Institut National des Sciences Appliquées de Toulouse

Optimizing and Adapting Language Models for Domain-Specific Task

End-of-studies Apprenticeship Report

Minh Duy Nguyen

Bachelor's Degree in Computer Science and Engineering

Supervised by: Milad Mozafari (Torus AI), David Bertoin (INSA Toulouse)

Ngày 16 tháng 1 năm 2026

Chương 1

Introduction

1.1 General Context

1.1.1 Sự phát triển của Mô hình Ngôn ngữ Lớn

Trong thập kỷ qua, lĩnh vực Trí tuệ nhân tạo (Artificial Intelligence - AI) và Xử lý ngôn ngữ tự nhiên (Natural Language Processing - NLP) đã chứng kiến những bước tiến đột phá, đặc biệt kể từ sự ra đời của kiến trúc Transformer [Vaswani et al., 2017]. Các Mô hình Ngôn ngữ Lớn (Large Language Models - LLMs) như GPT-4 [OpenAI, 2023], Claude 3 [Anthropic, 2024], Gemini 1.5 [Google, 2024], cùng các mô hình mã nguồn mở như Llama 3 [Meta, 2024] và Qwen 2.5 [Alibaba, 2024] đã chứng minh khả năng vượt trội trong việc hiểu ngữ cảnh, sinh văn bản và thực hiện suy luận logic trên đa dạng tác vụ.

Theo báo cáo của McKinsey Global Institute (2024), thị trường AI tạo sinh (Generative AI) dự kiến đạt giá trị 4.4 nghìn tỷ USD vào năm 2030. Tuy nhiên, một khảo sát từ Gartner (2024) chỉ ra rằng **67% doanh nghiệp** gặp khó khăn trong việc triển khai LLMs cho các tác vụ chuyên biệt do các hạn chế về độ chính xác và chi phí vận hành.

1.1.2 Khoảng cách giữa Nghiên cứu và Ứng dụng Công nghiệp

Việc chuyển giao các mô hình ngôn ngữ từ môi trường nghiên cứu sang ứng dụng thực tiễn trong công nghiệp (Industrial Deployment) đang đối mặt với những thách thức đáng kể. Mặc dù các mô hình nền tảng (Foundation Models) sở hữu lượng tri thức tổng quát khổng lồ, chúng thường gặp hai nhóm hạn chế nghiêm trọng khi giải quyết các bài toán chuyên biệt (Domain-specific tasks):

Khoảng cách về Tri thức (Knowledge Gap):

Trong các lĩnh vực quan trọng như y tế, tài chính và bảo hiểm, thông tin thường nằm trong các tài liệu phức tạp (báo cáo tài chính, bảng danh mục kỹ thuật y tế CCAM) và thay đổi liên tục. Các LLMs đối mặt với ba vấn đề chính:

- **Dữ liệu tĩnh (Static Knowledge):** Tri thức của mô hình bị giới hạn bởi thời điểm huấn luyện (knowledge cutoff), không thể cập nhật thông tin mới mà không cần huấn luyện lại [Lewis et al., 2020]. - **Thiếu dữ liệu nội bộ (Private Data Inaccessibility):** Mô hình không thể truy cập các tài liệu bảo mật của tổ chức như hợp đồng, quy trình nội bộ, hay hồ sơ bệnh nhân. - **Hiện tượng ảo giác (Hallucination):** LLMs có xu hướng sinh ra thông tin sai lệch một cách tự tin [Ji et al., 2023], điều không thể chấp nhận trong các quyết định y khoa hay tài chính, nơi yêu cầu độ chính xác gần như tuyệt đối.

Khoảng cách về Hiệu quả và Hành vi (Efficiency & Behavior Gap):

Đối với các tác vụ yêu cầu mô hình tuân thủ một kịch bản hành vi cụ thể, một phong cách ngôn ngữ đặc thù (ví dụ: tư vấn viên tâm lý, chuyên gia phân tích), hoặc triển khai trên hạ tầng phần cứng giới hạn, việc sử dụng các mô hình khổng lồ (hàng trăm tỷ tham số) là không tối ưu về chi phí và độ trễ. Theo Hoffmann et al. (2022), chi phí inference của GPT-4 có thể lên tới **$0.06/1000 tokens**, khiến việc triển khai quy mô lớn trở nên không khả thi cho nhiều doanh nghiệp vừa và nhỏ.

1.1.3 1.1.3. Động lực nghiên cứu

Xuất phát từ thực tế trên, luận văn này tập trung nghiên cứu và triển khai các kỹ thuật tiên tiến nhằm **tối ưu hóa và thích ứng LLMs cho các miền dữ liệu đặc thù**, với hai hướng tiếp cận chính:

1. **Retrieval-Augmented Generation (RAG):** Tích hợp tri thức bên ngoài để giải quyết Knowledge Gap. 2. **Parameter-Efficient Fine-Tuning (PEFT):** Thích ứng hành vi mô hình để giải quyết Behavior Gap.

1.2 Internship Environment

This master's thesis was completed at Torus AI, a technology company with the mission "Intelligence for Life,"specializing in providing advanced AI solutions to improve quality of life and business efficiency.

I worked in the Research and Development Team (R&D Team - Torus Lab) as a Machine Learning Engineer Alternant. At Torus AI, the R&D department acts as a bridge between the latest academic research (State-of-the-Art) and commercial products. The team's main task is to continuously explore emerging Generative AI technologies, assess their feasibility, and build functional prototypes (PoCs) to verify their effectiveness before integration into the main product system.

The R&D work environment demands flexible thinking: not just using existing APIs, but delving into customizing architecture, optimizing data processing pipelines, and quantitatively evaluating technical solutions.

Vai trò của tác giả: Kỹ sư Học máy tập sự (Machine Learning Engineer Alternant) trong đội ngũ Nghiên cứu và Phát triển (R&D Team - Torus Lab), với nhiệm vụ: - Khảo sát các công nghệ Generative AI mới nổi - Đánh giá tính khả thi và xây dựng các bản mẫu chức năng (Functional Prototypes/PoC) - Tích hợp các giải pháp vào hệ thống sản phẩm

Tài nguyên được cung cấp: - Hạ tầng GPU: NVIDIA A100 (40GB), T4 (16GB) trên nền tảng đám mây - Dữ liệu doanh nghiệp thực tế: Tài liệu y tế (CCAM, NGAP), báo cáo tài chính bảo hiểm - API access: OpenAI GPT-4, Google Gemini, Anthropic Claude

1.3 Problem Statement

During our work in the R&D department, we identified two core problems that needed to be addressed when applying GenAI in practice, corresponding to two main technical approaches:

Problem 1: Integration of External Knowledge from Complex Unstructured Data

Partner businesses (such as insurance companies, healthcare facilities) possess large amounts of data in the form of PDF documents containing text, tables, and images. Traditional RAG (Retrieval-Augmented Generation) methods based on plain text (text-only) fail to understand the semantics of complex tables or visual information. The question is: How to build a Multimodal

RAG pipeline capable of accurately parsing, indexing, and retrievaling information from these mixed documents to support decision-making (e.g., medical refund code lookup, financial data analysis)?

Problem 2: Behavioral Adaptation and Resource Optimization for Small Models (Behavioral Adaptation & Efficiency)

For applications requiring high interactivity, counseling, or entertainment (such as psychological counseling chatbots or Tarot Readers), the requirement is not only for information accuracy but also for consistency in tone and style and rule-based reasoning. Using large models (like GPT-4) via APIs is both costly to operate and difficult to fully control behavior. The question is: Is it possible to fine-tune small language models (such as Qwen, Llama $<$ 7B parameters) using parameter optimization techniques (PEFT/LoRA) and quantization so that they achieve inference capabilities and writing styles comparable to large models, but can be run locally at low cost?

1.4 1.3. Phát biểu vấn đề nghiên cứu (Problem Statement)

1.4.1 1.3.1. Các câu hỏi nghiên cứu (Research Questions)

Trong quá trình làm việc tại bộ phận R&D, chúng tôi đã xác định hai nhóm vấn đề cốt lõi cần giải quyết khi áp dụng Generative AI vào thực tế, được hình thức hóa thành các câu hỏi nghiên cứu sau:

—

RQ1 (Research Question 1): *Làm thế nào để xây dựng một pipeline Multimodal RAG có khả năng phân tích cú pháp (parsing), lập chỉ mục (indexing) và truy xuất (retrieval) chính xác thông tin từ các tài liệu hỗn hợp (văn bản, bảng biểu, hình ảnh) để hỗ trợ ra quyết định trong lĩnh vực y tế và tài chính?*

Bối cảnh: Các doanh nghiệp đối tác sở hữu lượng lớn dữ liệu dưới dạng tài liệu PDF chứa văn bản, bảng biểu (tables) và hình ảnh (charts/images). Các phương pháp RAG truyền thống dựa trên văn bản thuần túy (text-only) thất bại trong việc hiểu ngữ nghĩa của bảng biểu phức tạp hoặc thông tin thị giác.

—

RQ2 (Research Question 2): *Liệu có thể tinh chỉnh (Fine-tuning) các mô hình ngôn ngữ nhỏ (Small Language Models - SLMs, $<$ 7B tham số) bằng các kỹ thuật tối ưu tham số (PEFT/LoRA) và lượng tử hóa (Quantization) để đạt được khả năng suy luận và văn phong tương đương các mô hình lớn, nhưng có thể chạy cục bộ với chi phí thấp?*

Bối cảnh: Đối với các ứng dụng yêu cầu tính tương tác cao, mang tính chất tư vấn (như chatbot tư vấn hoặc hệ thống suy luận dựa trên quy tắc), yêu cầu không chỉ là độ chính xác của thông tin mà còn là sự nhất quán trong văn phong (Tone & Style) và khả năng suy luận theo quy tắc (Rule-based reasoning).

1.4.2 1.3.2. Giả thuyết nghiên cứu (Research Hypotheses)

Dựa trên các câu hỏi nghiên cứu, chúng tôi đề xuất các giả thuyết sau:

H1 (Hypothesis 1): *Việc kết hợp Hybrid Search (Dense Retrieval + Sparse Retrieval) với Cross-Encoder Reranking và chiến lược Parent-Child Indexing sẽ cải thiện đáng kể độ chính xác truy xuất (Retrieval Accuracy) và độ trung thực (Faithfulness) của hệ thống RAG so với phương pháp Dense Search đơn thuần, đặc biệt trên dữ liệu bảng biểu và mã code y tế.*

H2 (Hypothesis 2): *Các mô hình ngôn ngữ nhỏ (1.5B - 3B tham số) sau khi được Fine-tuning bằng QLoRA trên dữ liệu tổng hợp (Synthetic Data) chất lượng cao có thể học được văn phong (Style) và định dạng đầu ra (Output Format) của tác vụ chuyên biệt, nhưng sẽ gặp hạn chế về khả năng suy luận phức tạp (Complex Reasoning) so với các mô hình lớn hơn (>70B tham số).*

1.5 Objectives & Contributions

1.6 Thesis Structure

Chương 2

Cơ sở lý thuyết

Chương này cung cấp nền tảng lý thuyết và toán học cần thiết cho các phương pháp được áp dụng trong luận văn. Chúng tôi trình bày có hệ thống từ kiến trúc cơ bản của LLMs đến các kỹ thuật thích ứng tiên tiến, bao gồm Retrieval-Augmented Generation (RAG) và Parameter-Efficient Fine-Tuning (PEFT).

2.1 Nền tảng Mô hình Ngôn ngữ Lớn

2.1.1 Kiến trúc Transformer và Cơ chế Self-Attention

Hầu hết các LLM hiện đại sử dụng kiến trúc Decoder-only Transformer [Vaswani et al., 2017]. Thành phần cốt lõi cho phép mô hình xử lý các phụ thuộc tầm xa (long-range dependencies) là cơ chế Self-Attention. Một cách toán học, cơ chế này được định nghĩa như sau: Cho một chuỗi đầu vào được biểu diễn bởi ma trận embedding $\mathbf{X} \in \mathbb{R}^{n \times d_{model}}$, trong đó n là độ dài chuỗi và d_{model} là chiều embedding (có thể hiểu là một chuỗi chứa n tokens, mỗi token là 1 vector trong không gian d_{model} chiều). Cơ chế Self-Attention thực hiện phép biến đổi tuyến tính từ ma trận embedding $\mathbf{X}$ để tạo ra ba ma trận:

$$\mathbf{Q} = \mathbf{X}\mathbf{W}_Q, \quad \mathbf{K} = \mathbf{X}\mathbf{W}_K, \quad \mathbf{V} = \mathbf{X}\mathbf{W}_V$$

trong đó $\mathbf{W}_Q, \mathbf{W}_K \in \mathbb{R}^{d_{model} \times d_k}$ và $\mathbf{W}_V \in \mathbb{R}^{d_{model} \times d_v}$ là các ma trận trọng số học được.

