Legal Document Retrieval for BKAi SoICT 2024

Đinh Nhật Trường
22521575

Phạm Huỳnh Nhật Tân
22521309

Phạm Nguyễn Anh Tuấn
22521610

Abstract

Chúng tôi xây dựng hệ thống truy vấn văn bản luật pháp tiếng Việt nhằm hỗ trợ việc tìm kiếm và trích xuất thông tin từ các văn bản pháp luật tiếng Việt một cách hiệu quả. Hệ thống tập trung vào việc tối ưu hóa các phương pháp truy vấn, sử dụng các kỹ thuật xử lý ngôn ngữ tự nhiên (NLP) và mô hình học sâu (deep learning) để cải thiện khả năng tìm kiếm và trả về kết quả chính xác.

1 Giới thiệu

Việc truy vấn và tìm kiếm thông tin từ các tài liệu pháp lý luôn là một nhiệm vụ quan trọng trong lĩnh vực pháp lý. Đặc biệt, với sự phát triển của các hệ thống thông tin và dữ liệu pháp luật tiếng Việt, việc xây dựng một hệ thống *truy vấn văn bản luật pháp tiếng Việt* ngày càng trở nên cần thiết và cấp bách.

Trong bối cảnh đó, hệ thống truy vấn cần phải đảm bảo khả năng xử lý hiệu quả, chính xác và linh hoạt trước khối lượng dữ liệu lớn của các văn bản pháp lý. Các thách thức chính bao gồm sự phong phú và đa dạng của ngôn ngữ pháp lý, sự không nhất quán về cấu trúc, cũng như sự xuất hiện của các thuật ngữ đặc thù.

Chúng tôi tập trung vào việc phát triển một hệ thống retrieval nhằm nâng cao hiệu quả tìm kiếm thông tin trong kho dữ liệu luật pháp tiếng Việt. Hệ thống được xây dựng dựa trên việc kết hợp các kỹ thuật học máy, xử lý ngôn ngữ tự nhiên, và các mô hình deep learning nhằm cải thiện độ chính xác và khả năng trả về các văn bản liên quan.

Mục tiêu của đề tài là không chỉ tối ưu hóa khả năng truy vấn mà còn mang lại giải pháp khả thi và ứng dụng thực tiễn cho các hệ thống thông tin pháp lý, từ đó hỗ trợ các cơ quan, tổ chức trong việc tra cứu và áp dụng luật pháp một cách nhanh chóng và chính xác.

2 Cơ sở lý thuyết

Để xây dựng hệ thống *truy vấn văn bản luật pháp tiếng Việt*, chúng tôi sẽ trình bày các khái niệm cơ bản và lý thuyết liên quan, bao gồm xử lý ngôn ngữ tự nhiên (NLP), các mô hình học máy, và các kỹ thuật retrieval phổ biến.

2.1 Xử lý Ngôn ngữ Tự nhiên (NLP)

Xử lý ngôn ngữ tự nhiên (NLP) là một lĩnh vực quan trọng trong hệ thống tìm kiếm thông tin. Các phương pháp NLP giúp chuyển đổi dữ liệu ngôn ngữ tự nhiên thành dạng mà máy móc có thể xử lý. Trong nghiên cứu này, các kỹ thuật như tokenization, stemming, và stop-word removal sẽ được sử dụng để làm sạch dữ liệu và chuẩn hóa văn bản.

2.2 Mô hình học sâu và Retrieval

Các mô hình học sâu, đặc biệt là các mô hình transformer, đã chứng minh hiệu quả trong việc xử lý ngữ nghĩa và tìm kiếm thông tin. Chúng tôi sử dụng mô hình học sâu giúp trích xuất đặc trưng ngữ nghĩa từ văn bản và hỗ trợ việc tìm kiếm chính xác trong kho dữ liệu lớn. Mô hình này cung cấp khả năng xử lý ngôn ngữ đa ngữ, hỗ trợ tìm kiếm thông tin trong môi trường đa ngữ như văn bản pháp lý tiếng Việt.

