and effort.

2. Purpose of the project:

This project is dedicated to supporting the development of an application that can summarize PDF documents in an automatic manner. In simple terms, it means that users will get a summary concerning the documents they provide fast enough, hence saving them time and effort in reading and synthesizing information from long and complex documents.

3. Objectives of the project:

Since technology is rapidly advancing and updating, our lives are becoming more comfortable and modern. On this basis, accessing information from PDF documents can be a headache for many of us. From searching tools to information retrieval services online, everything will support us in finding information within no time, but reading and synthesizing its content from a PDF file is still time-consuming and laborious.

4. Contents and Objects of Study:

Project Content:

- Analyzing current methods and technology in text processing and content summarization
- Researching Natural Language Processing techniques and Machine Learning models for content summarization
- Mobilizing a system or an application that can, on its own accord, summarize the content of a PDF document.

- Evaluating the performance and quality of the result provided by the developed system or application.

Target Research Audience

- All researchers and developers involved in the automation of the process of content summarization from PDF documents.

5. Research Methodology:

Literature Review: This will entail a broad coverage of the existing methods and technologies in text processing and summarization of content.

Model Selection: Run a test for different machine learning models and NLP techniques in content summarization.

Design a system or application which, after making necessary deployments, will be able to summarize content from PDF documents.

Evaluation: This phase results in a performance and quality assessment of the developed system or application by means of testing and users' feedback.

6. Achievement of Project Goals:

In case the PDF content summarization application can be implemented successfully, huge benefits will ensue. For document users, it will save time and effort spent in searching for information within the documents, while assuring correctness and quality of answers so that work productivity in daily activities is improved. Applied to common troubles in the process of updating and grasping information fast and accurately, it gives practical benefits for users.

7. Project structure:

Structure of the thesis:

Introduction

Chapter 1: Theoretical Background

Chapter 2: System Analysis and Design

Chapter 3: Implementation and Evaluation

Conclusion and Future Development

CHAPTER 1 : THEORETICAL BASIS

1.1. Chapter introduction

In this chapter, we will unwind some of the important, not at all small theories inside a major field of our project. We will go through basic and advanced theories of AI, Machine Learning, and Deep Learning-three pillars that have made today's technology so mighty and did so in the past.

Over the next couple of weeks, we will dive into the most interesting and influential area of NLP that provided computers with a possibility to reorder and understand human language. We will take a closer look at yet another representation and encoding schemes for words and discover a genuinely powerful technology standing behind the processing of natural language text: large language models.

We will also cover the Transformer architecture with a self-attention mechanism, which comes enormously contributing toward the leaps and bounds of developments in a good many NLP tasks.

1.2. The relationship between AI, Machine Learning, and Deep Learning

Artificial Intelligence, Machine Learning, and Deep Learning have been the most discussed technologies across the commercial world as companies are leveraging them extensively to build intelligent machines and applications.

These three words are primarily used simultaneously to point out the very same things; however, the reality is that they do not. In the following figure 1.1, I have provided an example so that we understand it in a typical way about the main differences between Artificial Intelligence, Machine Learning, and Deep Learning.

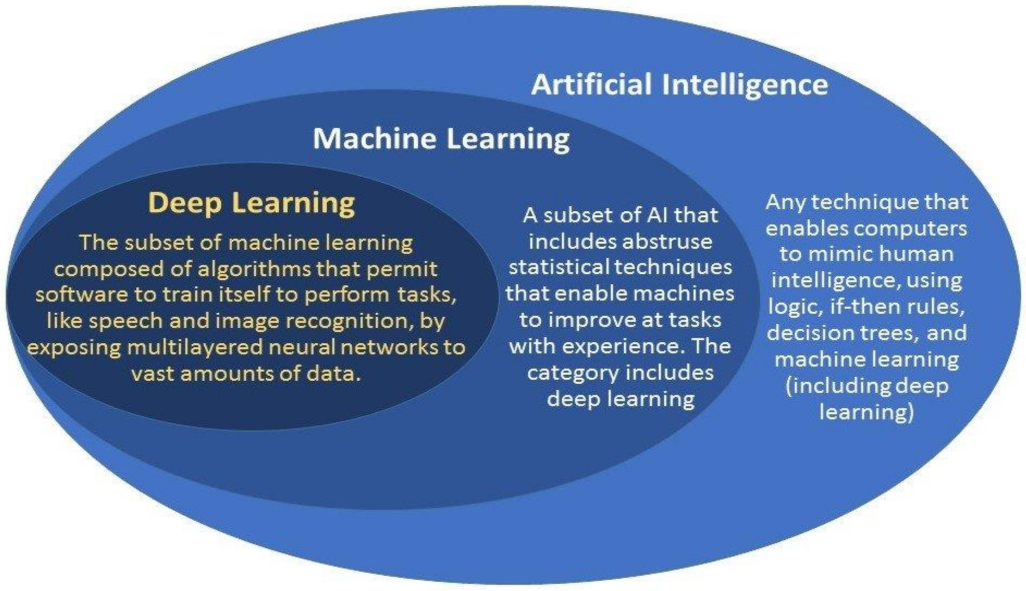

Artificial Intelligence
Machine Learning
Deep Learning
The subset of machine learning composed of algorithms that permit software to train itself to perform tasks, like speech and image recognition, by exposing multilayered neural networks to vast amounts of data.
A subset of AI that includes abstruse statistical techniques that enable machines to improve at tasks with experience. The category includes deep learning
Any technique that enables computers to mimic human intelligence, using logic, if-then rules, decision trees, and machine learning (including deep learning)

Figure 1.1: Relationship between AI, ML, DL

- Artificial Intelligence (AI) is the concept of creating intelligent machines.
- Machine Learning is a subset of artificial intelligence that helps us build applications based on AI.
- Deep Learning is a subset of machine learning that uses copious amounts of data and complex algorithms to train a model.

1.3. Natural Language Processing

1.3.1. Introduction

One more science of artificial intelligence, NLP stands for Natural Language Processing, which deals with the interaction between computers and a human through natural language. The goal of NLP is to read, decipher, understand, and make sense of human language in a meaningful way. One divides NLP into two major branches that are not entirely independent: speech processing and text processing.

Speech Processing: This technology develops and researches algorithms and computer programs that process human language in the form of speech, such as audio data. The significant applications of speech processing are speech recognition and speech synthesis.

- Text Processing: This branch deals with the analysis of text data. Notable applications in Text processing include search and information retrieval, Machine Translation, Automatic text summarization, and Automatic spell checking.

- NLP is part of everyday life, becoming increasingly so when language technology is applied to diversifying fields such as retailing by using chatbots for customer service or medicine for interpreting and summarizing electronic health records.

Figure 1.2 below illustrates various functions of NLP, such as sentiment analysis, language modeling, and named entity recognition.

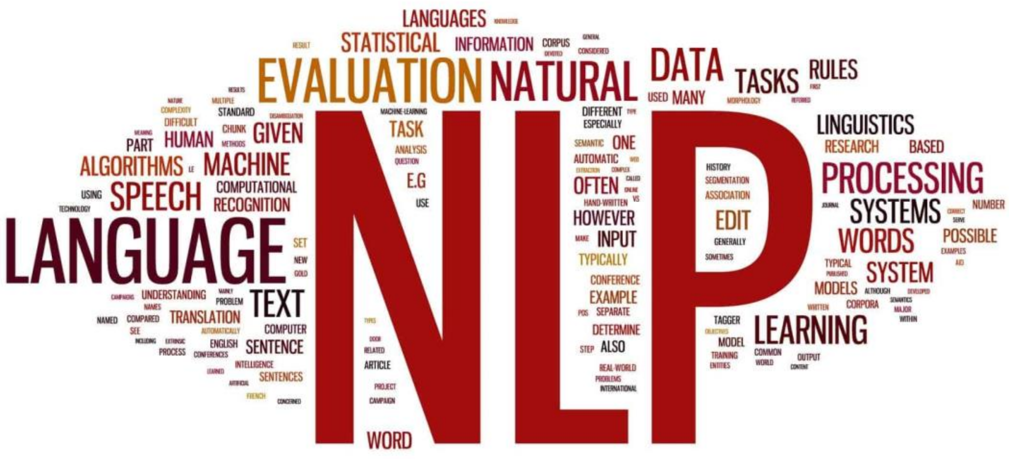


Figure 1.2: Natural language processing(NLP)

1.3.2. A few common techniques used in NLP

Now, let us focus on common NLP techniques we can adapt for our applications. Major techniques in extracting data from text are as follows:

- Tokenization: This is a step to break the text down into tokens, which could be word-level, subword-level, or character-level. It helps in getting insight into the structure of the text.
- Word Embeddings: These include dense vector representations of words in a multi-dimensional vector space, capturing the semantic relationship between words. Therefore, techniques such as Word2Vec and FastText are largely used.
- Seq2Seq Models: The toughest tasks, like machine translation, can also be done using seq2seq models for text summarization. This model contains an encoder for the conversion of source text into a context

vector and a decoder for the conversion of the context vector into summary text.

- Extractive Summarization: It involves the selection or extraction of key sentences or phases from above or below a given original text.
- Abstractive Summarization: This sort of summarization occurs when new sentences are generated with the core ideas intact from the source text; it is built with more complex models, including transformers like BERT or GPT.

1.4. Word Embeddings

1.4.1. Introduction

One of the most important concepts in NLP is word embedding, which enables the representation of text at the level of words, that is, single words in a text expressed as numerical vectors in a multidimensional space. This technology has phenomenally advanced the understanding of semantics and addressed many challenges in natural language processing.

1.4.2. Principle of operation

Word embeddings represent every word in a document as low-dimensional numerical vectors whereby related words are closed to each other in multidimensional space. With this approach, it enables the computer to understand the semantics of some of the words and phrases.

For instance, figure 1.3 shows a word represented as a 4-dimensional vector, although in real applications it is represented in higher dimensions.

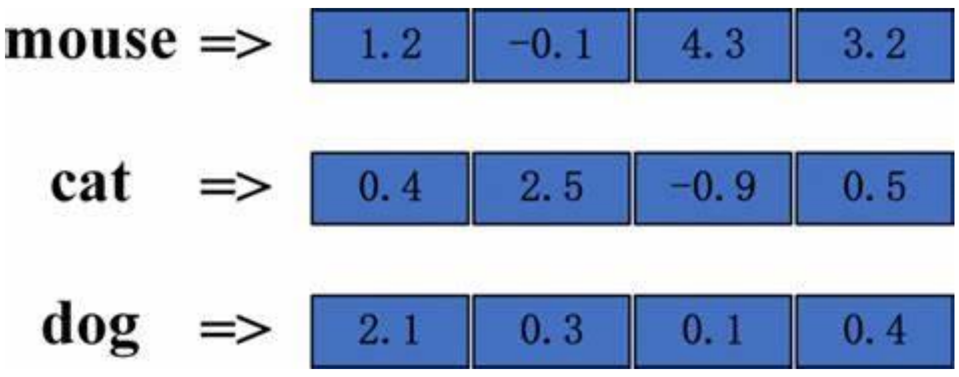


Figure 1.3: Transforming a word into a 4-dimensional vector

The vector representation of the words in multi-dimensional space, for a better view, is shown in figure 1.4. The words in multi-dimensional space are closer to each other when they have a very similar meaning; therefore, the semantic relationship of these kinds of words is depicted.

