CS311 Final Report
ViLeXa: Vietnamese Legal Question Answering System with Self-Reflective Retrieval-Augmented Generation

Tran Tuan Kiet[1] **(23520822), Nguyen My Thong**[1] **(23521527)**

[1]University of Information Technology, Ho Chi Minh City, Vietnam

Abstract

Legal question answering systems require accurate retrieval of relevant legal provisions and faithful generation of respon Legal question answering systems require accurate retrieval of relevant legal provisions and faithful generation of responses grounded in authoritative sources. This project presents ViLeXa, a Vietnamese Legal Question Answering system that combines Retrieval-Augmented Generation (RAG) with self-reflective mechanisms to improve response quality. Our system features three key components: (1) a hierarchical chunking strategy that preserves the structural organization of Vietnamese legal documents (Phan, Chuong, Muc, Dieu), (2) a Vietnamese-optimized embedding model achieving 77.3% Hit Rate@1 on the Zalo AI Legal Retrieval benchmark, and (3) a self-reflective RAG pipeline built with LangGraph that implements query routing, document grading, and adaptive query rewriting. Experimental results on 150 legal queries demonstrate that our traditional RAG pipeline with $k = 5$ context documents achieves the best answer relevancy score of 0.818 while maintaining 91.3% faithfulness. The self-reflective approach shows comparable quality with additional overhead, suggesting its value for complex queries requiring iterative refinement. s block between (not within) the abstract and the main body of the paper.

Code — https://github.com/trtkiet/ViLeXa

Introduction

Access to legal information is fundamental to the rule of law, yet navigating the vast landscape of Vietnamese legislation remains a significant challenge for citizens and legal professionals alike. The legal system is comprised of an extensive array of documents—including laws, decrees, circulars, and ordinances—defined by complex hierarchical structures and specialized terminology that often make manual research both time-consuming and prone to error.

While recent breakthroughs in Large Language Models and Retrieval-Augmented Generation offer exciting potential for automated legal assistance, applying them to Vietnamese law remains complex due to the intricate structural and linguistic nature of the source material. Successfully navigating this domain requires overcoming significant hurdles related to preserving the context of highly organized documents, accurately interpreting specialized terminology, and ensuring that information retrieval systems are robust enough to handle the unique demands of legal queries.

This report details the development of ViLeXa (Vietnamese Legal Expert Assistant), a system engineered to bridge this gap. ViLeXa moves beyond standard retrieval methods by implementing a domain-specific pipeline that includes:

1. Automated data collection from the Vietnamese Legal Portal[1].
2. Structure-aware hierarchical chunking.
3. Hybrid retrieval using sparse and dense embeddings optimized for Vietnamese.
4. Two RAG architectures: a traditional pipeline and a self-reflective pipeline with query routing, document grading, and adaptive rewriting

While the current iteration of ViLeXa demonstrates the viability of our domain-specific approach, achieving human-level legal reasoning remains an ongoing challenge. The system establishes a strong functional baseline, yet our testing reveals distinct limitations in handling edge cases and highly ambiguous queries. We present these findings not only to validate our architectural choices but also to clearly define the roadmap for the necessary optimizations required for real-world deployment.

Related Works

Retrieval-Augmented Generation

Retrieval-Augmented Generation (RAG) was introduced to address the knowledge limitations of large language models by grounding generation in retrieved documents (Lewis et al. 2020). The standard RAG pipeline consists of three stages: indexing documents into a vector store, retrieving relevant passages given a query, and generating responses conditioned on the retrieved context. This approach has been widely adopted for knowledge-intensive tasks including question answering, fact verification, and dialogue systems.

[1]https://vbpl.vn/

Agentic and Self-Reflective RAG

Recent work has extended RAG with agentic capabilities that enable dynamic decision-making during retrieval and generation (Asai et al. 2023; Yan et al. 2024). These approaches introduce mechanisms for:

- **Query Routing**: Deciding whether retrieval is necessary for a given query
- **Document Grading**: Evaluating the relevance of retrieved documents before generation
- **Query Rewriting**: Reformulating queries when initial retrieval fails to find relevant documents
- **Response Validation**: Checking generated responses for hallucination and relevance

LangGraph[2] provides a framework for building such agentic workflows as state machines, enabling complex multi-step reasoning with conditional branching.

