TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÙI HẢI DƯƠNG – 521H0220**

**TRẦN QUANG LUÂN -**

**CHATBOT VỀ LUẬT GIAO THÔNG ĐƯỜNG BỘ**

**DỰ ÁN CÔNG NGHỆ THÔNG TIN**

**KHOA HỌC MÁY TÍNH**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2025**

TỔNG LIÊN ĐOÀN LAO ĐỘNG VIỆT NAM

**TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

**KHOA CÔNG NGHỆ THÔNG TIN**



**BÙI HẢI DƯƠNG – 521H0220**

**TRẦN QUANG LUÂN - MSSV**

**CHATBOT VỀ LUẬT GIAO THÔNG ĐƯỜNG BỘ**

**DỰ ÁN CÔNG NGHỆ THÔNG TIN**

**HOA HỌC MÁY TÍNH**

Người hướng dẫn

**PGS. TS. Lê Anh Cường**

**THÀNH PHỐ HỒ CHÍ MINH, NĂM 2025**

**LỜI CẢM ƠN**

Chúng em xin chân thành cảm ơn PGS. TS. Lê Anh Cường đã tận tình hướng dẫn chúng em để có thể hoàn thành dự án này, giúp chúng em có thêm kiến thức và kinh nghiệm để có thể

*TP. Hồ Chí Minh, ngày 19 tháng 01 năm 2025.*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

**CÔNG TRÌNH ĐƯỢC HOÀN THÀNH**

**TẠI TRƯỜNG ĐẠI HỌC TÔN ĐỨC THẮNG**

Tôi xin cam đoan đây là công trình nghiên cứu của riêng tôi và được sự hướng dẫn khoa học của PGS. TS. Lê Anh Cường. Các nội dung nghiên cứu, kết quả trong đề tài này là trung thực và chưa công bố dưới bất kỳ hình thức nào trước đây. Những số liệu trong các bảng biểu phục vụ cho việc phân tích, nhận xét, đánh giá được chính tác giả thu thập từ các nguồn khác nhau có ghi rõ trong phần tài liệu tham khảo.

Ngoài ra, trong Dự án còn sử dụng một số nhận xét, đánh giá cũng như số liệu của các tác giả khác, cơ quan tổ chức khác đều có trích dẫn và chú thích nguồn gốc.

**Nếu phát hiện có bất kỳ sự gian lận nào tôi xin hoàn toàn chịu trách nhiệm về nội dung Dự án của mình**. Trường Đại học Tôn Đức Thắng không liên quan đến những vi phạm tác quyền, bản quyền do tôi gây ra trong quá trình thực hiện (nếu có).

*TP. Hồ Chí Minh, ngày 19 tháng 01 năm 2025*

*Tác giả*

*(Ký tên và ghi rõ họ tên)*

**TÊN ĐỀ TÀI**

**TÓM TẮT**

(Time New Romans – 13)

**TITLE**

**ABSTRACT**

(Time New Romans – 13)

**MỤC LỤC**

[DANH MỤC HÌNH VẼ v](#_Toc143173056)

[DANH MỤC BẢNG BIỂU vii](#_Toc143173057)

[DANH MỤC CÁC CHỮ VIẾT TẮT viii](#_Toc143173058)

[CHƯƠNG 1. MỞ ĐẦU VÀ TỔNG QUAN ĐỀ TÀI 1](#_Toc143173059)

[1.1 Lý do chọn đề tài 1](#_Toc143173060)

[1.2 Mục tiêu thực hiện đề tài 1](#_Toc143173061)

[CHƯƠNG 2. CƠ SỞ LÝ THUYẾT 2](#_Toc143173062)

[2.1 Mạng neural hồi quy 2](#_Toc143173063)

[2.1.1 Recurrent Neural Network (RNN) 2](#_Toc143173064)

[2.1.2 Long Short-term Memory (LSTM) 2](#_Toc143173065)