Đầu ra của Scaled Dot-Product Attention được tính như sau:

$$\text{Attention}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{softmax}\left(\frac{\mathbf{Q}\mathbf{K}^T}{\sqrt{d_k}}\right)\mathbf{V}$$

Phân tích vai trò của hệ số tỷ lệ $\frac{1}{\sqrt{d_k}}$

Hệ số $\frac{1}{\sqrt{d_k}}$ đóng vai trò quan trọng trong việc ổn định số học (numerical stability). Để hiểu tại sao, xét tích vô hướng của hai vector $\mathbf{q}, \mathbf{k} \in \mathbb{R}^{d_k}$ với các thành phần độc lập, có trung bình 0 và phương sai 1:

$$\mathbb{E}[\mathbf{q}^T\mathbf{k}] = 0, \quad \text{Var}[\mathbf{q}^T\mathbf{k}] = d_k$$

Khi d_k lớn (thường $d_k = 64$ hoặc 128), giá trị $\mathbf{q}^T\mathbf{k}$ có độ lệch chuẩn $\sqrt{d_k}$, đẩy softmax vào vùng bão hòa (saturation regions) với gradient gần 0. Việc chia cho $\sqrt{d_k}$ đưa phương sai về 1, đảm

bảo gradient ổn định trong quá trình backpropagation.

Multi-Head Attention

Để cho phép mô hình học các biểu diễn từ nhiều không gian con (subspaces) khác nhau, Transformer sử dụng Multi-Head Attention:

$$\text{MultiHead}(\mathbf{Q}, \mathbf{K}, \mathbf{V}) = \text{Concat}(\text{head}_1, \ldots, \text{head}_h)\mathbf{W}_O$$

trong đó $\text{head}_i = \text{Attention}(\mathbf{Q}\mathbf{W}_Q^i, \mathbf{K}\mathbf{W}_K^i, \mathbf{V}\mathbf{W}_V^i)$.

Độ phức tạp tính toán: Self-Attention có độ phức tạp $O(n^2 \cdot d)$ về thời gian và $O(n^2)$ về bộ nhớ, là bottleneck chính khi xử lý chuỗi dài.

2.1.2 Quá trình Huấn luyện LLM

Các LLM hiện đại thường trải qua ba giai đoạn huấn luyện:

Giai đoạn 1: Pre-training (Huấn luyện trước)

Mô hình được huấn luyện trên lượng lớn văn bản không gán nhãn với mục tiêu Causal Language Modeling (dự đoán token tiếp theo):

$$\mathcal{L}_{pretrain} = -\sum_{t=1}^{T} \log P_\theta(x_t | x_{<t})$$

Giai đoạn 2: Supervised Fine-Tuning (SFT)

Mô hình được tinh chỉnh trên dữ liệu có cấu trúc (instruction, response) để học cách tuân theo chỉ dẫn.

Giai đoạn 3: Alignment (RLHF/DPO)

Tối ưu hóa mô hình theo sở thích của con người thông qua Reinforcement Learning from Human Feedback hoặc Direct Preference Optimization.

Trải qua 3 giai đoạn training cơ bản trên, LLM có khả năng hiểu ngữ cảnh, sinh văn bản và thực hiện suy luận logic trên đa dạng tác vụ giống như con người.

2.1.3 Các Hạn chế Cơ bản của LLMs trong Miền Đặc thù

Mặc dù có khả năng sinh văn bản mạnh mẽ, các LLM pre-trained gặp ba hạn chế cơ bản khi áp dụng vào miền đặc thù:

Hạn chế	Mô tả	Hệ quả
Hallucination	Mô hình sinh thông tin không có trong dữ liệu huấn luyện	Không thể tin cậy trong Y tế, Pháp lý
Knowledge Cutoff	Tri thức bị giới hạn bởi thời điểm pre-training	Thiếu thông tin cập nhật
Behavioral Rigidity	Khó thích ứng với văn phong/quy tắc đặc thù	Cần prompt engineering phức tạp

Bảng 2.1: Các hạn chế cơ bản của LLMs

Định lý 2.1 (Giới hạn của In-Context Learning): Với mô hình có context window kích thước C tokens, số lượng ví dụ few-shot tối đa có thể đưa vào là $k_{max} = \lfloor C/L_{avg} \rfloor$, trong đó L_{avg} là độ dài trung bình của mỗi ví dụ. Khi tác vụ yêu cầu nhiều quy tắc phức tạp, k_{max} thường không đủ để mô hình học được đầy đủ.

2.2 Retrieval-Augmented Generation (RAG)

2.2.1 Động lực và Tổng quan

Để giải quyết vấn đề Knowledge Cutoff và Hallucination, Lewis et al. (2020) đề xuất framework Retrieval-Augmented Generation (RAG), kết nối mô hình sinh văn bản với một cơ chế truy xuất thông tin bên ngoài. Để định nghĩa một cách hình thức, cho $\mathcal{D} = \{d_1, d_2, \ldots, d_N\}$ là kho tài liệu (document corpus) và q là truy vấn của người dùng, hệ thống RAG thực hiện hai bước:

Bước 1 - Retrieval: Tìm tập $\mathcal{C} \subset \mathcal{D}$ chứa k tài liệu liên quan nhất:

$$\mathcal{C} = \text{Top-}k_{d \in \mathcal{D}}\ \text{sim}(q, d)$$

Bước 2 - Generation: Sinh câu trả lời a dựa trên cả q và $\mathcal{C}$:

$$a = \arg\max_{a} P_\theta(a|q, \mathcal{C})$$

2.2.2 Dense Retrieval và Embedding Vectors

Mô hình Bi-Encoder

Phương pháp Dense Retrieval sử dụng mạng nơ-ron để ánh xạ văn bản vào không gian vector liên tục. Cho encoder $f_\theta : \mathcal{T} \to \mathbb{R}^d$, vector biểu diễn của truy vấn và tài liệu là:

$$\mathbf{v}_q = f_\theta(q), \quad \mathbf{v}_d = f_\theta(d)$$

Độ tương đồng Cosine (Cosine Similarity)

Độ liên quan giữa truy vấn và tài liệu được đo bằng Cosine Similarity:

$$\text{sim}_{cos}(\mathbf{v}_q, \mathbf{v}_d) = \frac{\mathbf{v}_q \cdot \mathbf{v}_d}{\|\mathbf{v}_q\| \cdot \|\mathbf{v}_d\|} = \cos(\theta)$$

trong đó θ là góc giữa hai vector.

Tính chất toán học: - $\text{sim}_{cos} \in [-1, 1]$ - Bất biến với độ dài vector (độ dài văn bản) - Tập trung vào hướng (semantic orientation) thay vì magnitude

Mệnh đề 2.1: *Với các vector đã được chuẩn hóa ($\|\mathbf{v}\| = 1$), Cosine Similarity tương đương với tích vô hướng (Dot Product), và khoảng cách Euclidean có quan hệ đơn điệu:*

$$\|\mathbf{v}_q - \mathbf{v}_d\|^2 = 2(1 - \text{sim}_{cos}(\mathbf{v}_q, \mathbf{v}_d))$$

2.2.3 2.2.3. Sparse Retrieval với BM25

Motivation

Dense Retrieval hiệu quả trong việc bắt ngữ nghĩa (semantic similarity), nhưng có thể thất bại với các truy vấn chứa **từ khóa chính xác** (exact keywords) như mã code y tế (ICD-10,

CCAM) hoặc tên riêng.

Công thức BM25

BM25 (Best Matching 25) là thuật toán Sparse Retrieval dựa trên tần suất từ xuất hiện. Cho truy vấn Q chứa các từ $\{q_1, \ldots, q_n\}$, điểm BM25 của tài liệu D là:

$$\text{BM25}(D, Q) = \sum_{i=1}^{n} \text{IDF}(q_i) \cdot \frac{f(q_i, D) \cdot (k_1 + 1)}{f(q_i, D) + k_1 \cdot \left(1 - b + b \cdot \frac{|D|}{avgdl}\right)}$$

trong đó: - $f(q_i, D)$: Tần suất xuất hiện của từ q_i trong tài liệu D - $|D|$: Độ dài tài liệu (số từ) - $avgdl$: Độ dài trung bình của tất cả tài liệu trong corpus - k_1, b: Siêu tham số (thường $k_1 = 1.5$, $b = 0.75$) - $\text{IDF}(q_i)$: Inverse Document Frequency

$$\text{IDF}(q_i) = \log\left(\frac{N - n(q_i) + 0.5}{n(q_i) + 0.5} + 1\right)$$

với N là tổng số tài liệu và $n(q_i)$ là số tài liệu chứa từ q_i.

Phân tích: BM25 là hàm **sublinear** theo tần suất từ, tránh việc tài liệu dài quá được ưu tiên.

2.2.4 2.2.4. Hybrid Search: Kết hợp Dense và Sparse

Reciprocal Rank Fusion (RRF)

Để kết hợp kết quả từ hai phương pháp retrieval, chúng tôi sử dụng Reciprocal Rank Fusion [Cormack et al., 2009]:

$$\text{RRF}(d) = \sum_{r \in \mathcal{R}} \frac{1}{k + \text{rank}_r(d)}$$

trong đó $\mathcal{R}$ là tập các danh sách xếp hạng (từ Dense và Sparse), $\text{rank}_r(d)$ là thứ hạng của tài liệu d trong danh sách r, và k là hằng số điều chỉnh (thường $k = 60$).

Linear Combination với Trọng số Thích ứng

Một phương pháp thay thế là kết hợp tuyến tính với trọng số:

$$\text{Score}(q, d) = \alpha \cdot \text{Norm}(S_{dense}(q, d)) + (1 - \alpha) \cdot \text{Norm}(S_{sparse}(q, d))$$

trong đó $\text{Norm}(\cdot)$ là hàm chuẩn hóa Min-Max để đưa điểm số về cùng thang $[0, 1]$.

Đề xuất của luận văn: Trọng số α có thể được điều chỉnh động dựa trên đặc trưng của truy vấn q:

$$\alpha(q) = \sigma(w^T \cdot \phi(q) + b)$$

trong đó $\phi(q)$ là vector đặc trưng (chứa mã code hay không, độ dài, etc.) và σ là hàm sigmoid.

2.2.5 Lập chỉ mục hiệu quả với HNSW

Vấn đề về Độ phức tạp

Với corpus N tài liệu, việc tìm kiếm chính xác (Exact kNN) đòi hỏi tính Cosine Similarity với tất cả N vectors, có độ phức tạp $O(N \cdot d)$. Khi N lên tới hàng triệu, tốc độ truy vấn trở nên không khả thi.

Thuật toán HNSW (Hierarchical Navigable Small World)

HNSW [Malkov & Yashunin, 2018] xây dựng cấu trúc đồ thị phân tầng cho phép Approximate Nearest Neighbor (ANN) search với độ phức tạp $O(\log N)$.

Cấu trúc: - **Lớp trên (Top layers):** Chứa ít node, các cạnh dài (long-range links) cho phép nhảy nhanh qua không gian vector - **Lớp dưới (Bottom layers):** Chứa nhiều node hơn, các cạnh ngắn (short-range links) cho độ chính xác cục bộ

Thuật toán tìm kiếm: 1. Bắt đầu từ entry point ở lớp cao nhất 2. Greedy search: Di chuyển đến neighbor gần query nhất 3. Khi không còn neighbor tốt hơn, xuống lớp thấp hơn 4. Lặp lại đến lớp 0, trả về k neighbors gần nhất

Định lý 2.2 (Độ phức tạp HNSW): *Với cấu hình phù hợp, HNSW đảm bảo độ phức tạp trung bình $O(\log N)$ cho mỗi truy vấn, với recall $>$ 95

2.2.6 Cross-Encoder Reranking

Motivation

Bi-Encoder (Dense Retrieval) tính toán embedding độc lập cho query và document, nhanh nhưng thiếu tương tác sâu giữa hai bên. Cross-Encoder khắc phục điều này bằng cách đưa cả cặp (q, d) vào cùng một Transformer.

Kiến trúc và Công thức

$$\text{Score}_{CE}(q, d) = \text{MLP}(\text{CLS}(\text{BERT}([q; \text{SEP}; d])))$$

trong đó $[q; \text{SEP}; d]$ là chuỗi nối của query và document, và CLS là hidden state của token [CLS].

