MINIRAG:
TỐI ƯU HÓA RAG CHO MÔ HÌNH NGÔN NGỮ NHỎ (SLM)

Nguyễn Dương Hải - 250101015

Tóm tắt

- Lớp: CS2205.CH201
- Link Github của nhóm: https://github.com/iamndh34/CS2205-CH201-MiniRAG.git
- Link YouTube video: https://youtu.be/Er1naAiJNes?si=4BzX5QTfW2U4eudQ

Nguyễn Dương Hải - 250101015

Giới thiệu

Bối cảnh thực tế:

- Xu hướng chuyển dịch AI từ Cloud về Edge (Laptop, Mobile) để bảo mật dữ liệu riêng tư.
- Sự trỗi dậy của các Small Language Models (SLMs) như Phi-3, Gemma-2B, Qwen-1.5B.

Hạn chế của RAG truyền thống (Vector-based):

- **Tốn kém tài nguyên:** Phải lưu trữ vector embeddings (float32) tốn RAM/Disk.
- **Độ trễ cao:** Quá trình tính toán embedding và tìm kiếm KNN (K-Nearest Neighbors) đòi hỏi GPU mạnh.
- **"Hộp đen" (Black-box):** Khó giải thích tại sao hệ thống chọn đoạn văn bản đó để trả lời.

Câu hỏi nghiên cứu: *Làm thế nào để tạo ra một cơ chế RAG hiệu quả, chính xác mà không cần GPU mạnh hay Vector Database?*

Mục tiêu

Mục tiêu 1: Xây dựng Pipeline Lập chỉ mục (Indexing)

- Thiết kế thuật toán tự động chuyển đổi văn bản thô (Unstructured Data) thành cấu trúc **Đồ thị hỗn hợp (Heterogeneous Graph)**.
- Tối ưu hóa dung lượng lưu trữ của đồ thị so với Vector DB.

Mục tiêu 2: Phát triển Thuật toán Truy xuất (Retrieval)

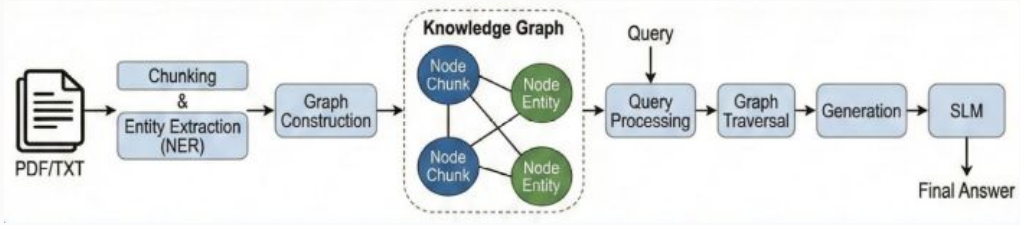
- Nghiên cứu phương pháp **Tìm kiếm Topo (Topology Search)**: Sử dụng các thực thể (Entities) làm cầu nối để tìm ngữ cảnh.
- Loại bỏ hoàn toàn bước tính toán Vector Embedding trong quá trình truy vấn.

Mục tiêu 3: Tối ưu hóa & Đánh giá thực nghiệm

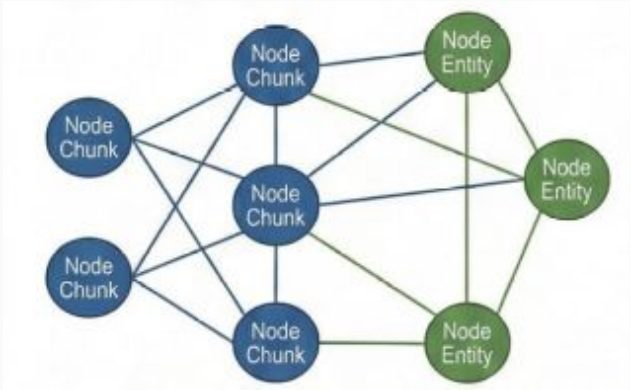
- Tích hợp hệ thống với các SLM phổ biến.
- Đánh giá hiệu năng trên phần cứng giới hạn (CPU Only, Low VRAM).

Nội dung và Phương pháp

Phương pháp 1 - Kiến trúc Đồ thị (Graph Construction)