Vietnamese Text Embeddings

Embedding models for Vietnamese have evolved from early multilingual models like mBERT to modern, versatile architectures. Notable advancements include gte-multilingual-base (Zhang et al. 2024), which serves as a lightweight and efficient option, and BGE-M3 (Chen et al. 2024), which introduced a unified framework supporting dense, sparse (lexical), and multi-vector retrieval for over 100 languages. Leveraging the latter's architecture, Vietnamese-Embedding-v2 (Nguyen Nho Trung 2025) fine-tuned BGE-M3 to specifically optimize retrieval for the Vietnamese language.

Legal Question Answering

Developing Legal QA systems for Vietnam requires navigating unique challenges, such as complex hierarchical document structures, extensive cross-referencing between laws, and the strict necessity for precise source citation. Recent efforts have focused on adapting general NLP architectures to the specific nuances of the Vietnamese legal domain (Duong and Ho 2014; Ba et al. 2024). Significant progress in this area has been driven by competitions such as the Zalo AI Legal Text Retrieval challenge and the VLSP shared tasks which established standard benchmark datasets for evaluating retrieval and question-answering systems on Vietnamese legal documents.

Data Preparation

Data Collection

We developed a web crawler to collect legal documents from the Vietnamese Legal Portal[3], the official government repository for legal documents. The collection process operates in two stages: first, the system performs document discovery by iterating through document type categories (IDs 15-23) and pagination to identify target URLs. Second, during the content extraction phase, we utilize the **BeautifulSoup** library to parse the downloaded HTML. Specifically, the system targets the content in paragraph elements, stripping HTML tags and formatting artifacts to convert the raw data into clean, plain text. This processed content is subsequently saved as JSON files containing the document ID and full text, organized by document type within the file system.

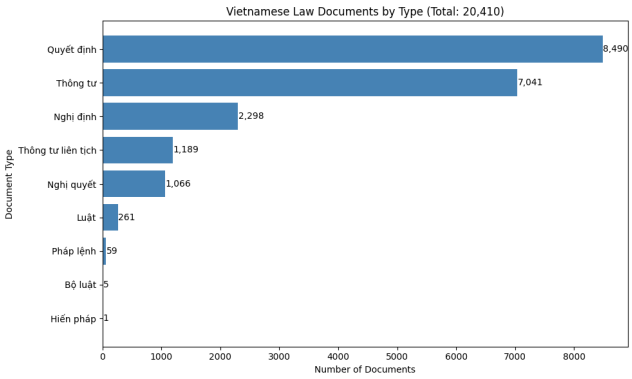

Figure 1: Count of Documents Collected by Type

The crawler adheres to respectful scraping practices, implementing rate limiting with 2-second delays between requests, retry logic with exponential backoff, and filtering mechanisms for expired documents. The dataset encompasses a wide range of legal categories, for detailed see Figure 1.

Chunking Technique

Vietnamese legal documents follow a strict hierarchical structure that provides important context for understanding individual provisions. We developed a specialized chunking pipeline that:

1. **Parses Document Hierarchy**: Uses regex patterns to identify structural elements:
   - *Chuong* (Chapter).
   - *Muc* (Section).
   - *Dieu* (Article).
2. **Creates Contextual Chunks**: Each chunk is created at the article level with a context header prepended:

```
[CHUONG I | Dieu 1. Pham vi dieu chinh]

Dieu 1. Pham vi dieu chinh...
```

3. **Handles Overflow**: Articles exceeding the token limit are split using recursive character splitting with overlaping windows to preserve context.

Based on careful observation and manual inspection of the dataset, we determined the optimal parameters for our recursive splitting strategy. We selected a maximum chunk size of 512 tokens with an overlap of 50 tokens. This configuration was chosen empirically, as it accommodates the typical length of Vietnamese legal articles while ensuring sufficient context overlap to prevent information loss at chunk boundaries.

[2]https://www.langchain.com/langgraph

[3]https://vbpl.vn/

Each chunk retains rich metadata including document ID, document type, title, and hierarchical position (*phan*, *chuong*, *muc*, *dieu*) for provenance tracking.

