**VIETNAM GENERAL CONFEDERATION OF LABOR**

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**REPORT**

**COMPUTER VISION**

*Instructor*: **PhD. PHAM VAN HUY**

*Student*: **BUI HAI DUONG - 521H0220**

**PHAN MINH HOANG – 521H0501**

**LA NGUYEN QUOC THINH – 521H0513**

*Class*: **21H50302**

**HO CHI MINH CITY, 2025**

**VIETNAM GENERAL CONFEDERATION OF LABOR**

**TON DUC THANG UNIVERSITY**

**FACULTY OF INFORMATION TECHNOLOGY**



**REPORT**

**COMPUTER VISION**

*Instructor*: **PhD. PHAM VAN HUY**

*Student*: **BUI HAI DUONG - 521H0220**

**PHAN MINH HOANG – 521H0501**

**LA NGUYEN QUOC THINH – 521H0513**

*Class*: **21H50302**

**HO CHI MINH CITY, 2025**

ACKNOWLEDGEMENT

To complete this essay, besides our own efforts, we have received a lot of help in terms of knowledge, experience, and skills from the school and teachers. First and foremost, we would like to express my special gratitude to PhD Pham Van Huy - the lecturer of Introduction to Computer Vision course who has taught us valuable knowledge of the subject. That knowledge is the foundation for us to continue learning and effectively apply it to this essay. Additionally, We would like to thank the teacher for allowing us to complete this essay, which has helped us to further develop my understanding of the subject. Thank you for your guidance and support in helping us to complete this essay to the best of my ability. Moreover, we would also like to express our gratitude to the school and the teachers who have compiled the Introduction to Computer Vision materials, providing me with useful resources for research and essay writing. Thank you sincerely!

**COMPLETION OF THESIS**

**AT TON DUC THANG UNIVERSITY**

We here by certify that this thesis is my/our own work and was conducted under the guidance of PhD. Pham Van Huy. The research and results presented in this thesis are truthful and have not been published previously in any form. The data presented in tables and figures used for analysis, comments, and evaluations were collected by the author from various sources and are clearly cited in the reference section.

Moreover, this thesis includes some comments, evaluations, and data from other authors and organizations, which are properly cited and referenced.

If any misconduct is detected, I fully take responsibility for the content of my thesis. Ton Duc Thang University is not liable for any copyright infringement that may occur during the thesis completion process.

*Ho Chi Minh City, October 22, 2023*

*Author*

*(signature and full name)*

ACKNOWLEDGEMENT AND EVALUATION SECTION BY INSTRUCTOR

**Instructor's Acknowledgement Section**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh City, 2024

(signature and full name)

**Instructor's Evaluation Section**

\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_

Ho Chi Minh City, 2024

(signature and full name)

SUMMARY

The task of this documents is about how to deal with the Image captioning task, introducing two main approaching ways are Merging and Injection architecture to solve the problems. This document separated into 5 main chapters:

Chapter 1: Introduction – This chapter covers main general ideas to deal with the image captioning task.

Chapter 2: Preprocessing – Describe about how raw images and captions is preprocessed

Chapter 3: Model architecture – Showing the main architecture used in the implementation and the mathematical behind.

Chapter 4: Environmental setup – Showing the base setup before training and evaluating the model.

Chapter 5: Result – Showing some result after the training process.

Table of Contents

[ACKNOWLEDGEMENT i](#_Toc190005987)

[ACKNOWLEDGEMENT AND EVALUATION SECTION BY INSTRUCTOR iii](#_Toc190005988)

[SUMMARY iv](#_Toc190005989)

[**TABLE OF FIGURES** 1](#_Toc190005990)

[ABBREVIATIONS 2](#_Toc190005991)

[**CHAPTER 1: INTRODUCTION AND TOPIC OVERVIEW** 3](#_Toc190005992)

[**1.1 Chosen topic** 3](#_Toc190005993)

[**1.2 Objectives of implementing the topic** 3](#_Toc190005994)

[**CHAPTER 2: APPLICATION METHODS** 4](#_Toc190005995)

[**2.1 Approaching ways** 4](#_Toc190005996)

