

ĐẠI HỌC QUỐC GIA THÀNH PHỐ HỒ CHÍ MINH TRƯỜNG ĐẠI HỌC CÔNG NGHÊ THÔNG TIN KHOA KHOA HỌC MÁY TÍNH

BÁO CÁO ĐỒ ÁN

Xây dựng hệ thống PDF Question Answering

dựa trên RAG

Môn học: Xử lý ngôn ngữ tự nhiên

Giảng viên hướng dẫn: TS. Nguyễn Trọng Chỉnh

Thầy Nguyễn Đức Vũ

Thành viên nhóm 21520331 – Châu Thiên Long 21520229 – Tăng Minh Hiển

21520334 – Nguyễn Thái Thành Long

Thành phố Hồ Chí Minh, tháng 1 năm 2024

NHẬN XÉT CỦA GIẢNG VIÊN HƯỚNG DẪN

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

.............................................................................................................................................

........., ngày ... tháng 2 năm 2024

Người nhận xét

(Ký tên và ghi rõ họ tên)

# MỤC LỤC

[DANH MỤC HÌNH ẢNH 1](#_bookmark0)

[DANH MỤC BẢNG BIỂU 2](#_bookmark1)

[DANH MỤC PHỤ LỤC 3](#_bookmark2)

[LỜI CẢM ƠN 4](#_bookmark3)

[LỜI MỞ ĐẦU 5](#_bookmark4)

[Chương 1: TỔNG QUAN ĐỀ TÀI 6](#_bookmark5)

1. [Giới thiệu đề tài 6](#_bookmark6)
2. [Tổng quan đề tài 8](#_bookmark7)
3. [Ý nghĩa đề tài 8](#_bookmark8)
4. [Mục tiêu đồ án 8](#_bookmark9)
   1. [Mục tiêu môn học 8](#_bookmark10)
   2. [Mục tiêu đồ án 9](#_bookmark11)

[Chương 2: CƠ SỞ LÝ THUYẾT - HIỆN THỰC ĐỀ TÀI 10](#_bookmark12)

1. [Retrieval-Augmented Generation (RAG) 10](#_bookmark14)
2. [Split Document 12](#_bookmark15)
   1. [Cài đặt: 13](#_bookmark16)
3. [Embedding model 13](#_bookmark17)
   1. [Cài đặt 14](#_bookmark19)
4. [Retriever 15](#_bookmark20)
   1. [Cài đặt: 17](#_bookmark21)
5. [Large Language Model Generation 17](#_bookmark22)
   1. [Cài đặt: 19](#_bookmark25)

[Chương 3: XÂY DỰNG BỘ DỮ LIỆU 21](#_bookmark26)

1. [Nguồn dữ liệu 21](#_bookmark27)
2. [Cách xây dựng 21](#_bookmark28)
3. [Phân tích dữ liệu 23](#_bookmark29)

[Chương 4: ĐỘ ĐO ĐÁNH GIÁ – KẾT QUẢ 25](#_bookmark31)

1. [Độ đo đánh giá 25](#_bookmark32)
   1. [Context Precision 26](#_bookmark33)
   2. [Context Recall 26](#_bookmark34)
   3. [Faithfulness 26](#_bookmark35)
   4. [Answer Correctness 26](#_bookmark36)
2. [Kết quả thử nghiệm 27](#_bookmark37)
   1. [Kết quả 27](#_bookmark38)
   2. [Nhận xét 27](#_bookmark40)

[Chương 5: KẾT LUẬN 29](#_bookmark41)

1. [Ưu điểm 29](#_bookmark42)
2. [Nhược điểm 29](#_bookmark43)
3. [Hướng phát triển 29](#_bookmark44)

[Chương 6: TÀI LIỆU THAM KHẢO 31](#_bookmark45)

[Phụ lục 32](#_bookmark46)

# DANH MỤC HÌNH ẢNH

[Hình 2. Sơ đồ tổng quan hệ thống hỏi đáp file PDF dựa trên RAG 10](#_bookmark13)

[Hình 2-1: Mô hình Encoder – Decoder của mô hình LLM BAAI/bge-large-en 14](#_bookmark18)

[Hình 2-2: Tóm tắt các bước mô hình Zephyr được huấn luyện trên bộ UltraChat và tối ưu](#_bookmark24) [dựa trên AI Feedback 19](#_bookmark24)

[Hình 3-1: Đại diện các mẫu dữ liệu đã được thu thập từ 3 bài báo 23](#_bookmark30)

# DANH MỤC BẢNG BIỂU

[Bảng 2-1: Tham số cài đặt của mô hình LLM zephyr-7b-beta 18](#_bookmark23)

[Bảng 4-1: Kết quả thử nghiệm của hệ thống truy vấn dựa trên bộ dữ liệu 27](#_bookmark39)

# DANH MỤC PHỤ LỤC

[Bảng phụ lục 2: Bảng phân công công việc 32](#_bookmark47)

# LỜI CẢM ƠN

Để hoàn thành đồ án này, nhóm chúng em xin gửi lời cảm ơn chân thành nhất đến với trường Đại học Công nghệ Thông tin cùng khoa *Khoa học máy tính* đã tạo mọi điều kiện tốt nhất để tất cả sinh viên chúng em được trải nghiệm và học tập với môn học *Xử lý ngôn ngữ tự nhiên* (Mã môn CS221). Trong xuyên suốt khoảng thời gian học tập, chúng em đã được tiếp thu không chỉ kiến thức về chuyên môn về Công nghệ thông tin, về chuyên ngành Khoa học máy tính mà còn là kiến thức về kỹ năng trình bày và làm việc nhóm. Đây là những kiến thức vô cùng quan trọng và cần thiết để hoàn thành đồ án và làm hành trang của chúng em cho công việc trong tương lai

Tiếp đến, là lời cảm ơn sâu sắc đến với TS. Nguyễn Trọng Chỉnh, thầy Nguyễn Đức Vũ, giảng viên giảng dạy lý thuyết. Dưới sự hướng dẫn, định hướng về chuyên môn cùng với những lời góp ý sâu sắc và giúp đỡ tận tình của các thầy đã giúp sức rất nhiều cho nhóm chúng em để hoàn thành đồ án môn học lần này. Cùng với đó, qua quá trình làm việc và học tập với các thầy, chúng em cũng được thu nạp thêm những tri thức mới về xử lý ngôn ngữ tự nhiên cũng như các kiến thức về nghiên cứu các bài báo khoa học. Chúng em vô cùng biết ơn với những bài học quý báu đó.

Nhóm chúng em đã dành nhiều thời gian để cố gắng tìm hiểu và hoàn thành đồ án môn học tốt nhất trong khả năng trong khả năng nhóm chúng em. Dẫu vậy, khó có thể những thiếu sót vì những thiếu sót nhất định về kiến thức, kỹ năng. Vì thế, qua đồ án này, nhóm em cũng xin lắng nghe những góp ý bổ sung từ thầy để nhóm có thể hoàn thiện hơn ở đề tài này.