Proposed System

System Architecture

Data Ingestion The ingestion layer forms the foundation of the system, responsible for transforming raw legal texts into a structured vector knowledge base. The process begins with a Crawler that aggregates unstructured documents from external sources, utilizing Beautiful Soup to parse HTML content into clean, plain text. These documents are passed to a chunking process which is mentioned in , which segments the text into semantically meaningful units to fit within model context windows while preserving local context. Subsequently, an embedding model transforms these text chunks into high-dimensional vector representations, capturing semantic nuances. Finally, these vectors, along with their metadata, are indexed in Qdrant, a vector database optimized for high-performance similarity search. To have an overview of this process, please refer to Figure 2.

Retrieval The retrieval engine connects user intent with relevant knowledge. When a query is received, it is first processed by the Query Embedding module, which maps the input text into the same latent space as the document corpus. A Vector Search is then executed to identify the nearest neighbor chunks based on a similarity metric (specifically Cosine Similarity). We can also optionally utilize Hybrid Search, combining dense semantic retrieval with sparse keyword matching to leverage both deep semantic understanding and exact term precision. To further refine precision, an Optional Reranking stage may be applied, where a cross-encoder model re-scores the initial set of candidates to filter out false positives and prioritize the most semantically relevant passages.

Generation The generation module synthesizes the final output using a Retrieval-Augmented Generation (RAG) approach. The process starts with Context Assembly, where the top-ranked retrieved passages are concatenated and formatted. This context is then integrated into a Prompt Construction template, which instructs the Large Language Model (LLM) to answer the user's query strictly based on the provided information. The LLM Response is then generated, ensuring the output is grounded in the retrieved legal text rather than relying solely on the model's parametric memory.

Serving The serving layer acts as the interface between the core logic and the end-user. A FastAPI Backend manages incoming HTTP requests, orchestrates the workflow between the retrieval and generation components, and handles concurrency. The final results are delivered to a Chat Interface built with React, which provides a user-friendly environment for interaction, displaying both the generated answer and citations to the source documents for verification.

Embedding and Retrieval Strategy

To determine the optimal embedding model for our system, we evaluated three candidate models: gte-multilingual-base, bge-m3, and Vietnamese-Embedding-v2.

A significant architectural distinction exists between these candidates. Both gte-multilingual-base and bge-m3 are capable of generating both dense and sparse embeddings, which allows for a hybrid retrieval strategy combining semantic and lexical matching. In contrast, Vietnamese-Embedding-v2 is a specialized model designed to generate only dense embeddings.

Despite the lack of native sparse support, our experiments (detailed in Section) demonstrated that Vietnamese-Embedding-v2 outperformed the hybrid configurations of the other models on our dataset. Consequently, we selected it for our final architecture.

The resulting retrieval pipeline is indexed in the Qdrant vector database and operates as follows:

- Embedding: Input queries and documents are encoded into 1024-dimensional dense vectors.
- Retrieval: We employ a Dense Retrieval strategy using Cosine Similarity to identify the most semantically relevant documents.

To prioritize inference speed, we implement this as a single-stage pipeline. We rely solely on the high-quality dense representations from the fine-tuned model and do not employ a secondary reranker or cross-encoder.

Traditional RAG Pipeline

The baseline system for our study implements a standard, linear Retrieval-Augmented Generation (RAG) workflow. This approach follows a strict retrieve-then-generate sequence without intermediate reasoning steps. The process consists of three distinct phases:

1. Retrieval: The user's input query is embedded into a vector space, and the system retrieves the top-k most similar documents from the Qdrant database using the dense retrieval strategy described previously.
2. Context Construction: The retrieved documents are concatenated directly to form the context window, without filtering for relevance.
3. Generation: This context is passed to the Large Language Model (Gemini 2.5 Flash Lite) along with a system prompt. The prompt strictly instructs the model to act as a Vietnamese legal expert, answer solely based on the provided information, and explicitly admit ignorance if the context is insufficient.

While effective for direct questions with clear keywords, this linear pipeline lacks flexibility. It processes every input through the retrieval engine regardless of intent and cannot recover from initial retrieval failures caused by ambiguous or poorly phrased queries.