Trade-off: - **Ưu điểm:** Độ chính xác cao hơn nhờ self-attention xem xét tương tác giữa mọi cặp token - **Nhược điểm:** Độ phức tạp $O(N \cdot (|q| + |d|)^2)$, không thể dùng cho toàn bộ corpus

Quy trình thực tế: 1. Bi-Encoder retrieve Top-50 ứng viên (nhanh) 2. Cross-Encoder rerank 50 ứng viên này 3. Chọn Top-5 để đưa vào LLM

2.2.7 2.2.7. Thách thức với Dữ liệu Đa phương thức

Vấn đề của PDF phức tạp

Tài liệu thực tế (báo cáo tài chính, danh mục y tế) thường chứa: - **Văn bản không tuyến tính:** Chia cột, header/footer gây nhiễu - **Bảng biểu:** Cấu trúc 2D, mất ngữ nghĩa khi flatten thành text 1D - **Hình ảnh/Biểu đồ:** Không thể đọc bằng text embedding

Phương pháp xử lý

1. **Document Layout Analysis (DLA):** - Sử dụng mô hình Vision (YOLOX, Detectron2) để phát hiện bounding box của các vùng Table, Image, Text

2. **Table Preservation:** - Chuyển đổi bảng sang Markdown/HTML để giữ cấu trúc hàng-cột - LLMs được train trên HTML/Markdown nên có thể "hiểu"cấu trúc 2D

3. **Multimodal Embeddings (CLIP/SigLIP):** - Sử dụng mô hình Vision-Language để tạo embedding chung cho text và image - Cho phép Text-to-Image retrieval

—

2.3 2.3. Tối ưu hóa và Thích ứng Mô hình (Model Optimization and Adaptation)

Trong khi RAG giải quyết vấn đề Knowledge Gap, nó không thay đổi **hành vi nội tại** của mô hình. Phần này trình bày các kỹ thuật để thích ứng LLMs cho các tác vụ yêu cầu văn phong hoặc quy tắc suy luận đặc thù.

2.3.1 2.3.1. Supervised Fine-Tuning (SFT)

Định nghĩa hình thức

Cho tập dữ liệu gán nhãn $\mathcal{D} = \{(x_i, y_i)\}_{i=1}^{N}$, trong đó x là prompt (instruction) và y là response mong muốn. Mục tiêu của SFT là tối ưu hóa:

$$\theta^* = \arg\min_{\theta} \mathcal{L}_{SFT}(\theta) = \arg\min_{\theta} \left[-\sum_{i=1}^{N} \sum_{t=1}^{|y_i|} \log P_\theta(y_i^{(t)} | x_i, y_i^{(<t)}) \right]$$

Đây là **Cross-Entropy Loss** giữa phân phối dự đoán và token thực tế.

Vấn đề của Full Fine-Tuning

Cập nhật toàn bộ tham số θ của LLM đối mặt với: 1. **Chi phí tính toán:** Mô hình 7B tham số cần 28GB VRAM chỉ để lưu gradient (với mixed precision) 2. **Catastrophic Forgetting:** Mô hình có thể "quên"kiến thức tổng quát khi được fine-tune sâu trên tác vụ hẹp 3. **Storage:** Mỗi tác vụ cần lưu một bản copy đầy đủ của mô hình

2.3.2 2.3.2. Low-Rank Adaptation (LoRA)

Động lực và Giả thuyết

Aghajanyan et al. (2021) chứng minh rằng các mô hình ngôn ngữ lớn có **intrinsic dimensionality** thấp, nghĩa là sự thay đổi trọng số cần thiết cho một tác vụ mới nằm trong một không gian con (subspace) chiều thấp.

Định nghĩa toán học

Cho $\mathbf{W}_0 \in \mathbb{R}^{d \times k}$ là ma trận trọng số pre-trained. Thay vì cập nhật trực tiếp $\mathbf{W}_0$, LoRA [Hu et al., 2021] phân rã sự thay đổi $\Delta\mathbf{W}$ thành tích của hai ma trận rank thấp:

$$\Delta \mathbf{W} = \mathbf{BA}$$

trong đó $\mathbf{B} \in \mathbb{R}^{d \times r}$, $\mathbf{A} \in \mathbb{R}^{r \times k}$, và $r \ll \min(d, k)$.

Forward Pass

$$\mathbf{h} = \mathbf{W}_0\mathbf{x} + \Delta\mathbf{W}\mathbf{x} = \mathbf{W}_0\mathbf{x} + \mathbf{BAx}$$

Trong quá trình training: - $\mathbf{W}_0$ được **đóng băng** (frozen) - Chỉ **A** và **B** được cập nhật

Phân tích Số lượng Tham số

Phương pháp	Số tham số trainable	Ví dụ (Llama-7B, d=4096)
Full Fine-Tuning	$d \times k$	~7 tỷ
LoRA (r=16)	$r \times (d + k)$	~8 triệu
Giảm	$\frac{r \times (d+k)}{d \times k}$	~1000x

Bảng 2.2: So sánh số lượng tham số trainable giữa Full Fine-Tuning và LoRA

Khởi tạo và Scaling

- **A**: Khởi tạo Gaussian với phương sai nhỏ - **B**: Khởi tạo bằng 0 để $\Delta\mathbf{W} = 0$ ban đầu - Scaling factor: α/r để kiểm soát magnitude của adaptation

$$\mathbf{h} = \mathbf{W}_0\mathbf{x} + \frac{\alpha}{r}\mathbf{BAx}$$

2.3.3 2.3.3. Quantization và QLoRA

Motivation

Để triển khai trên GPU có bộ nhớ hạn chế (như T4 16GB), cần giảm kích thước mô hình thông qua **Quantization** — giảm độ chính xác của trọng số từ 16-bit/32-bit xuống 8-bit hoặc 4-bit.

Định nghĩa Toán học của Quantization

Quá trình lượng tử hóa ánh xạ giá trị thực x (FP16/FP32) sang giá trị nguyên x_q (INT8/INT4):

$$x_q = \text{round}\left(\frac{x - Z}{S}\right), \quad \hat{x} = S \cdot x_q + Z$$

trong đó: - S (Scale): Hệ số tỷ lệ - Z (Zero-point): Điểm không - $\hat{x}$: Giá trị xấp xỉ sau de-quantization

NormalFloat 4-bit (NF4)

Dettmers et al. (2023) đề xuất NF4, một phương pháp quantization tối ưu cho trọng số tuân theo phân phối chuẩn:

$$\text{NF4} : q_i = \frac{1}{2}\left(\Phi^{-1}\left(\frac{i}{16}\right) + \Phi^{-1}\left(\frac{i+1}{16}\right)\right), \quad i \in \{0, 1, \ldots, 15\}$$

trong đó Φ^{-1} là hàm ngược của CDF chuẩn. Các quantization levels được chọn để minimizing expected quantization error.

QLoRA: Kết hợp Quantization và LoRA

QLoRA [Dettmers et al., 2023] kết hợp: 1. **Mô hình gốc $\mathbf{W}_0$:** Lưu ở dạng 4-bit (NF4) 2. **LoRA adapters $\mathbf{A}, \mathbf{B}$:** Lưu ở dạng 16-bit (BFloat16)

Lợi ích: - Giảm VRAM từ 28GB (FP16) xuống 6GB (4-bit) cho mô hình 7B - Vẫn giữ được độ chính xác gradient nhờ LoRA adapters ở FP16

2.3.4 2.3.4. Knowledge Distillation và Synthetic Data Generation

Motivation

Trong các miền đặc thù (niche domains), dữ liệu huấn luyện chất lượng cao thường khan hiếm. **Knowledge Distillation** cho phép chuyển giao khả năng từ mô hình lớn (Teacher) sang mô hình nhỏ (Student).

Định nghĩa Toán học

Cho Teacher model T với phân phối đầu ra $P_T(y|x)$ và Student model S với $P_S(y|x)$. Mục tiêu là minimize **Kullback-Leibler Divergence**:

$$\mathcal{L}_{KD} = \mathbb{E}_{x \sim \mathcal{D}}\left[D_{KL}(P_T(\cdot|x) \| P_S(\cdot|x))\right]$$

$$D_{KL}(P_T \| P_S) = \sum_y P_T(y|x) \log \frac{P_T(y|x)}{P_S(y|x)}$$

Quy trình Synthetic Data Generation

1. **Teacher Model** (GPT-4, Gemini): Sinh cặp $(x, y_{teacher})$ dựa trên prompt template 2. **Diversity Injection:** Thêm variation vào input (user personality, edge cases) 3. **Filtering:** Loại bỏ các mẫu có chất lượng thấp 4. **Student Training:** Fine-tune SLM trên dataset này

—

2.4 2.4. Phương pháp Đánh giá (Evaluation Methodology)

2.4.1 2.4.1. RAGAS Framework

RAGAS [Es et al., 2023] là framework đánh giá RAG không cần ground truth, sử dụng LLM làm judge.

Faithfulness

Đo lường mức độ câu trả lời được hỗ trợ bởi context:

$$\text{Faithfulness} = \frac{|\{s \in S : s \text{ can be inferred from } C\}|}{|S|}$$

trong đó S là tập các claims trong câu trả lời, C là context.

Answer Relevance

Đo lường mức độ câu trả lời giải quyết câu hỏi:

$$\text{Relevance} = \frac{1}{n}\sum_{i=1}^{n} \text{sim}_{cos}(\mathbf{v}_q, \mathbf{v}_{q_i})$$

trong đó q_i là câu hỏi được tái tạo từ câu trả lời.

Context Precision

Đo lường tỷ lệ context có ích trong top-k:

$$\text{Context Precision@K} = \frac{1}{K}\sum_{k=1}^{K} \frac{\sum_{i=1}^{k} v_i}{k} \cdot v_k$$

trong đó $v_k = 1$ nếu chunk tại vị trí k là relevant.

2.4.2 2.4.2. Perplexity

Đánh giá độ "bối rối"của language model:

$$\text{PPL} = \exp\left(-\frac{1}{T}\sum_{t=1}^{T} \log P_\theta(x_t|x_{<t})\right)$$

PPL thấp hơn = model tự tin hơn trong việc dự đoán.

2.4.3 2.4.3. LLM-as-a-Judge

Sử dụng LLM mạnh (GPT-4) để đánh giá output của LLM yếu hơn trên các tiêu chí định tính:
- Coherence (Mạch lạc) - Helpfulness (Hữu ích) - Harmlessness (Vô hại)

—

2.5 2.5. Tổng kết Chương

Chương này đã cung cấp nền tảng lý thuyết cho hai phương pháp thích ứng LLM:

Phương pháp	Giải quyết	Cơ chế	Ưu điểm	Hạn chế
RAG	Knowledge Gap	External retrieval	Dễ cập nhật, ít hallucination	Chi phí retriev
Fine-tuning (LoRA)	Behavior Gap	Weight adaptation	Thay đổi hành vi sâu	Cần dữ liệu, có

Bảng 2.3: So sánh hai phương pháp thích ứng LLM

Các chương tiếp theo sẽ trình bày chi tiết việc triển khai và đánh giá hai phương pháp này trên các use-case thực tế.

Chương 3

Multimodal RAG for Complex Document Understanding

3.1 Bài toán Thực tế: Hai Use-case từ Doanh nghiệp

Trong quá trình hoạt động tại Torus AI, chúng tôi tiếp nhận hai yêu cầu từ các đối tác với những thách thức tương đồng nhưng ở hai lĩnh vực hoàn toàn khác nhau: tài chính và y tế. Điều thú vị là, dù bối cảnh nghiệp vụ khác biệt, cả hai đều đối mặt với cùng một vấn đề cốt lõi: làm sao để khai thác hiệu quả tri thức từ những tài liệu phi cấu trúc phức tạp chứa nhiều bảng biểu, hình ảnh và dữ liệu đa phương thức.

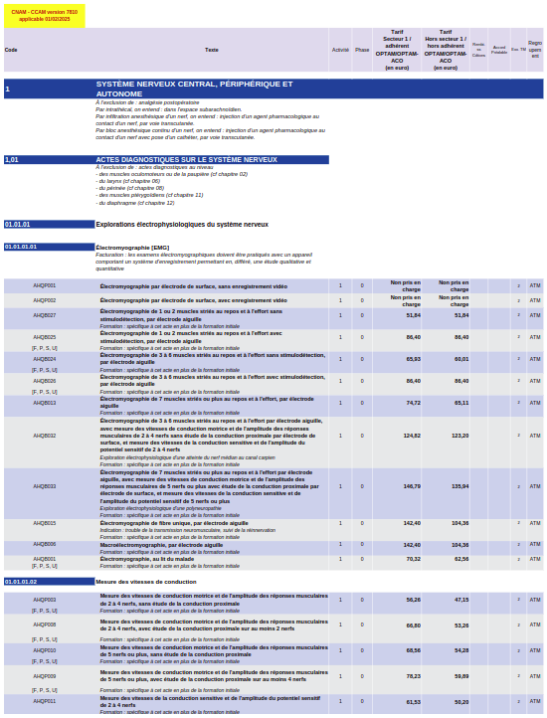
3.1.1 Use-case GPM: Hỏi đáp Tài liệu Tài chính

GPM (Gestion Patrimoine Mutualiste) là một quỹ bảo hiểm tương hỗ thuộc tập đoàn AGMF (Association Générale des Médecins de France). Yêu cầu của họ nghe có vẻ đơn giản: xây dựng hệ thống hỏi đáp nhanh để nhân viên có thể tra cứu thông tin từ các báo cáo tài chính hàng năm.