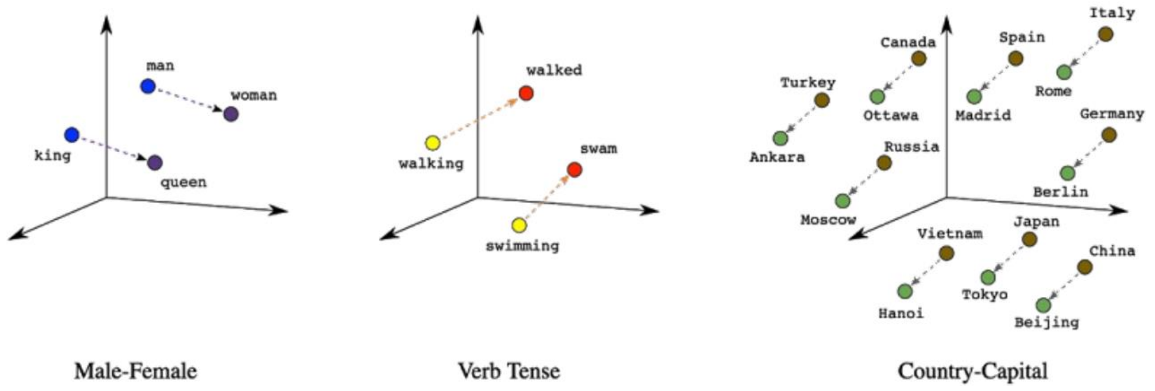


Figure 1.4: The words are visualized in a three-dimensional space.

1.4.3. Popular Embedding Methods

There are a variety of techniques for generating word embeddings, not all of which work well with every task. Some require enough data; others are domain-dependent or language-complexity-dependent. Here is how some of the popular word-embedding techniques work:

- Word2Vec: This is a model learning representation of words based on the context they appear in. It uses a neural network to predict all the other words in a window's center. There are two Word2Vecs: Skip-gram and CBOW.
- GloVe: It represents words using the combination of community information of words with their frequency of occurrence in the text.
- FastText: An extension to Word2Vec that represents words using subwords as a part of words to manage new and rare words.

1.4.4. Transformers

The Transformer is a deep learning model designed to address a variety of tasks in natural language processing and speech processing, such as machine translation, language generation, classification, entity recognition, speech recognition, and text-to-speech conversion. The architecture of the Transformer model consists of two main components: the encoder and the decoder, as illustrated in figure 1.5. The encoder is responsible for learning representation vectors for sentences, aiming to capture all relevant information. In contrast, the decoder transforms these representation vectors into the target language.

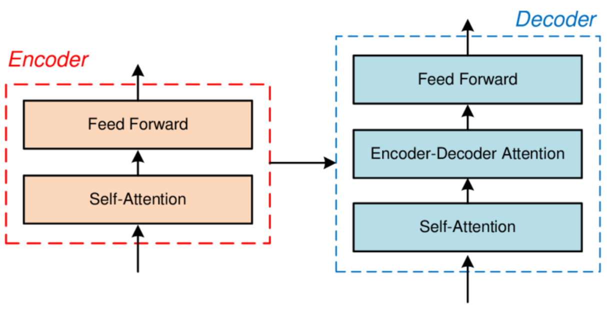


Figure 1.5: Two primary components of the transformer architecture.

Unlike other models, the architecture of the Transformer model (figure 1.6) does not process elements in a sequence sequentially. When the input data is a natural language sentence, the Transformer does not need to process the beginning of the sentence before the end. Due to this characteristic, the Transformer can leverage the parallel computing capability of GPUs and significantly reduce processing time.

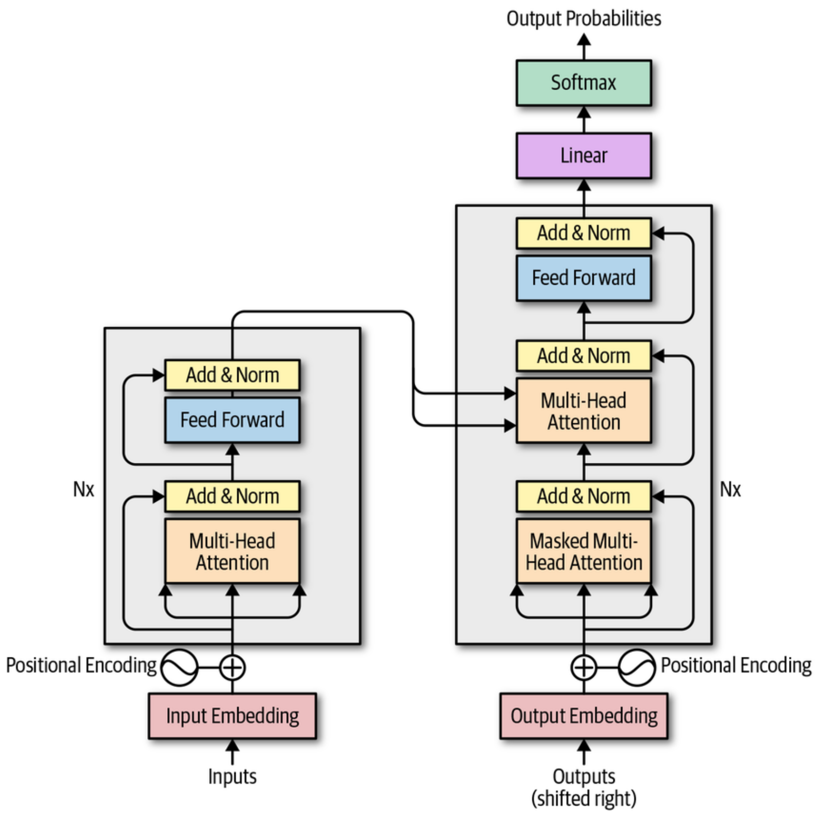


Figure 1.6: The architecture of the transformer model.

1.4.5. Self-attention

The self-attention mechanism is a fundamental component of the Transformer model, allowing it to effectively understand and process the relationships between words in a sequence. This mechanism enhances the model's ability to focus on relevant parts of the input when encoding a specific word. Figure 1.7 below illustrates the step-by-step calculations and implementation of self-attention

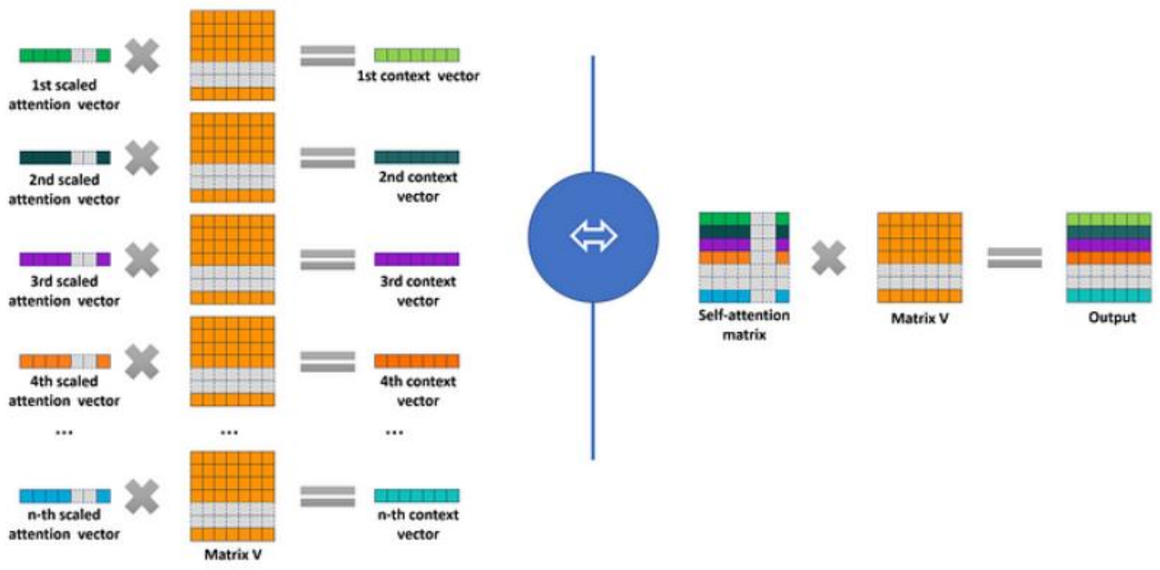


Figure 1.7: Self-Attention Mechanism Implementation

Implementation of Self-Attention:

1. Vector Creation: For each word in the sequence, create three vectors (Query, Key, Value) by multiplying the word's token representation (1x512) with three learned weight matrices (each 512x64). This results in three vectors per word, each sized (1x64).
2. Score Calculation: Compute the score (weight) for each word by performing the dot product between its query vector and the key vectors of all words. This score determines the relevance of other words to the word being encoded.
3. Score Normalization: Normalize the score by dividing it by the square root of the key vector's dimension (which is eight in this case).
4. SoftMax Application: Apply the SoftMax function to the scores to convert them into probabilities, ensuring that all scores sum to one.
5. Weighted Value Vectors: Multiply each value vector by its corresponding score.
6. Final Representation: Sum the weighted value vectors to obtain the final representation of the word being encoded.
7. Attention Map Creation: Repeat the above steps for all words in the sequence to generate an attention map that fully encodes the input using self-attention.

1.5. The Encoder

1.5.1. Positional Encoding

Since Transformers lack the recurrence mechanism found in RNNs, they utilize positional encodings to convey information about the position of each token in the sequence, essential for comprehending the order and structure of words within a sentence.

Researchers have proposed using a combination of sine and cosine functions to generate positional encodings. This approach enables the model to handle sentences of any length by creating positional vectors that encode the position of each token.

Figure 1.8 illustrates how positional encoding operates, with each dimension represented by unique sine and cosine waves of varying frequencies and offsets. The values of these functions range from -1 to 1, offering a continuous and differentiable method for representing each position in the sequence. This enables the model to effectively incorporate positional information without relying on recurrence mechanisms.

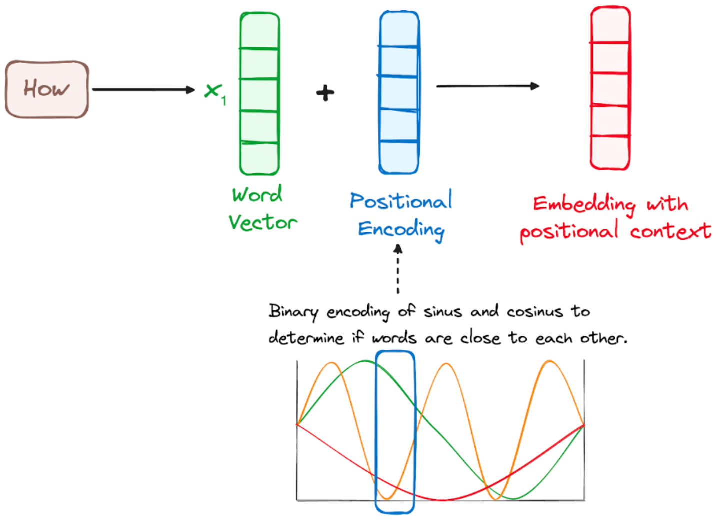

How
x_1
+
Word Vector
Positional Encoding
Embedding with positional context
Binary encoding of sinus and cosinus to determine if words are close to each other.

Figure 1.8: Positional encoding

1.5.2. Multi-Headed Attention

Multi-head attention focuses on the relevance of each word with respect to others in a sentence, represented as attention vectors. For each word, multiple attention vectors are determined to capture the contextual relationship between words in the sentence.

However, a single attention vector tends to weigh the word itself more heavily, rather than its interactions with other words. To address this, the multi-head attention mechanism computes multiple attention vectors for each word, and then takes a weighted average to compute the final attention vector for each word.

Figure 1.9 illustrates how the multi-head attention mechanism operates, enhancing the model's ability to understand complex relationships by allowing it to consider multiple aspects of word interactions simultaneously.