2.3 Kỹ thuật Retrieval

Các kỹ thuật retrieval được sử dụng trong đề tài này bao gồm:

- **Semantic Search**: Tìm kiếm dựa trên ngữ nghĩa, giúp hệ thống hiểu và so sánh giữa các văn bản dựa trên ý nghĩa.
- **Matching Models**: Các mô hình học sâu sẽ được huấn luyện để tính toán mức độ tương đồng giữa câu hỏi và văn bản nhằm xác định các tài liệu liên quan.
- **Re-Rank Models**: Mô hình xếp hạng lại nhằm nâng cao thứ tự và độ chính xác của các tài liệu liên quan.

3 Giới thiệu Dữ liệu

Dữ liệu phục vụ cho hệ thống truy vấn văn bản luật pháp tiếng Việt được cung cấp từ các nguồn chính nhằm đảm bảo độ phong phú và chính xác cho việc tìm kiếm thông tin pháp lý.

3.1 Training Data (Dữ liệu Huấn Luyện)

- **Mô tả:** Đây là tập dữ liệu chính chứa cặp truy vấn và văn bản liên quan được gán nhãn, dùng để huấn luyện các mô hình học máy.
- **Số lượng:** 119,456 cặp truy vấn-văn bản.
- **Nội dung:** Tập dữ liệu này giúp cung cấp các mẫu mẫu truy vấn và văn bản liên quan, từ đó huấn luyện mô hình nhận diện và hiểu ngữ nghĩa của truy vấn, tăng khả năng tìm kiếm chính xác.

3.2 Public Test Data (Dữ liệu Kiểm Thử Công Khai)

- **Mô tả:** Dùng để kiểm thử và đánh giá hiệu quả của mô hình. Kết quả từ dữ liệu này sẽ được công khai để so sánh hiệu quả.
- **Số lượng:** 10,000 truy vấn.
- **Nội dung:** Cũng bao gồm cặp truy vấn và văn bản liên quan, giúp đo lường khả năng tổng quát và độ chính xác của mô hình đối với dữ liệu chưa thấy trước.

3.3 Kho Dữ liệu Văn bản Pháp luật

- **Mô tả:** Là kho dữ liệu chính, chứa các văn bản pháp luật từ nhiều nguồn khác nhau như luật, nghị định, thông tư, giúp cung cấp nền tảng cho việc tìm kiếm thông tin pháp lý.
- **Số lượng:** Tổng cộng 261,597 văn bản pháp luật.
- **Sử dụng:** Các tập dữ liệu như training data và public test data đều sử dụng chung kho dữ liệu này, đảm bảo sự nhất quán và độ tin cậy của thông tin.

4 Phương pháp luận

4.1 Tiền xử lý dữ liệu

4.1.1 Token hóa và làm sạch dữ liệu

- **Token hóa:** Văn bản sẽ được chia nhỏ thành các token, bao gồm từ ngữ hoặc cụm từ, giúp phân tách dữ liệu thành các đơn vị nhỏ hơn để dễ dàng xử lý.
- **Làm sạch:** Loại bỏ các ký tự đặc biệt, dấu câu, và từ dừng không quan trọng như "là", "các", "những", "và", v.v. để giảm nhiễu và tập trung vào thông tin quan trọng.
- **Chuẩn hóa văn bản:** Các từ dạng khác nhau (chẳng hạn như số ít, số nhiều) sẽ được chuẩn hóa thành một dạng duy nhất để đảm bảo tính nhất quán.