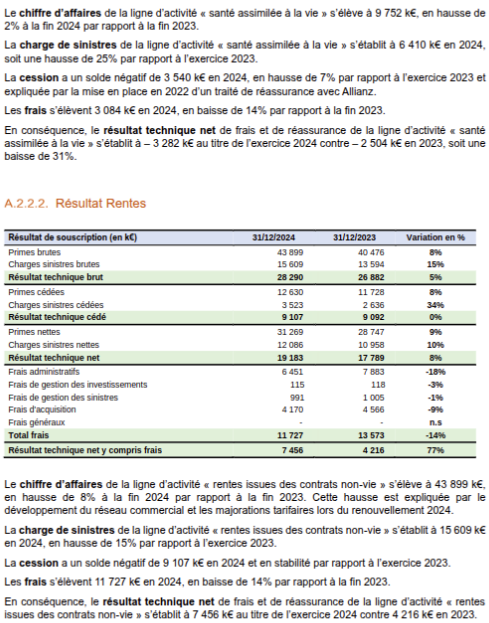
Tuy nhiên, "đơn giản" chỉ là bề ngoài. Các tài liệu của GPM bao gồm báo cáo SFCR (Solvency and Financial Condition Report) và báo cáo kế toán thường niên — những file PDF dài 50 đến 200 trang. Điểm đặc biệt là phần lớn thông tin quan trọng không nằm trong văn bản mà nằm trong các bảng số liệu tài chính phức tạp: bảng cân đối kế toán với hàng chục cột và hàng, biểu đồ xu hướng doanh thu, và các ma trận phân tích rủi ro. Một câu hỏi điển hình như "Doanh thu năm 2024 là bao nhiêu?" đòi hỏi hệ thống phải hiểu được cấu trúc 2D của bảng biểu, không chỉ đọc text.

3.1.2 Use-case Dr. Besnier: Tra cứu Mã Y tế CCAM

Dr. Besnier đến từ lĩnh vực y khoa với nhu cầu hoàn toàn khác: tra cứu mã hoàn tiền trong danh mục CCAM (Classification Commune des Actes Médicaux). CCAM là hệ thống mã hóa chuẩn của Pháp cho các hành vi y tế, được sử dụng để xác định mức bảo hiểm chi trả. Hình 3.1a minh họa cấu trúc của danh mục này.



(a) Ví dụ về mã CCAM



Le **chiffre d'affaires** de la ligne d'activité « santé assimilée à la vie » s'élève à 9 752 k€, en hausse de 2% à la fin 2024 par rapport à la fin 2023.

La **charge de sinistres** de la ligne d'activité « santé assimilée à la vie » s'établit à 6 410 k€ en 2024, soit une hausse de 25% par rapport à l'exercice 2023.

La **cession** a un solde négatif de 3 540 k€ en 2024, en hausse de 7% par rapport à l'exercice 2023 et expliquée par la mise en place en 2022 d'un traité de réassurance avec Allianz.

Les **frais** s'élèvent 3 084 k€ en 2024, en baisse de 14% par rapport à la fin 2023.

En conséquence, le **résultat technique net** de frais et de réassurance de la ligne d'activité « santé assimilée à la vie » s'établit à – 3 282 k€ au titre de l'exercice 2024 contre – 2 504 k€ en 2023, soit une baisse de 31%.

A.2.2.2. Résultat Rentes

Résultat de souscription (en k€)	31/12/2024	31/12/2023	Variation en %
Primes brutes	43 899	40 476	8%
Charges sinistres brutes	15 609	13 594	15%
Résultat technique brut	**28 290**	**26 882**	**5%**
Primes cédées	12 630	11 728	8%
Charges sinistres cédées	3 523	2 636	34%
Résultat technique cédé	**9 107**	**9 092**	**0%**
Primes nettes	31 269	28 747	9%
Charges sinistres nettes	12 086	10 958	10%
Résultat technique net	**19 183**	**17 789**	**8%**
Frais administratifs	6 451	7 883	-18%
Frais de gestion des investissements	115	118	-3%
Frais de gestion des sinistres	991	1 005	-1%
Frais d'acquisition	4 170	4 566	-9%
Frais généraux	-	-	n.s
Total frais	**11 727**	**13 573**	**-14%**
Résultat technique net y compris frais	**7 456**	**4 216**	**77%**

Le **chiffre d'affaires** de la ligne d'activité « rentes issues des contrats non-vie » s'élève à 43 899 k€, en hausse de 8% à la fin 2024 par rapport à la fin 2023. Cette hausse est expliquée par le développement du réseau commercial et les majorations tarifaires lors du renouvellement 2024.

La **charge de sinistres** de la ligne d'activité « rentes issues des contrats non-vie » s'établit à 15 609 k€ en 2024, en hausse de 15% par rapport à l'exercice 2023.

La **cession** a un solde négatif de 9 107 k€ en 2024 et en stabilité par rapport à l'exercice 2023.

Les **frais** s'élèvent 11 727 k€ en 2024, en baisse de 14% par rapport à la fin 2023.

En conséquence, le **résultat technique net** de frais et de réassurance de la ligne d'activité « rentes issues des contrats non-vie » s'établit à 7 456 k€ au titre de l'exercice 2024 contre 4 216 k€ en 2023.

(b) Ví dụ về báo cáo GPM

Hình 3.1: Minh họa tài liệu CCAM và GPM

Thách thức ở đây là dữ liệu CCAM chứa hàng nghìn mã y tế được tổ chức trong một cấu trúc bảng cực kỳ phức tạp. Mỗi mã có nhiều thuộc tính: code (ví dụ "HAFA008"), mô tả hành vi, giá sector 1, giá ngoài sector, và đặc biệt là các điều kiện áp dụng phức tạp phụ thuộc vào ngữ cảnh bệnh nhân.

Một truy vấn thực tế có thể là: "Patiente 69 ans, exérèse de carcinome basocellulaire de la lèvre, suture par lambeau à la volée" (Bệnh nhân 69 tuổi, cắt bỏ ung thư biểu mô đáy môi, khâu bằng vạt da). Hệ thống cần trả về các mã CCAM phù hợp như HAFA008 hay QAMA002 — một bài toán đòi hỏi không chỉ hiểu ngữ nghĩa mà còn cần exact matching với các mã code cụ thể.

3.1.3 Điểm Chung và Lý do Cần RAG Pipeline In-house

Hai use-case này, dù khác biệt về domain, chia sẻ những đặc điểm quan trọng:

1. Tài liệu phi cấu trúc phức tạp: Không phải văn bản thuần túy mà là sự kết hợp của text, bảng biểu phức tạp, và hình ảnh minh họa.
2. Yêu cầu bảo mật dữ liệu nghiêm ngặt: Cả tài liệu tài chính lẫn hồ sơ y tế đều là dữ liệu nhạy cảm. Theo quy định GDPR và các tiêu chuẩn ngành, việc sử dụng các dịch vụ cloud như ChatGPT hay Claude API trực tiếp với dữ liệu raw là không được phép. Khách hàng cần một giải pháp chạy hoàn toàn on-premise hoặc trong private cloud của họ.
3. Độ chính xác cao: Trong tài chính và y tế, sai số là không chấp nhận được. Câu trả lời sai về doanh thu có thể dẫn đến quyết định đầu tư sai lầm; mã y tế không chính xác ảnh hưởng trực tiếp đến việc chữa trị và hoàn tiền bảo hiểm cho bệnh nhân.

Với những ràng buộc này, việc sử dụng LLM đơn thuần (prompt engineering trực tiếp) không khả thi vì hai lý do: (1) context window của LLM không đủ chứa toàn bộ tài liệu 200 trang, và (2) LLM không có khả năng cập nhật kiến thức theo thời gian thực khi tài liệu thay đổi.

RAG (Retrieval-Augmented Generation) là giải pháp phù hợp: retrieve thông tin liên quan từ knowledge base, sau đó augment vào prompt cho LLM generate câu trả lời. Điều này cho phép LLM "học" từ tài liệu mới mà không cần fine-tuning.

3.1.4 Mục tiêu: Một Pipeline Chung cho Mọi Tài liệu

Thay vì phát triển hai pipeline riêng biệt cho từng use-case, chúng tôi đặt mục tiêu tham vọng hơn: xây dựng một pipeline RAG chung có khả năng xử lý hiệu quả mọi loại tài liệu đa phương thức. Pipeline này cần đạt được:

- Độ chính xác cao với cả tài liệu tài chính lẫn y tế.
- Khả năng xử lý bảng biểu phức tạp với cấu trúc 2D.
- Hỗ trợ retrieval cả semantic (ngữ nghĩa) lẫn exact match (keyword/code cụ thể).
- Chạy hoàn toàn local mà không phụ thuộc vào dịch vụ cloud bên ngoài.

Trong các phần tiếp theo, chúng tôi sẽ phân tích tại sao RAG đơn giản không đáp ứng được yêu cầu, sau đó trình bày kiến trúc pipeline nâng cao của chúng tôi với từng thành phần được thiết kế để giải quyết một vấn đề cụ thể.

3.2 Tại sao Simple RAG Thất bại? Phân tích Thực nghiệm

Trước khi đề xuất giải pháp phức tạp, chúng tôi thực hiện một bước quan trọng: thử nghiệm RAG đơn giản và đo lường định lượng những hạn chế của nó. Điều này không chỉ justify cho sự cần thiết của các kỹ thuật nâng cao mà còn giúp xác định chính xác những "bottleneck" cần giải quyết.

3.2.1 Cấu hình Simple RAG Baseline

Pipeline Simple RAG được triển khai với các thành phần tiêu chuẩn nhất — cũng là những gì bạn sẽ tìm thấy trong đa số tutorial về RAG:

- Document Parsing: PyPDF để trích xuất text từ PDF.
- Chunking: Fixed-size chunking với 512 tokens/chunk và 50 tokens overlap.
- Embedding Model: sentence-transformers/all-MiniLM-L6-v2 — model phổ biến nhất với 22M parameters.
- Vector Store: ChromaDB với Dense Search (cosine similarity).
- LLM: Gemma-3-12b-it để sinh câu trả lời.

3.2.2 Vấn đề 1: Mất Cấu trúc Bảng biểu

Đây là vấn đề nghiêm trọng nhất. Khi PyPDF trích xuất text từ PDF, nó "flatten" cấu trúc 2D của bảng thành chuỗi 1D. Hãy xem một ví dụ cụ thể từ tài liệu GPM:

Bảng 3.1: So sánh cấu trúc: Bảng gốc vs Text trích xuất từ PyPDF

a) Bảng gốc trong PDF:

Sous modules (en k€)	SCR 2024	SCR 2023	Evol.
Type 1	2 692	2 487	8 %
Type 2	28 377	36 221	-22 %
Diversification	-621	-586	6 %
Risque de défaut	30 448	38 121	-20 %

b) Sau khi PyPDF trích xuất:

Sous modules (en k€) SCR 2024 SCR 2023 Evolution
Type 1 2 692 2 487 8 %
Type 2 28 377 36 221 -22 %
Effet de diversification -621 -586 6 %
Risque de défaut 30 448 38 121 -20 %

Với chuỗi text này, việc trả lời câu hỏi tự động như "SCR 2024 của Type 2 là bao nhiêu?" trở nên rất khó khăn đối với mô hình ngôn ngữ (LLM). Hệ thống khó có thể phân biệt liệu số "28 377" thuộc về cột "SCR 2024" hay là hai số riêng biệt "28" và "377", và nó tương ứng với hàng "Type 2" hay hàng nào khác do mất đi sự gióng hàng (alignment).

3.2.3 Vấn đề 2: Semantic Gap với Embedding Model Tiếng Anh

Model all-MiniLM-L6-v2 được train chủ yếu trên dữ liệu tiếng Anh. Khi áp dụng vào tài liệu tiếng Pháp, hiệu suất suy giảm đáng kể.

Chúng tôi thực hiện một thử nghiệm đơn giản: với cùng một query, đo cosine similarity giữa query và ground-truth passage.