[2.2 Mô hình Transformer 3](#_Toc143173066)

[2.2.1 Encoder và Decoder 3](#_Toc143173067)

[2.2.2 Attention 3](#_Toc143173068)

[CHƯƠNG 3. MÔ HÌNH ĐỀ XUẤT 4](#_Toc143173069)

[CHƯƠNG 4. THỰC NGHIỆM 4](#_Toc143173070)

[4.1 Dữ liệu thực nghiệm 4](#_Toc143173071)

[4.2 Cài đặt thực nghiệm 4](#_Toc143173072)

[CHƯƠNG 5. KẾT LUẬN 5](#_Toc143173073)

[5.1 Kết luận 5](#_Toc143173074)

[5.2 Hướng phát triển 5](#_Toc143173075)

[TÀI LIỆU THAM KHẢO 6](#_Toc143173076)

# DANH MỤC HÌNH VẼ

[Hình 2.1: Scaled Dot-Product Attention 3](#_Toc142677545)

# DANH MỤC BẢNG BIỂU

[Bảng 4.1: Thống kê kiểu thực thể trong tập VLSP 2016 4](#_Toc142677565)

# DANH MỤC CÁC CHỮ VIẾT TẮT

|  |  |
| --- | --- |
| BERT | Bidirectional Encoder Representations from Transformers |
| GEC | Grammatical Error Correction |
| MLM | Masked Language Model |
| NLP | Natural Language Processing |
| NSP | Next Sentence Prediction |

# MỞ ĐẦU VÀ TỔNG QUAN ĐỀ TÀI

## Lý do chọn đề tài

Trong năm 2025, luật giao thông đã có nhiều thay đổi lớn và đến đời sống của người dân rất nhiều đặc biệt là các hình phạt được tăng lên được quy định trong Nghị định 168 có hiệu lực vào 1/1/2025. Mặc dù giao thông đường bộ gắn liền với mọi người hàng ngày nhưng việc người dân nắm bắt được hết luật giao thông đường bộ là rất khó do có rất nhiều quy định được ban hành trong các Luât, thông tư và nghị định và không phải ai cũng có thể bỏ thời gian, công sức để học và tiếp thu được hết những tài liệu đó. Chính vì vậy việc xây dựng và áp dụng chatbot về luật giao thông đường bộ là một giải pháp tốt, mọi người dễ tiếp cận và tiết kiệm được thời gian, công sức, giúp nâng cao kiến thức và ý thức của mọi người khi tham gia giao thông.

## Mục tiêu thực hiện đề tài

Mục tiêu của đề tài này là nghiên cứu, thiết kế và xây dựng một chatbot thông minh, có khả năng tự động trả lời các câu hỏi liên quan đến luật giao thông, đồng thời hỗ trợ giải đáp những tình huống cụ thể mà người dùng có thể gặp phải trong quá trình tham gia giao thông. Chatbot này không chỉ giúp nâng cao nhận thức pháp luật giao thông mà còn cung cấp thông tin nhanh chóng, chính xác và dễ hiểu, góp phần hỗ trợ người dùng tuân thủ các quy định và giảm thiểu các vi phạm giao thông.

# PHƯƠNG PHÁP SỬ DỤNG

## Phương pháp tiếp cận

Thông thường sẽ có hai cách tiếp cận cho bài toán về chatbot này đó chính là Finetune mô hình có sẵn hoặc sử dụng Retrieval Augmented Generation. Cả hai cách tiếp cận này đều sử dụng dữ liệu có sẵn của chúng ta để trả lời những câu hỏi của người dùng khi tương tác với chatbot nhưng cách thực hiện của chúng lại khác nhau.

### Finetune mô hình

Finetune mô hình có sẵn là việc sử dụng các mô hình đã được huấn luyện (pre-trained model) dựa trên bộ dữ liệu rất lớn. Những mô hình đó sẽ được sử dụng và được huấn luyện tiếp trên dữ liệu của chúng ta và từ đó mô hình có thể được sử dụng trong task cụ thể của chúng ta.