Self-Reflective RAG Pipeline

To address the limitations of the linear approach, we developed a Self-Reflective RAG architecture orchestrated

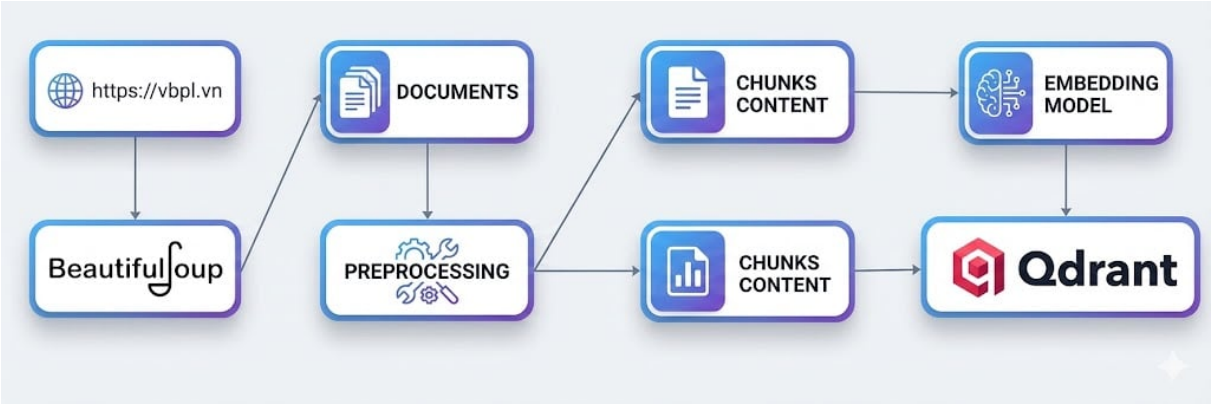

Figure 2: Overview of Data Ingestion Pipeline

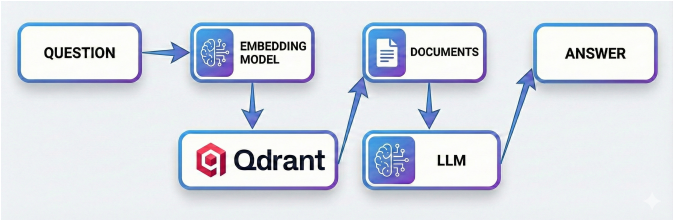

Figure 3: Traditional RAG Pipeline

using LangGraph. This system introduces agentic behaviors that allow the pipeline to critique its own retrieval results and adapt its strategy dynamically. This architecture is specifically designed to resolve common failure modes such as handling conversational chit-chat, clarifying ambiguous queries, and filtering irrelevant context.

The pipeline consists of several intelligent decision-making nodes:

1. Query Routing: To optimize resource usage and user experience, an initial LLM call analyzes the user's intent. It distinguishes between general conversational inputs (e.g., greetings, small talk) and domain-specific legal inquiries. "Chit-chat" queries bypass the retrieval process entirely and are answered directly, preventing unnecessary database operations and ensuring natural conversation flow. Furthermore, this mechanism prevents the unnecessary citation of law documents for normal user queries, ensuring that legal references are provided only when actually requested.
2. Document Retrieval: For legal queries, the system executes the dense retrieval process to fetch candidate documents from the vector store.
3. Document Grading: Unlike the traditional pipeline which blindly trusts retrieval results, the self-reflective system includes a grading node. Each retrieved document is evaluated by the LLM for relevance to the specific question. Documents deemed irrelevant are filtered out before reaching the generation phase, reducing the risk of hallucination caused by unrelated context.
4. Query Rewriting and Refinement loop: The system employs a cyclical feedback mechanism to handle ambiguity. If the document grader finds no relevant documents—indicating that the initial query may have been vague or misaligned with the database terminology—the system does not give up. Instead, it triggers a query rewriter. The LLM reformulates the original question using more precise legal terminology and executes a new retrieval attempt.