Cuối lời, chúng em xin được chân thành cảm ơn tất cả vì đã giúp đỡ chúng em hoàn thành thuận lợi đồ án lần này.

TP. Hồ Chí Minh, ngày … tháng 2 năm 2024

# LỜI MỞ ĐẦU

Trí tuệ nhân tạo đang ngày càng phát triển và là xu hướng công nghệ mới được áp dụng để xây dựng nhiều ứng dụng, đáp ứng được nhiều nhu cầu ngày càng nâng cao từ người dùng. Trong đó, để đáp ứng được nhu cầu học hỏi, nắm bắt và hiểu được các tri thức, thông tin khác nhau từ con người, các hệ thống đã ra đời và được phát triển để trở thành một công cụ hỗ trợ con người đáp ứng các nhu cầu trên.

Để xây dựng các hệ thống này, đóng một vai trò không nhỏ tạo ra các hệ thống hỏi đáp chính là các mô hình ngôn ngữ, vốn là hạt nhân để đưa các hệ thống này có thể đưa ra câu trả lời gần nhất với mong muốn từ con người. Ngày nay, với sức mạnh của các hệ thống máy tính hiện đại, các mô hình ngôn ngữ đã được tạo ra nhằm đáp ứng với nhu cầu xử lý các nhu cầu phức tạp, đặc biệt với tác vụ tạo văn bản của hệ thống hỏi đáp. Với số lượng tham số cực lớn được huấn luyện bởi một bộ dữ liệu lớn đã tạo ra nhiều mô hình ngôn ngữ lớn (Large Language Model) như GPT3.5, Meena, Bard,...

Từ nhu cầu ngày càng cao của nhiều người dùng mong muốn có cho mình hệ thống hỏi đáp với mong muốn nắm được tri thức trong một lĩnh vực cụ thể, cùng với nhu cầu tìm hiểu các mô hình ngôn ngữ lớn hiện nay, nhóm đã quyết định lựa chọn đề tại xây dựng hệ thống PDF Question Answering dựa trên RAG là chủ đề cho đồ án lần này của nhóm chúng em.

Do có những hạn chế về kiến thức và kinh nghiệm thực tiễn để xây dựng ứng dụng chủ đề này nên nhóm sẽ mắc một vài khuyết điểm. Vì thế nhóm mong đợi những góp ý từ thầy cũng như các bạn để cải thiện chất lượng sản phẩm đề tài cũng như nâng cao kiến thức chuyên môn về môn học.

## Chương 1: TỔNG QUAN ĐỀ TÀI

Đề mục của chương 1 gồm

* Giới thiệu đề tài
* Tổng quan đề tài
* Ý nghĩa đề tài
* Mục tiêu đồ án

Nội dung chương 1 nhằm đưa ra góc nhìn tổng quan về đề tài của nhóm, chi tiết về các đồ án của nhóm. Cuối cùng là phạm vi đề tài và mục tiêu thực hiện đồ án của nhóm.

### Giới thiệu đề tài

Trí tuệ nhân tạo đang ngày càng phát triển mạnh mẽ và có tác động sâu rộng đến nhiều lĩnh vực xung quanh chúng ta hiện nay. Nhiều công trình nghiên cứu từ các nhà Khoa học máy tính đã và đang được áp dụng một cách rộng rãi vào thực tiễn hiện nay. Có thể kể đến như ứng dụng nhận diện vật thể, khuôn mặt; nhận diện giọng nói; phân tích cảm xúc qua bình luận trực tuyến,... Và nổi lên trong khoảng thời gian ngắn dạo gần đây và trở thành một ứng dụng có số lượng người dùng kỷ lục trên thế giới hiện nay, chính là hệ thống hỏi đáp tự động như ChatGPT, Microsoft Bing, Google Bard,... Đây đang là một cuộc chay đua về công nghệ giữa những nhà công nghệ lớn để cho ra đời hệ thống hỏi đáp hiện đại hơn và có thể trả lời nhiều vấn đề khác nhau trong thế giới này.

Một điểm mà các mô hình hỏi đáp này có thể thông minh và trả lời được nhiều câu hỏi khác nhau từ người dùng chính là Mô hình Ngôn ngữ lớn (Large Language Model). Trước đây, với các giải thuật Máy học, người ta đã có thể sử dụng các mô hình ngôn ngữ để giải quyết nhiều bài toán như Phân tích cảm xúc, Gán nhãn từ loại hoặc Tạo văn bản để phục vụ hệ thống hỏi đáp. Tuy nhiên, các mô hình vẫn còn hạn chế do số lượng dữ liệu hạn chế được huấn luyện của mô hình và tài nguyên để huấn luyện nên các mô hình vẫn các tác vụ với độ chính xác còn chưa cao và chưa giải quyết được các bài toán mong muốn của con người ở độ chính xác cao nhất. Tuy nhiên, ngày nay, với mô hình ngôn

ngữ lớn, mô hình được huấn luyện với bộ dữ liệu lớn và số lượng tham số lớn, các mô hình này đã là một cuộc cách mạng trong giải quyết các bài toán lớn hiện nay và trong đó có sự thành công của các Chatbot hiện nay.

Hiện nay, bên cạnh sử dụng các hệ thống Chatbot, người lập trình cũng có thể tự xây dựng cho mình một hệ thống Chatbot riêng phục vụ cho nhu cầu cá nhân của bản thân. Có thể sử dụng các mô hình từ các đơn vị như OpenAI (mô hình GPT3.5, GPT4.0), Google (LaMDA), Google AI (Bard),... Tuy nhiên, đây là các mô hình mà người dùng có thể sử dụng miễn phí ở một mức độ nhất định. Nếu vượt qua số tokens được gọi qua API thì phải trả thêm phí cho các dịch vụ này. Thêm nữa, các dịch vụ này khi ta đưa dữ liệu vào thì các đơn vị có bản quyền dịch vụ này có thể dùng dữ liệu mà người dùng gửi vào để có thể phục vụ cho một mục đích nào khác ngoài tầm kiểm soát của người lập trình. Đây là yếu tố ảnh hưởng lớn đến bảo mật thông tin đối với các thông tin quan trọng cần đảm bảo bảo mật cao như thông tin nội bộ trong một doanh nghiệp hoặc thông tin của cơ quan chính phủ, nhà nước, quân đội,... Vì thế, để đảm bảo tính bảo mật cũng như có thể tiết kiệm chi phí, hiện nay có nhiều mã nguồn mở mà người lập trình có thể sử dụng để xây dựng các hệ thống hỏi đáp cho riêng mình.