[**2.1.1 Fine-tuning** 4](#_Toc190005997)

[**2.1.2 Retrieval Augmented Generation (RAG)** 4](#_Toc190005998)

[**2.1.3 Pros and cons of the two methods** 5](#_Toc190005999)

[**2.1.4 Appropriate approaches** 6](#_Toc190006000)

[**2.2 Model** 7](#_Toc190006001)

[**2.2.1 Embedding model** 7](#_Toc190006002)

[**2.2.2 Generative model** 9](#_Toc190006003)

[**2.3 System components** 10](#_Toc190006004)

[**2.3.1 Retrieval Module** 10](#_Toc190006005)

[**2.3.2 Generative Module** 10](#_Toc190006006)

[**2.3.2.1 Generative Module** 10](#_Toc190006007)

[**CHAPTER 3: EXPERIMENTAL SETUP** 11](#_Toc190006008)

[**4.1 Encoder** 11](#_Toc190006009)

[**4.2 Decoder** 12](#_Toc190006010)

[**4.3 Attention** 13](#_Toc190006011)

[**CHAPTER 4: EXPERIMENTAL RESULTS** 15](#_Toc190006012)

[**5.1 Dataset** 15](#_Toc190006013)

[**5.2 Code implementation** 15](#_Toc190006014)

[**5.2.1 Model** 15](#_Toc190006015)

[**5.2.2 Configuration** 15](#_Toc190006016)

[**CHAPTER 5: RESULT** 16](#_Toc190006017)

[**REFERENCES** 18](#_Toc190006018)

**TABLE OF FIGURES**

[Figure 1. Injection architecture 2](#_Toc186957164)

[Figure 2. Merging architecture 3](#_Toc186957165)

[Figure 3. Injection architecture workflow 7](#_Toc186957166)

[Figure 4. LSTM architecture 8](#_Toc186957167)

[Figure 5. Result 1 12](#_Toc186957168)

[Figure 6. Result 2 12](#_Toc186957169)

[Figure 7. Result 3 13](#_Toc186957170)

[Figure 8. Result 4 13](#_Toc186957171)

# ABBREVIATIONS

|  |  |
| --- | --- |
| CNN | Convolutional Neural Network |
| RNN | Recurrent Neural Network |
| LSTM | Long-short Term Memory |
| NLP | Natural Language Processing |

**CHAPTER 1: INTRODUCTION AND TOPIC OVERVIEW**

**1.1 Chosen topic**

In 2025, traffic laws have undergone significant changes, greatly impacting people's daily lives, especially with increased penalties as stipulated in Decree 168, which took effect on January 1, 2025. Although road traffic is an essential part of everyday life, it is challenging for citizens to fully grasp all traffic regulations due to the large number of laws, circulars, and decrees issued. Not everyone has the time and effort to study and absorb all these legal documents. Therefore, developing and implementing a chatbot for road traffic laws is an effective solution, making legal information more accessible while saving time and effort. This helps improve people's knowledge and awareness when participating in traffic.

**1.2** **Objectives of implementing the topic**

The objective of this study is to research, design, and develop an intelligent chatbot capable of automatically answering questions related to traffic laws while also assisting users in resolving specific situations they may encounter while participating in traffic. This chatbot not only helps improve awareness of traffic regulations but also provides quick, accurate, and easy-to-understand information, supporting users in complying with the rules and minimizing traffic violations.

**CHAPTER 2: APPLICATION METHODS**

**2.1 Approaching ways**

There are typically two approaches to solving the chatbot problem: fine-tuning a pre-trained model or using Retrieval-Augmented Generation (RAG). Both approaches leverage our available data to answer user queries when interacting with the chatbot, but they differ in their implementation.

**2.1.1 Fine-tuning**

Fine-tuning a pre-trained model involves utilizing models that have already been trained on large datasets. These models are then further trained on our specific dataset, allowing them to adapt and perform well on our particular task. This approach enables the model to leverage prior knowledge while refining its understanding based on domain-specific data.