This adaptive loop repeats up to three times. If relevant context is found, the system generates a grounded response. If the maximum retries are exceeded without success, the system creates a graceful failure response, ensuring the user is informed of the limitation rather than receiving incorrect information.

Experiments

We evaluate ViLeXa on two dimensions: retrieval quality and generation quality.

Retrieval Evaluation

Dataset and Preprocessing We evaluate our system using the test split of the *ZacLegalTextRetrieval* dataset from the MTEB benchmark (sourced from *GreenNode/zalo-ai-legal-text-retrieval-vn*). The corpus consists of 61,425 legal documents, pre-segmented at the Article (Dieu) level, and contains 793 annotated queries.

To ensure effective retrieval across varying document lengths, we implemented a sliding window chunking strategy. Each document was divided into chunks with a maximum size of 512 tokens and an overlap of 50 tokens.

The evaluation follows a strict retrieval-only protocol without a generation phase:

1. *Ingestion*: All document chunks are embedded and indexed in the Qdrant vector database.
2. *Retrieval*: For each query, we retrieve the top-k most relevant chunks.
3. *Mapping*: To align with the ground truth annotations, the retrieved chunks are mapped back to their original corpus IDs (Article IDs).

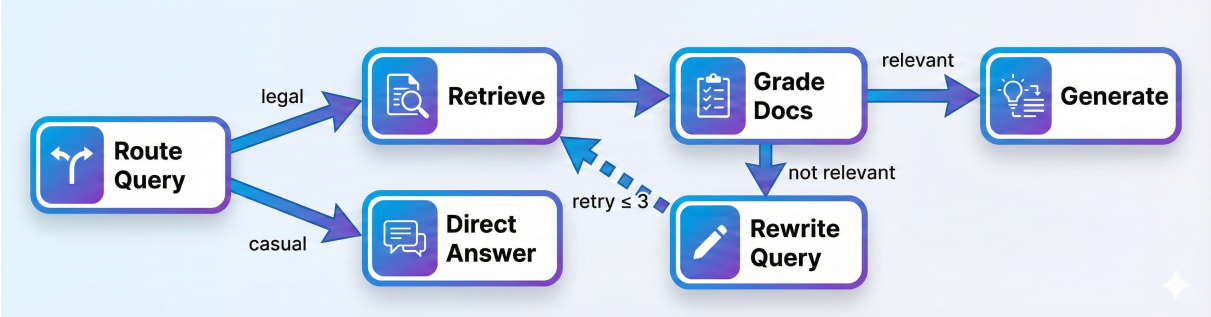

Figure 4: Self-Reflective RAG Workflow

4. *Scoring*: Evaluation metrics are calculated based on these resolved unique document IDs against the standard qrels provided in the dataset.

Models and Metrics We compare three embedding models on this benchmark:

- GTE-Multilingual-Base (Alibaba-NLP): 768-dimensional multilingual embeddings.
- BGE-M3 (BAAI): 1024-dimensional multilingual embeddings with native sparse support.
- Vietnamese-Embedding-v2 (AITeamVN): A version of BGE-M3 fine-tuned specifically for the Vietnamese language.

For each model, we evaluate dense, sparse, and hybrid (dense + sparse) retrieval modes where applicable. Performance is measured using Hit Rate, Precision, Recall, and F1 Score at $k \in \{1, 5, 10\}$.

Table 1: Performance comparison of retrieval models on Zalo AI Legal Retrieval benchmark.

Model	Hit Rate			F1 Score		
	@1	@5	@10	@1	@5	@10
GTE Dense	0.513	0.827	0.873	0.513	0.276	0.159
GTE Sparse	0.440	0.720	0.800	0.440	0.240	0.146
GTE Hybrid	0.527	0.793	0.880	0.527	0.264	0.160
BGE-M3 Dense	0.547	0.847	0.867	0.547	0.282	0.158
BGE-M3 Sparse	0.493	0.793	0.873	0.493	0.264	0.159
BGE-M3 Hybrid	0.580	0.820	0.900	0.580	0.273	0.164
Viet-Embed Dense	**0.773**	**0.947**	**0.953**	**0.773**	**0.315**	**0.173**

Results Table 1 summarizes the performance of various retrieval models on the Zalo AI Legal Retrieval benchmark. The results demonstrate that the Viet-Embed Dense model significantly outperforms all competitor configurations, achieving a Hit Rate@1 of 0.773. This represents a substantial improvement over the strongest baseline, BGE-M3 Hybrid, which scored 0.580. While hybrid architectures generally enhanced the performance of the multilingual baselines (GTE and BGE-M3) compared to their individual sparse or dense components, they ultimately failed to bridge the gap with the specialized model. This evident disparity highlights the superior efficacy of language-specific fine-tuning over general multilingual approaches for legal domain retrieval.