4.1.2 Chunking ngữ nghĩa

- **Embedding văn bản:** Mỗi câu hoặc đoạn nhỏ trong văn bản sẽ được chuyển đổi thành các vector embedding sử dụng mô hình embeddings_model:

$$E_i = embeddings\_model(s_i) \quad (1)$$

- **Tính tương tự ngữ nghĩa:** Đo độ tương tự giữa các vector liên tiếp bằng cách sử dụng cosine similarity:

$$sim(E_i, E_{i+1}) = \frac{E_i \cdot E_{i+1}}{\|E_i\|\|E_{i+1}\|} \quad (2)$$

- **Xác định ranh giới chunk:** Dựa trên danh sách các giá trị cosine similarity, các chunk ngữ nghĩa sẽ được xác định, giúp phân đoạn văn bản thành các đoạn có ý nghĩa rõ ràng.
- **Lợi ích:** Tạo ra các đoạn chunk có ngữ nghĩa cụ thể, giúp mô hình dễ dàng nắm bắt các mối quan hệ ngữ nghĩa, nâng cao hiệu quả tìm kiếm thông tin.

4.2 Lựa chọn đặc trưng

4.2.1 Embeddings Ngữ Nghĩa với Jin-AI

Sử dụng các mô hình học sâu như Jin-AI đã được fine-tuned trên dữ liệu tiếng Việt pháp lý. Những mô hình này giúp trích xuất các đặc trưng ngữ nghĩa từ văn bản, tăng cường khả năng tìm kiếm dựa trên ý nghĩa.

- **Đầu vào văn bản:** Văn bản đầu vào được token hóa thành các token:

  Encoded_Input $=$ Tokenizer$(x,$ padding=True, truncation=Tr

  Trong đó, x là văn bản đầu vào.

- **Lấy embeddings từ Jin-AI:** Sau khi token hóa, Jin-AI tính toán embeddings cho từng token:

$$E_i = f_{\text{Jin-AI}}(t_i) \quad \text{for each token } t_i$$

  - E_i là embeddings của token t_i. - $f_{\text{Jin-AI}}$ là mô hình Jin-AI.

- **Mean Pooling (Tính trung bình embeddings):** Để lấy embeddings đại diện cho toàn bộ văn bản, tính trung bình embeddings của các token:

$$E_{\text{mean}} = \frac{1}{N} \sum_{i=1}^{N} E_i$$

- E_{mean} là embeddings ngữ nghĩa của toàn bộ văn bản. - N là số token trong văn bản.

- **Normalization (Chuẩn hóa):** Chuẩn hóa embeddings bằng L2 norm:

$$E_{\text{normalized}} = \frac{E_{\text{mean}}}{\|E_{\text{mean}}\|_2}$$

- Đảm bảo embeddings đều được chuẩn hóa để mô hình Jin-AI hoạt động hiệu quả.

4.3 Mô hình và kỹ thuật Retrieval

4.3.1 Semantic Retrieval

Trong **Semantic Retrieval**, **Cosine Similarity** được sử dụng để đo lường độ tương đồng giữa các vector nhúng của truy vấn và tài liệu. Các vector này phản ánh ngữ nghĩa của văn bản, giúp tìm kiếm tài liệu phù hợp không chỉ dựa trên từ khóa mà còn trên ý nghĩa.

Công thức Cosine Similarity Công thức cosine similarity giữa hai vector A và B được tính như sau:

$$\text{cosine\_similarity}(A, B) = \frac{A \cdot B}{||A|| \times ||B||}$$

Trong đó:

- A và B là các vector nhúng đại diện cho truy vấn và tài liệu.
- $A \cdot B$ là tích vô hướng giữa hai vector.
- $||A||$ và $||B||$ là chuẩn L2 (độ dài) của các vector.

Ứng dụng Trong **Semantic Retrieval**, cosine similarity giúp so sánh độ tương đồng giữa vector truy vấn và các tài liệu. Tài liệu có cosine similarity cao sẽ được xếp hạng cao hơn trong kết quả tìm kiếm, cải thiện độ chính xác của việc truy xuất thông tin.

4.3.2 Matching Models sử dụng Jin-AI trong Semantic Retrieval

Embedding với Jin-AI Jin-AI có thể được sử dụng để chuyển các đoạn văn bản (truy vấn và tài liệu) thành các vector nhúng (embeddings) ngữ nghĩa. Mỗi văn bản sẽ được đưa qua mô hình Jin-AI để tạo ra embeddings đại diện cho ngữ nghĩa của nó. Công thức tính embeddings có thể như sau:

$$E_{\text{query}} = \text{Jin-AI}(x_{\text{query}}), \quad E_{\text{doc}} = \text{Jin-AI}(x_{\text{doc}})$$

Trong đó:

- x_{query} và x_{doc} là văn bản truy vấn và tài liệu đầu vào.
- E_{query} và E_{doc} là các vector nhúng ngữ nghĩa tương ứng của truy vấn và tài liệu.