A diagram of a function

Description automatically generated

Figure 1. Sơ lược về quá trình fine-tuning mô hình

Hình trên đây thể hiện cho quá trình Finetune mô hình. Từ những pre-train model, chúng ta sẽ huấn luyện tiếp trên dữ liệu của chúng ta. Thông qua quá trình backpropagation, trọng số và bias của mô hình trong các lớp sẽ được cập nhật, từ đó sẽ phù hợp trong tác vụ mà chúng ta thực hiện.

### Retrieval Augmented Generation (RAG)

Cách hoạt động của RAG thì khác so với việc Finetune mô hình. Về cách thức hoạt động, RAG sẽ đưa vào mô hình ngôn ngữ lớn (LLM) câu hỏi của người dùng và các tài liệu liên quan để LLM phản hồi lại kết quả cho người dùng.

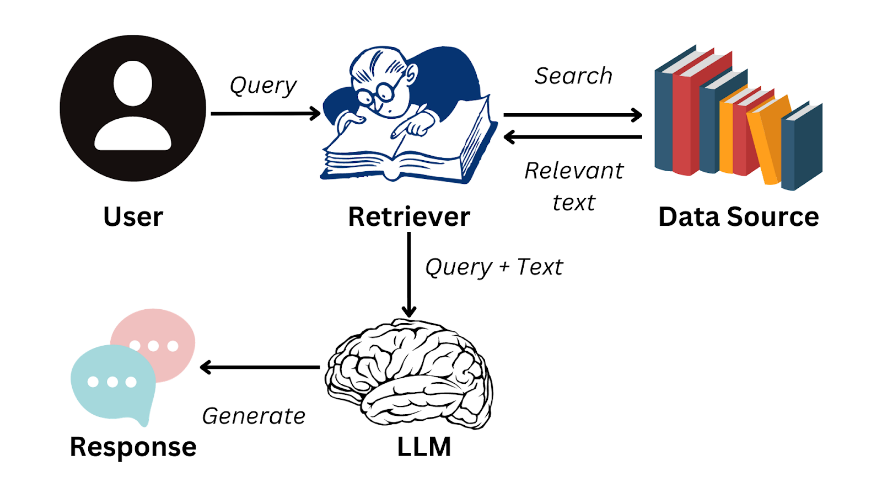


Figure 2. Sơ lược hệ thống RAG

Hình trên thể hiện quy trình hoạt động của RAG. Dữ liệu của chúng ta đầu tiên sẽ được lưu trữ trong một hoặc nhiều database. Khi người dùng thực hiện nhập vào một câu hỏi, chúng ta sẽ thực hiện tìm kiếm các tài liệu mà chúng ta đã lưu trước đó trong database có liên quan đến câu hỏi của người dùng. Câu hỏi của người dùng và các tài liệu đã được truy vấn sẽ được truyền trực tiếp vào trong mô hình để mô hình có thể tự trả lời dựa trên những thông tin đã tìm được.

### Điểm mạnh và điểm yếu của hai phương pháp

Điểm mạnh:

* Finetune pre-trained model:
  + Mô hình được finetune sẽ phù hợp với tác vụ chuyên biệt.
  + Tận dụng được những dữ liệu đã được huấn luyện trước đó.
  + Vì không cần lấy từ các dữ liệu bên ngoài và chỉ dựa và những gì đã được huấn luyện để thực hiện tác vụ nên tốc độ phản hồi rất nhanh.
* RAG:
  + Dữ liệu mới sẽ được cập nhật một cách dễ dàng bằng cách đưa vào trong database mà không cần cập nhật lại mô hình.
  + Không phải mất thời gian huấn luyện mô hình nên sẽ tiết kiệm được thời gian và chi phí.
  + Kết quả được dựa vào trực tiếp từ nguồn dữ liệu nên sẽ đưa ra kết quả chính xác và không gây ra hiện tượng ảo giác.