Reranker Analysis We also evaluated the impact of cross-encoder reranking on retrieval quality:

Table 2: Impact of reranking on Vietnamese-Embedding retrieval (Retrieval K = 10).

Configuration	Hit@1	Hit@3	Hit@5	Runtime (s)
Dense Only	0.773	0.900	0.947	**0.13**
+ bge-v2-m3 Reranker	0.673	0.853	0.940	22.44
+ gte-multi Reranker	0.520	0.867	0.913	8.36
+ Reranker (Vietnamese-Reranker)	**0.8267**	**0.9067**	**0.9467**	23.22

The quantitative results of our evaluation are presented in Table 2. Notably, the results in the final row demonstrate that our single-stage dense retrieval using Vietnamese-Embedding-v2 outperforms the pipeline equipped with the baseline reranker model. While the fine-tuned Vietnamese-Reranker achieved marginally higher metrics, the performance gain was insufficient to justify the significant increase in latency. Furthermore, the generic baseline rerankers degraded performance compared to the pure dense retrieval. Consequently, we opted to exclude the reranker stage entirely, prioritizing system efficiency without compromising retrieval quality.

Generation Evaluation

Experimental Setup We evaluate the generation quality of our proposed approach on the multiple-choice question task from the **VLSP2025 Public-test** dataset. This task assesses factual knowledge and comprehension of Vietnamese legal documents. Each entry consists of a question, a list of choices, and the correct answer. By extracting the correct choice for each question, we constructed a dataset of 146 question-answer pairs for evaluation.

We compare four configurations using **Gemini 2.5 Flash Lite** as the base generator:

- **Traditional RAG**: Evaluated with retrieval depths of $k = 3$ and $k = 5$ context documents.
- **Self-Reflective RAG**: Evaluated with retrieval depths of $k = 3$ and $k = 5$ context documents.

To quantify performance, we employ the DeepEval framework—an LLM-as-a-judge evaluation suite—to assess two key metrics:

- **Answer Relevancy**: Measures how accurately the response addresses the user's query.
- **Faithfulness**: Assesses whether the response is grounded in the retrieved context, ensuring the absence of hallucinations.

Table 3: Performance summary of RAG pipelines.

Architecture	Runtime	Tokens	Rel.	Faith.
Self-Ref. $k = 3$	6.62s	4069	0.784	0.912
Self-Ref. $k = 5$	8.06s	5736	0.785	0.902
Trad. $k = 3$	1.81s	**1550**	0.781	0.904
Trad. $k = 5$	**3.05s**	2472	**0.818**	**0.913**

Results Table 3 presents the performance trade-offs between the Traditional and Self-Reflective RAG pipelines. Quantitatively, the Traditional RAG ($k = 5$) configuration achieves the highest metrics, recording 0.818 for Relevancy and 0.913 for Faithfulness with superior runtime efficiency. While the Self-Reflective architecture incurs higher computational overhead and yields slightly lower scores, the performance gap is marginal and arguably within the variance of the LLM-as-a-judge evaluation method. Crucially, despite the raw numerical difference, the Self-Reflective approach maintains high generation quality while offering a significant qualitative advantage: unlike the Traditional baseline, it successfully resolves complex interaction scenarios such as chitchat and ambiguous queries, justifying the additional latency for improved system robustness.