Với nhu cầu sử dụng file PDF nhiều tại bậc Đại học, cũng như đây là định dạng phổ biến trong nhiều cơ quan trong việc lưu hành tài liệu ở trong trường học cũng như thực tế ở cơ quan, doanh nghiệp, nhóm nhận thấy khi có số lượng file PDF lớn, để nắm bắt tất cả thông tin của tất cả các file này bằng cá nhân thật sự gặp nhiều khó khăn. Vì thế nhu cầu về một hệ thống truy vấn để có thể dễ dàng lấy được các thông tin từ các tập tin này đã được nảy ra từ thực tế này và đang trở nên ngày càng phổ biến. Bên cạnh đó, khi làm việc tại doanh nghiệp, các tài liệu này thường ở dạng mật và cần đảm bảo tính bảo mật cao, không gây thất thoát, tránh lan tràn ra ngoài và đảm bảo kinh tế tốt do không trả tiền thêm khi dùng nhiều truy vấn cho các API trả phí, vì thế nhu cầu xây dựng hệ thống hỏi đáp dùng mô hình LLM mã nguồn mở đang ngày càng phổ biến trong xây dựng các hệ thống truy vấn. Nhận thấy nhu cầu này, nhóm đã quyết định lựa chọn đề tài này để thực hiện đồ án cho môn học Xử lý ngôn ngữ tự nhiên (CS221).

### Tổng quan đề tài

Tên đề tài: Xây dựng hệ thống PDF Question Answering dựa trên RAG (Retrieval Augmented Generation)

* Giải quyết bài toán Question Answering.
* Phục vụ cho domain cụ thể là truy vấn trên các tài liệu có định dạng PDF ( Portable Document Format).
* Lĩnh vực, chủ đề file PDF dùng để đánh giá mô hình: Bài báo khoa học.

### Ý nghĩa đề tài

Đồ án hướng đến giải quyết nhu cầu truy vấn các tài liệu trên file PDF dựa trên mô hình RAG (Retrieval Augmented Generation). Đây là nhu cầu mà một số nhà lập trình đang hướng đến học tập và xây dựng các hệ thống tương tự nhằm phục vụ mục đích cá nhân, cải thiện hiệu quả làm việc khi người dùng có số lượng lớn các tài liệu khác nhau.

Thêm nữa, các tài liệu PDF là tài liệu phổ biến trong nhiều môi trường làm việc khác nhau như trong các công ty, doanh nghiệp nhiều ngành nghề đều sử dụng định dạng tài liệu này để truyền các thông báo hoặc tổng hợp kiến thức của các doanh nghiệp.

Mặt khác, hệ thống hỏi đáp với ngày nay đang hướng đến sử dụng các mô hình ngôn ngữ lớn open-source (Open-source Large Language Model). Việc sử dụng này đem lại nhiều ưu điểm như đảm bảo được tính bảo mật đến các tài liệu mà người lập trình sử dụng để xây dựng hệ thống. Bên cạnh đó, tiết kiệm được chi phí xây dựng thay vì dùng các nền tảng đã phổ biến như OpenAI, Microsoft Bing,... Bên cạnh đó, với cộng đồng các nhà Khoa học Máy tính rộng lớn thì các mô hình mã nguồn mở này luôn phát triển không ngừng và ngày càng được tối ưu phù hợp với các nhu cầu khác nhau từ người dùng. Cuối cùng, người lập trình có thể thay đổi và tối ưu các tham số của mô hình tùy thuộc vào tài nguyên và nhu cầu sử dụng của người dùng.

### Mục tiêu đồ án

### Mục tiêu môn học

* + - Học tập và áp dụng thực tiễn các công nghệ hiện đại gồm mô hình ngôn ngữ lớn (LLM) và các kỹ thuật liên quan đến xử lý ngôn ngữ đã học trong môn học để hoàn thành đồ án môn học.
    - Phát triển ứng dụng có tính thực tiễn cao, đang được sử dụng rộng rãi bởi một số lượng người nhất định và có thể thao tác dễ dàng, tiện lợi.
    - Củng cố và phát huy khả năng làm việc nhóm của các thành viên.

### Mục tiêu đồ án

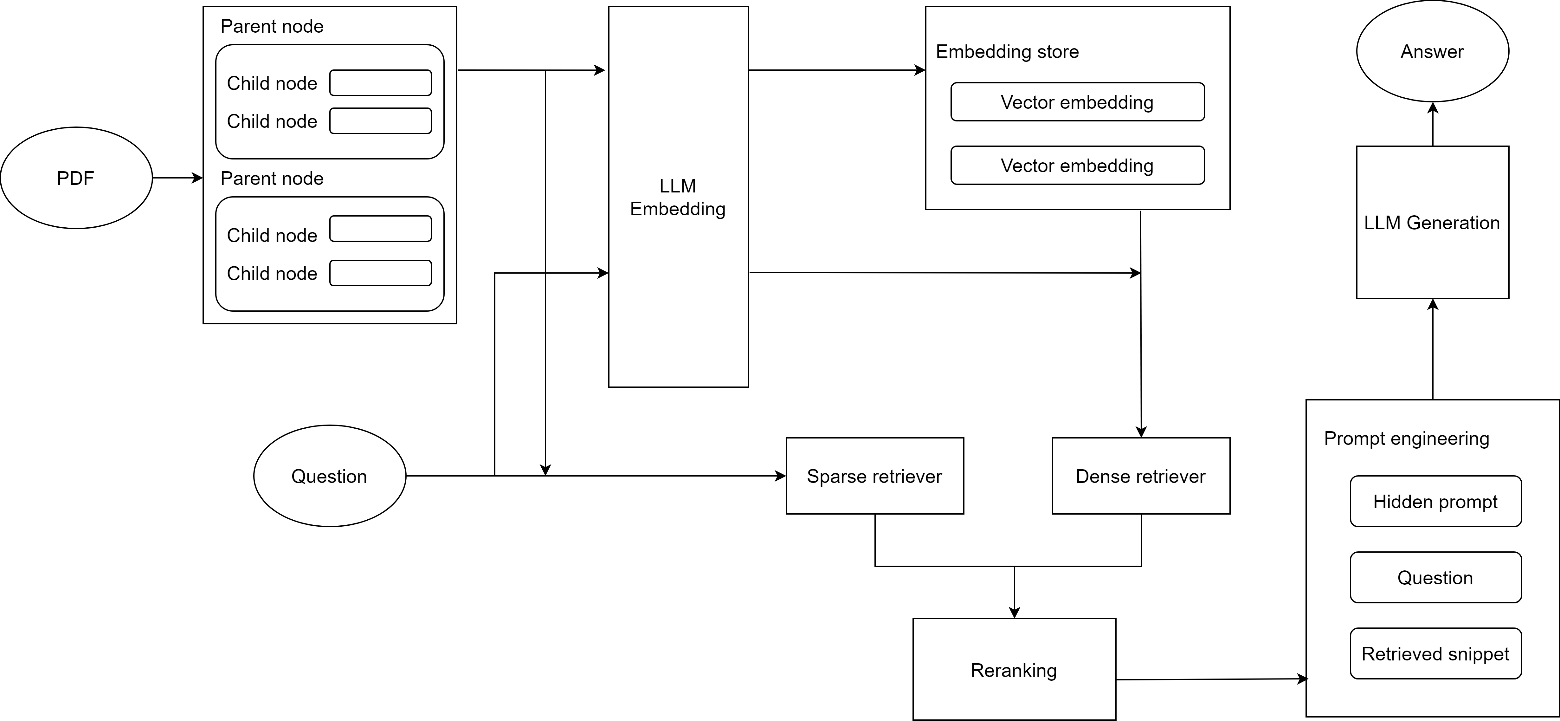
* + - Xây dựng hệ thống PDF Question Answering dựa trên RAG.
    - Người dùng có thể thêm các file PDF bên ngoài và có thể thực hiện hỏi đáp trực tiếp trên file này.
    - Có thể thực hiện hỏi đáp với chủ đề các kiến thức liên quan đến các bài báo khoa học từ các bài báo khoa học có định dạng PDF được đưa vào hệ thống.