Điểm yếu:

* Finetune pre-trained model:
  + Việc huấn luyện lại mô hình tốn rất nhiều thời gian và chi phí.
  + Muốn cập nhật dữ liệu mới cho mô hình thì phải huấn luyện mô hình trên dữ liệu mới đó.
  + Khi thực hiện tác vụ thì mô hình có thể sẽ bị ảo tưởng từ những kiến thức mình đã học và đưa ra kết quả thiếu chính xác.
* RAG:
  + Việc đưa ra kết quả sẽ chậm, phụ thuộc vào quá trình truy xuất thông tin từ câu hỏi.
  + Nếu thông tin truy xuất kém thì có thể dẫn tới kết quả trả về thiếu xót hoặc không chính xác.
  + Vì quy trình của RAG phức tạp do có nhiều thành phần kết hợp lại với nhau.

### Phương pháp tiếp cận phù hợp

Vì tác vụ cần làm là về luật giao thông đường bộ nên các sẽ cần đáp ứng các tiêu chí sau:

* Dữ liệu pháp luật phải là dữ liệu mới nhất và dễ dàng cập nhật.
* Kết quả trả về phải chính xác dựa theo dữ liệu pháp luật.

Việc finetune mô hình cũng có thể cập nhật được dữ liệu mới những sẽ tốn rất nhiều thời gian và chi phí. Bên cạnh đó mô hình được finetune có khả năng sẽ trả lời không chính xác do mô hình có thể bị hiện tượng ảo tưởng.

Với RAG thì việc cập nhật dữ liệu này sẽ rất dễ dàng và cũng rất nhanh chóng và việc đưa trực tiếp tài liệu lưu trữ vào cho mô hình trả lời sẽ chính xác hơn và không gây ra hiện tượng ảo giác. Chính vì vậy RAG sẽ là giải pháp tiếp cận phù hợp.

## Mô hình sử dụng

### Mô hình embedding

Chức năng chính của mô hình embedding là chuyển đổi các token như ảnh, video, chữ cái hoặc âm thành sang biểu diễn vector, từ đó sẽ giúp việc xử lí dữ liệu trong học máy một cách dễ dàng hơn.

Trong RAG thì việc xử lý embedding này sẽ được thực hiện trong khâu truy xuất thông tin, bằng cách đưa các chunk được chia nhỏ từ các văn bản chính, từ đó embedding chúng thành dạng vector và lưu trữ trong database. Khi các câu hỏi của người dùng được đưa vào thì chúng sẽ được chuyển qua dạng vector và thực hiện so sánh với các vector đã được lưu trong database và truy xuất ra những dữ liệu liên quan.

Dưới đây là thống kê về kết quả của các mô hình embedding dựa trên 20% của tập huấn luyện luật Zalo, chưa được huấn luyện trên các mô hình dưới đây. Kết quả thể hiện đánh giá về Acurracy, Precision và 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 |

Dựa vào thống kê ở trên, ta có thể thấy được mô hình embedding cho ra kết quả cao nhất đó là halong\_embedding (768). Vậy nên halong\_embedding (768) sẽ là lựa chọn để sử dụng làm mô hình embedding.

### Mô hình tạo sinh

Mô Mô hình tạo sinh là các mô hình ngôn ngữ lớn (LLM) có khả năng tạo ra nội dung mới dựa trên dữ liệu đã được huấn luyện và đầu vào từ người dùng. Những mô hình này được đào tạo trên các tập dữ liệu khổng lồ, bao gồm nhiều chủ đề khác nhau, nhằm hiểu và tạo ra văn bản tự nhiên.