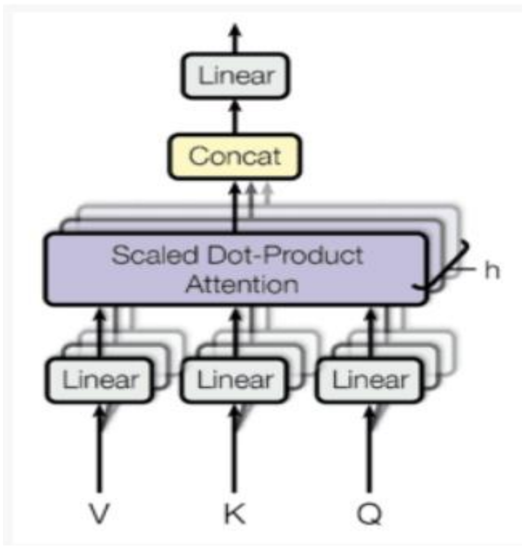


Figure 1.9: Multi-Headed Attention

1.5.3. Decoder

The decoder in a Transformer model is responsible for converting the vector received from the encoder into text. Its architecture closely resembles that of the encoder, with the key difference being the addition of a masked multi-head attention layer. This layer is positioned in the middle of the decoder and is designed to learn the relationships between the word currently being translated and the words in the source sentence.

The masked multi-head attention ensures that the model focuses on relevant parts of the input sentence while generating the output, allowing it to effectively decode the information into coherent and contextually appropriate text.

1.6. Large Language Models (LLMs)

1.6.1. Langue model

A language model is a machine learning model designed to predict and generate text that appears coherent and contextually appropriate. An example of a language model in action is the autocomplete function, which suggests the next word or phrase as you type.

These models operate by estimating the probability of a token or sequence of tokens appearing within a larger sequence. Consider the following sentence: "When I hear rain on my roof, I _______ in my kitchen." Assuming a token is a word, a language model would predict the likelihood of various words or phrases that could fill the blank. For instance, the model might assign the following probabilities:

- cook soup: 9.4%
- warm up a kettle: 5.2%
- cower: 3.6%
- nap: 2.5%
- relax: 2.2%

A "sequence of tokens" can refer to a sentence or a series of sentences. In other words, a language model can compute the probability of different sentences or paragraphs. Estimating the probability of what comes next in a sequence is valuable for various applications, such as text generation, language translation, and question answering.

By predicting the most probable next words, language models can generate coherent and contextually appropriate text, making them crucial in numerous natural language processing tasks.

1.6.2. What are Large Language Models?

Large language models (LLMs) represent an advancement in artificial intelligence, specifically tailored to comprehend and produce human-like text. These models, typically constructed on deep learning frameworks, have significantly impacted natural language processing (NLP) tasks by their capability to analyze, understand, and generate text on a large scale.

At the heart of LLMs lie neural network architectures like Transformers, which utilize self-attention mechanisms to effectively process input sequences. Trained on extensive text data from diverse sources, these models acquire intricate language patterns, relationships, and nuances.

The evolution of LLMs has been characterized by substantial growth in model size. Examples include OpenAI's GPT series, Google's BERT, and Facebook's RoBERTa, boasting hundreds of millions to billions of parameters. This scale enables them to capture and generate highly coherent and contextually relevant text.

LLMs find utility across various domains, including NLP tasks like understanding, text generation, sentiment analysis, translation, summarization, and question answering. They power intelligent assistants, chatbots, recommendation systems, and translation services, among other applications.

1.6.3. *Architecture of Large Language Models*

The architecture of LLM consists of various neural network layers, such as recurrent layers, feedforward layers, embedding layers, and attention layers, as illustrated in figure 1.10. These layers collaborate to process the input text and generate output predictions.

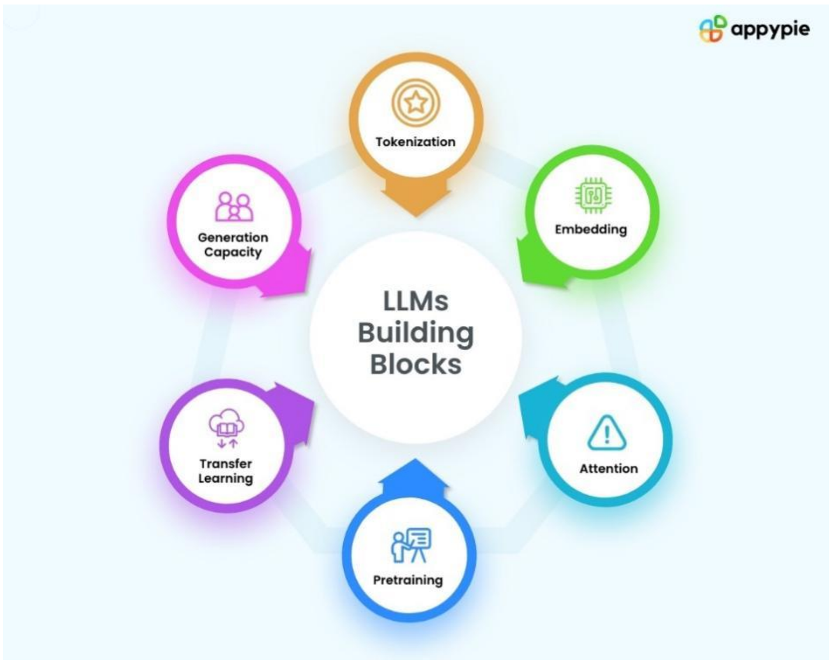


Figure 1.10: LLMs building blocks

While some LLMs are constructed with different architectures, most are built from Transformer models, sharing similar architectures with some variations. The key building blocks include:

- Embedding layer: Converts each word in the input text into high-dimensional vector representations, capturing semantic and syntactic information to help the model understand the context.
- Feedforward layers: Comprise multiple fully connected layers that apply nonlinear transformations to input embedding vectors, facilitating the learning of abstract information.
- Recurrent layers: Interpret information from the input text sequentially, maintaining hidden states updated at each time step to capture dependencies between words.
- Attention layers: Crucial for selectively focusing on various parts of the input text, allowing the model to pay attention to relevant sections and make accurate predictions.

A notable feature distinguishing LLMs and enhancing them compared to traditional Transformer models is the use of Prompts. This condition is crucial for

LLMs to learn effectively and mitigate hallucination phenomena by adjusting input questions or output results to provide relatively accurate results.

CHAPTER 2 : SYSTEM ANALYSIS AND DESIGN

2.1. Use case chart

2.1.1. Agent

The diagram below illustrates all the agents of the PDF Summarization Application system. Each agent represents a role within the system. Currently, the system has only one agent, which is the User.

Table 2.1: Agent of the application

Agent	**Description**
User	Upload files and view summary history

2.1.2. Use case chart

The system's use case diagram is depicted in figure 2.1 below.

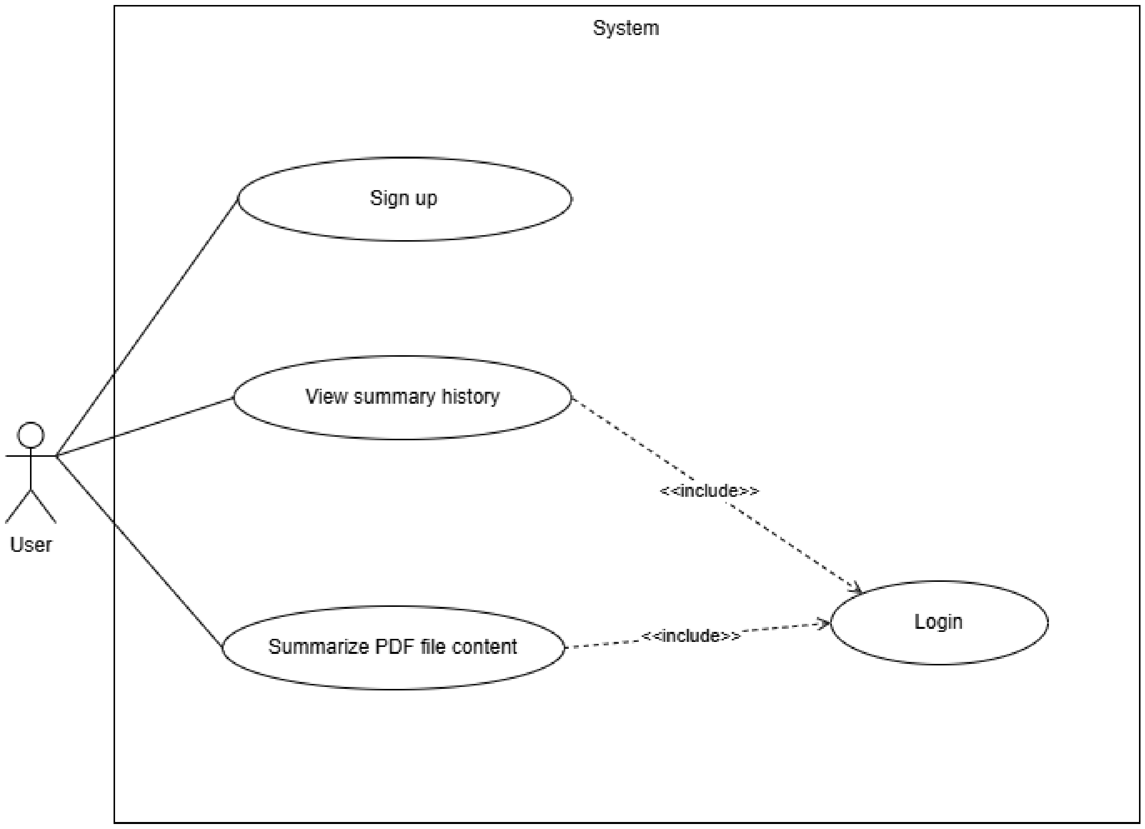


Figure 2.1: System use case

2.2. Databases

2.2.1. MongoDB

MongoDB is an open-source NoSQL database management system developed by MongoDB Inc. It stores data in structured documents called BSON (Binary JSON), which is a binary form of JSON, enhancing performance and storage capabilities compared to traditional JSON data.

MongoDB offers various features and applications in software development, including:

- High flexibility: MongoDB allows storing data with non-uniform structures, making it easy to change data structures without the intervention of database administrators.
- Easy scalability: MongoDB can easily scale by adding new nodes to the database cluster, improving system scalability without downtime.
- High integrity and reliability: MongoDB supports features such as data backup and recovery and sharding to ensure data integrity and reliability.
- Good integration with programming languages and frameworks: MongoDB provides drivers for popular programming languages such as JavaScript, Python, Java, and C#, facilitating integration with other applications and frameworks.
- Support for complex queries: MongoDB supports complex queries using a JSON-based query language, allowing execution of queries based on complex conditions and flexible data organization.

MongoDB is commonly used for web applications, mobile applications, data analysis, IoT, and many other applications due to its flexibility, scalability, and high performance.

For example, figure 2.2 below displays the MongoDB logo image.

Figure 2.2: MongoDB Database Management System

2.2.2. Database schema

MongoDB represents a paradigm shift in database management systems as it belongs to the NoSQL category, breaking away from the traditional fixed-schema approach seen in relational databases like MySQL and PostgreSQL. It stores data in BSON (Binary JSON) format, which is akin to JSON-like documents. The beauty of this system is that each document can have its own unique structure.

One of the key advantages of MongoDB is its inherent flexibility. It allows for easy modification of data structures without the need to alter the database schema. This proves to be a boon for applications that are in the rapid development phase and require a high degree of flexibility. However, there are scenarios where the definition of a schema can be beneficial. It can serve as a control mechanism for data and help maintain the integrity of data within the MongoDB system.

2.3. Management source code

I manage my source code through GitHub, applying Git Flow (figure 2.3 below) to set up various branches and the methods to merge them back together following an optimal process.