## Chương 2: CƠ SỞ LÝ THUYẾT - HIỆN THỰC ĐỀ TÀI

Nội dung chương 2 gồm các mục:

* + - Retrieval-Augmented Generation (RAG)
    - Split Document
    - Embedding model
    - Retriever
    - Large Language Model Gerneration

Nội dung chương 2 là trình bày về cơ sở lý thuyết về cách xây dựng hệ thống PDF Question Answering PDF dựa trên mô hình RAG. Bên cạnh đó, chương 2 cũng trình bày các bước mà nhóm đã hiện thực đề tài. Phía trên các đề mục chính là lần lượt các bước triển khai cho xây dựng hệ thống đề tài của nhóm.



*Hình 2. Sơ đồ tổng quan hệ thống hỏi đáp file PDF dựa trên RAG*

### Retrieval-Augmented Generation (RAG)

LLMs (Large Language Models) là các mô hình đã được train trên một lượng lớn dữ liệu với hàng tỉ tham số giúp hoàn thành các tác vụ như dịch câu, hoàn thành câu hay

hỏi đáp. Tuy nhiên khi ứng dụng LLM để xây dựng hệ thống hỏi đáp sẽ gặp nhiều hạn chế như:

* Có thể trình bày những thông tin lỗi thời hoặc chung chung khi người dùng mong muốn nhận được phản hồi cụ thể phù hợp với hiện tại.
* Phản hồi đưa ra không rõ nguồn gốc, không có tính xác thực, không được dẫn chứng cụ thể để chứng minh tính đúng đắn.
* Đưa ra những phản hồi nhầm lẫn do một thuật ngữ có thể sử dụng trong nhiều lĩnh vực khác nhau.
* Phản hồi thông tin sai sự thật dù cho không có câu trả lời đúng.

RAG (Retrieval Augmented Generation) là quá trình tối ưu hóa đầu ra của LLMs nhằm cải thiện các khuyết điểm trên. RAG mở rộng các khả năng vốn đã mạnh mẽ của LLMs sang các lĩnh vực cụ thể mà không cần phải train lại mô hình. Một số lợi ích khi sử dụng RAG như:

* Tính cập nhật: khác với tập dữ liệu tĩnh được train cho các LLMs, RAG cho phép cung cấp một cách linh hoạt các nguồn thông tin như các nghiên cứu, bài báo, mạng xã hội,.. để cập nhật các thông tin mới nhất.
* Tính xác thực: RAG sử dụng các nguồn tài liệu được người dùng xác thực, tin tưởng để nâng cao tính đúng đắn của các phản hồi.
* Tiết kiệm chi phí: việc cập nhật thông tin thường xuyên mà không cần phải retrain khiến RAG tiết kiệm rất nhiều chi phí so với cách xử lí LLMs truyền thống.

Quá trình thực hiện RAG được tổng quát qua các bước sau:

* Chuẩn bị dữ liệu: Tổng hợp dữ liệu từ các nguồn, xử lí và biểu diễn thông tin. Sử dụng embedding language model để tạo các vector embedding và lưu trữ trên các hệ quản trị dữ liệu.
* Truy vấn thông tin: Sử dụng các độ đo hay phương pháp ước lượng để tìm ra thông tin phù hợp nhất với truy vấn đầu vào dựa theo các vector embedding đang được lưu trữ.
* Tạo LLM prompt: RAG tăng cường đầu vào cho LLM bằng cách thêm các ngữ liệu có liên quan đã được truy xuất vào prompt giúp LLM tạo ra các câu trả lời chính xác dựa trên truy vấn của người dùng.
* Cập nhật dữ liệu: tùy vào nhu cầu của người sử dụng mà dữ liệu đầu vào có thể được cập nhật thường xuyên để đảm bảo tính chính xác và xác thực cho LLM.

Với các phân tích trên, RAG thực sự phù hợp để sử dụng cho hệ thống hỏi đáp file PDF với tính linh hoạt trong việc thay đổi nguồn dữ liệu thường xuyên và trong từng lĩnh vực cụ thể.

### Split Document

RAG thực hiện truy xuất thông tin phù hợp nhất từ các nguồn do người dùng cung cấp nhằm tăng tính hiệu quả của LLM, vì vậy việc tố chức và lưu trữ thông tin đầu vào có thể ảnh hưởng trực tiếp đến hiệu quả của việc truy vấn nói riêng cũng như toàn bộ mô hình nói chung. Khi chia nhỏ văn bản đầu vào cần xem xét các yếu tố sau:

* Độ dài đoạn văn càng nhỏ thì vector embedding càng biểu diễn chính xác ngữ

nghĩa của đoạn văn. Nếu đoạn văn quá dài sẽ làm mất mát thông tin trong quá trình tạo vector embedding dẫn đến hiệu quả truy xuất kém.

* Mặt khác đoạn văn được truy xuất phải đủ độ dài để chứa đựng thông tin cần thiết

làm ngữ cảnh cho LLM tạo câu trả lời.

Phân cấp văn bản thành các parent-node và child-node giúp cân bằng được hai vấn đề trên. Đầu tiên văn bản được chia thành các parent-node với độ dài đủ lớn để chứa được toàn bộ ngữ cảnh hoàn chỉnh cho một câu trả lời từ LLM. Tiếp đến chia nhỏ các parent-node thành các child-node, tạo các vector embedding dựa trên child-node với độ dài vừa đủ để không bị mất mát thông tin. Sau đó thực hiện truy vấn dựa các vector

embedding vừa tạo, tra cứu ids của các parent-node tương ứng để trả về ngữ cảnh cho việc tạo prompt và câu trả lời.