Trong bối cảnh hệ thống RAG các mô hình tạo sinh phổ biến đến từ các tổ chức như OpenAI, Google, Meta, cũng như các mô hình mã nguồn mở trên GitHub hoặc HuggingFace.

Đặc biệt, mô hình Gemini của Google cung cấp các phiên bản như Gemini 1.5 Flash và Gemini 1.5 Pro với khả năng xử lý ngữ cảnh dài, lần lượt hỗ trợ cửa sổ ngữ cảnh lên đến 1 triệu và 2 triệu token. Điều này cho phép tạo ra các prompt chi tiết và phức tạp.

Để sử dụng các mô hình này, người dùng cần có khóa API được cung cấp thông qua tài khoản Google. Việc lựa chọn mô hình phù hợp sẽ dựa trên yêu cầu cụ thể của dự án, bao gồm độ phức tạp của nhiệm vụ, giới hạn ngữ cảnh và ngân sách.

## Hệ thống và các thành phần

Thông thường trong hệ thống RAG sẽ có hai phần chính đó chính là Module truy xuất và Module để tạo sinh. Trong đó Module truy xuất sẽ thực hiện công việc truy xuất ra các văn bản có độ liên quan nhất định dựa trên câu hỏi của người dùng và Module tạo sinh sẽ dựa vào những văn bản đã được truy xuất ra trả ra kết quả phản hồi cho người dùng dựa trên câu hỏi của người dùng và các văn bản đã được truy xuất ra trước đó ở Module truy xuất. Hệ thống RAG này sẽ được áp dụng một số kỹ thuật nâng cao để có thể tối ưu việc trả ra kết quả cho người dùng.

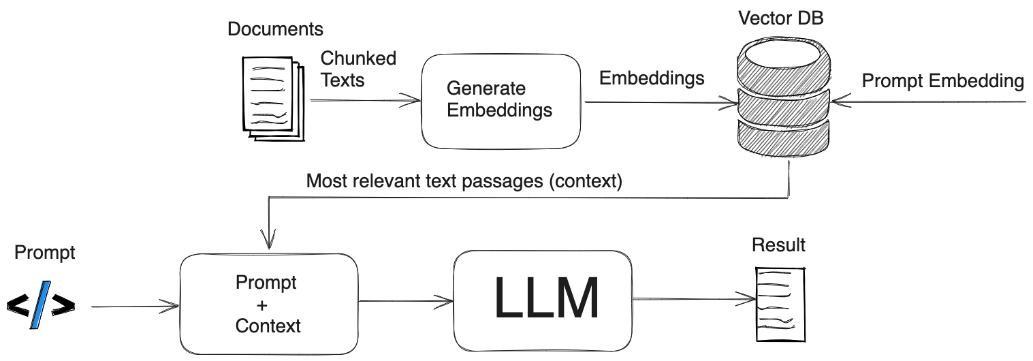


Figure 3. Quy trình hoạt động cơ bản của hai module trong RAG

## Chi tiết các thành phần

### Module truy xuất

#### Phân đoạn (Chunking)

Đầu vào của quá trình chunking là những văn bản thu thập được thông qua việc tìm kiếm trên các website, sách báo và các nguồn thông tin đáng tin cậy, dữ liệu được lưu trữ trong các dạng file như pdf, docx, markdown sẽ được chunking để có thể dễ dàng lưu trữ và truy xuất. Đầu ra của quá trình này là những chunk được chunking từ các văn bản ban đầu.