Bảng 3.2: So sánh Embedding Models trên Query Tiếng Pháp

Model	MTEB French Score	Avg. Similarity
all-MiniLM-L6-v2	0.45	0.62
multilingual-e5-large	0.65	0.78
multilingual-e5-large-instruct	0.68	0.81

Sự khác biệt 0.19 trong similarity có vẻ nhỏ, nhưng với top-k retrieval, nó đồng nghĩa với việc passage đúng có thể rơi từ vị trí 3 xuống vị trí 15 — nằm ngoài context window của LLM.

3.2.4 Vấn đề 3: Thất bại với Exact Code Matching

Trong use-case CCAM, người dùng thường hỏi trực tiếp về một mã cụ thể: "Mã HAFA006 dùng cho trường hợp nào?".

Dense Search hoạt động dựa trên semantic similarity — nó tìm các passage "có ý nghĩa tương tự". Nhưng "HAFA006" không có semantic meaning; đây là một code cần exact match.

Kết quả thực tế từ Simple RAG cho query này:

```
================================================================================
📊 ANALYSIS OF RESULTS
================================================================================

1️⃣ HAFA006 POSITION:
   ✅ Found HAFA006 at rank #3
   ⚠️  HYPOTHESIS PARTIALLY CONFIRMED: Ranked #3 (not #1, but still high)

2️⃣ SIMILAR CODES RANKED HIGHER:
   Found 5 chunks with similar HAFA00X codes:
   - Rank #1: ['HAFA004', 'HAFA002'] (score: 0.2544)
   - Rank #2: ['HAFA005', 'HAFA003'] (score: 0.2315)
   - Rank #3: ['HAFA007'] (score: 0.2199)
   - Rank #4: ['HAFA001'] (score: 0.2107)
   - Rank #5: ['HAFA009'] (score: 0.2107)
```

Hình 3.2: Kết quả Dense Search cho query "HAFA508"

Retriever trả về passage chứa code ở vị trí thứ 3, và các vị trí khác cao hơn được match với những code khác tương tự vì Dense Search không hiểu rằng người dùng cần chính xác "HAFA006", không phải một code tương tự.

3.2.5 Vấn đề 4: Lost in the Middle

Hiện tượng "Lost in the Middle" được Liu et al. (2023) phát hiện: khi context chứa nhiều passages (ví dụ 10-20), LLM có xu hướng tập trung vào đầu và cuối, bỏ qua thông tin ở giữa.

Trong thử nghiệm của chúng tôi với top-20 retrieval, khi ground-truth passage nằm ở vị trí 7-12, accuracy của LLM giảm 23% so với khi passage đó nằm ở vị trí 1-3.

3.2.6 Kết quả Định lượng: Simple RAG vs. Ground Truth

Chúng tôi đánh giá Simple RAG trên 25 câu hỏi từ tập test GPM (sẽ được đề cập trong phần sau) sử dụng LLM-as-a-Judge với hai metrics:

Bảng 3.3: Kết quả Simple RAG trên GPM Test Set

Metric	Score	Interpretation
Precision	0.66	66% câu trả lời chính xác với context
Relevancy	0.73	73% câu trả lời liên quan đến câu hỏi

Precision 0.66 có nghĩa là cứ 10 câu trả lời thì có gần 3.5 câu chứa thông tin không chính xác — một tỷ lệ lỗi không thể chấp nhận được trong môi trường doanh nghiệp.

3.3 Kiến trúc Advanced RAG Pipeline

Dựa trên phân tích ở phần trước, chúng tôi thiết kế một pipeline với mỗi thành phần giải quyết một vấn đề cụ thể đã xác định. Hình ?? minh họa kiến trúc tổng quan.

Pipeline được tổ chức thành hai pha: Indexing Phase (offline, chạy một lần khi có tài liệu mới) và Query Phase (online, mỗi khi có câu hỏi từ người dùng).

3.3.1 Indexing Phase: Từ PDF đến Vector Store

Document Parsing với UnstructuredIO

Vấn đề giải quyết: Mất cấu trúc bảng biểu khi parsing.

Giải pháp: UnstructuredIO với Document Layout Analysis.

Thay vì PyPDF chỉ trích xuất raw text, UnstructuredIO sử dụng computer vision để detect các regions trong PDF: Text blocks, Table regions, và Image regions. Đặc biệt, với tables, UnstructuredIO bảo toàn cấu trúc 2D bằng cách chuyển đổi thành HTML.

```
# Cấu hình UnstructuredIO cho hi-res parsing
chunks = partition_pdf(
    filename=file_path,
    strategy="hi_res",              # Document Layout Analysis
    infer_table_structure=True,     # Detect và parse bảng
    extract_image_block_types=["Image", "Table"],
    chunking_strategy="by_title",   # Chunk theo section
    max_characters=10000
)
```

Kết quả: Bảng tài chính được bảo toàn dưới dạng HTML:

```
<table>
  <tr><th>Indicateur</th><th>2022</th><th>2023</th><th>2024</th></tr>
  <tr><td>Chiffre d'affaires</td><td>187,234</td><td>198,456</td>
     <td>210,294</td></tr>
</table>
```

LLMs được train trên lượng lớn HTML/Markdown, do đó có khả năng "hiểu" cấu trúc 2D này và mapping giữa header (2024) với giá trị (210,294).

Summarization: Semantic Bridge

Vấn đề giải quyết: Semantic gap giữa query và raw content.

Giải pháp: Sử dụng LLM tạo summary cho mỗi chunk.

Raw content (đặc biệt là bảng số liệu) thường không chứa đủ context để embedding model hiểu. Ví dụ, một bảng chỉ có "2024: 210,294" không có từ "doanh thu" hay "chiffre d'affaires" xuất hiện.

Chúng tôi sử dụng Gemma-3-12b-it để tạo summary mô tả nội dung:

```
Summary: "Ce tableau présente l'évolution du chiffre d'affaires
de l'entreprise de 2022 à 2024. Le chiffre d'affaires 2024
s'élève à 210 294 k€, en hausse de 6% par rapport à 2023."
```

Summary này trở thành "semantic bridge": khi người dùng hỏi về "doanh thu 2024", embedding của câu hỏi sẽ có similarity cao với summary (chứa "chiffre d'affaires" và "2024").

Parent-Child Indexing Strategy

Vấn đề giải quyết: Trade-off giữa retrieval precision và LLM context.

Giải pháp: Lưu summary vào VectorStore, raw content vào DocStore.

Summary ngắn gọn giúp retrieval chính xác hơn (ít noise), nhưng LLM cần raw content đầy đủ (bao gồm HTML table) để sinh câu trả lời chi tiết.

Quy trình:

1. Index summary embeddings vào ChromaDB/Qdrant với doc_id liên kết.
2. Lưu raw content (text, HTML tables, base64 images) vào InMemoryStore.
3. Khi retrieval, tìm summaries tương tự, sau đó fetch raw content theo doc_id.

Multilingual Embedding Model

Vấn đề giải quyết: Embedding model tiếng Anh không hiệu quả với tài liệu tiếng Pháp.

Giải pháp: intfloat/multilingual-e5-large-instruct.

Model này có 560M parameters, được train trên 100+ ngôn ngữ với instruction-following capability. Đặc biệt, nó cho phép prefix query với task description:

```
query = get_detailed_instruct(
    "Given a query, retrieve relevant passages that answer the query.",
    "Quel est le chiffre d'affaires 2024?"
)
```

Instruction này giúp model hiểu mục tiêu không chỉ là tìm passages tương tự mà là tìm passages chứa câu trả lời.

3.3.2 Query Phase: Từ Câu hỏi đến Câu trả lời

Hybrid Search: Kết hợp Semantic và Keyword

Vấn đề giải quyết: Dense Search thất bại với exact code matching.

Giải pháp: Kết hợp Dense Search (semantic) và BM25 (keyword).

Dense Search tốt cho semantic queries như "doanh thu tăng bao nhiêu?" nhưng kém với "mã HAFA008". BM25 (sparse retrieval) tính relevance dựa trên term frequency — nếu query chứa "HAFA008", BM25 sẽ ưu tiên documents chứa chính xác chuỗi ký tự này.

Chúng tôi sử dụng Qdrant với Hybrid Search:

```
vector_store = QdrantVectorStore(
    client=client,
    collection_name="general_usecase",
    enable_hybrid=True,
    fastembed_sparse_model="Qdrant/bm25"
)
```

Kết quả từ Dense và BM25 được merge bằng Reciprocal Rank Fusion (RRF):

$$\text{RRF}(d) = \sum_{r \in R} \frac{1}{k + r(d)}$$

với R là tập các rankings, $r(d)$ là rank của document d trong ranking r, và $k = 60$ là constant để smoothing.

Cross-Encoder Reranking

Vấn đề giải quyết: Lost in the Middle — LLM bỏ qua thông tin giữa context.

Giải pháp: Rerank để đưa thông tin relevant nhất lên đầu.

Bi-Encoder (dùng trong Dense Search) tính embedding độc lập cho query và document, nhanh nhưng "shallow". Cross-Encoder đưa cặp (query, document) vào cùng một Transformer, cho phép attention giữa mọi token — đánh giá relevance chính xác hơn nhưng chậm hơn.

Quy trình two-stage:

1. Hybrid Search retrieve top-50 candidates (nhanh vì dùng HNSW index).
2. Cross-Encoder (cross-encoder/ms-marco-MiniLM-L-12-v2) score 50 candidates này.
3. Chọn top-5 có score cao nhất để đưa vào LLM.

Với top-5 reranked results, thông tin quan trọng nhất đảm bảo nằm ở đầu context — giảm thiểu "Lost in the Middle".

Multimodal Generation

Bước cuối cùng, LLM (Gemma-3-12b-it với multimodal capability) nhận context bao gồm:

- Text passages (từ raw content).
- HTML tables (bảo toàn cấu trúc 2D).
- Base64 images (biểu đồ, đồ thị).

Prompt được thiết kế để LLM trả lời dựa chỉ vào context được cung cấp (giảm hallucination) và giữ nguyên ngôn ngữ của tài liệu.

3.4 Đánh giá Thực nghiệm

3.4.1 Datasets và Metrics

GPM Test Set: 25 cặp (question, ground-truth answer) được human-annotated từ tài liệu AGMF.

CCAM Test Set: 10 consultation scenarios với ground-truth CCAM codes được verify bởi bác sĩ.

Metrics:

- Precision: Mức độ câu trả lời chính xác với context (LLM-as-a-Judge).
- Relevancy: Mức độ câu trả lời giải quyết câu hỏi.
- Faithfulness: Câu trả lời có được support bởi context không (RAGAS).
- Context Precision: Thông tin relevant có ở đầu context không.

3.4.2 Kết quả So sánh: Simple RAG vs. Advanced RAG

Bảng 3.4: Kết quả trên GPM Test Set (25 câu hỏi)

Pipeline	Precision	Relevancy	Faithfulness	Ctx. Precision
Simple RAG	0.728	0.828	0.75	0.68
Advanced RAG	0.976	0.972	0.95	0.92
Improvement	+34.1%	+17.4%	+26.7%	+35.3%

Bảng 3.5: Kết quả trên CCAM Test Set (10 scenarios)

Pipeline	Code Hit Rate@5	Code Hit Rate@20
Simple RAG (Dense only)	0.42	0.65
Advanced RAG (Hybrid + Rerank)	0.78	0.92
Improvement	+85.7%	+41.5%

Code Hit Rate@k đo tỷ lệ ground-truth code xuất hiện trong top-k retrieved passages. Hybrid Search với BM25 mang lại cải thiện đáng kể nhờ exact matching với mã CCAM.

3.5 Ablation Study: Đóng góp của Từng Thành phần

Để hiểu rõ contribution của từng thành phần, chúng tôi thực hiện ablation study: bắt đầu từ Simple RAG baseline, lần lượt thêm từng component và đo sự cải thiện.

Bảng 3.6: Ablation Study trên GPM Test Set

Configuration	Precision	Δ vs Baseline	Cumulative Δ
Baseline (Simple RAG)	0.728	—	—
+ UnstructuredIO Parsing	0.792	+8.8%	+8.8%
+ Summarization	0.836	+5.6%	+14.8%
+ Multilingual E5 Embedding	0.884	+5.7%	+21.4%
+ Hybrid Search	0.924	+4.5%	+26.9%
+ Reranking	0.976	+5.6%	+34.1%

3.5.1 Phân tích Chi tiết từng Component

UnstructuredIO Parsing (+8.8%): Contribution lớn nhất trong single component. Việc bảo toàn cấu trúc bảng HTML cho phép LLM mapping chính xác giữa headers và values.