Một khía cạnh khác cần xem xét là sự chồng chéo giữa các node. Trong mỗi node, một đoạn ngữ cảnh có thể hoàn chỉnh ý nghĩa trong khi các đoạn liền kề lại bị thiếu xót thông tin, điều này xảy ra do việc chia nhỏ văn bản không thể hoàn toàn đảm bảo dựa trên ngữ nghĩa của từng đoạn văn, điểm chia cắt có thể làm ý nghĩa của ngữ cảnh bị đứt đoạn. Đặt các node chồng chéo lên nhau giúp hạn chế sự mất mát cũng như bổ sung thông tin giữa các node, giúp cải thiện chất lượng đầu ra của mô hình.

### Cài đặt:

* + - Sử dụng thư viện PyPDFLoader để đọc file PDF với nhiều định dạng hỗ trợ
    - Thư viện RecursiveCharacterTextSplitter hỗ trợ chia nhỏ văn bản đầu vào với các tham số chunk\_size (kích thước đoạn văn, tính theo số lượng kí tự) và chunk\_overlap

(mức độ chồng chéo giữa các đoạn, tính theo số lượng kí tự)

import PyPDFLoader

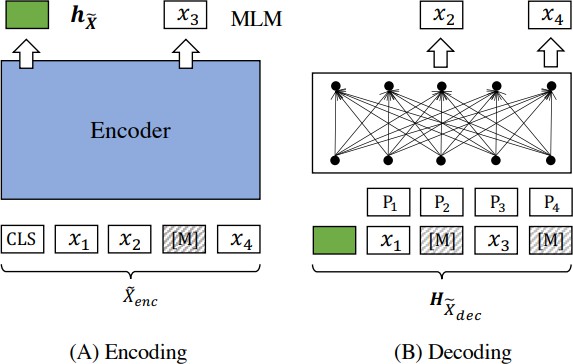
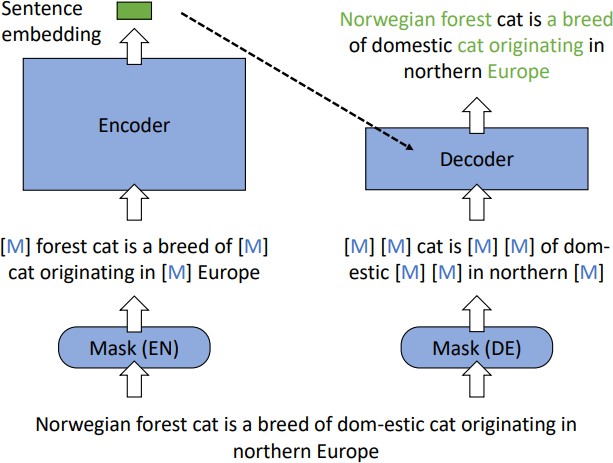
import RecursiveCharacterTextSplitter loader = PyPDFLoader(file\_path) documents = loader.load()

parent\_splitter = RecursiveCharacterTextSplitter(parent\_chunk\_size, parent\_chunk\_overlap)

child\_splitter = RecursiveCharacterTextSplitter(child\_chunk\_size, child\_chunk\_overlap)

### Embedding model

Sử dụng LLM [BAAI/bge-large-en](https://github.com/FlagOpen/FlagEmbedding) tạo vector embedding từ các child-node với số chiều là 1024 và độ dài chuỗi 512. BGE là embedding model được pretrain dựa trên [RetroMAE](https://arxiv.org/abs/2205.12035) (Pre-Training Retrieval-oriented Language Models Via Masked Auto- Encoder) sau đó train trên tập dữ liệu quy mô lớn với phương pháp học tương phản.

*Hình 2-1: Mô hình Encoder – Decoder của mô hình LLM BAAI/bge-large-en*

* Encoding: một câu đầu vào X sẽ được che giấu thông tin ngẫu nhiên (15-30%)

bằng mã thông báo đặc biệt [M] trở thành Xenc chuẩn bị cho giai đoạn mã hóa, phần lớn thông ban đầu vẫn sẽ được giữ nguyên. [CLS] được chọn làm trạng thái ẩn cuối cùng cho sentence embedding. Sử dụng BERT làm bộ encoder với 12 lớp và 768 chiều ẩn giúp nắm bắt ngữ nghĩa chuyên sâu của câu. Đầu vào X qua bộ encoder trở thành sentence embedding hX.

* Decoding: câu đầu vào được che giấu thông tin với tỉ lệ cao hơn (50-70%) cho giai đoạn giải mã. Đầu vào bị giấu thông tin sẽ kết hợp với sentence embedding, dựa vào đó

câu gốc sẽ được bộ giải mã xây dựng lại. Việc kết hợp đầu vào đã che giấu thông tin và sentence embedding được thực hiện theo thứ tự:

HXdec ← [hX , ex1 + p1, ..., exN + pN ]

Trong đó exi biểu thị embedding của xi, pi biểu thị position embedding được thêm vào.

Cuối cùng bộ giải mã được học để xây dựng lại câu gốc X bằng cách tối ưu hàm mất mát sau:

Ldec = ∑𝑥𝑖 ∈ masked 𝐶𝐸 (𝑥𝑖|𝑑𝑒𝑐(𝐻𝑋𝑑𝑒𝑐))

Trong đó CE là hàm cross entroy loss.

### Cài đặt

Sử dụng thư viện HuggingFaceBgeEmbeddings để load model với các tham số model\_kwargs: device – cuda xử lí trên gpu và encode\_kwargs: normalize\_embedding – False

import HuggingFaceBgeEmbeddings model\_name = “BAAI/bge-large-en” model\_kwargs = {“device”: ”cuda”}

encode\_kwargs = {“normalize\_embeddings”: False}

embeddings = HuggingFaceBgeEmbeddings( model\_name = model\_name,

model\_kwargs = model\_kwargs, encode\_kwargs = encode\_kwargs)

### Retriever

Retriever đóng vai trò quan trọng khi quyết định phần lớn chất lượng của câu trả lời dựa vào tính đúng đắn của văn bản được truy vấn. Việc tìm kiếm các văn bản phù hợp nhất cần được xem xét trên nhiều khía cạnh như ngữ nghĩa hay hình thức. Kết hợp sparse retriever và dense retriever để tạo thành ensemble retriever giúp cải thiện hiệu suất truy xuất tốt hơn các thuật toán đơn lẻ vì chúng có thể bổ sung cho nhau điểm mạnh ở từng phương pháp.

* Dense retriever: tập hợp các vector embedding, tính toán độ đo cosine, cho kết quả

tốt trong việc tìm kiếm các tài liệu liên quan dựa trên sự tương đồng về ngữ nghĩa.