Tính Tương Đồng Ngữ Nghĩa Sau khi có được các embeddings từ Jin-AI, chúng sẽ được so sánh với nhau để đo lường mức độ tương đồng ngữ nghĩa giữa truy vấn và các tài liệu. **Cosine Similarity** là một phương pháp phổ biến được sử dụng để đo lường sự tương đồng này:

$$\text{cosine\_similarity}(E_{\text{query}}, E_{\text{doc}}) = \frac{E_{\text{query}} \cdot E_{\text{doc}}}{||E_{\text{query}}|| \times ||E_{\text{doc}}||}$$

Tính toán cosine similarity giúp xác định mức độ tương đồng giữa truy vấn và tài liệu, với các tài liệu có giá trị similarity cao sẽ được ưu tiên.

4.3.3 Re-Rank Models

1. Mục tiêu Ứng dụng mô hình rerank BERT (BAAI/bge-reranker-v2-m3) nhằm cải thiện thứ hạng tài liệu từ kho dữ liệu văn bản pháp lý. Tối ưu hóa kết quả tìm kiếm thông tin pháp lý bằng cách xếp hạng lại tài liệu dựa trên điểm số rerank từ mô hình.

2. Phương pháp thực hiện

- **a. Truy xuất tài liệu** Dữ liệu văn bản từ kho pháp lý được truy xuất thông qua các hệ thống tìm kiếm như Elasticsearch hoặc các nguồn dữ liệu khác. Tài liệu được lưu trữ dưới dạng văn bản, đảm bảo sẵn sàng cho quá trình xử lý và đánh giá.
- **b. Tạo cặp truy vấn-tài liệu** Mỗi tài liệu được kết hợp với truy vấn liên quan thành cặp $[\text{question}, \text{text}]$ để làm đầu vào cho mô hình rerank. Việc tạo cặp này giúp mô hình hiểu và đánh giá mức độ liên quan của tài liệu với từng truy vấn cụ thể.

- **c. Token hóa và tính điểm rerank** Mô hình BAAI/bge-reranker-v2-m3 được sử dụng để tính điểm rerank cho mỗi cặp truy vấn-tài liệu. Dữ liệu được token hóa và đưa vào mô hình để tính toán điểm số rerank, đánh giá mức độ liên quan của mỗi tài liệu so với truy vấn.

  Công thức điểm rerank:

  $$\text{Rerank Score} = f(\text{question, text})$$

  Trong đó:

  - f là hàm đầu ra từ mô hình reranker.
  - question là truy vấn đầu vào.
  - text là tài liệu đầu vào.

- **d. Sắp xếp lại tài liệu** Các tài liệu được sắp xếp theo điểm số rerank từ cao đến thấp. Tài liệu có điểm số rerank cao nhất sẽ được trả về trước, giúp lọc và xếp hạng tài liệu quan trọng và có liên quan nhất cho việc tìm kiếm thông tin pháp lý.

4.4 Đánh giá mô hình(mRR@10)

mRR@10 là chỉ số đánh giá hiệu quả của các hệ thống tìm kiếm thông tin, đặc biệt trong các hệ thống *Semantic Retrieval*, nơi mức độ chính xác của kết quả tìm kiếm được quan tâm. Chỉ số này tập trung vào thứ tự xuất hiện của tài liệu phù hợp trong danh sách kết quả trả về, đặc biệt là trong 10 kết quả đầu tiên.