Có nhiều phương pháp chunking khác nhau tuỳ thuộc vào bài toán và loại văn bản cần xử lý. Có rất nhiều phương pháp chunking được sử dụng trong văn bản với các mục đích khác nhau. Các phương pháp chunking thông dụng thường được sử dụng một cách phổ biến:

* Fixed Fixed-size chunking: Chia văn bản thành các đoạn có kích thước cố định.
* Recursive chunking: Tách văn bản theo các dấu hiệu phân đoạn (đoạn văn, tiêu đề...) một cách đệ quy đến khi đạt kích thước mong muốn.
* Document-based chunking: Áp dụng cho các văn bản có cấu trúc đặc biệt (Markdown, mã nguồn, bảng biểu...).
* Semantic chunking: Dựa vào mức độ tương đồng ngữ nghĩa giữa các câu để gom nhóm các câu liên quan.
* Agentic chunking: Sử dụng mô hình ngôn ngữ lớn (LLM) để quyết định cách nhóm các đoạn văn bản theo ngữ cảnh.

#### Lưu trữ (Storing)

Storing là quá trình lưu trữ các chunk đã được phân đoạn từ quá trình chunking. Tuỳ thuộc vào bài toán mà chúng ta sẽ chia thành các không gian lưu trữ khác nhau. Để tăng tốc quá trình truy xuất văn bản thì các chunk sẽ được lưu trữ trong không gian vector (Vectorstore). Quá trình nãy sẽ bao gồm những phần:

* Embedding: Sử dụng mô hình embedding để chuyển đổi các chunk của văn bản thành dạng vector để lưu trữ trong Vectorstore
* Vectorstore: Có rất nhiều các Vectorstore hiện nay có hỗ trợ lưu trữ các vector trên cloud như FAISS, ChromaDB, Pinecone, … Bên cạnh đó cũng có thể sử dụng các Vectorstore cục bộ để lưu trữ các vector ví dụ như PostgresSQL, …

#### Chuyển đổi câu hỏi (Query transformation)

Đây là kỹ thuật dùng để biến đổi các câu hỏi của người dùng sao cho các câu hỏi đó được biến đổi thành các câu hỏi ngắn gọn hơn, súc tích hơn, giúp cho quá trình truy xuất các chunk một cách tối ưu hơn.

Kỹ thuật này được áp dụng bằng cách sử dụng các mô hình ngôn ngữ lớn, thực hiện các kỹ thuật prompting để biến các câu hỏi của người dùng thành các câu hỏi phù hợp hơn. Một số kỹ thuật để chuyển đồi câu query:

* Step back prompting là phương pháp biến đổi những câu hỏi cụ thể thành những câu hỏi chung hơn và bao hàm ý nghĩa rộng hơn, cải thiện khả năng truy vấn văn bản.
* Query rewriting là phương pháp biến những query ban đầu trở nên cụ thể và chi tiết hơn. Phương pháp này sẽ giúp tăng độ chính xác trong quá trình truy xuất văn bản.
* Sub-query decomposition là phương pháp chia nhỏ nhưng câu query phức tạp thành những câu query thành phần, đơn giản hơn. Quá trình này giúp quá trình truy xuất trở nên đơn giản và hiệu quả hơn.

#### Các kỹ thuật truy xuất

Thông thường sẽ có nhiều loại văn bản khác nhau nên cũng sẽ tuỳ vào đó mà lựa cọn các kỹ thuật truy xuất sao cho phù hợp. Các kỹ thuật truy vấn hiệu quả:

* Parent document retrieval: Phân tách các văn bản gốc thành các chunk nhỏ. Mỗi văn bản gốc sẽ có một id riêng và các chunk của chúng sẽ trỏ tới văn bản gốc khi các chunk đó được truy xuất thông qua câu query của người dùng. Kỹ thuật này sẽ tối ưu trong việc truy xuất toàn bộ ngữ cảnh của văn bản như các văn bản luật, văn bản y khoa, …
* Hybrid retrieval: Truy xuất dự trên từ khoá (sparse retrieval) và dựa trên ngữ nghĩa (dense retrieval). Phương pháp này tận dụng hai phương pháp truy xuất trên để tối ưu khả năng truy xuất chính xác tới các chunk cụ thể dựa và query của người dùng. Sparse retrieval là phương pháp truyền thống dùng để đo sự tương đồng từ query của người dùng dựa vào độ khớp của các từ khoá. Dense retrieval là sử dụng mô hình embedding để truy vấn văn bản trong không gian vector dựa trên độ đo về sự tương đồng với câu query của người dùng. Việc kết hợp hai phương pháp này, hay còn gọi là Hybrid retrieval sẽ làm tăng khả năng truy xuất chính xác một cách đáng kể, cải thiện đầu ra của kết quả.
* Multi-query retrieval: Biến đổi câu query ban đầu của người dùng thành nhiều biến thể mang ý nghĩa tương đồng, từ đó đưa ra nhiều kết quả truy xuất hơn.