* Sparse retriever: tính toán TF-IDF, đơn giản, hiệu quả khi tìm kiếm các tài liệu liên quan dựa trên từ khóa.

Đối với Dense retriever, sử dụng Chroma làm embedding database (hay vector database) giúp hỗ trợ lưu trữ, quản lí và truy vấn các vector embedding được tạo dựng ở bước trước, giúp dễ dàng tìm kiếm các tài liệu phù hợp nhất với câu truy vấn. Có thể sử dụng các độ đo khác nhau như L2 (Euclidean Distance), Inner Product (IP), hay Cosine. Khả năng biểu diễn ngữ nghĩa của các vector embedding sẽ quyết định đến chất lượng truy vấn của Dense retriever.

𝐴.𝐵

∑𝑛

𝐴𝑖𝐵𝑖

Cosine(A,B) =

= 𝑖=1

|𝐴|.|𝐵|

√∑𝑛 𝐴2.√∑𝑛 𝐵2.

𝑖=1 𝑖 𝑖=1 𝑖

Đối với Sparse retriever, sử dụng BM25 ranking để xếp hạng các văn bản dựa trên tần suất của các từ và chuẩn hóa độ dài của văn bản để xác định mức độ liên quan của một đoạn văn bản với một truy vấn nhất định. BM25 tuân theo các quy luật xác suất với giả định rằng các tài liệu có liên quan và không liên quan tuân theo các phân phối thống kê khác nhau.

Score(D,Q) = ∑𝑛

𝐼𝐷𝐹( 𝑞 ) . 𝑓(𝑞𝑖,𝐷).(𝑘1+1)

𝑖=1

𝑖 𝑓(𝑞 ,𝐷)+𝑘 .(1−𝑏+𝑏. |𝐷|

𝑎𝑣𝑔𝑑𝑙

𝑖 1

Trong đó:

IDF = log((N — n + 0.5) / (n + 0.5)) với N là tổng số tài liệu và n là số tài liệu có chứa từ khóa qi, biểu thị tần số nghịch đảo của tài liệu có từ khóa qi

f(qi,D) biểu thị tần suất xuất hiện của từ khóa qi trong tài liệu D

|D| biểu thị độ dài của tài liệu

avgdl là độ dài trung bình của tài liệu

k1 và b là các hằng số giúp điều chỉnh tác động của độ bão hòa tần số của qi và chuẩn hóa độ dài tài liệu

Reranking list: khi kết hợp các phương pháp truy vấn khác nhau cần có một thước đo giúp xếp hạng lại mức độ phù hợp của các văn bản được truy vấn. Sử dụng Reciprocal Rank Fusion score:

RRFscore(d ∈ D) = ∑ 1

𝑟∈R

𝑘+𝑟(𝑑)

Trong đó:

k là hằng số xếp hạng

d là tập kết quả truy vấn của r

r(d) là xếp hạng của d trong truy vấn r

### Cài đặt:

* + - Sử dụng thư viện Chroma giúp lưu trữ và hỗ trợ truy vấn vector embedding.
    - Thư viện ParentDocumentRetriever giúp hỗ trợ phân cấp văn bản để truy vấn theo Dense retriever.
    - Thư viện BM25Retriever truy vấn văn bản theo Sparse retriever.
    - Thư viện EnsembleRetriever cho phép kết hợp các retriever theo trọng số tương

ứng.

import Chroma

import ParentDocumentRetriever import BM25Retriever

import EnsembleRetriever import InMemoryStore

vector\_store = Chroma(collection\_name = ”split\_parents”,embedding\_function = embeddings, collection\_metadata = {“hnsw:space”:”cosine”})

store = InMemoryStore()

Dense\_retriever = ParentDocumentRetriever( vectorstore = vector\_store,

docstore= store,

child\_splitter = child\_splitter, parent\_splitter = parent\_splitter)

Sparse\_retriever = BM25Retriever.from\_documents(docs)

Ensemble\_retriever = EnsembleRetriever(retrievers=[Dense\_retriever,Sparse\_retriever], weights=[0.5,0.5])

### Large Language Model Generation

Sử dụng LLM [zephyr-7b-beta](https://arxiv.org/abs/2310.16944) làm model generation, tạo câu trả lời hoàn chỉnh dựa trên câu hỏi và văn bản được truy vấn. Zephyr là mô hình được fine-tuning dựa trên [Mistral 7b](https://arxiv.org/abs/2310.06825), vượt trội so với các model 7b khác ở hầu hết các benchmarks. Mô hình được xây dựng dựa trên kiến trúc transformer với một số điểm khác biệt cải tiến:

- Sliding window attention: thay cho cơ chế attention trên toàn bộ độ dài chuỗi, với

kích thước window W, trạng thái ẩn ở vị trí i của lớp k (gọi là hi) tiếp nhận thông tin từ tất cả các layer trước đó trong giữa i – W và i. Như vậy hi có thể truy cập đến thông tin từ lớp đầu tiên với khoảng cách lên đến Wxk tokens. Ở lớp cuối cùng, kích thước window W=4096 và phạm vi attention khoảng 131k tokens. Thay đổi trên giúp cải thiện tốc độ của mô hình gấp 2 lần so với cơ chế attention thông thường.

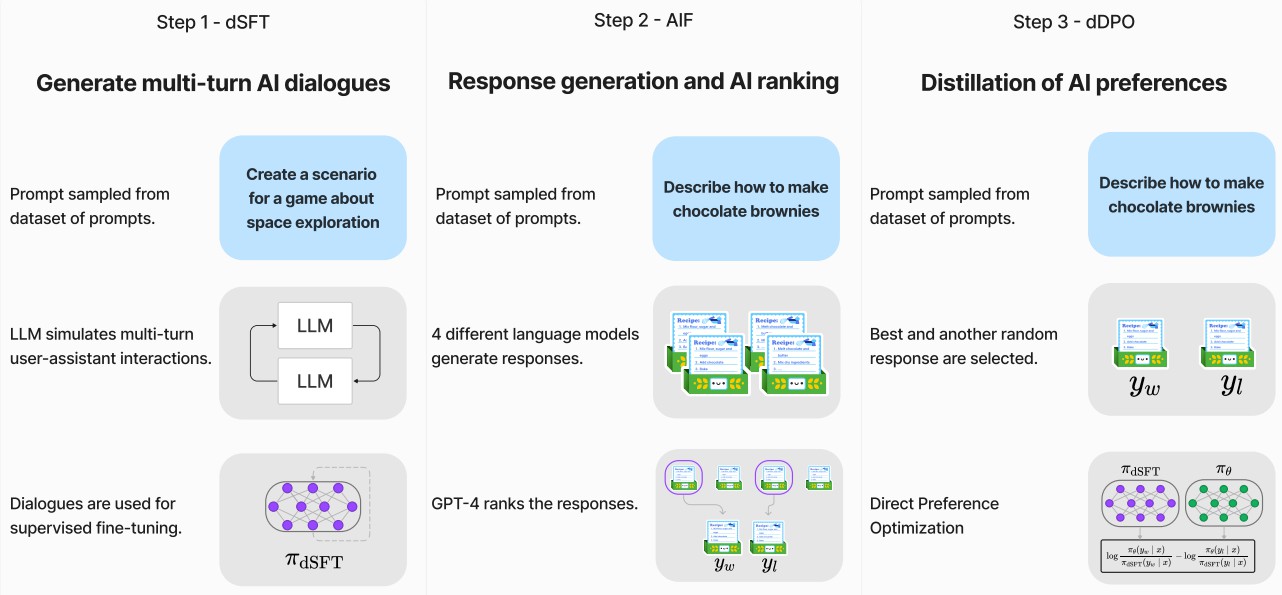
*Bảng 2-1: Tham số cài đặt của mô hình LLM zephyr-7b-beta*