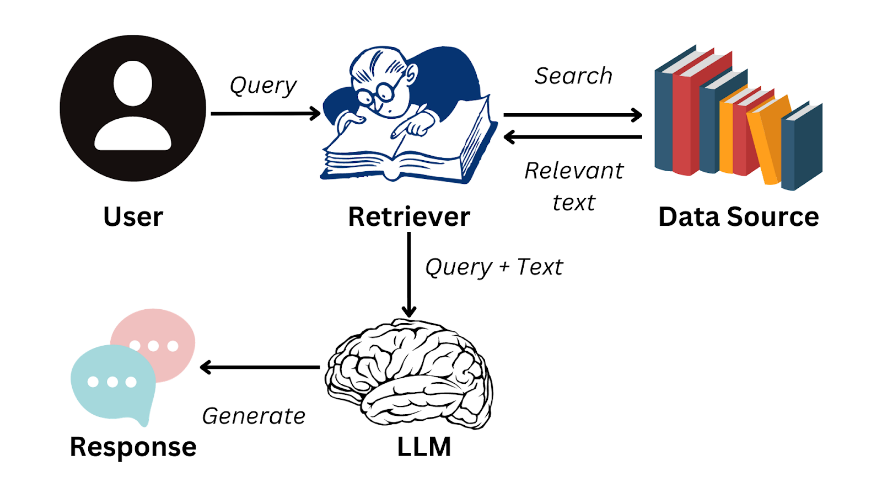
A diagram of a function

Description automatically generated

The image above illustrates the fine-tuning process of a model. Starting from pre-trained models, we continue training them on our specific dataset. Through the backpropagation process, the model's weights and biases across layers are updated, allowing it to better adapt to the specific task we aim to perform.

**2.1.2 Retrieval Augmented Generation (RAG)**

The operation of RAG differs from fine-tuning a model. In terms of functionality, RAG feeds a Large Language Model (LLM) with both the user's query and relevant retrieved documents, enabling the LLM to generate a response based on the provided context. This approach enhances the accuracy and relevance of responses without requiring extensive retraining of the model.



The image above illustrates the workflow of Retrieval-Augmented Generation (RAG). First, our data is stored in one or more databases. When a user submits a query, the system searches for relevant documents within the stored database. The retrieved documents, along with the user's query, are then fed directly into the model, enabling it to generate a response based on the retrieved information. This approach ensures that the model provides accurate and up-to-date answers without requiring extensive fine-tuning.

**2.1.3** **Pros and cons of the two methods**

Pros:

Fine-tuning a Pre-trained Model:

* The fine-tuned model is optimized for a specific task, ensuring high accuracy.
* Leverages previously trained data, reducing the need for training from scratch.
* Since the model relies solely on learned knowledge without external retrieval, response time is very fast.

Retrieval-Augmented Generation (RAG):

* New data can be updated easily by adding it to the database without retraining the model.
* Eliminates the need for model retraining, saving both time and computational costs.
* Responses are based directly on retrieved data, reducing hallucination and improving factual accuracy.

Cons:

Fine-tuning a Pre-trained Model:

* Retraining a model requires significant time and computational resources.
* Updating new information requires additional training on the latest dataset.
* The model may suffer from hallucination, generating inaccurate responses based on outdated or incomplete learned knowledge.

Retrieval-Augmented Generation (RAG):

* Response time is slower due to the need to retrieve and process relevant information.
* Poor retrieval quality can lead to incomplete or inaccurate responses.
* The system is more complex, involving multiple integrated components (retrieval, ranking, and generation).

**2.1.4** **Appropriate approaches**

Since the task involves traffic law regulations, the solution must meet the following criteria:

* Legal data must be up-to-date and easily maintainable.
* Responses must be accurate and strictly based on official legal documents.

While fine-tuning a model allows for updating new data, it requires significant time and computational costs. Additionally, fine-tuned models may suffer from hallucinations, potentially generating inaccurate responses.

On the other hand, RAG provides a more efficient and reliable solution since:

* Updating legal data is quick and straightforward by simply adding new documents to the database.
* Responses are generated based on directly retrieved legal documents, ensuring accuracy, and minimizing hallucination risks.

Given these advantages, RAG is the most suitable approach for building a chatbot that provides accurate and up-to-date legal information.