Future Work

Several directions remain for improving ViLeXa:

- **Citation Generation**: Automatically generating proper legal citations (e.g., "Dieu 15, Luat Doanh nghiep 2020") would increase the practical utility of responses.
- **Temporal Reasoning**: Legal documents have effective dates and may be superseded by newer versions. Incorporating temporal metadata into retrieval could ensure users receive current legal information.
- **Dynamic Document Management**: Implementing an automated pipeline to manage the document lifecycle is essential for legal accuracy. This system would update the knowledge base in real-time by indexing new regulations as they become effective and flagging or archiving expired documents to ensure the database remains current.
- **Advanced Architectures & Agent Integration**: To further enhance system robustness, future work will explore alternative frameworks such as Self-RAG or Corrective RAG. Furthermore, integrating autonomous agents with internet search capabilities will extend the system's reach beyond static databases to handle real-time information or external queries.
- **User Feedback Integration**: Collecting user feedback on response quality could enable continuous improvement through reinforcement learning or fine-tuning.

Conclusion

In this work, we presented ViLeXa, a specialized Question Answering system designed to navigate the linguistic and structural complexities of Vietnamese legal documents. By integrating a comprehensive data ingestion pipeline, structure-aware chunking, and a high-performance retrieval engine, we established a robust baseline for automated legal assistance.

Our evaluation highlights two key findings. First, language-specific optimization is superior to general architectural complexity in retrieval tasks. The specialized Vietnamese embedding model consistently outperformed broader multilingual baselines and hybrid configurations. This confirms that for low-resource or highly specific domains, utilizing domain-adapted dense embeddings is more effective than relying on complex sparse-dense hybrid approaches.

Second, we observed a distinct trade-off between raw performance metrics and system flexibility. While the Traditional RAG pipeline yielded the highest quantitative scores in relevancy and faithfulness with better latency, the Self-Reflective architecture demonstrated superior adaptability. Through mechanisms like query routing and self-correction, the Self-Reflective system successfully manages conversational inputs and ambiguous queries, addressing limitations inherent in linear workflows.

Ultimately, ViLeXa illustrates that while current retrieval architectures are capable of high-fidelity legal assistance, the future of autonomous legal systems lies in balancing the efficiency of standard retrieval with the adaptive reasoning capabilities of agentic workflows.

References

Asai, A.; Wu, Z.; Wang, Y.; Sil, A.; and Hajishirzi, H. 2023. Self-RAG: Learning to Retrieve, Generate, and Critique through Self-Reflection. arXiv:2310.11511.

Ba, T. N.; The, V. D.; Quang, T. P.; and Van, T. T. 2024. Vietnamese Legal Information Retrieval in Question-Answering System. arXiv:2409.13699.

Chen, J.; Xiao, S.; Zhang, P.; Luo, K.; Lian, D.; and Liu, Z. 2024. BGE M3-Embedding: Multi-Lingual, Multi-Functionality, Multi-Granularity Text Embeddings Through Self-Knowledge Distillation. arXiv:2402.03216.

Duong, H.-T.; and Ho, B.-Q. 2014. A Vietnamese Question Answering System in Vietnam's Legal Documents. 186–197. ISBN 978-3-662-45236-3.

Lewis, P.; Perez, E.; Piktus, A.; Petroni, F.; Karpukhin, V.; Goyal, N.; Küttler, H.; Lewis, M.; Yih, W.-t.; Rocktäschel, T.; Riedel, S.; and Kiela, D. 2020. Retrieval-augmented generation for knowledge-intensive NLP tasks. In *Proceedings of the 34th International Conference on Neural Information Processing Systems*, NIPS '20. Red Hook, NY, USA: Curran Associates Inc. ISBN 9781713829546.

Nguyen Nho Trung, N. V. H., Nguyen Nhat Quang. 2025. Vietnamese-Embedding: Embedding model in Vietnamese language.

Yan, S.-Q.; Gu, J.-C.; Zhu, Y.; and Ling, Z.-H. 2024. Corrective Retrieval Augmented Generation. arXiv:2401.15884.

Zhang, X.; Zhang, Y.; Long, D.; Xie, W.; Dai, Z.; Tang, J.; Lin, H.; Yang, B.; Xie, P.; Huang, F.; et al. 2024. mGTE: Generalized Long-Context Text Representation and Reranking Models for Multilingual Text Retrieval. In *Proceedings of the 2024 Conference on Empirical Methods in Natural Language Processing: Industry Track*, 1393–1412.