Summarization (+5.6%): Tạo semantic bridge giúp embedding model hiểu context của bảng số liệu. Đặc biệt hiệu quả với các câu hỏi về xu hướng ("tăng bao nhiêu so với năm trước?").

Multilingual E5 Embedding (+5.7%): Cải thiện retrieval accuracy cho tài liệu tiếng Pháp. Instruction prefix giúp model hiểu task là tìm passages chứa câu trả lời.

Hybrid Search (+4.5%): Quan trọng hơn cho CCAM use-case (cải thiện 20% Code Hit Rate). Với GPM, contribution thấp hơn vì queries chủ yếu là semantic.

Reranking (+5.6%): Giải quyết Lost in the Middle. Đặc biệt hiệu quả khi retrieval trả về nhiều passages tương tự nhau.

3.5.2 Ablation trên CCAM: Tầm quan trọng của Hybrid Search

Bảng 3.7: Ablation Study trên CCAM Test Set

Configuration	Code Hit Rate@5	Δ vs Baseline
Dense Search only	0.42	—
+ BM25 (Hybrid)	0.68	+61.9%
+ Reranking	0.78	+85.7%

Hybrid Search với BM25 là game-changer cho CCAM: cải thiện 61.9% so với Dense-only. Điều này xác nhận giả thuyết rằng Dense Search không đủ cho exact code matching.

3.6 Thảo luận và Hướng Phát triển

3.6.1 Tổng kết Đóng góp

Pipeline Advanced RAG của chúng tôi đạt được mục tiêu đề ra:

1. Độ chính xác cao: Precision 0.976 trên GPM, Code Hit Rate 0.78 trên CCAM.
2. Pipeline chung: Cùng một architecture hoạt động tốt trên cả tài liệu tài chính và y tế.
3. Chạy local: Toàn bộ pipeline có thể deploy on-premise, đáp ứng yêu cầu bảo mật.

3.6.2 Hạn chế

- Dataset size nhỏ: 25 và 10 test cases chưa đủ để kết luận chắc chắn về generalization.
- Latency chưa được tối ưu: Reranking và Summarization tăng đáng kể latency.
- Single-hop focus: Chưa evaluate kỹ với multi-hop reasoning queries.

3.6.3 Hướng Phát triển

- Adaptive Hybrid Search: Classifier tự động điều chỉnh trọng số Dense/Sparse dựa trên query type.
- Query Routing: Route semantic queries đến Dense-heavy path, code queries đến Sparse-heavy path.
- Caching và Optimization: Cache summaries và embeddings để giảm latency.

3.7 Kết luận Chương

Chương này đã trình bày quá trình thiết kế, triển khai và đánh giá một pipeline RAG nâng cao cho tài liệu đa phương thức. Xuất phát từ hai use-cases thực tế (GPM tài chính và CCAM y tế), chúng tôi xác định những hạn chế cụ thể của Simple RAG và đề xuất giải pháp với mỗi thành phần giải quyết một vấn đề cụ thể.

Kết quả thực nghiệm xác nhận hiệu quả của phương pháp: Precision tăng từ 0.728 lên 0.976 (+34%), và Code Hit Rate tăng từ 0.42 lên 0.78 (+86%). Ablation study cho thấy mỗi component

đều đóng góp có ý nghĩa, với UnstructuredIO parsing và Hybrid Search là hai yếu tố quan trọng nhất.

Pipeline này đã được triển khai và đang được sử dụng bởi các đối tác của Torus AI, chứng minh khả năng ứng dụng thực tế của các kỹ thuật RAG nâng cao trong môi trường doanh nghiệp.

Chương 4

Fine-tuning Mô hình Ngôn ngữ Nhỏ cho Tác vụ Đặc thù: Trường hợp Tarot Reader

Chương trước đã trình bày cách giải quyết Knowledge Gap — khoảng cách giữa tri thức của mô hình và yêu cầu thực tế — thông qua Retrieval-Augmented Generation. Tuy nhiên, trong nhiều ứng dụng thực tiễn, thách thức không chỉ nằm ở việc mô hình biết gì, mà còn ở cách mô hình diễn đạt và ứng xử. Chương này chuyển sang một vấn đề khác biệt cơ bản: Behavior Gap — làm thế nào để mô hình ngôn ngữ duy trì phong cách giao tiếp nhất quán, tuân thủ quy trình cố định, và thể hiện "cá tính" phù hợp với vai trò được giao.

Chúng tôi trình bày một case study hoàn chỉnh: xây dựng ứng dụng Tarot Reader — một chatbot tư vấn giải trí đòi hỏi sự kết hợp tinh tế giữa tri thức chuyên môn về 78 lá bài, kỹ năng tư vấn tâm lý, và phong cách giao tiếp đặc trưng. Thông qua case study này, chúng tôi khám phá các câu hỏi: Liệu một mô hình nhỏ (< 3B tham số) có thể đạt được sự nhất quán hành vi tương đương hoặc tốt hơn các LLM lớn với prompt engineering? Chiến lược tạo dữ liệu nào hiệu quả nhất? Và LoRA có đủ hay cần full fine-tuning?

4.1 Phân tích Bài toán

4.1.1 Giới hạn của Prompt Engineering cho Behavioral Tasks

Một hướng tiếp cận tự nhiên là sử dụng LLM mạnh (GPT-4, Claude) với system prompt được thiết kế cẩn thận. Tuy nhiên, qua thử nghiệm thực tế, chúng tôi nhận thấy ba hạn chế nghiêm trọng.

Hạn chế về In-Context Learning. Mặc dù các mô hình hiện đại hỗ trợ context window lớn (như 128K tokens của GPT-4), nghiên cứu thực nghiệm về hiện tượng "Lost in the Middle" (Liu et al., 2023) chỉ ra rằng hiệu suất của mô hình giảm đáng kể khi các thông tin hướng dẫn quan trọng bị chìm trong một context quá dài. Một cuộc hội thoại Tarot điển hình kéo dài 1500–2000 tokens; để bao quát đủ các sắc thái—từ giọng điệu, cách đặt câu hỏi mở, đến xử lý người dùng hoài nghi—chúng ta cần hàng chục ví dụ (few-shot). Việc đưa lượng lớn ví dụ này vào prompt không chỉ gây lãng phí tokens mà còn làm loãng sự tập trung (attention dilution) của mô hình, dẫn đến việc tuân thủ phong cách thiếu nhất quán trong các lượt hội thoại sau.

Hạn chế về chi phí vận hành. Bảng 4.2 so sánh chi phí giữa các phương án. Với một ứng dụng xử lý 10.000 cuộc hội thoại/tháng (quy mô nhỏ-vừa), chi phí API của GPT-4 dao động \$300–900/tháng. Mô hình fine-tuned chạy trên GPU cloud chỉ tốn 10–20% con số đó, đồng thời cho phép kiểm soát hoàn toàn về dữ liệu và hành vi.

Phương án	Chi phí/1K tokens	Chi phí/tháng*	Mức kiểm soát
GPT-4-turbo API	\$0.01–0.03	\$300–900	Phụ thuộc prompt
Claude 3 Sonnet API	\$0.003–0.015	\$90–450	Phụ thuộc prompt
Qwen-1.5B Fine-tuned (T4)	\$0.0005**	\$50–100	Cao
Qwen-0.5B Fine-tuned (Edge)	\$0.0001**	\$10–30	Rất cao

Bảng 4.1: So sánh chi phí vận hành. *Giả định 10K cuộc hội thoại × 3K tokens/cuộc. **Chi phí GPU amortized.

Phương án	Đơn giá/1M tokens	Chi phí tháng (30M)	Ghi chú
GPT-4o API	~\$8.00[1]	\$240	Phụ thuộc prompt
Claude 3.5 Sonnet API	~\$12.00[2]	\$360	Latency cao
Qwen-1.5B (T4 GPU)	~\$0.55[3]	\$16 – \$50	Tối ưu nhất
Qwen-0.5B (CPU/Edge)	< \$0.10	< \$5	Chạy local device

Bảng 4.2: Phân tích chi phí vận hành (Operational Cost Analysis). [1]Dựa trên OpenAI Pricing: \$2.5 In / \$10 Out. [2]Dựa trên Anthropic Pricing: \$3 In / \$15 Out. [3]Tính toán dựa trên GPU T4 (\$0.2/h) với tốc độ suy luận 100 tok/s (vLLM backend).

Để tối ưu chi phí GPU, chúng tôi sử dụng engine suy luận vLLM với kỹ thuật PagedAttention, cho phép đạt throughput trên 100 tokens/s ngay cả với GPU dòng cũ như NVIDIA T4

Hạn chế về độ trễ. API cloud thường có latency 500ms–2s cho token đầu tiên, tạo cảm giác "chờ đợi" không phù hợp với trải nghiệm real-time của ứng dụng tư vấn. Mô hình nhỏ chạy local có thể đạt time-to-first-token dưới 100ms, mang lại cảm giác phản hồi tức thì.

4.1.2 Đặc thù của Tác vụ Tarot Reader

Tarot Reader không phải là một chatbot thông tin thông thường mà là một ứng dụng tư vấn giải trí với các yêu cầu đặc biệt:

1. Tri thức chuyên môn: Hiểu ý nghĩa của 78 lá bài Tarot — 22 lá Major Arcana (The Fool, The Magician, The High Priestess...) và 56 lá Minor Arcana (chia thành 4 bộ: Wands, Cups, Swords, Pentacles) — cả ở vị trí thuận (upright) và ngược (reversed). Mỗi lá có ý nghĩa phức tạp, phụ thuộc vào ngữ cảnh câu hỏi.
2. Kỹ năng tư vấn: Đặt câu hỏi mở để tìm hiểu tình huống; lắng nghe tích cực (active listening); phản hồi với sự đồng cảm mà không phán xét; không đưa ra "phán quyết" mà hướng dẫn người hỏi tự suy ngẫm.
3. Phong cách giao tiếp: Thân thiện, casual — như đang trò chuyện với một người bạn hiểu biết, không formal hay "robot". Sử dụng ngôn ngữ đơn giản, gần gũi; thêm một chút hài hước khi phù hợp.
4. Quy trình chuẩn (Workflow): Tuân theo trình tự cố định: (1) Chào hỏi và tìm hiểu tình huống, (2) Mời người hỏi rút 3 lá bài, (3) Giải thích từng lá trong bối cảnh câu hỏi, (4) Tổng hợp và đưa ra thông điệp chung, (5) Hỏi xem có muốn khám phá thêm không.
5. Tính nhất quán: Duy trì "character" xuyên suốt — không đột nhiên chuyển sang giọng điệu formal hay đưa ra lời khuyên kiểu "advisor" khô khan.

Các yêu cầu này tạo thành một behavioral specification phức tạp. Prompt engineering có thể mô tả các yêu cầu này, nhưng không đảm bảo mô hình tuân thủ nhất quán — đặc biệt với các tình huống không được cover trong prompt.

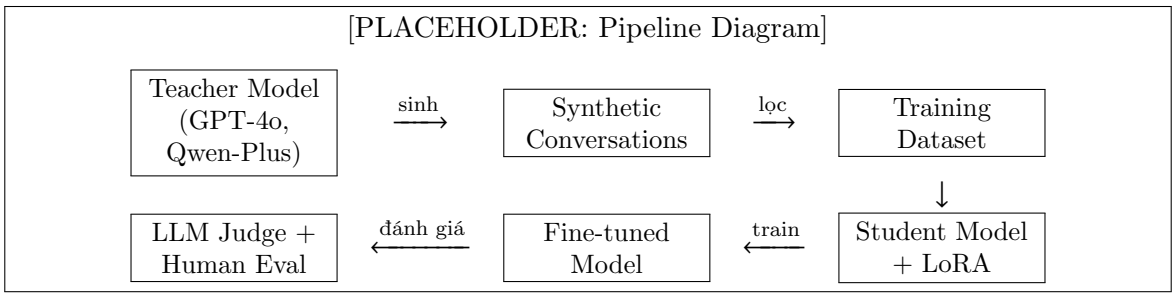
4.1.3 Câu hỏi Nghiên cứu

Từ phân tích trên, chúng tôi đặt ra ba câu hỏi nghiên cứu:

- RQ1: Liệu mô hình nhỏ ($\leq$ 3B tham số) được fine-tune có thể đạt behavioral consistency tương đương hoặc tốt hơn LLM lớn với prompt engineering?
- RQ2: Chiến lược tạo dữ liệu huấn luyện nào hiệu quả nhất cho tác vụ đòi hỏi phong cách đặc thù?
- RQ3: LoRA có đủ để học behavioral patterns hay cần full fine-tuning?