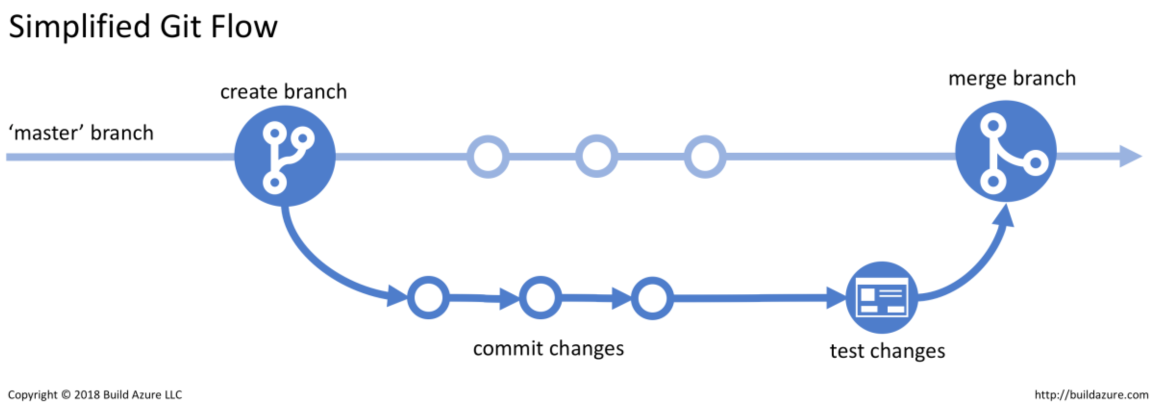


Figure 2.3: Git flow

Here is an overview of the Gitflow workflow for my project:

- Main: The main branch is readily available in git and is the branch that contains the initial source code of the application and the versions ready to be released for users to use.
- Features: These are based on the develop branch. Each time we develop a new feature, we need to create a branch to write source code for each feature.

2.4. System Architecture

2.4.1. Overview

The architecture of the application is depicted in figure 2.4 below

Application		React Native
Backend	node JS	JS
Database	mongoDB.	
AI	python™	jupyter

Figure 2.4: System architecture overview

The technical stack includes:

- Artificial Intelligence (AI): Python
- Backend: Node.js
- Application: React Native

2.4.2. Building AI using python

The development of the AI component for our PDF summarization application involves fine-tuning a pre-trained model, specifically viT5, to effectively summarize content from PDF files. This section outlines the process of building and fine-tuning the AI model.

Step 1: Environment Setup

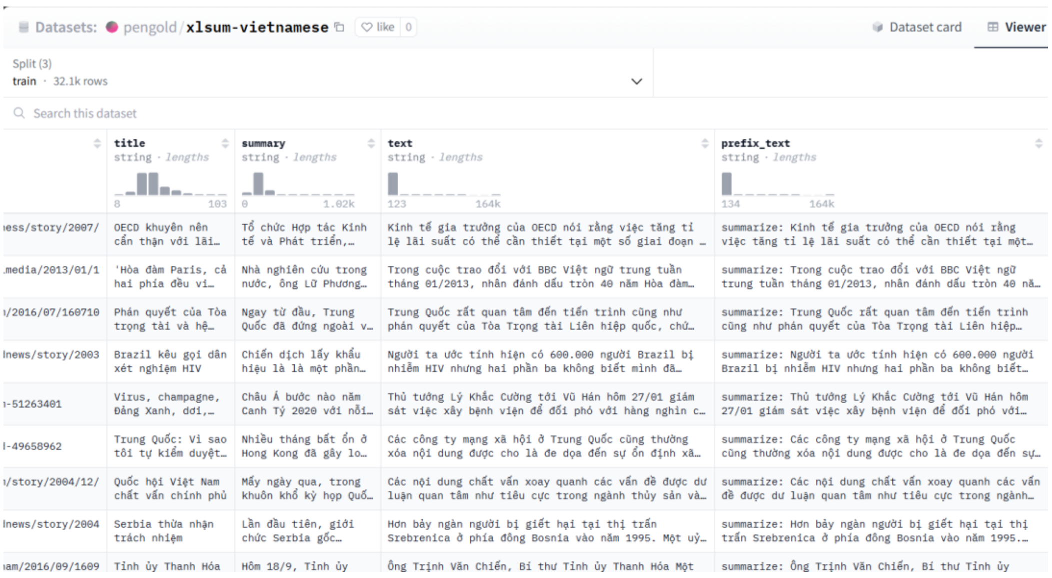
The initial step in developing the AI component is setting up the necessary development environment. This includes installing Python and essential libraries such as PyTorch, and the Hugging Face Transformers library, which provide the foundational tools required for building and training our AI model.

Step 2: Data Collection and Preparation

For effective training of the AI model, a comprehensive dataset of Vietnamese text documents is essential. In this project, the dataset chosen is sourced from Hugging Face, specifically the xlsum-vietnamese dataset. This dataset consists of pairs of input texts and their corresponding summaries, making it ideal for our summarization task. The texts are meticulously cleaned and formatted to ensure consistency and quality in the training data, which is crucial for the successful fine-tuning of the model.

This dataset (figure 2.5 below) includes three subsets: train, test, and validation, structured as follows:

- **Training Set**: Contains 11,800 rows with features 'summary' and 'text'.
- **Validation Set**: Contains 2,017 rows with features 'summary' and 'text'.
- **Test Set**: Contains 2,028 rows with features 'summary' and 'text'.



	title string · *lengths* 8 – 103	summary string · *lengths* 0 – 1.02k	text string · *lengths* 123 – 164k	prefix_text string · *lengths* 134 – 164k
ıess/story/2007/	OECD khuyên nên cẩn thận với lãi…	Tổ chức Hợp tác Kinh tế và Phát triển,…	Kinh tế gia trưởng của OECD nói rằng việc tăng tỉ lệ lãi suất có thể cần thiết tại một số giai đoạn …	summarize: Kinh tế gia trưởng của OECD nói rằng việc tăng tỉ lệ lãi suất có thể cần thiết tại một…
.media/2013/01/1	'Hòa đàm Paris, cả hai phía đều vi…	Nhà nghiên cứu trong nước, ông Lữ Phương…	Trong cuộc trao đổi với BBC Việt ngữ trung tuần tháng 01/2013, nhân đánh dấu tròn 40 năm Hòa đàm…	summarize: Trong cuộc trao đổi với BBC Việt ngữ trung tuần tháng 01/2013, nhân đánh dấu tròn 40 nă…
ı/2016/07/160710	Phán quyết của Tòa trọng tài và hệ…	Ngay từ đầu, Trung Quốc đã đứng ngoài v…	Trung Quốc rất quan tâm đến tiến trình cũng như phán quyết của Tòa Trọng tài Liên hiệp quốc, chứ…	summarize: Trung Quốc rất quan tâm đến tiến trình cũng như phán quyết của Tòa Trọng tài Liên hiệp…
lnews/story/2003	Brazil kêu gọi dân xét nghiệm HIV	Chiến dịch lấy khẩu hiệu là là một phần…	Người ta ước tính hiện có 600.000 người Brazil bị nhiễm HIV nhưng hai phần ba không biết mình đã…	summarize: Người ta ước tính hiện có 600.000 người Brazil bị nhiễm HIV nhưng hai phần ba không biết…
ı-51263401	Virus, champagne, Đảng Xanh, dơi,…	Châu Á bước nào năm Canh Tý 2020 với nỗi…	Thủ tướng Lý Khắc Cường tới Vũ Hán hôm 27/01 giám sát việc xây bệnh viện để đối phó với hàng nghìn c…	summarize: Thủ tướng Lý Khắc Cường tới Vũ Hán hôm 27/01 giám sát việc xây bệnh viện để đối phó với…
l-49658962	Trung Quốc: Vì sao tôi tự kiểm duyệt…	Nhiều tháng bất ổn ở Hong Kong đã gây lo…	Các công ty mạng xã hội ở Trung Quốc cũng thường xóa nội dung được cho là đe dọa đến sự ổn định xã…	summarize: Các công ty mạng xã hội ở Trung Quốc cũng thường xóa nội dung được cho là đe dọa đến sự…
ı/story/2004/12/	Quốc hội Việt Nam chất vấn chính phủ	Mấy ngày qua, trong khuôn khổ kỳ họp Quố…	Các nội dung chất vấn xoay quanh các vấn đề được dư luận quan tâm như tiêu cực trong ngành thủy sản và…	summarize: Các nội dung chất vấn xoay quanh các vấn đề được dư luận quan tâm như tiêu cực trong ngành…
lnews/story/2004	Serbia thừa nhận trách nhiệm	Lần đầu tiên, giới chức Serbia gốc…	Hơn bảy ngàn người bị giết hại tại thị trấn Srebrenica ở phía đông Bosnia vào năm 1995. Một uỷ…	summarize: Hơn bảy ngàn người bị giết hại tại thị trấn Srebrenica ở phía đông Bosnia vào năm 1995.…
ıam/2016/09/1609	Tỉnh ủy Thanh Hóa	Hôm 18/9, Tỉnh ủy	Ông Trịnh Văn Chiến, Bí thư Tỉnh ủy Thanh Hóa Một	summarize: Ông Trịnh Văn Chiến, Bí thư Tỉnh ủy

Figure 2.5: Vietnamese Datasets on Hugging Face

For the project focused on summarizing PDF content, several key data preprocessing steps were undertaken:

- Data Filtering:
  - **Word Count Filtering**: Focus on texts with 300-400 words. Exclude texts with more than 500 words to comply with the model's 512 token limit.
- Tokenization:
  - **Splitting Text**: Break down the text into individual tokens (words, subwords, or characters) that the model can process.
- Removing Unnecessary Characters:
  - **Whitespace Removal**: Strip out unnecessary whitespace to ensure clean input data.
  - **Special Character Removal**: Remove non-textual characters and special symbols to maintain data consistency.

The distribution of content word count is described in figure 2.6 below:

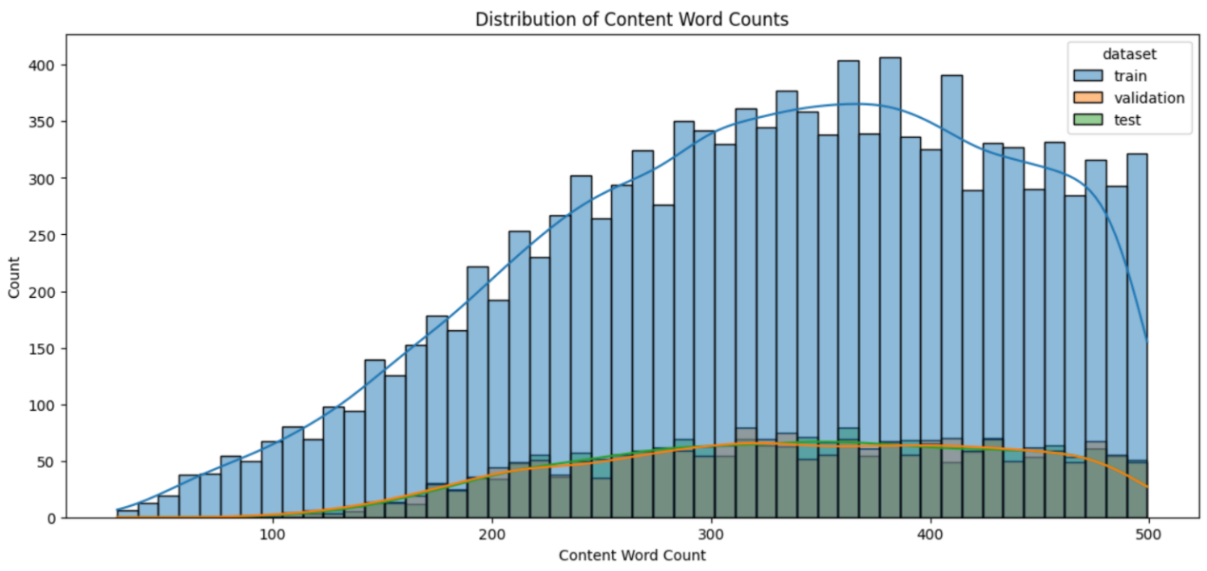


Figure 2.6: Distribution of content data consistency

The distribution of content word count in our dataset (figure 2.7 below) primarily falls within the range of 300-400 words. To optimize the input for our model, we have excluded any texts exceeding 500 words, as the model's input can only accommodate up to 512 tokens. This preprocessing step ensures that the data aligns with the model's requirements, enhancing the efficiency and accuracy of the summarization process.