|  |  |
| --- | --- |
| Parameter | Value |
| Dim | 4096 |
| N\_layers | 32 |
| Head\_dim | 128 |
| Hidden\_dim | 14336 |
| N\_heads | 32 |
| N\_kv\_heads | 8 |
| Window\_size | 4096 |
| Context\_len | 8192 |
| Vocab\_size | 32000 |

* Rolling buffer cache: giới hạn không gian attention cố định giúp giới hạn kích thước bộ nhớ đệm. Bộ nhớ đệm có kích thước cố định là W, key và value ở bước thứ i sẽ được lưu trữ ở vị trí i mod W trong bộ đệm. khi i>W, các giá trị trước đó trong bộ nhớ đệm sẽ bị ghi đè và kích thước của bộ đệm sẽ không tang.
* Pre-fill and chunking: thông thường khi tạo sinh một chuỗi ta cần phải dự đoán từng bước một vì mỗi token được sinh ra sẽ điều chỉnh dựa trên các token trước đó. Tuy

nhiên, với prompt được cho trước , có thể điền trước vào bộ nhớ đệm (k,v), nếu prompt có kích thước lớn có thể chia nhỏ ra phù hợp với kích thước bộ đệm. Theo đó ta chỉ cần tính toán attention trên bộ nhớ đệm và trên từng phần chia nhỏ của prompt.

Zephyr được huấn luyện trên bộ UltraChat và tối ưu dựa trên AI Feedback.



*Hình 2-2: Tóm tắt các bước mô hình Zephyr được huấn luyện trên bộ UltraChat và tối ưu dựa trên AI Feedback*

* Bước 1: mô hình ban đầu được huấn luyện để phản hồi prompt của người dùng. Gọi xo … xj là tập các prompt chuẩn bị cho huấn luyện. Với mỗi x, phản hồi ban đầu là y

= πT(·|x ), mô hình sẽ học cách tinh chỉnh bằng cách xây dựng prompt mới x1 = πT(·|x,y ). Cuối cùng ta đạt được tập dữ liệu C = {(x1, y1), . . . ,(xJ , yJ )}

* Bước 2: dựa trên feedback của GPT4 trên model ban đầu so với các model khác

như Claude, Falcon, Llama,.. chọn ra tập phản hồi có điểm số cao nhất và tập ngẫu nhiên có điểm số thấp để cho quá trình học tương phản.

* Bước 3: tối ưu hóa model dựa trên tập dữ liệu tương phản vừa có được.

### Cài đặt:

* + - Sử dụng thư viện LlamaCpp để load model và chạy trên gpu.
    - Thư viện PromptTemplate giúp tạo prompt cho model.

import LlamaCpp

import CallbackManager

import StreamingStdOutCallbackHandler import PromptTemplate

callback\_manager = CallbackManager([StreamingStdOutCallbackHandler()]) n\_gpu\_layer = 50

n\_batch = 512

llm = LlamaCpp( model\_path = model\_path, n\_gpu layers = n\_gpu layers, n\_batch = n\_batch,

callback\_manager = callback\_manager, verbose = True, n\_ctx = 2048)

prompt\_template = """Use the following pieces of information to answer the user's question.

If you don't know the answer, just say that you don't know, don't try to make up an answer.

Context: {context} Question: {question}

Only return the helpful answer below and nothing else. Helpful answer:

"""

prompt = PromptTemplate(template=prompt\_template, input\_variables=['context', 'question'])

## Chương 3: XÂY DỰNG BỘ DỮ LIỆU

Đề mục của chương 3 bao gồm:

* Nguồn dữ liệu
* Cách xây dựng
* Phân tích dữ liệu

Nội dung của chương 2 này sẽ tóm tắt cách nhóm thu thập bộ dữ liệu cho đồ án. Mục đích của bộ dữ liệu này nhằm thực hiện đánh giá hiệu năng của hệ thống mà nhóm xây dựng. Cuối cùng, là phần phân tích dữ liệu của nhóm để thấy tổng quan dữ liệu nhóm sử dụng có số lượng và nội dung như thế nào.

### Nguồn dữ liệu

Mục đích của bộ dữ liệu được nhóm xây dựng là kiểm tra hệ thống có khả năng phản hồi tốt các yêu cầu, truy vấn mà nhóm đặt ra hay không. Để đánh giá, nhóm đã tự xây dựng bộ dữ liệu để kiểm tra hiệu năng mô hình mà nhóm đã xây dựng.

Như domain đã đề cập của đề tài của nhóm, nhóm đã thực hiện lấy các tài liệu có định dạng PDF từ nguồn Internet để xây dựng bộ dữ liệu cho đồ án. Bên cạnh nhu cầu xây dựng hệ thống hỏi đáp từ file PDF, một nhu cầu khác mà nhóm muốn đạt được là hệ thống hỏi đáp có thể đưa ra các câu trả lời mang các kiến thức là những lý thuyết, kiến thức có được qua bài báo khoa học. Do đó, nhóm đã tìm kiếm các bài báo khoa học và sử dụng định dạng tài liệu là PDF và đưa vào xây dựng bộ dữ liệu này.