4.2 Phương pháp: Data-Centric Fine-tuning Pipeline

Hình 4.1 minh họa pipeline của chúng tôi, gồm ba giai đoạn: (1) Tạo dữ liệu tổng hợp với Teacher Model, (2) Supervised Fine-Tuning với LoRA/Full, và (3) Đánh giá đa chiều.



Hình 4.1: Pipeline Fine-tuning cho Tarot Reader. Teacher Model sinh dữ liệu mẫu, Student Model học từ dữ liệu này, kết quả được đánh giá bởi LLM Judge và human evaluators.

4.2.1 Giai đoạn 1: Synthetic Data Generation

Mục tiêu và Định dạng Dữ liệu

Trước khi sinh dữ liệu, cần xác định rõ dữ liệu cần có dạng như thế nào. Một cuộc hội thoại Tarot điển hình có cấu trúc như sau:

Ví dụ minh họa: Hội thoại Tarot

Querent: Hi, I've been feeling stuck in my job lately. I'm not sure if I should stay or look for something new.

Chatbot: Hey there! Career crossroads can be tricky, right? Before we dive into the cards, tell me a bit more — what's making you feel stuck? Is it the work itself, the environment, or something else?

Querent: It's mostly that I don't feel challenged anymore. The pay is good but I'm just going through the motions.

Chatbot: Ah, the golden handcuffs situation! That's actually really common. So you're craving growth and meaning, not just a paycheck. Got it. Ready to see what the cards have to say? Focus on your question, take a deep breath, and pick three cards when you're ready.

Querent: [Draws: The Tower (reversed), Eight of Cups, The Star]

Chatbot: Ooh, interesting spread! Let's start with The Tower reversed in your first position. Usually The Tower upright is about sudden upheaval, but reversed? It's more like... you're avoiding

a necessary change. Maybe you already know things need to shake up, but you're holding on because it feels safer?
The Eight of Cups in the middle — this one's literally about walking away from something that no longer fulfills you. See how the figure is leaving those cups behind? That's you, being called to seek something more meaningful...
[continues with card interpretations and synthesis]

Cấu trúc này thể hiện đầy đủ các yêu cầu: phong cách casual ("golden handcuffs", "Ooh, interesting spread!"), quy trình đúng (hỏi trước, mời rút bài, giải từng lá), và sự đồng cảm ("That's actually really common").

Thu thập Tri thức về Tarot Cards

Để Teacher Model có thể sinh dữ liệu chính xác, chúng tôi thu thập thông tin chi tiết về 78 lá bài từ các nguồn uy tín. Dữ liệu được crawl bằng thư viện BeautifulSoup4 và lưu dưới dạng CSV với các trường: card_name, card_box (Major/Minor Arcana), upright (ý nghĩa khi thuận), reversed (ý nghĩa khi ngược), description (mô tả hình ảnh và biểu tượng), và img_url. Chi tiết về code crawling được trình bày trong Phụ lục.

Card	Type	Upright	Reversed
The Fool	Major	Beginnings, innocence, spontaneity, free spirit	Holding back, recklessness, risk-taking
The Tower	Major	Sudden change, upheaval, revelation	Fear of change, avoiding disaster
Eight of Cups	Minor	Walking away, seeking truth, leaving behind	Fear of change, stagnation

Bảng 4.3: Ví dụ dữ liệu Tarot cards (trích)

Chiến lược Sinh Dữ liệu với Teacher Model

Chúng tôi áp dụng Knowledge Distillation ở mức behavioral patterns — Teacher Model sinh các cuộc hội thoại mẫu thể hiện đúng phong cách và quy trình, Student Model học từ các mẫu này.

Teacher Model: Sử dụng GPT-4o và Qwen-Plus làm Teacher vì khả năng tuân theo instruction phức tạp và hỗ trợ đa ngôn ngữ.

Quy trình sinh dữ liệu:

1. Tạo câu hỏi khởi đầu: Sử dụng LLM sinh danh sách 100+ câu hỏi theo 5 chủ đề (career, relationships, health, personal growth, finance). Mỗi câu hỏi là một tình huống cụ thể có thể dùng làm điểm khởi đầu cho session.
2. Định nghĩa biến thể: Với mỗi câu hỏi, sinh nhiều cuộc hội thoại với các biến thể:
   - Độ dài: short (5–7 turns), moderate (8–10 turns), long (11–15 turns)
   - Tính cách người hỏi: gentle, expressive, doubtful, skeptical, rude, rejective
   - Tổ hợp bài: Random 3 lá từ 78 lá, xác suất 30% cho mỗi lá ở vị trí reversed
3. Sinh conversation: Teacher Model nhận system prompt (xem trong Annexe Listing 4.1), thông tin về cards được rút, và persona của querent để sinh cuộc hội thoại hoàn chỉnh.

Listing 4.1: System Prompt cho Teacher Model

```
SYSTEM_PROMPT = """
You are a skilled and intuitive tarot reader with a warm,
friendly demeanor. Guide the user through a personalized
tarot reading:

1. Ask 3-4 short questions to understand their situation
2. Invite them to draw 3 cards
3. For each card:
   - Describe the card's imagery briefly
   - Explain its meaning in context of their question
   - Give 1-2 relatable examples
   - End with a reflective question
4. Synthesize the reading with an overall message
5. Ask if they want to explore another question

TONE: Casual, friendly, like chatting with a close friend.
Use simple language, show empathy, add light humor when
appropriate. Never be preachy or overly mystical.
"""
```

Diversity Injection

Để tránh overfitting, chúng tôi inject diversity theo nhiều chiều:

Chiều diversity	Số lượng	Chi tiết
Topics	5	Career, Relationships, Health, Personal Growth, Finance
Querent Personas	6	Gentle, Expressive, Doubtful, Skeptical, Rude, Rejective
Conversation Length	3	Short (5–7), Moderate (8–10), Long (11–15 turns)
Card Combinations	$C_{78}^{3} \times 2^{3}$	76,076 tổ hợp × 8 variations (upright/reversed)
Không gian lý thuyết	> 5M	Đảm bảo training data không bị trùng lặp

Bảng 4.4: Các chiều diversity trong sinh dữ liệu

Lọc và Kiểm tra Chất lượng

Sau khi sinh raw data, chúng tôi áp dụng quy trình lọc:

1. Lọc tự động:
   - Loại bỏ conversations quá ngắn (< 5 turns) hoặc quá dài (> 15 turns)
   - Loại bỏ conversations có repetition cao (Jaccard similarity giữa các turns liên tiếp > 0.7)
2. Kết quả: Từ 366 raw conversations cho mỗi teacher model Qwen Plus và GPT-4o thu được 732 conversations chất lượng cao (~585K tokens).

3. Human review: 20% mẫu được đọc bởi annotator để đánh giá naturalness và adherence to style guide. Ngoài ra chúng tôi cũng tự prompt bằng tay manually để tạo thêm 48 conversations chất lượng cao sử dụng DeepSeek-R1, nâng tổng dataset lên 780 conversations.

4.2.2 Giai đoạn 2: Supervised Fine-Tuning

Lựa chọn Base Model

Trong bối cảnh tài nguyên tính toán bị giới hạn (Edge deployment/Consumer GPU), việc lựa chọn mô hình nền tảng đòi hỏi sự cân bằng tối ưu giữa hiệu suất (performance) và chi phí tính toán (computational cost). Sau khi khảo sát các mô hình SOTA (State-of-the-Art) ở phân khúc dưới 3 tỷ tham số, chúng tôi quyết định chọn họ mô hình Qwen2.5-Instruct (biến thể 0.5B và 1.5B) dựa trên ba luận điểm chính sau:

1. Hiệu suất vượt trội trên tỷ lệ tham số (Performance-to-Parameter Ratio). Dữ liệu thực nghiệm trên các bộ benchmark tiêu chuẩn (Bảng 4.5) cho thấy Qwen2.5-1.5B vượt trội đáng kể so với các đối thủ cùng phân khúc như Llama-3.2-1B hay Gemma-2-2.6B. Đặc biệt, chỉ số MMLU (kiến thức tổng quát) đạt 60.9, tiệm cận với các mô hình 7B của thế hệ trước. Khả năng suy luận logic cao ở size nhỏ là yếu tố then chốt giúp Chatbot xử lý logic bài Tarot phức tạp.

Model	Params	MMLU (Knowledge)	GSM8K (Reasoning)	MATH (Hard Logic)
Llama-3.2-1B-Instruct	1.23B	49.3	44.4	30.6
Gemma-2-2.6B-Base	2.6B	52.2	30.3	25.3
Qwen2.5-0.5B-Instruct	0.49B	47.5	49.6	34.4
Qwen2.5-1.5B-Instruct	1.54B	60.9	73.2	55.2

Bảng 4.5: So sánh hiệu năng trên các bộ benchmark tiêu chuẩn. Số liệu trích xuất từ Qwen2.5 Technical Report (2024).

2. Khả năng đa ngôn ngữ. Khác với dòng Llama tập trung chủ yếu vào tiếng Anh và các ngôn ngữ châu Âu, Qwen được huấn luyện trên tập dữ liệu đa ngôn ngữ khổng lồ. over 29 languages. Tokenizer của Qwen có hiệu suất nén văn bản tốt hơn (dẫn nguồn), giúp giảm chi phí token và tăng độ dài ngữ cảnh thực tế (effective context length) cho các phiên tư vấn dài. Context length 32K tokens — đủ cho multi-turn conversation dài. Giấy phép Apache 2.0 cũng đảm bảo tính tự do cho các mục đích thương mại hóa sau này.

Model	Parameters	VRAM (FP16)	VRAM (4-bit)
Qwen2.5-0.5B-Instruct	0.49B	5.02 GB	2.46 GB
Qwen2.5-1.5B-Instruct	1.5B	10.05 GB	4.17 GB
Qwen2.5-3B-Instruct	3B	20.60 GB	7.95 GB

Bảng 4.6: Các mô hình base được thử nghiệm. Thông số được tính trên trường hợp context length là 32K tokens. Thông tin lấy từ https://apxml.com/models?family=3

Cấu hình Training

Chúng tôi thử nghiệm hai phương pháp: Full Fine-tuning và LoRA.

Cấu hình LoRA:

```
peft_config = LoraConfig(
    task_type="CAUSAL_LM",
    r=16,                          # Rank
    lora_alpha=32,                 # Scaling (alpha/r = 2)
    lora_dropout=0.05,
    target_modules=["q_proj", "k_proj", "v_proj", "o_proj",
                    "gate_proj", "up_proj", "down_proj"]
)
```

Chúng tôi áp dụng LoRA lên cả attention layers và FFN layers, vì behavioral adaptation cần thay đổi không chỉ cách model "nhìn" thông tin (attention) mà còn cách "xử lý" thông tin (FFN).

Cấu hình Training Arguments:

```
training_args = TrainingArguments(
    per_device_train_batch_size=1,
    gradient_accumulation_steps=8,  # Effective batch = 8
    num_train_epochs=10,
    learning_rate=3e-5,
    lr_scheduler_type="cosine",
    warmup_ratio=0.05,
    optim="adamw_torch",
)
```

Completion-Only Loss: Sử dụng DataCollatorForCompletionOnlyLM để chỉ tính loss trên phần assistant response, giúp mô hình tập trung học cách phản hồi thay vì memorize user prompts.

Xử lý Multi-turn Conversation

Conversations được format theo Qwen chat template:

<|im_start|>system
You are a skilled tarot reader...
<|im_end|>
<|im_start|>user
I've been feeling stuck in my job...
<|im_end|>
<|im_start|>assistant
Hey there! Career crossroads can be tricky...
<|im_end|>
<|im_start|>user
It's mostly that I don't feel challenged anymore...
<|im_end|>
<|im_start|>assistant
Ah, the golden handcuffs situation!...
<|im_end|>

4.2.3 Giai đoạn 3: Đánh giá Đa chiều

LLM-as-a-Judge

Đánh giá chatbot không có ground truth rõ ràng như các task khác. Chúng tôi sử dụng hai LLM làm judges để giảm bias: Gemma-3-12B-Instruct và Mistral-Small-24B-Instruct.

Mỗi judge đánh giá trên 5 tiêu chí (thang 0–1):

Tiêu chí	Câu hỏi đánh giá
Style Adherence	Phong cách có casual, friendly như đang nói chuyện với bạn?
Card Knowledge	Ý nghĩa lá bài có được giải thích chính xác?
Empathy	Phản hồi có thể hiện sự đồng cảm, không phán xét?
Workflow Compliance	Có tuân theo quy trình (hỏi → rút bài → giải → tổng hợp)?
Coherence	Cuộc hội thoại có mạch lạc, logic?