### Module tạo sinh (Generative module)

Sau khi truy xuất được các chunk liên quan, câu hỏi của người dùng và các chunk liên quan này sẽ được đưa vào trong mô hình LLM để sinh ra câu trả lời và phản hồi lại cho người dùng. Các bước thực hiện chính trong module tạo sinh:

* Tạo prompt đầu vào: Kết hợp giữa câu hỏi và các chunk đã truy xuất được.
* Sử dụng LLM để trả lời câu hỏi: Sử dụng các mô hình như T5, ViT5, GPT, Gemini, …
* Các kỹ thuật được sử dụng để tối ưu đầu ra:
  + Fewshot Chain of Thought (Fewshot CoT): Hướng dẫn mô hình suy luật từng bước để đưa ra câu trả lời, cung cấp thêm các ví dụ cụ thể để mô hình có thể thực hiện theo.
  + Memory-Augmented RAG: Lưu trữ lại lịch sử cuộc hội thoại để tăng tính tương tác và chính xác cho các câu trả lời.

# THỰC NGHIỆM

## Dữ liệu sử dụng

Dữ liệu được sử dụng trong nghiên cứu này được thu thập từ Thư viện Pháp luật, bao gồm 115 tệp tài liệu định dạng DOCX. Trong số này, có 61 tài liệu luật ban hành trước năm 2025 và 48 luật mới có hiệu lực từ ngày 1/1/2025. Các tài liệu này bao gồm Luật, Thông tư và Nghị định, cung cấp một nguồn thông tin phong phú và đa dạng cho nghiên cứu.

Trong quá trình tiền xử lý thì toàn bộ văn bản sẽ được chuyển thành chữ viết thường và loại bỏ toàn bộ khoảng trắng thừa.

## Xây dựng Vectorstore và hệ thống truy xuất

Từ Các văn bản luật giao thông thường được cấu trúc với các phần như Tiêu đề, Tóm tắt, Các điều mục, Tên điều mục và Nội dung điều mục. Để tối ưu hóa quá trình truy xuất thông tin, chúng tôi áp dụng phương pháp Parent Document Retrieval. Cụ thể, từ các văn bản luật gốc, nội dung được chia nhỏ theo từng điều mục. Mỗi điều mục đóng vai trò là "Parent Document" và chứa các đoạn văn bản ("chunk") liên quan.

Quá trình chunking được thực hiện bằng phương pháp Recursive Chunking, tức là tách các văn bản chính thành các đoạn nhỏ hơn dựa trên cấu trúc phân cấp của tài liệu. Điều này giúp tạo ra các chunk có kích thước phù hợp cho việc lưu trữ và truy xuất hiệu quả.

Với việc sử dụng thư viện Langchain để hỗ trợ xây dựng hệ thống Parent Document Retrieval và lưu trữ dữ liệu. Cụ thể, Langchain cung cấp các công cụ để triển khai kỹ thuật này một cách hiệu quả.

Để lưu trữ cả các vector đã được embedding và văn bản gốc, chúng ta sẽ sử dụng PostgreSQL kết hợp với tiện ích mở rộng pgvector. Sự kết hợp này cho phép lưu trữ và truy xuất vector hiệu quả, đồng thời tận dụng các tính năng mạnh mẽ của PostgreSQL như ACID compliance và khả năng phục hồi theo thời gian.