**2.2 Model**

**2.2.1 Embedding model**

The main function of the embedding model is to convert tokens such as images, videos, characters, or sounds into vector representations, which helps make data processing in machine learning easier.

In RAG, this embedding process occurs during the information retrieval step. Text chunks are extracted from the main documents, then embedded into vectors and stored in a database. When user queries are input, they are converted into vectors and compared with the vectors stored in the database to retrieve relevant data.

Below is a summary of the results of embedding models based on 20% of the Zalo law training set, which has not been trained on the models listed below. The results show evaluations of Accuracy, Precision, and Recall:

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Accuracy@1 | Accuracy@3 | Accuracy@5 | Accuracy@10 |
| vietnamese-bi-encoder | 0.8169 | 0.9108 | 0.9437 | 0.9640 |
| sup-SimCSE-VietNamese-phobert-base | 0.5540 | 0.7308 | 0.7981 | 0.8748 |
| halong\_embedding  (768) | 0.8294 | 0.9233 | 0.9437 | 0.9687 |
| halong\_embedding (512) | 0.8138 | 0.9233 | 0.9390 | 0.9703 |
| halong\_embedding (256) | 0.7934 | 0.8967 | 0.9280 | 0.9593 |
| halong\_embedding (128) | 0.7840 | 0.8951 | 0.9264 | 0.9515 |
| halong\_embedding (64) | 0.6980 | 0.8435 | 0.8920 | 0.9358 |

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Model | Precision@1 | Precision@3 | Precision@5 | Precision@10 |
| vietnamese-bi-encoder | 0.8169 | 0.3099 | 0.1931 | 0.0987 |
| sup-SimCSE-VietNamese-phobert-base | 0.5540 | 0.2473 | 0.1621 | 0.0892 |
| halong\_embedding  (768) | 0.8294 | 0.3146 | 0.1931 | 0.0991 |
| halong\_embedding (512) | 0.8138 | 0.3146 | 0.1922 | 0.0992 |
| halong\_embedding (256) | 0.7934 | 0.3062 | 0.1900 | 0.0981 |
| halong\_embedding (128) | 0.7840 | 0.3046 | 0.1894 | 0.0975 |
| halong\_embedding (64) | 0.6980 | 0.2864 | 0.1815 | 0.0954 |
| Model | Recall@1 | Recall@3 | Recall@5 | Recall@10 |
| vietnamese-bi-encoder | 0.8020 | 0.9045 | 0.9390 | 0.9601 |
| sup-SimCSE-VietNamese-phobert-base | 0.5446 | 0.7246 | 0.7903 | 0.8693 |
| halong\_embedding  (768) | 0.8146 | 0.9178 | 0.9390 | 0.9640 |
| halong\_embedding (512) | 0.7989 | 0.9178 | 0.9343 | 0.9656 |
| halong\_embedding (256) | 0.7786 | 0.8920 | 0.9233 | 0.9546 |
| halong\_embedding (128) | 0.7707 | 0.8889 | 0.9210 | 0.9476 |
| halong\_embedding (64) | 0.6854 | 0.8365 | 0.8842 | 0.9311 |

Based on the statistics above, we can see that the embedding model yielding the highest result is halong\_embedding (768). Therefore, halong\_embedding (768) will be the chosen model for use as the embedding model.

**2.2.2 Generative model**

Generative models are large language models (LLMs) capable of creating new content based on the data they have been trained on and user input. These models are trained on vast datasets that cover a wide range of topics to understand and generate natural language text.

In the context of the RAG system, popular generative models come from organizations like OpenAI, Google, Meta, as well as open-source models on GitHub or HuggingFace.

In particular, Google's Gemini model offers versions like Gemini 1.5 Flash and Gemini 1.5 Pro, which can handle long contexts, supporting context windows of up to 1 million and 2 million tokens, respectively. This allows for the creation of detailed and complex prompts.

To use these models, users need an API key provided through a Google account. The choice of model will depend on the specific requirements of the project, including task complexity, context limits, and budget.

**2.3 System components**

Typically, in a RAG system, there are two main components: the Retrieval Module and the Generation Module. The Retrieval Module is responsible for retrieving documents that are most relevant to the user’s query, while the Generation Module uses the retrieved documents to generate a response based on the user’s question and the previously retrieved texts. This RAG system is enhanced with advanced techniques to optimize the results provided to the user.