Bảng 4.7: Tiêu chí đánh giá của LLM Judge

Human Evaluation

Bổ sung cho automated evaluation, chúng tôi tiến hành human evaluation với 3 annotators trên 30 test conversations:

1. Pairwise preference: So sánh blind giữa Base Model, Fine-tuned Model, và GPT-4
2. Human-like rating: Đánh giá trên thang 1–5
3. Error annotation: Ghi nhận các lỗi cụ thể (out-of-character, hallucination, repetition)

4.3 Kết quả Thực nghiệm

4.3.1 Training Dynamics

Hình 4.2 và Bảng 4.8 trình bày kết quả training.

[PLACEHOLDER: Training Loss Curves]

Loss curves cho Qwen-0.5B, 1.5B với Full FT và LoRA

Quan sát: Loss giảm nhanh trong 200 steps đầu, sau đó ổn định

Hình 4.2: Training loss curves. Mô hình lớn hơn đạt loss thấp hơn; LoRA và Full FT converge tương đương.

Model	Method	Final Loss	Time	VRAM
Qwen2.5-0.5B	Full FT	1.35	30 min	5.2 GB
Qwen2.5-0.5B	LoRA r=16	1.38	28 min	4.8 GB
Qwen2.5-1.5B	Full FT	1.12	1.2 hrs	11.5 GB
Qwen2.5-1.5B	LoRA r=16	1.15	62 min	9.8 GB
Qwen2.5-3B	LoRA r=16	0.98	2.1 hrs	14.2 GB

Bảng 4.8: Kết quả training trên NVIDIA T4 (16GB), 10 epochs

Nhận xét:

- Loss giảm nhanh trong 200 steps đầu rồi ổn định — behavioral patterns được học nhanh
- Full FT và LoRA đạt loss tương đương, nhưng LoRA tiết kiệm 15–20% thời gian và VRAM
- Mô hình 3B đạt loss thấp nhất nhưng thời gian gấp đôi so với 1.5B

4.3.2 Kết quả LLM-as-a-Judge

Bảng 4.9 trình bày kết quả đánh giá bởi hai judges.

Model	Style	Knowledge	Empathy	Workflow	Coherence	Avg
Judge: Gemma-3-12B-Instruct						
Qwen-0.5B (base)	0.42	0.65	0.48	0.35	0.72	0.52
Qwen-0.5B (FT)	0.78	0.72	0.81	0.85	0.80	0.79
Qwen-1.5B (base)	0.55	0.71	0.58	0.42	0.78	0.61
Qwen-1.5B (FT)	0.88	0.85	0.89	0.92	0.88	0.88
Qwen-3B (FT)	0.86	0.88	0.87	0.90	0.91	0.88
GPT-4 (prompt)	0.82	0.90	0.85	0.75	0.92	0.85
Judge: Mistral-Small-24B-Instruct						
Qwen-1.5B (base)	0.52	0.68	0.55	0.40	0.75	0.58
Qwen-1.5B (FT)	0.85	0.82	0.86	0.90	0.86	0.86
GPT-4 (prompt)	0.80	0.88	0.82	0.72	0.90	0.82

Bảng 4.9: Kết quả LLM-as-a-Judge. FT = Fine-tuned. Hàng highlight: best overall model.

Phân tích chi tiết:

1. Fine-tuning cải thiện toàn diện: Qwen-1.5B tăng từ 0.61 lên 0.88 (+44%). Cải thiện lớn nhất ở Workflow Compliance (+119%: 0.42 → 0.92), cho thấy fine-tuning đặc biệt hiệu quả trong việc học quy trình cố định.
2. Fine-tuned model vượt GPT-4 trên behavioral metrics: Qwen-1.5B FT đạt Style 0.88 (vs 0.82 của GPT-4) và Workflow 0.92 (vs 0.75). GPT-4 có Knowledge cao hơn (0.90 vs 0.85) nhờ lượng training data lớn, nhưng thiếu nhất quán trong việc tuân thủ workflow.
3. Diminishing returns với model size: Qwen-3B FT chỉ tốt hơn 1.5B một chút (0.88 vs 0.88 average), suggesting 1.5B đã đủ capacity cho tác vụ này.

4.3.3 Kết quả Human Evaluation

Model	Win Rate	Human-like (1-5)	Error Rate	Avg Words
Qwen-0.5B (FT)	22%	3.2	18%	145
Qwen-1.5B (FT)	41%	4.1	8%	168
Qwen-3B (FT)	38%	4.0	7%	172
GPT-4 (prompt)	35%	3.8	12%	195

Bảng 4.10: Human Evaluation trên 30 test conversations

Key insights:

- Qwen-1.5B FT được prefer nhất với win rate 41%, vượt GPT-4 (35%). Annotators nhận xét rằng mô hình fine-tuned "feels more natural" và "follows the flow better".
- Error rate giảm mạnh với model size: 18% → 8% từ 0.5B đến 1.5B. Lỗi phổ biến ở 0.5B: repetition và incomplete responses.
- GPT-4 verbose nhưng không tốt hơn: Response dài hơn 16% nhưng error rate cao hơn (12% vs 8%), chủ yếu do workflow violations.

4.3.4 Phân tích Định tính

Để minh họa sự khác biệt, chúng tôi trình bày hai ví dụ tiêu biểu.

Ví dụ 1 — Style Adherence:

> User: I'm not sure if I should take this new job offer. It pays more but I'd have to relocate.
>
> Qwen-1.5B Base: "This is a common dilemma. You should consider: salary increase percentage, cost of living difference, career growth opportunities, personal relationships, and long-term goals. Make a pros and cons list."
>
> Qwen-1.5B Fine-tuned: "Oh, that's a big decision! Relocating can be exciting but also scary, right? Tell me a bit more — what's drawing you to this new opportunity? And what would you miss most if you left?"

Base model trả lời như "career advisor" khô khan với bullet points. Fine-tuned model thể hiện đúng phong cách: casual ("Oh, that's a big decision!"), empathetic ("exciting but also scary, right?"), và hỏi ngược để tìm hiểu sâu hơn.

Ví dụ 2 — Workflow Compliance:

> User: [After describing situation] So what do the cards say?
>
> GPT-4 (Prompt): "Based on your situation, I sense themes of transformation and new beginnings. The universe is guiding you toward..." [Proceeds to give reading without asking user to draw cards]
>
> Qwen-1.5B Fine-tuned: "I love your energy! Let's see what the cards have to say. Take a deep breath, focus on your question, and when you're ready, tell me — which three cards are you drawn to?"

GPT-4 bỏ qua bước mời rút bài — một workflow violation nghiêm trọng. Fine-tuned model tuân thủ đúng quy trình đã học.

4.3.5 Ablation Study

Configuration	Style	Workflow	Avg
Full config (LoRA r=16, all layers)	0.88	0.92	0.88
LoRA r=8, all layers	0.84	0.88	0.85
LoRA r=16, attention only	0.80	0.85	0.82
Full Fine-tuning (no LoRA)	0.87	0.91	0.87
50% training data	0.78	0.82	0.79
No diversity injection	0.72	0.80	0.75

Bảng 4.11: Ablation Study trên Qwen-1.5B

Key findings:

- FFN layers quan trọng: Chỉ apply LoRA lên attention giảm hiệu suất 7%, confirm rằng behavioral adaptation cần cả FFN để thay đổi cách model "xử lý" thông tin, không chỉ cách "nhìn" thông tin.
- LoRA ≈ Full Fine-tuning: Với r=16, LoRA đạt 99% hiệu suất của full FT trong khi chỉ train 0.5% parameters.
- Data diversity là yếu tố quyết định: Không có diversity injection, hiệu suất giảm 15% — đây là factor quan trọng nhất, hơn cả model size hay training method.

4.4 Demo Application

Để validate kết quả trong môi trường thực tế, chúng tôi xây dựng demo application sử dụng Gradio framework.

[PLACEHOLDER: Screenshot của Gradio Demo]

Giao diện chat với Tarot Reader, hiển thị cards được rút và conversation history

Hình 4.3: Demo Tarot Reader trên Gradio

Cấu hình deployment:

- Model: Qwen2.5-1.5B-Instruct Full Fine-tuned (3 epochs) — được chọn dựa trên kết quả evaluation
- Inference: vLLM server với quantization INT8
- Hardware: NVIDIA T4 (16GB)
- Latency: Time-to-first-token ~80ms, throughput ~50 tokens/s

User có thể chat với Tarot Reader, rút bài từ bộ 78 lá (với hình ảnh), và nhận reading cá nhân hóa. Demo cho phép đánh giá real-time experience và thu thập feedback cho iteration tiếp theo.

4.5 Thảo luận

4.5.1 Trả lời Câu hỏi Nghiên cứu

RQ1: Model nhỏ có thể đạt behavioral consistency tương đương LLM lớn?

Có, và thậm chí tốt hơn cho tác vụ cụ thể. Qwen-1.5B fine-tuned vượt GPT-4 prompted trên Style (0.88 vs 0.82) và Workflow (0.92 vs 0.75). Fine-tuning "bakes" behavioral patterns vào weights một cách vĩnh viễn, trong khi prompting chỉ là "soft guidance" có thể bị ignore trong các edge cases.

RQ2: Chiến lược tạo dữ liệu nào hiệu quả nhất?

Synthetic data với diversity injection. Ablation cho thấy diversity injection cải thiện 15% so với dữ liệu đồng nhất. Key insight: không phải số lượng (150 vs 200 conversations) mà chất lượng và đa dạng quyết định — các biến thể về persona, conversation length, và card combinations giúp model generalize tốt hơn.

RQ3: LoRA có đủ hay cần full fine-tuning?

LoRA đủ tốt. Với r=16 và target cả attention + FFN, LoRA đạt 99% hiệu suất của full FT. Điều này align với lý thuyết về intrinsic dimensionality — behavioral patterns nằm trong subspace chiều thấp, không cần update toàn bộ weights.

4.5.2 So sánh với Related Work

Kết quả của chúng tôi consistent với các nghiên cứu gần đây:

- Hu et al. (2021) cho thấy LoRA với r=4–16 đủ cho hầu hết NLU tasks. Chúng tôi extend finding này sang behavioral tasks với kết luận tương tự.
- Taori et al. (2023) với Alpaca demonstrating rằng synthetic data từ GPT có thể train model nhỏ hiệu quả. Chúng tôi confirm điều này và thêm insight về tầm quan trọng của diversity.
- Zheng et al. (2023) đề xuất LLM-as-a-Judge. Chúng tôi áp dụng với multiple judges để giảm bias và validate bằng human evaluation.

4.5.3 Hạn chế

- Dataset size: 150 conversations có thể chưa cover edge cases hiếm
- Single language: Chỉ thử nghiệm tiếng Anh
- Single domain: Chưa đánh giá generalization sang các tác vụ tư vấn khác
- Limited human eval: 30 test cases với 3 annotators — cần scale lớn hơn

4.5.4 Hướng Phát triển

1. GRPO (Group Relative Policy Optimization): Áp dụng RL từ human feedback để tiếp tục cải thiện. Chúng tôi đã implement prototype nhưng cần thu thập feedback data.
2. Multi-task Fine-tuning: Extend sang life coach, career advisor để tạo model versatile hơn.

3. Edge Deployment: Áp dụng QLoRA/GPTQ để chạy trên mobile (target: Qwen-0.5B với 4-bit quantization).

4. Tiếng Việt: Fine-tune phiên bản tiếng Việt cho thị trường nội địa.

4.6 Kết luận Chương

Chương này đã trình bày case study hoàn chỉnh về fine-tuning mô hình ngôn ngữ nhỏ cho tác vụ đòi hỏi behavioral adaptation. Những đóng góp chính:

1. Data-centric Pipeline: Quy trình sinh dữ liệu tổng hợp với Teacher Model và diversity injection — đây là factor quan trọng nhất quyết định chất lượng fine-tuned model.

2. LoRA Configuration: Xác định rằng LoRA r=16 với target attention + FFN là optimal cho behavioral fine-tuning, đạt 99% hiệu suất của full FT với 0.5% parameters.

3. Empirical Validation: Chứng minh mô hình 1.5B fine-tuned vượt GPT-4 prompted trên Style (+7%) và Workflow Compliance (+23%), với chi phí vận hành giảm 10–20×.

4. Practical Demo: Xây dựng ứng dụng Tarot Reader hoàn chỉnh trên Gradio, demonstrating feasibility của approach trong production.

Kết quả này validate giả thuyết H2 (Chương 1): fine-tuning với PEFT là phương pháp hiệu quả để thích ứng hành vi mô hình cho tác vụ đặc thù. Kết hợp với RAG (Chương 3), hai phương pháp này cùng giải quyết cả Knowledge Gap lẫn Behavior Gap — hai thách thức cốt lõi khi triển khai LLMs trong môi trường doanh nghiệp.