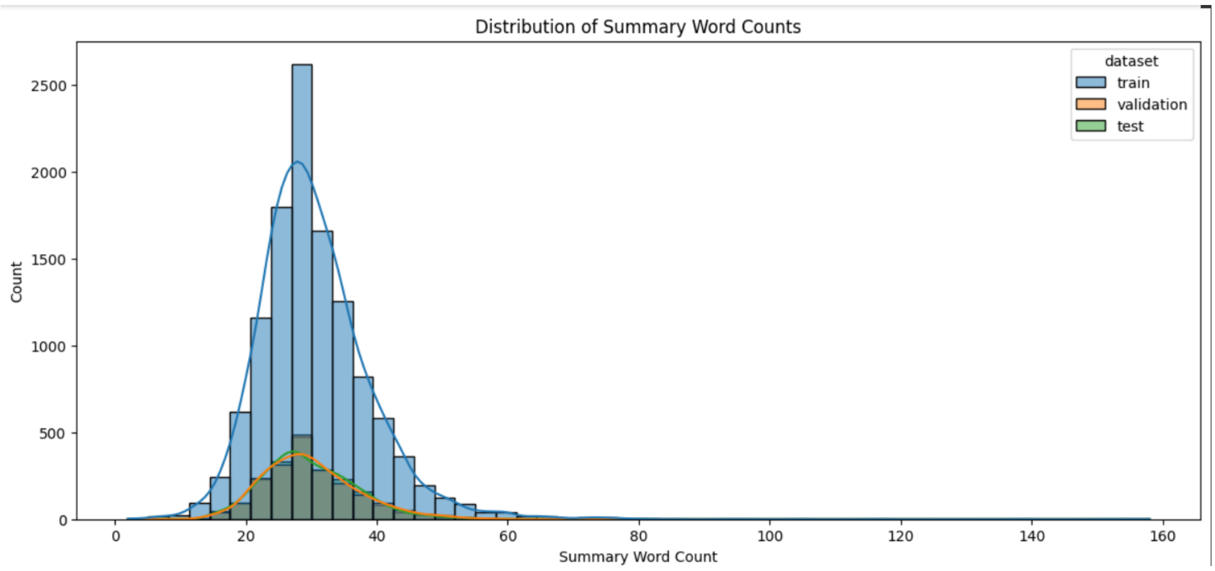
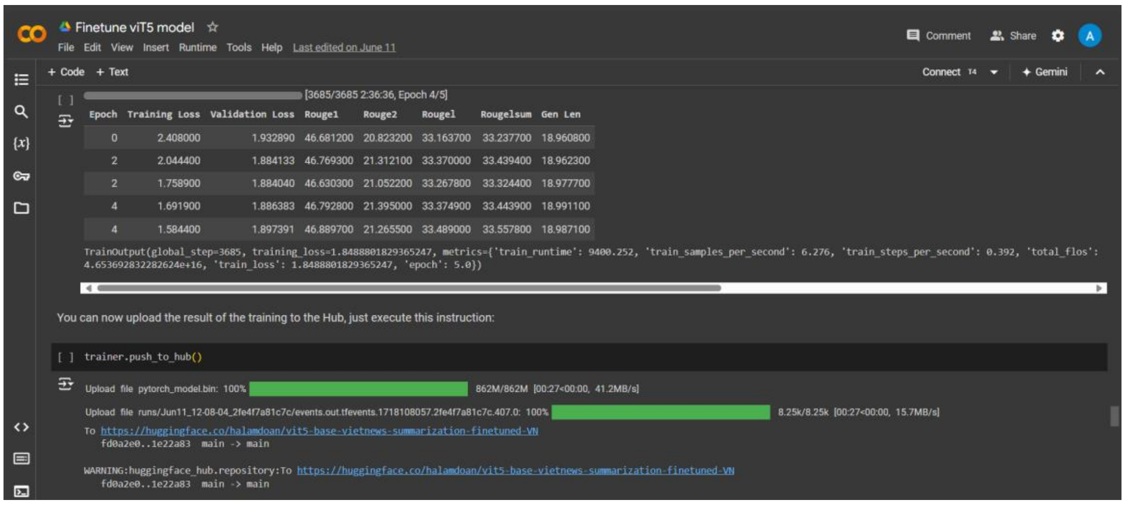


Figure 2.7: Distribution of summary word counts

Training model:

Google Colaboratory, originally an internal tool at Google, has been made available to the public for data science and machine learning development. This platform allows users to train data models efficiently using its robust computational resources. The model training process is described in figure 2.8 below.


Finetune viT5 model
File Edit View Insert Runtime Tools Help Last edited on June 11
Comment Share
+ Code + Text
Connect T4
Gemini

[3685/3685 2:36:36, Epoch 4/5]

Epoch	Training Loss	Validation Loss	Rouge1	Rouge2	Rougel	Rougelsum	Gen Len
0	2.408000	1.932890	46.681200	20.823200	33.163700	33.237700	18.960800
2	2.044400	1.884133	46.769300	21.312100	33.370000	33.439400	18.962300
2	1.758900	1.884040	46.630300	21.052200	33.267800	33.324400	18.977700
4	1.691900	1.886383	46.792800	21.395000	33.374900	33.443900	18.991100
4	1.584400	1.897391	46.889700	21.265500	33.489000	33.557800	18.987100

```
TrainOutput(global_step=3685, training_loss=1.8488801829365247, metrics={'train_runtime': 9400.252, 'train_samples_per_second': 6.276, 'train_steps_per_second': 0.392, 'total_flos':
4.653692832282624e+16, 'train_loss': 1.8488801829365247, 'epoch': 5.0})
```

You can now upload the result of the training to the Hub, just execute this instruction:

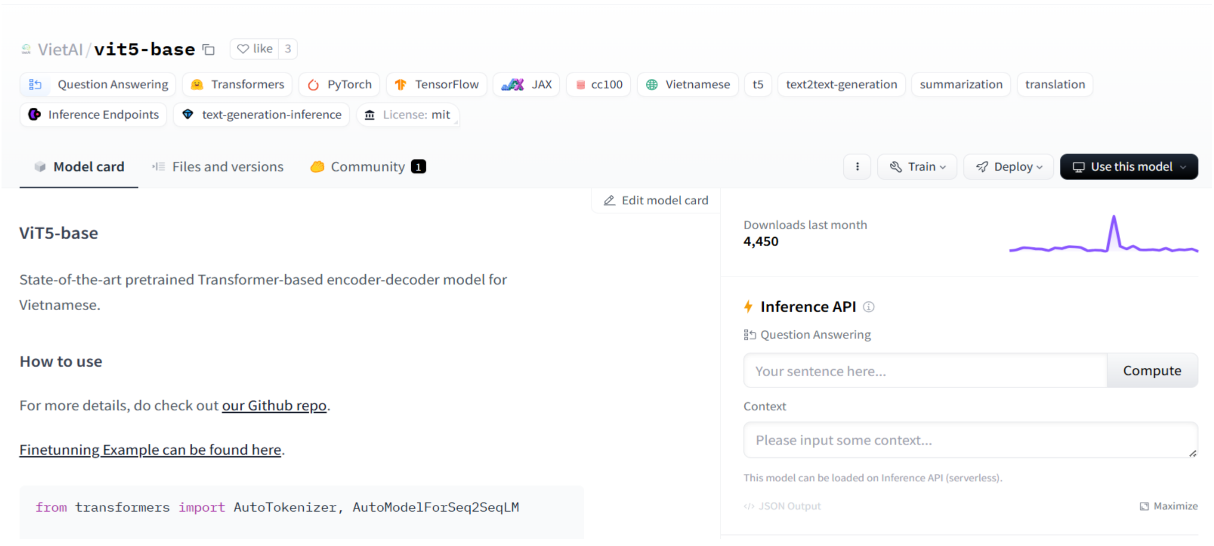
```
trainer.push_to_hub()
```

```
Upload file pytorch_model.bin: 100% 862M/862M [00:27<00:00, 41.2MB/s]
Upload file runs/Jun11_12-08-04_2fe4f7a81c7c/events.out.tfevents.1718108057.2fe4f7a81c7c.407.0: 100% 8.25k/8.25k [00:27<00:00, 15.7MB/s]
To https://huggingface.co/halamdoan/vit5-base-vietnews-summarization-finetuned-VN
   fd0a2e0..1e22a83  main -> main

WARNING:huggingface_hub.repository:To https://huggingface.co/halamdoan/vit5-base-vietnews-summarization-finetuned-VN
   fd0a2e0..1e22a83  main -> main
```

Figure 2.8: Training model with Colab

The ViT5 model (figure 2.9 bellow) is a pretrained, Transformer-based encoder-decoder framework developed specifically for processing the Vietnamese language. It employs a T5-style self-supervised pretraining strategy and has been trained on a vast corpus of diverse and high-quality Vietnamese texts. Demonstrating significant advancements over previous models, ViT5 sets a new benchmark for state-of-the-art performance in Vietnamese Text Summarization tasks. For fine-tuning purposes, ViT5 can be customized for specific summarization requirements by training on datasets pertinent to the target text domain.



ViT5-base

State-of-the-art pretrained Transformer-based encoder-decoder model for Vietnamese.

How to use

For more details, do check out our Github repo.

Finetunning Example can be found here.

```
from transformers import AutoTokenizer, AutoModelForSeq2SeqLM
```

Figure 2.9: viT5 model on Hugging Face

2.4.3. Building application using React Native

React Native is an open-source JavaScript framework used for building mobile applications. It allows developers to create cross-platform apps with native performance and appearance, using a single codebase. React Native is often compared to other frameworks like Flutter and Xamarin, but its strength lies in its ability to leverage the vast ecosystem of React and JavaScript.

Advantages:

- Ease of Learning and Use: React Native has a syntax similar to React, along with extensive documentation and community support, making it easy for developers to learn and get started. Cross-Platform
- Compatibility: With React Native, developers can write code once and deploy it on both iOS and Android, significantly reducing development time and effort.
- Performance: React Native is designed to deliver near-native performance by using native components, making it capable of handling complex applications smoothly.
- Rich Ecosystem and Community: Being an extension of React, it benefits from a vast ecosystem of libraries, tools, and a large community that provides extensive support and resources.

Disadvantages:

- Performance Overheads: While React Native provides near-native performance, there are still some overheads compared to fully native applications, which can affect very performance-sensitive applications.
- Limited Access to Native APIs: Although React Native provides many native modules, some platform-specific APIs may not be available out of the box, requiring the development of custom native modules.
- Frequent Updates: The fast-paced updates of React Native can sometimes lead to compatibility issues with existing code and libraries, requiring developers to frequently update and refactor their codebase.

2.4.4. Building backend using NodeJs

Node.js is a powerful open-source JavaScript runtime built on Chrome's V8 JavaScript engine. It allows developers to build scalable, high-performance applications by leveraging event-driven, non-blocking I/O model. Node.js is widely used for building server-side applications and APIs due to its lightweight nature and extensive ecosystem of libraries and frameworks.

Advantages:

- Asynchronous and Non-blocking I/O: Node.js utilizes an event-driven architecture, enabling it to handle a large number of concurrent connections efficiently without blocking the execution of other operations.
- Scalability: Node.js applications are highly scalable, making them suitable for building real-time applications and microservices architectures.
- Large Ecosystem: Node.js has a vast ecosystem of libraries and frameworks available through npm (Node Package Manager), allowing developers to easily integrate third-party modules and tools into their applications.
- Community Support: Node.js has a large and active community of developers, providing extensive support, documentation, and resources.