**2.3.1 Retrieval Module**

**2.3.1.1 Chunking**

The input of the chunking process consists of documents collected through searches on websites, books, and reliable information sources. Data stored in file formats like PDF, DOCX, and Markdown will be chunked for easier storage and retrieval. The output of this process is the chunks created from the original texts.

There are various chunking methods depending on the problem and the type of text being processed. Many chunking methods are used in text for different purposes. Common chunking methods that are widely used include:

* Fixed-size chunking: Divides the text into segments of fixed size.
* Recursive chunking: Splits the text according to segmenting markers (paragraphs, headings, etc.) recursively until the desired size is achieved.
* Document-based chunking: Used for documents with special structures (Markdown, source code, tables, etc.).
* Semantic chunking: Groups sentences based on semantic similarity to cluster related sentences.
* Agentic chunking: Uses a large language model (LLM) to decide how to group text segments based on context.

**2.3.1.2 Storing**

Storing is the process of saving the chunks that have been segmented during the chunking process. Depending on the problem, we will divide the data into different storage spaces. To speed up the retrieval process, the chunks are stored in vector spaces (Vectorstore). This process includes the following parts:

* Embedding: Using an embedding model to convert the chunks of text into vectors for storage in the Vectorstore.
* Vectorstore: There are many Vectorstores today that support storing vectors in the cloud, such as FAISS, ChromaDB, Pinecone, etc. Additionally, local Vectorstores can also be used to store vectors, such as PostgresSQL, etc.

**2.3.1.3 Query transform**

This is a technique used to transform user queries into shorter, more concise questions, helping to optimize the process of retrieving chunks more effectively.

This technique is applied by using large language models to perform prompting techniques that convert user queries into more suitable questions. Some techniques for query transformation include:

* Step-back prompting: A method that transforms specific questions into broader, more general questions, improving the ability to query documents.
* Query rewriting: A method that makes initial queries more specific and detailed. This approach helps increase accuracy in the document retrieval process.
* Sub-query decomposition: A method that breaks down complex queries into simpler, component queries. This process simplifies and enhances the retrieval process.

**2.3.1.4 Retrieval methods**

Typically, there are various types of documents, so the choice of retrieval techniques will depend on the nature of the documents. Effective query techniques include:

* Parent document retrieval: This involves breaking down original documents into smaller chunks. Each original document has its own ID, and the chunks will point to the original document when retrieved through the user's query. This technique is optimized for retrieving the entire context of documents, such as legal texts, medical documents, etc.
* Hybrid retrieval: A combination of keyword-based retrieval (sparse retrieval) and semantic-based retrieval (dense retrieval). This method leverages both techniques to optimize the accuracy of retrieving specific chunks based on the user's query. Sparse retrieval is the traditional approach that measures the similarity between the user’s query and documents based on keyword matching. Dense retrieval uses an embedding model to query documents in vector space based on the similarity to the user’s query. Combining these two methods, known as Hybrid retrieval, significantly enhances the accuracy of the retrieval process, improving the results.
* Multi-query retrieval: Transforms the user’s original query into multiple variations with similar meanings, thereby generating more retrieval results.

**2.3.2 Generative Module**

After retrieving the relevant chunks, the user’s question and the related chunks are fed into the LLM to generate an answer and respond to the user. The main steps in the generation module are:

* Creating the input prompt: Combine the user’s question with the retrieved chunks.
* Using the LLM to answer the question: Use models such as T5, ViT5, GPT, Gemini, etc.
* Techniques used to optimize the output:
* Few-shot Chain of Thought (Few-shot CoT): Guides the model through step-by-step reasoning to provide an answer, offering additional specific examples for the model to follow.
* Memory-Augmented RAG: Stores the conversation history to improve interactivity and accuracy in responses.

**CHAPTER 3:** **EXPERIMENTAL SETUP**

**3.1 Data**

The data used in this study were collected from the Legal Library, consisting of 115 DOCX formatted documents. Among these, 61 documents are laws enacted before 2025, and 48 are new laws that came into effect on January 1, 2025. These documents include Laws, Circulars, and Decrees, providing a rich and diverse source of information for the research.