Việc sử dụng Langchain cùng với PostgreSQL và pgvector giúp hệ thống trở nên linh hoạt, hiệu quả và dễ dàng mở rộng trong quá trình triển khai và vận hành.

## Mô hình ngôn ngữ lớn

LLM sẽ được sử dụng trong ba thành phần chính đó là Query transformation, Routing và Module tạo sinh. Temparature = 0 được sử dụng trong các mô hình này để trả về kết quả chính xác nhất.

Phương pháp prompting sử dụng trong cả ba thành phần đều là Fewshot CoT, vừa hướng dẫn mô hình từng bước và vừa đưa ra các ví dụ cụ thể để mô hình làm theo. Phương pháp này cải thiện đầu ra của mô hình giúp cho kết quả trở nên chính xác hơn.

Query transformation sẽ biến đổi các câu hỏi của người dùng như sau:

* Loại bỏ những từ ngữ chỉ mới hoặc cũ như “luật mới” hay “luật cũ” để tăng hiệu quả truy xuất.
* Với những câu hỏi về luật, nếu những câu hỏi đó quá chi tiết thì sẽ thực hiện hiện Step back prompting để biến câu hỏi đó trở nên bao quát, tăng khả năng truy xuất và cung cấp thêm những thông tin phụ.
* Với những câu hỏi không phải về luật thì giữ nguyên câu hỏi gốc.

Phần Routing sẽ hướng tới ba hướng khác nhau. Các hướng Routing sẽ bao gồm:

* Database chứa các văn bản luật giao thông cũ trước năm 2025 nếu câu hỏi của người dùng hỏi về luật hiện tại hoặc đề cập tới luật mới.
* Database chứa các văn bản luật giao thông mới có hiệu lực từ 1/1/2025 nếu câu hỏi của người dùng đề cập tới luật cũ.
* Để mô hình LLM trả lời trực tiếp khi câu hỏi của người dùng là normal chatting.

Về mô hình được sử dụng trong Module tạo sinh, đầu ra sẽ được quy định thông qua prompt như sau:

* Prompt sẽ được cung cấp câu query của người dùng và các văn bản Parent được trỏ tới từ các chunk liên quan tới câu hỏi của người dùng. Các văn bản này sẽ được đính với metadata và metadata sẽ là thành phần quan trọng trong việc format đầu ra của phản hồi cho người dùng.
* Prompt sẽ được cung cấp lịch sử đoạn chat để có thể tương tác với người dùng và trả lời câu hỏi một cách chính xác hơn. Thông tin trong lịch sử đoạn chat cũng có thể được sử dụng để thực hiện so sánh.
* Format chính của đầu ra sẽ là:
  + Nguồn văn bản: Tên tài liệu
  + Tên văn bản: Tiêu đề của văn bản
  + Điều: Tiều đề điều mục
  + Nội dung:
    - Nội dung của điều mục
* Bên cạnh những quy ước chính trên thì sẽ có cả các tiêu chí phụ làm tăng tính thẩm mỹ của phản hồi người dùng:
  + Viết hoa chữ cái đầu dòng
  + Ngắt dòng giữa các tiểu mục: a), b), c), …

# KẾT QUẢ THỰC NGHIỆM

# TÀI LIỆU THAM KHẢO

Tiếng Việt

…

Tiếng Anh

Hochreiter, S., & Schmidhuber, J. (1997). Long Short-term Memory. *Neural Computation*, *9*, 1735–1780. https://doi.org/10.1162/neco.1997.9.8.1735

Vaswani, A., Shazeer, N., Parmar, N., Uszkoreit, J., Jones, L., Gomez, A. N., Kaiser, L., & Polosukhin, I. (2023). *Attention Is All You Need* (arXiv:1706.03762). arXiv. https://doi.org/10.48550/arXiv.1706.03762