Disadvantages:

- Callback Hell: Asynchronous programming in Node.js often leads to callback hell, where deeply nested callbacks can make the code difficult to read and maintain.
- Performance Limitations: While Node.js excels in handling I/O-bound tasks, it may not be the best choice for CPU-intensive operations due to its single-threaded nature.
- Debugging: Debugging Node.js applications can be challenging, especially when dealing with asynchronous code and complex event-driven architectures.

2.4.5. Building system using MVC architecture

The Model-View-Controller (MVC) architecture is a widely used design pattern for developing web applications. It divides an application into three interconnected components: the Model, the View, and the Controller. This architectural pattern provides a structured and organized approach to building scalable and maintainable applications. Components of MVC Architecture:

- Model: The Model represents the data and business logic of the application. It encapsulates the application's data and defines how it interacts with the database. In Node.js applications, the Model is often responsible for querying the database, performing data validation, and executing business logic.

- View: The View is responsible for presenting the data to the user. It generates the user interface based on the data provided by the Controller and renders it to the client's browser. In Node.js applications, the View is typically implemented using templating engines like EJS, Pug, or Handlebars.
- Controller: The Controller acts as an intermediary between the Model and the View. It handles user input, processes requests from the client, and updates the Model accordingly. In Node.js applications, the Controller is responsible for routing requests, calling appropriate methods on the Model, and rendering the View with the updated data.

Additionally, the components of the MVC architecture are further detailed in figure 2.10 below, illustrating the architecture of the MVC model.

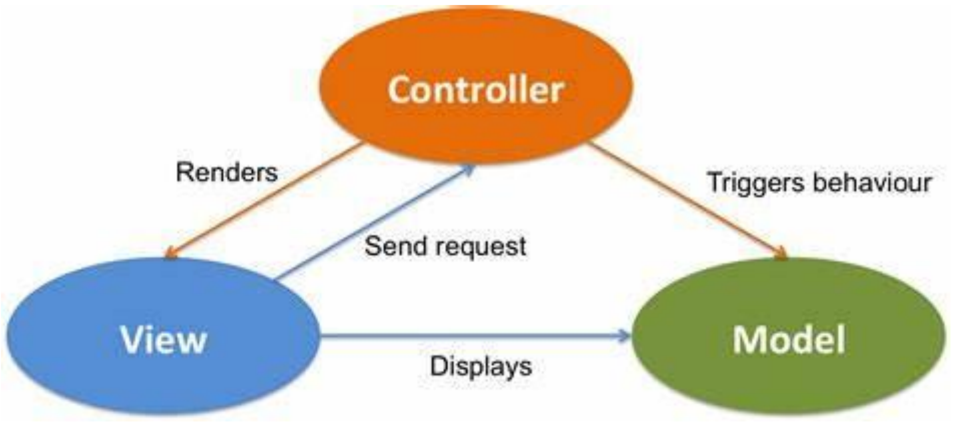


Figure 2.10: MVC architecture

Advantages of MVC Architecture:

- Separation of Concerns: MVC promotes the separation of concerns, allowing developers to focus on specific aspects of the application independently. This separation enhances code reusability, maintainability, and testability.
- Modularity: MVC encourages modular development, enabling developers to divide the application into smaller, manageable components. Each component can be developed, tested, and maintained separately, facilitating collaboration among team members.
- Scalability: By separating the presentation logic (View) from the business logic (Model) and application logic (Controller), MVC architecture provides scalability and flexibility, allowing applications to evolve and grow over time.

- Code Reusability: MVC facilitates code reusability by abstracting common functionalities into reusable components. This reduces code duplication and promotes consistency across the application.

2.4.6. Interacting with the Server using RESTful API

A Web API is a method used to allow different applications to communicate and exchange data. The data returned by a Web API is typically in JSON or XML format via the HTTP or HTTPS protocol.

Key Features:

- Full support for RESTful methods: Web APIs provide comprehensive support for all RESTful methods, including GET, POST, PUT, and DELETE, making it easy and quick to build HTTP services.
- Complete support for HTTP components: Web APIs also support various HTTP components such as URI, request/response, headers, caching, versioning, and content format.
- Automating tasks: With Web APIs, we can automate task management, workflow updates, leading to increased productivity and efficiency.
- Flexible integration capabilities: It allows easy content retrieval from a website if permitted, enhancing user experience. APIs act as a gateway, allowing companies to share selected information while avoiding unwanted requests.

RESTful API is a widely adopted standard for designing APIs in web applications (Web services) to streamline resource management. It emphasizes managing system resources such as text files, images, sounds, videos, or dynamic data, and facilitates the transmission of formatted resource states through HTTP.

Additionally, the figure 2.11 below provides a visual representation of RESTful API.

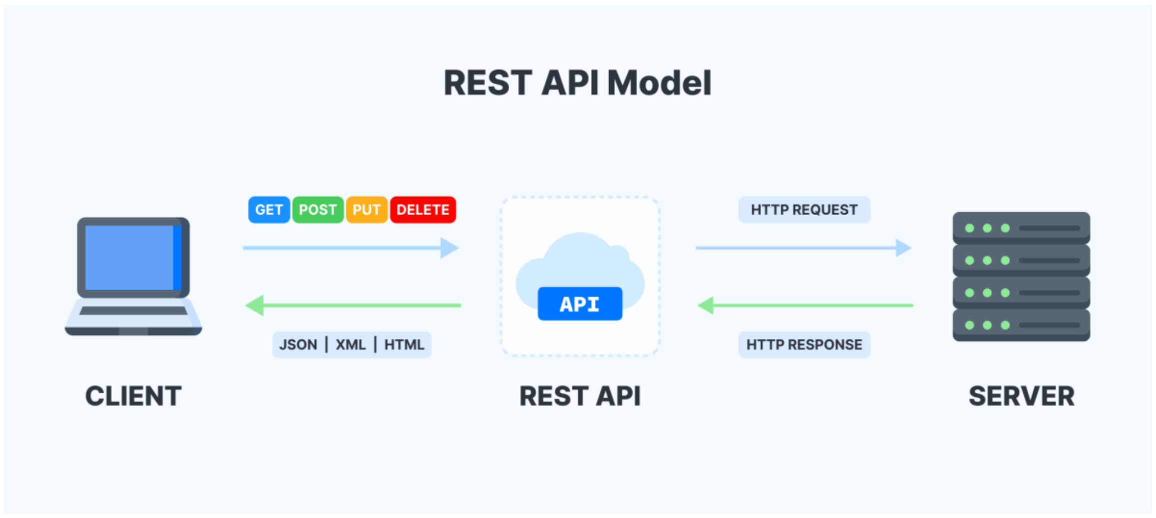


Figure 2.11: RestfulAPI

The most important function of REST is to specify the use of HTTP methods (such as GET, POST, PUT, DELETE...) and URL formatting for web applications to manage resources. RESTful does not define application logic code and is not limited by the application programming language; any language or framework can be used to design a RESTful API.

Operation Mechanism REST primarily operates based on the HTTP protocol. The basic operations outlined below utilize specific HTTP methods:

- GET: Returns a Resource or a list of Resources.
- POST: Creates a new Resource.
- PUT: Updates information for a Resource.
- PATCH: Updates information for a part of a Resource.
- DELETE: Deletes a Resource.

Some status codes when requesting an API are described in figure 2.12 below

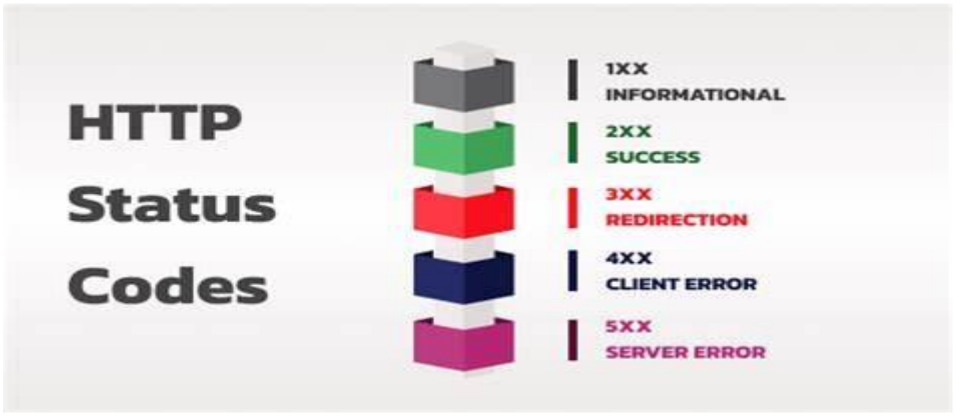


Figure 2.12: Status code in REST

- 200 OK: Indicates success for GET, PUT, PATCH, or DELETE methods.
- 304 Not Modified: The client can use cached data.
- 400 Bad Request: The request is invalid.
- 401 Unauthorized: Authentication is required for the request.
- 403 Forbidden: Access is forbidden, permission denied.
- 404 Not Found: The requested resource could not be found at the given URI.
- 405 Method Not Allowed: The method is not allowed for the current user.
- 410 Gone: The resource no longer exists, and the previous version is no longer supported.
- 429 Too Many Requests: The request is rejected due to rate limiting.

Implementing API in the Project:

Designing REST API URIs: The URLs are designed following a convention as shown below. This design approach emphasizes a clear principle of using request methods to indicate the API's task. The URI does not necessarily need to contain verbs like create, get, update, delete. The resource name should be in plural form.

POST: /v1/api/summarization -> Summarizing PDF file content

GET: /v1/api/summarization -> Get all history PDF summarization

GET: /v1/api/product/:id -> Get details of a specific summarization with the given id

Some Conventions for API Implementation

- Use appropriate status codes: If the API encounters an error, return the correct status code instead of always returning status 2xx.
- Avoid using underscore (_), use hyphen (-) instead.
- Use lowercase in URIs.

Avoid using file extensions in URIs (e.g., .html, .xml, .json, etc.).

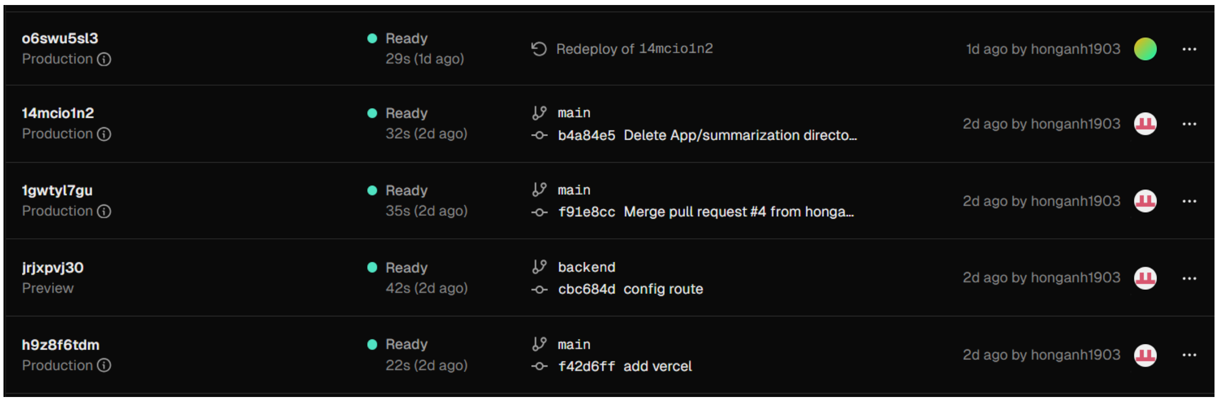
2.4.7. Deploy with Vercel

a) CI/CD with Vercel

Continuous Integration and Continuous Deployment (CI/CD) is a crucial methodology for managing and deploying source code continuously, playing a vital role in web application development. Vercel, a leading cloud service, provides a robust and user-friendly CI/CD solution.