During the preprocessing phase, the entire text is converted to lowercase and all unnecessary whitespace is removed.

**3.2 Build vectorstore and retrieval system**

Traffic law documents are typically structured with sections such as Title, Summary, Articles, Article Titles, and Article Contents. To optimize the information retrieval process, we apply the Parent Document Retrieval method. Specifically, the original law documents are divided into sections based on individual articles. Each article serves as a Parent Document and contains related text segments chunks.

The chunking process is carried out using the Recursive Chunking method, which breaks down the main text into smaller sections based on the hierarchical structure of the document. This helps create appropriately sized chunks for effective storage and retrieval.

By using the Langchain library to assist in building the Parent Document Retrieval system and data storage, Langchain provides tools to implement this technique efficiently.

To store both the embedded vectors and the original text, we use PostgreSQL combined with the pgvector extension. This combination allows for efficient vector storage and retrieval while leveraging PostgreSQL's powerful features, such as ACID compliance and time-based recovery capabilities.

Using Langchain along with PostgreSQL and pgvector makes the system flexible, efficient, and easily scalable during implementation and operation.

**3.3 LLM**

The LLM will be used in three main components: Query transformation, Routing, and the Generation Module. A temperature setting of 0 is used in these models to return the most accurate results.

The prompting method used in all three components is Few-shot CoT, which guides the model step by step while providing specific examples for the model to follow. This method improves the model's output, making the results more accurate.

Query transformation will modify the user’s questions as follows:

Remove terms such as "new law" or "old law" to improve retrieval efficiency.

For law-related questions, if they are too detailed, Step-back prompting will be used to broaden the question, increasing retrieval capability, and providing additional relevant information.

For questions not related to the law, the original query will be kept unchanged.

Routing will lead to three different directions. The Routing directions include:

* A database containing old traffic law documents prior to 2025, if the user’s question is about current laws or mentions new laws.
* A database containing traffic law documents that are effective from January 1, 2025, if the user's question refers to old laws.
* The LLM model will directly answer if the user's question is related to normal chatting.

In the Generation Module, the output will be determined by the following prompt:

* The prompt will include the user's query and the Parent documents linked to chunks relevant to the user’s question. These documents will be attached with metadata, and the metadata will be crucial in formatting the output response for the user.
* The prompt will include the chat history to interact with the user and answer the question more accurately. The information in the chat history can also be used for comparisons.

The main format for the output will be:

* Source document: Document name
* Document title: Title of the document
* Article: Title of the article
* Content:
  + Content of the article

Along with the main format, additional criteria will enhance the aesthetics of the user response:

* Capitalizing the first letter of each line
* Breaking lines between sub-sections: a), b), c), ...

**CHAPTER 4:** **EXPERIMENTAL RESULTS**

**5.1 Dataset**

*Dataset name:* Flickr8k

*Detail about this dataset:*

A new benchmark collection for sentence-based image description and search, consisting of 8,000 images that are each paired with five different captions which provide clear descriptions of the salient entities and events, etc... The images were chosen from six different Flickr groups, and tend not to contain any well-known people or locations, but were manually selected to depict a variety of scenes and situations.

**5.2 Code implementation**

**5.2.1 Model**

Encoder model: Resnet-50

Decoder model: LSTM

Decoder model: LSTM-Attention

**5.2.2 Configuration**

Word embedding dimensions: 300

Attention dimensions: 256

Decoder dimensions: 512

Dropout: 0.5

Encoder learning rate: 1e-4

Decoder learning rate:  3e-4

Numbers of Epoch: 40

Optimizer: Adam

Loss function: Cross Entropy Loss

**REFERENCES**

[1] Nguyen Thanh Huyen, A Guide to Image Captioning (2021), Viblo.

[2] Trung Đức, Image Captioning – Chú thích dữ liệu hình ảnh bằng công nghệ học sâu (2022), VinBigdata.

[3] G. Sairam, M. Mandha, P. Prashanth, P. Swetha et al, Image Captioning using CNN and LSTM (2023), ieeexplore.