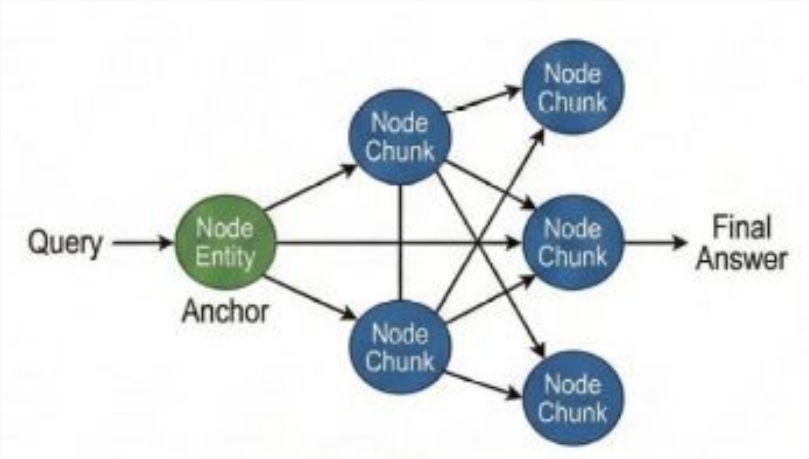
- **Input:** Tài liệu văn bản (PDF, TXT, Markdown).
- **Quy trình xử lý:**
  - **Chunking:** Chia văn bản thành các đoạn nhỏ (Block size: 128-256 tokens) để đảm bảo ngữ cảnh cô đọng.
  - **Entity Extraction (NER):** Sử dụng mô hình NER nhẹ (như Spacy hoặc Gliner-small) để trích xuất danh từ riêng, thuật ngữ kỹ thuật.
- **Cấu trúc Đồ thị hỗn hợp:**
  - **Node Chunk:** Đại diện cho nội dung văn bản gốc.
  - **Node Entity:** Đại diện cho các từ khóa/thực thể trích xuất được.
  - **Edge (Cạnh):** Biểu diễn mối quan hệ "chứa đựng" (Chunk chứa Entity) hoặc quan hệ đồng xuất hiện (Co-occurrence).



Nội dung và Phương pháp

Phương pháp 2 - Quy trình Truy xuất (Retrieval Strategy)

- **Bước 1: Query Processing**
  - Phân tích câu hỏi người dùng -> Trích xuất từ khóa chính (Query Entities).
- **Bước 2: Graph Traversal (Duyệt đồ thị)**
  - Xác định các **Node Entity** trên đồ thị trùng khớp với từ khóa câu hỏi (Nodes neo - Anchor nodes).
  - Thực hiện lan truyền (Propagation) từ Node Entity sang các **Node Chunk** liên kết trực tiếp.
  - *Ưu điểm:* Tìm kiếm chính xác dựa trên từ khóa thực tế, không bị nhiễu bởi khoảng cách vector mơ hồ.
- **Bước 3: Ranking & Contextualization**
  - Xếp hạng các Node Chunk dựa trên số lượng kết nối với từ khóa trong câu hỏi.
  - Ghép nội dung các Chunk hàng đầu vào Prompt của SLM.



Nội dung và Phương pháp

Phương pháp 3 - Tích hợp & Sinh câu trả lời (Generation)

- **Mô hình ngôn ngữ (SLM):**
  - Sử dụng các model < 3B tham số (Ví dụ: Microsoft Phi-3-mini, Google Gemma-2b).
  - Chạy Quantized (int4/int8) để tối ưu bộ nhớ.
- **Cấu trúc Prompt:**
  - Input: `[Context từ Graph] + [Câu hỏi người dùng]`
  - Instruction: "Trả lời câu hỏi dựa trên thông tin được cung cấp trong Context."
- **Cơ chế Minh bạch (Explainability):**
  - Hệ thống có thể trích xuất lại đường đi trên đồ thị: *Query -> Entity A -> Chunk B -> Answer*.

Kết quả dự kiến

Kế hoạch thực nghiệm & Đánh giá

- **Môi trường thử nghiệm:**
  1. Hardware: Laptop cá nhân (CPU 11th Gen Intel® Core™ i7-1165G7 × 8, RAM 16GB).
  2. Dataset: Tập dữ liệu hỏi đáp về kế toán Việt Nam theo TT99/2025/TT-BTC.
- **Tiêu chí đánh giá:**
  1. **Độ chính xác:** So sánh câu trả lời với RAG Vector truyền thống.
  2. **Tốc độ:** Thời gian từ lúc hỏi đến lúc nhận câu trả lời.
  3. **Tài nguyên:** Mức tiêu thụ RAM và dung lượng ổ cứng lưu Index.

Kết quả dự kiến

- **Sản phẩm:**
  - Bộ mã nguồn Python hoàn chỉnh triển khai MiniRAG cho truy vấn tài khoản kế toán.
  - Demo ứng dụng Chatbot chạy 100% offline trên Laptop.
- **Đóng góp khoa học:**
  - Chứng minh tính khả thi của việc dùng Graph thay thế Vector cho bài toán RAG trên thiết bị yếu.
  - Đề xuất phương pháp tối ưu hóa bộ nhớ cho các ứng dụng AI cục bộ.

Tài liệu tham khảo

[NeurIPS 2020] Patrick S. H. Lewis et al.: *Retrieval-Augmented Generation for Knowledge-Intensive NLP Tasks*.

[arXiv 2024] Yuqi Zhu et al.: *Graph RAG: Retrieval Augmented Generation with Knowledge Graphs*.

[arXiv 2025] Tianyu Fan et al.: *MiniRAG: Towards Extremely Simple Retrieval-Augmented Generation*.

[NAACL 2019] Jacob Devlin et al.: *BERT: Pre-training of Deep Bidirectional Transf*