During the Continuous Integration (CI) process, Vercel automatically tests and builds source code from the GitHub repository. This ensures that every change is automatically verified, maintaining the stability and quality of the application. Vercel's Continuous Deployment (CD) process further automates the deployment of new application versions to the production environment quickly and reliably.

By leveraging Vercel's CI/CD capabilities (figure 2.13 below), development teams can ensure seamless integration, consistent testing, and rapid deployment, leading to more stable and high-quality web applications.



o6swu5sl3 Production ⓘ	● Ready 29s (1d ago)	Redeploy of 14mcio1n2	1d ago by honganh1903
14mcio1n2 Production ⓘ	● Ready 32s (2d ago)	main b4a84e5 Delete App/summarization directo...	2d ago by honganh1903
1gwtyl7gu Production ⓘ	● Ready 35s (2d ago)	main f91e8cc Merge pull request #4 from honga...	2d ago by honganh1903
jrjxpvj30 Preview	● Ready 42s (2d ago)	backend cbc684d config route	2d ago by honganh1903
h9z8f6tdm Production ⓘ	● Ready 22s (2d ago)	main f42d6ff add vercel	2d ago by honganh1903

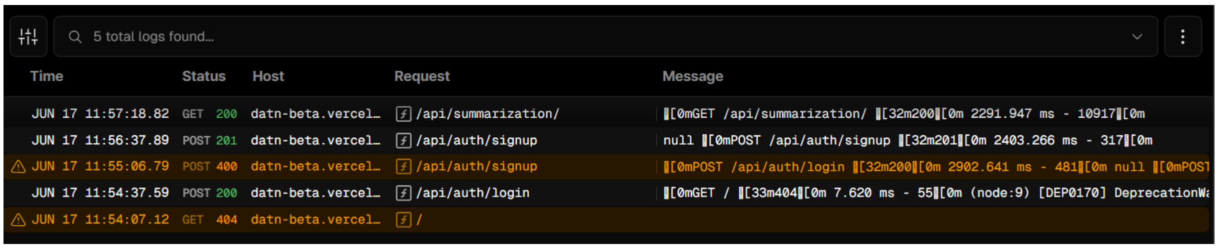
Figure 2.13: CI/CD with Vercel

b) Logging with Vercel

Vercel provides real-time interactive logs, allowing for immediate monitoring of events and interactions within your application. This feature enables quick detection and resolution of issues as they arise.

Additionally, Vercel supports trace logs, which allow for tracking HTTP requests through your application. This capability helps in understanding the handling and response times of various requests, providing insights into the application's performance and behavior.

Figure 2.14 below is an example of Vercel logging with my project:



```
5 total logs found...

Time                  Status    Host                 Request                   Message
JUN 17 11:57:18.82    GET  200  datn-beta.vercel…    /api/summarization/       ▮[0mGET /api/summarization/ ▮[32m200▮[0m 2291.947 ms - 10917▮[0m
JUN 17 11:56:37.89    POST 201  datn-beta.vercel…    /api/auth/signup          null ▮[0mPOST /api/auth/signup ▮[32m201▮[0m 2403.266 ms - 317▮[0m
⚠ JUN 17 11:55:06.79  POST 400  datn-beta.vercel…    /api/auth/signup          ▮[0mPOST /api/auth/login ▮[32m200▮[0m 2902.641 ms - 481▮[0m null ▮[0mPOST
JUN 17 11:54:37.59    POST 200  datn-beta.vercel…    /api/auth/login           ▮[0mGET / ▮[33m404▮[0m 7.620 ms - 55▮[0m (node:9) [DEP0170] DeprecationWa
⚠ JUN 17 11:54:07.12  GET  404  datn-beta.vercel…    /
```

Figure 2.14: Logging with Vercel

CHAPTER 3 : SYSTEM DEPLOYMENT AND CONSTRUCTION

3.1. Interface and functions of the system

3.1.1. Login

Below is figure 3.1, which displays the login screen of the application.

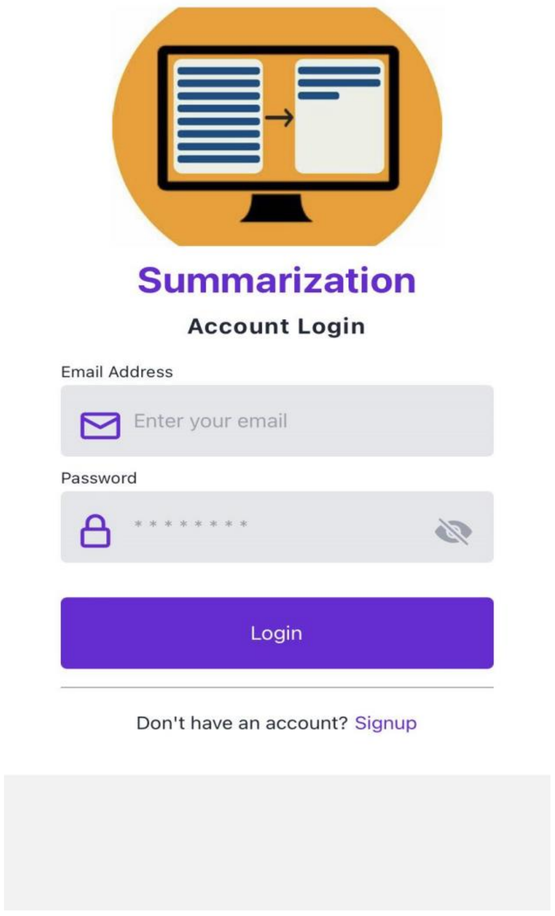


Figure 3.1: Login screen

Table 3.1: Login Function Information Details

Screen	Login		
Description	The system allows only users with registered accounts to access its features. Therefore, every action requires users to log in before proceeding. At this point, users can log in or register for a new account.		
Screen Access	Upon accessing the application, users who are not logged in will be directed to the [Login] screen.		
Screen Content			
Item	**Type**	**Data**	**Description**

Email	Text		The field reserved for users to input their email address.
Password	Password		The field reserved for users to input their password.
Login	Button		Sign in button.
Sign Up	Button		Register button if you don't have an account.
Screen Actions			
Action Name	**Description**	**Success**	**Failure**
Click [Login] button	Logging into the application.	Navigate to the homepage of the application.	Display the message: "Incorrect email/password. Please try again.
Click [Signup] button	Register for a new account.	Navigate to the registration page.	

3.1.2. Sign in

The image below, labeled as figure 3.1, illustrates the application's login screen

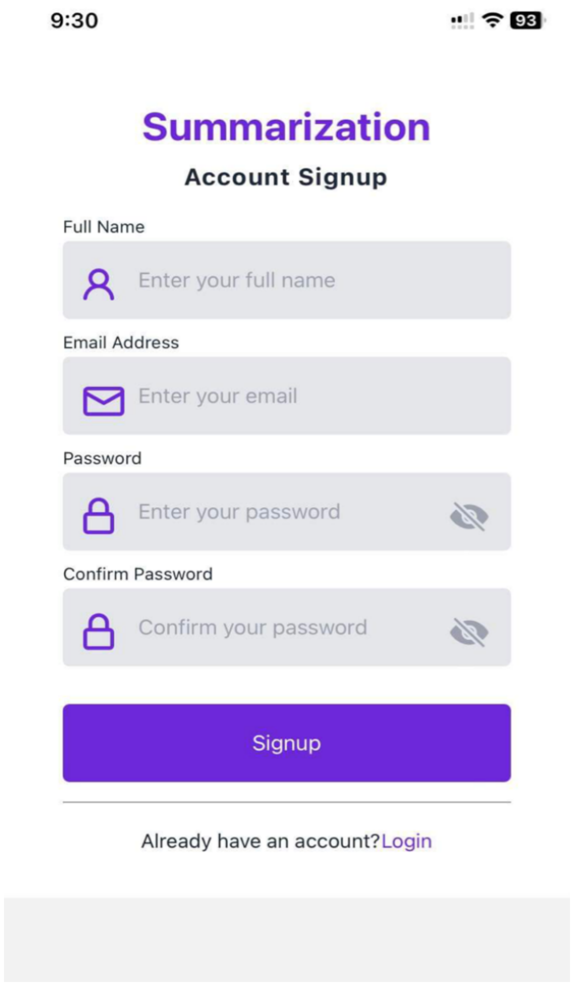


Figure 3.2: Sign up screen

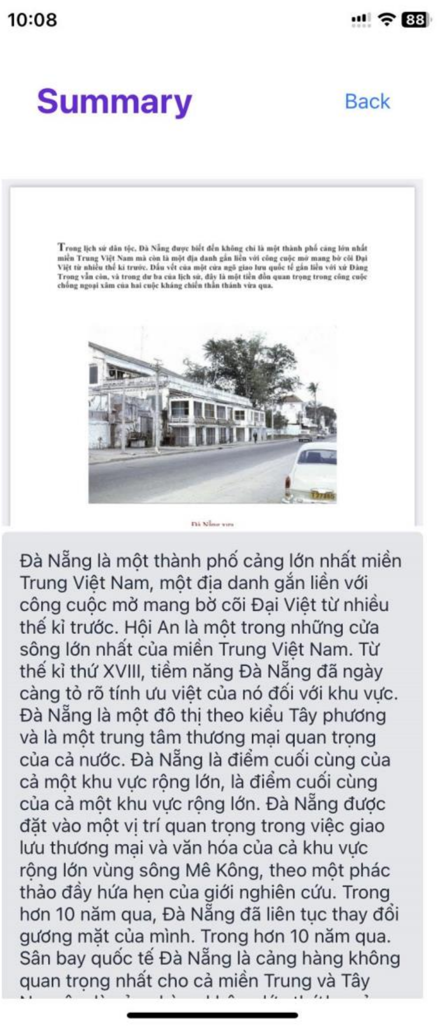
Table 3.2: Signup Function Information Details

Screen	Sign up		
Description	Users need to have a registered account to utilize the system's functionalities. Hence, any action necessitates logging in beforehand. At this juncture, users can either sign in or create a new account.		
Screen Access	Select "Sign Up" on the login screen.		
Screen Content			
Item	**Type**	**Data**	**Description**
Email	Text		The field reserved for users to input their email address.

Full name	Text		The field reserved for users to input their full name.
Password	Password		The field reserved for users to input their password.
Confirm password	Password		The field reserved for users to input their confirm password.
Login	Button		Sign up button.
Login	Button		Login button if you already have an account.
Screen Actions			
Action Name	**Description**	**Success**	**Failure**
Click [Signup] button	Creating a new account.	Navigate to the homepage of the application.	Display the message: "Incorrect credentials. Please try again”.
Click [Login] button	Login to application.	Navigate to the login page.	

3.1.3. Summarization

Figure 3.3 below illustrates the summary display screen



Trong lịch sử dân tộc, Đà Nẵng được biết đến không chỉ là một thành phố cảng lớn nhất miền Trung Việt Nam mà còn là một địa danh gắn liền với công cuộc mở mang bờ cõi Đại Việt từ nhiều thế kỉ trước. Dấu vết của một cửa ngõ giao lưu quốc tế gắn liền với xứ Đàng Trong vẫn còn, và trong dư ba của lịch sử, đây là một tiền đồn quan trọng trong công cuộc chống ngoại xâm của hai cuộc kháng chiến thần thánh vừa qua.