4.5 Định nghĩa:

- **Reciprocal Rank (RR):** Được tính bằng nghịch đảo của vị trí của tài liệu phù hợp đầu tiên trong danh sách kết quả. Nếu tài liệu phù hợp đầu tiên xuất hiện ở vị trí k, thì:

  $$\text{RR} = \frac{1}{k}$$

  Nếu không có tài liệu phù hợp trong 10 kết quả trả về, thì $\text{RR} = 0$.

- **Mean Reciprocal Rank (mRR):** Được tính bằng trung bình của các *Reciprocal Rank (RR)* từ các truy vấn. Công thức tổng quát là:

  $$\text{mRR} = \frac{1}{N} \sum_{i=1}^{N} \text{RR}_i$$

  Trong đó:

  - N là tổng số truy vấn.
  - RR_i là Reciprocal Rank của truy vấn i.

- **mRR@10:** Chỉ số này đặc biệt chỉ tính đến các kết quả từ 1 đến 10. Nếu tài liệu phù hợp không nằm trong 10 kết quả đầu tiên, **mRR@10** sẽ không tính đến truy vấn đó.

Ý nghĩa của mRR@10:

- **mRR@10** giúp đánh giá hiệu quả của mô hình tìm kiếm trong việc trả về các tài liệu phù hợp sớm trong danh sách kết quả, đặc biệt là trong 10 kết quả đầu tiên. Hệ thống tìm kiếm tốt là hệ thống có khả năng đưa tài liệu phù hợp lên đầu danh sách.

- Chỉ số này đặc biệt hữu ích trong việc đánh giá các hệ thống *Semantic Retrieval*, nơi mô hình không chỉ tìm kiếm theo từ khóa mà còn dựa trên sự hiểu biết ngữ nghĩa để trả về các kết quả phù hợp với truy vấn người dùng.

article [utf8]inputenc graphicx booktabs xcolor caption array

5 Kết quả thực nghiệm

Trong quá trình đánh giá, chúng tôi đã thực hiện thực nghiệm trên tập dữ liệu gồm 119,456 mẫu, trong đó 9,456 mẫu cuối được trích ra làm tập test. Kết quả thu được được thể hiện trong Bảng 1.

	non chunk	**chunk**	**chunk+finetune +rerank BGE**
Jinaai_8194	44%	47%	57%
Jinaaig_1024	46%	48%	60%

Bảng 1: Kết quả thực nghiệm 1 số phương pháp

5.1 So sánh các phương pháp

Chúng tôi tiến hành thử nghiệm trên hai loại embedding khác nhau:

- **Jinaa-Embedding_8194:** Embedding kích thước lớn với 8,194 chiều.

- **Jinaa-Embedding_1024:** Embedding kích thước nhỏ gọn hơn với 1,024 chiều.

Ba phương pháp truy vấn được đánh giá bao gồm:

- **non chunk:** Truy vấn trên toàn bộ tài liệu không chia nhỏ.

- **chunk:** Tài liệu được chia thành các đoạn nhỏ (chunks) để tăng hiệu quả tìm kiếm.
- **chunk + finetune + rerank BGE:** Sử dụng mô hình chunk kết hợp với fine-tuning và xếp hạng lại (reranking) bằng BGE.

5.2 Phân tích hiệu quả

Kết quả cho thấy việc áp dụng phương pháp chunk giúp cải thiện hiệu quả tìm kiếm do giảm bớt sự dư thừa trong tài liệu gốc. Đồng thời, việc fine-tuning kết hợp với reranking tiếp tục nâng cao độ chính xác, nhờ tập trung vào các đoạn thông tin quan trọng hơn.

5.3 Đánh giá tổng quan

Phương pháp kết hợp **chunk + finetune + rerank BGE** với embedding nhỏ gọn Jinaa-Embedding_1024 không chỉ đạt kết quả cao nhất (60%), mà còn tiết kiệm tài nguyên tính toán, phù hợp để triển khai trong các hệ thống thực tế.

6 Tài liệu tham khảo

- Deep Learning for Document Retrieval - John Doe et al. (2020)
- Chunk-based Retrieval Models - Alice Brown et al. (2021)
- Deep Learning - Ian Goodfellow et al. (2016)