Đà Nẵng là một thành phố cảng lớn nhất miền Trung Việt Nam, một địa danh gắn liền với công cuộc mở mang bờ cõi Đại Việt từ nhiều thế kì trước. Hội An là một trong những cửa sông lớn nhất của miền Trung Việt Nam. Từ thế kì thứ XVIII, tiềm năng Đà Nẵng đã ngày càng tỏ rõ tính ưu việt của nó đối với khu vực. Đà Nẵng là một đô thị theo kiểu Tây phương và là một trung tâm thương mại quan trọng của cả nước. Đà Nẵng là điểm cuối cùng của cả một khu vực rộng lớn, là điểm cuối cùng của cả một khu vực rộng lớn. Đà Nẵng được đặt vào một vị trí quan trọng trong việc giao lưu thương mại và văn hóa của cả khu vực rộng lớn vùng sông Mê Kông, theo một phác thảo đầy hứa hẹn của giới nghiên cứu. Trong hơn 10 năm qua, Đà Nẵng đã liên tục thay đổi gương mặt của mình. Trong hơn 10 năm qua. Sân bay quốc tế Đà Nẵng là cảng hàng không quan trọng nhất cho cả miền Trung và Tây

Figure 3.3: Summarization screen

Table 3.3: Summarization Function Information Details

Screen	Summarization		
Description	Users can access this section to view their summary history and create new summaries after uploading a file.		
Screen Access	Select "Upload file" on the login screen.		
Screen Content			
Item	**Type**	**Data**	**Description**
File	PDF file		Display the content of the PDF file.

Summarization text	Text		Display the summary content of the file
Back	Button		Button to return to the previous page.
Screen Actions			
Action Name	**Description**	**Success**	**Failure**
Click [Back] button	Return to the previous page.	Navigate to the previous of the application.	

3.2. Evaluation method.

I have chosen Rouge to evaluate the model.

3.2.1. What is Rouge?

Rouge is a framework used to assess the quality of text summaries generated by a machine learning model. It provides various metrics to evaluate different aspects of the summarization process. These metrics include:

- Rouge-N: Measures the overlap of N-grams (sequences of N words) between the generated summary and the reference summary.
- Rouge-L: Computes the longest common subsequence between the generated summary and the reference summary, taking into account the length of the subsequence.
- Rouge-W: Calculates the weighted harmonic mean of precision and recall of overlapping units, where the weight is determined by the length of the unit.

Rouge allows us to quantify the effectiveness of the summarization process by providing detailed information on the quality of the generated summaries. It can be integrated into our automated testing tools to ensure continuous evaluation and performance monitoring.

By using Rouge, we aim to assess the performance of our system in generating accurate and informative summaries, thereby gaining insights into the effectiveness of our summarization model.

3.2.2. *Evaluation criteria for model*

ROUGE-1, ROUGE-2, and ROUGE-L

- Description: ROUGE (Recall-Oriented Understudy for Gisting Evaluation) metrics measure the overlap between the generated summaries and reference summaries.
- Components:
  - ROUGE-1: Evaluates the overlap of unigrams (single words).
  - ROUGE-2: Evaluates the overlap of bigrams (two consecutive words).
  - ROUGE-L: Evaluates the longest common subsequence to consider sentence-level structure similarity.
- Purpose: Higher ROUGE scores indicate better quality and content coverage in the generated summaries.

Sentence Tokenization

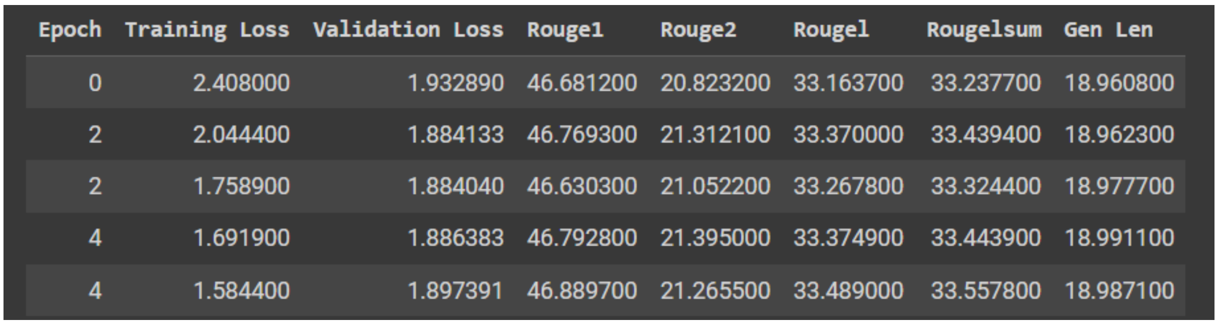
- Description: Ensures that the ROUGE metric operates at the sentence level for accurate evaluation.
- Implementation: Uses nltk.sent_tokenize to split the text into sentences.
- Purpose: Proper tokenization helps in accurately comparing sentence boundaries in generated and reference summaries.

Average Length of Generated Summaries

- Description: Measures the average length of the summaries generated by the model.
- Calculation: Computes the length of each summary by counting non-padding tokens and averaging them.
- Purpose: Provides insight into the brevity or verbosity of the generated summaries, helping to understand the model's summarization behavior.

3.2.3. *Evaluation of Training Results*

The training of the text summarization model was evaluated based on the training loss, validation loss, and various ROUGE metrics over multiple epochs. Figure 3.4 below is a summary of the results:



Epoch	Training Loss	Validation Loss	Rouge1	Rouge2	Rougel	Rougelsum	Gen Len
0	2.408000	1.932890	46.681200	20.823200	33.163700	33.237700	18.960800
2	2.044400	1.884133	46.769300	21.312100	33.370000	33.439400	18.962300
2	1.758900	1.884040	46.630300	21.052200	33.267800	33.324400	18.977700
4	1.691900	1.886383	46.792800	21.395000	33.374900	33.443900	18.991100
4	1.584400	1.897391	46.889700	21.265500	33.489000	33.557800	18.987100

Figure 3.4: Rouge score

Training and Validation Loss:

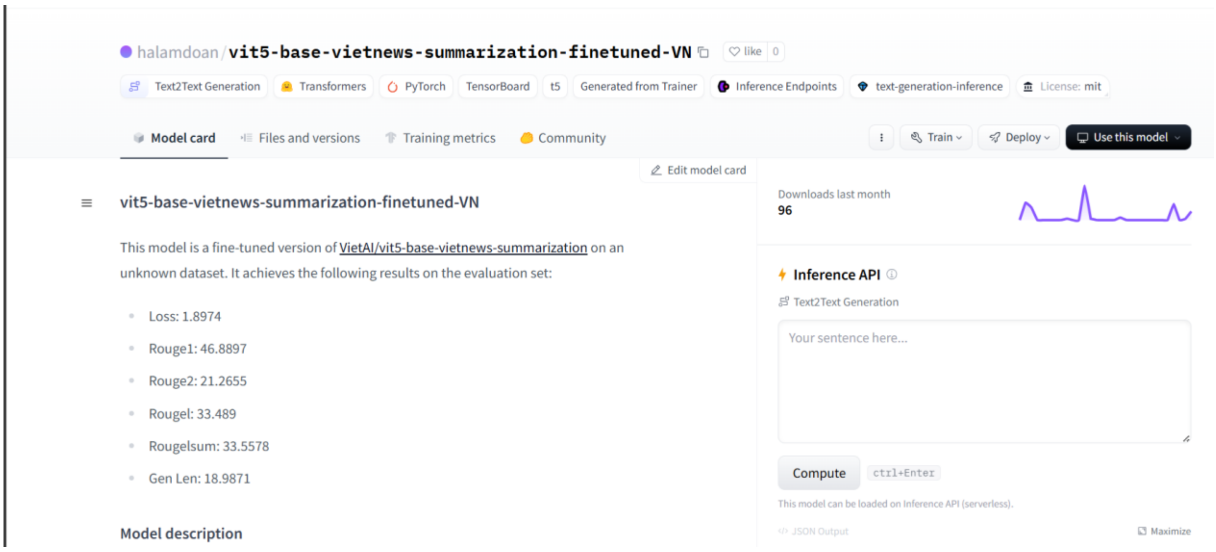
- Training Loss: There is a consistent decrease in training loss from 2.4080 to 1.5844 over the epochs, indicating that the model is learning and improving its performance on the training data.
- Validation Loss: The validation loss decreases initially from 1.9329 to 1.8840 by the second epoch but slightly increases towards the fourth epoch to 1.8974. This could indicate a slight overfitting as the model becomes more specialized to the training data.

ROUGE Scores:

- ROUGE-1: The ROUGE-1 score slightly improves from 46.6812 to 46.8897, showing an improvement in capturing the overall content.
- ROUGE-2: The ROUGE-2 score increases from 20.8232 to 21.3950, indicating a better performance in capturing bi-gram level content, although with some fluctuations.
- ROUGE-L: The ROUGE-L score increases from 33.1637 to 33.4890, reflecting an improvement in maintaining the sequence and structure of the content.
- ROUGE-Lsum: This score shows similar trends as ROUGE-L, suggesting consistency in summarization quality.
- Generation Length (Gen Len): The average length of the generated summaries remains consistent around 18.96 to 18.99 tokens across the epochs, indicating that the model maintains a stable length for its generated summaries.

The training results indicate that the model is effectively learning to generate summaries with high content retention and structure maintenance, as evidenced by the improving ROUGE scores. The consistent training and validation loss, along with stable generation lengths, suggest a well-balanced training process with minimal overfitting. The slight increase in validation loss towards the end needs to be monitored, and potentially addressed, to ensure robust performance on unseen data. Overall, the model shows promising summarization capabilities with strong performance metrics.

After training, the model will be pushed to Hugging Face, as shown in Figure 3.5 below.



vit5-base-vietnews-summarization-finetuned-VN

This model is a fine-tuned version of VietAI/vit5-base-vietnews-summarization on an unknown dataset. It achieves the following results on the evaluation set:

- Loss: 1.8974
- Rouge1: 46.8897
- Rouge2: 21.2655
- Rougel: 33.489
- Rougelsum: 33.5578
- Gen Len: 18.9871

This model can be loaded on Inference API (serverless).

Model description

Figure 3.5: Model has been uploaded to Hugging Face.

CONCLUSION AND DEVELOPMENT DIRECTION

Conclusion

During the development of the PDF content summarization application using NodeJs and React Native, I have achieved several significant milestones and overcome numerous challenges. The key achievements are as follows:

- Successfully fine-tuned the ViT5 pre-trained model for summarizing PDF file content.
- Achieved notable ROUGE scores, indicating effective summarization capabilities.
- Automated the content summarization process, reducing the time required for manual content review.
- Implemented a user-friendly interface for uploading and viewing summarized content.
- Overcame integration challenges between NodeJs and React Native to create a seamless application experience.

Despite these successes, some limitations were observed:

- The system needs further optimization for faster processing.
- The quality of summaries can be further improved for better accuracy.

Development direction

Due to the limited timeframe for product development, the system currently has basic functionalities. There are several potential future enhancements for this system:

- Deploy the system to a specific web address for broader public access.
- Improve system processing speed.
- Enable querying documents in addition to summarizing them.

REFERENCES

[1] ViT5 Document, date accessed: 05/2024
Url: https://arxiv.org/abs/2205.06457

[2] Illustrated Guide to Transformers, date accessed: 05/2024
Url: https://towardsdatascience.com/illustrated-guide-to-transformers-step-by-step-explanation-f74876522bc0

[3] What are large language models (LLMs)?, date accessed: 05/2024
Url: https://www.ibm.com/topics/large-language-models,

[4] Transformers Architecture, date accessed: 05/2024
Url: https://medium.com/inside-machine-learning/what-is-a-transformer-d07dd1fbec04,

[5] Official Node.js website, date accessed: 05/2024
Url: https://nodejs.org/docs/latest/api/

[6] React Native Document, date accessed: 05/2024
Url: https://reactnative.dev/docs/environment-setup,

[7] Text Summarization in NLP, date accessed: 05/2024
Url: https://www.geeksforgeeks.org/text-summarization-in-nlp/

APPENDIX1

APPENDIX2