Dưới đây là tựa đề các bài báo mà nhóm đã sử dụng (link các bài báo được dẫn kèm theo tựa đề bài báo dưới dạng hyperlink):

* [RAGAS: Automated Evaluation of Retrieval Augmented Generation](https://arxiv.org/pdf/2309.15217.pdf%E2%80%8D%E2%80%8D)
* [A Survey on Retrieval-Augmented Text Generation](https://arxiv.org/pdf/2202.01110.pdf)
* [Transformer-based Dual Relation Graph for Multi-label Image Recognition](https://openaccess.thecvf.com/content/ICCV2021/papers/Zhao_Transformer-Based_Dual_Relation_Graph_for_Multi-Label_Image_Recognition_ICCV_2021_paper.pdf)

### Cách xây dựng

Từ các bài báo trên, nhóm đã xây dựng bộ dữ liệu gồm 3 thuộc tính: question, context, ground\_truth. Mục tiêu của bộ dữ liệu này nhằm đưa ra các đánh giá về hiệu quả của hệ thống hỏi đáp thông qua các độ đo được trình bày ở chương 4. Trong đó:

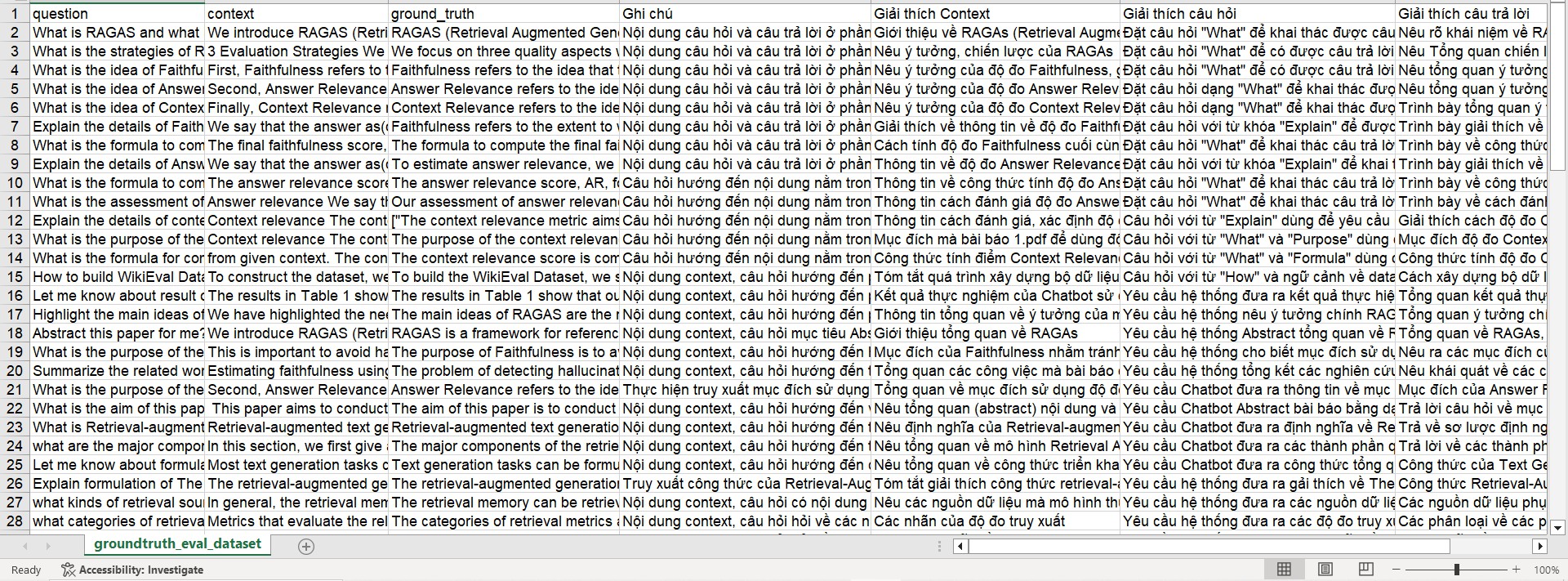
* Thuộc tính Context gồm các câu, đoạn văn được lấy ra từ bài báo nhằm đặt làm ngữ cảnh để đặt ra các mẫu câu hỏi và câu trả lời mẫu để đánh giá. Để tạo Context, nhóm dựa trên các đề mục chính thường có trong bài báo gồm: Abstract, Introduction, Related Work, Experiments, Conclusion, từ đó lấy ra các đoạn ngữ liệu từ từng đề mục này để đưa ra các ngữ cảnh cho các câu hỏi, nằm trong thuộc tính Question.
* Thuộc tính Question sẽ là câu hỏi được đưa ra dựa trên nội dung của Context. Việc đặt câu hỏi sẽ dựa trên cấu trúc gồm từ để hỏi (giới hạn ở các cụm từ “What”, “How”, “Explain”, “Abstract”, “Summarize” và cụm từ chỉ yêu cầu “Let’s me know”, “Explain the details of”) và nội dung cần hỏi. Với ý nghĩa tiếng Việt tương ứng với lần lượt các câu hỏi “Là gì?”, “Như thế nào”, “Hãy giải thích...”, “Phân tách vấn đề...”, “Tóm tắt...” nội dung được hỏi. Bên cạnh đó, khi có Context về chủ đề so sánh, sử dụng cụm từ “Compare” và nội dung của 2 chủ thể cần so sánh hoặc cần nội dung kết luận của bài báo, thì sử dụng mẫu câu “Conclude this paper”. Còn 1 số từ được dùng đặt câu hỏi như: “Limitations....” dùng để chỉ các hạn chế của một vấn đề nào đó trong bài báo; “Effects of...” dùng để hỏi hiệu quả của một phương pháp trong bài báo; “Highlight the idea...” dùng để đưa ra yêu cầu tóm tắt ý tưởng của mô hình trong bài báo hoặc một công thức, độ đo,...
* Thuộc tính grouth\_truth: thuộc tính này có ý nghĩa là câu trả lời chuẩn tương ứng từ nội dung của thuộc tính Context và thuộc tính Question. Nội dung của Grouth truth bên cạnh lấy từ nội dung trả lời trực tiếp tương ứng bên trong bài báo, mà nhóm còn tham khảo câu trả lời từ ChatGPT.

Trong quá trình xây dựng bộ dữ liệu, các thành viên cũng đã ghi nhận và tiến hành đưa ra các giải thích cho các thuộc tính context, question và grouth\_truth. Do đó, trong bộ dữ liệu sẽ có thêm các cột gồm “Chú thích”, “Giải thích Context”, “Giải thích Question” và giải thích “Grouth Truth”. Các cột này không được đưa vào để hệ thống để

xử lý mà chỉ có mục đích diễn giải cụ thể thông tin trên từng dòng dữ liệu trong bộ dữ liệu.Với cột “Chú thích” là cột nhằm định vị được nội dung context, câu hỏi và câu trả lời ở đâu trong văn bản, cột “Giải thích context” có nhiệm vụ giải thích nội dung context là gì, cột “Giải thích Question” có nhiệm vụ giải thích nội dung câu hỏi, “Giải thích Grouth Truth” có nhiệm vụ giải thích nội dung câu trả lời chuẩn.

### Phân tích dữ liệu

Qua quá trình xây dựng, nhóm thu được nhóm các mẫu dữ liệu gồm 60 mẫu dữ liệu và các mẫu dữ liệu này, qua các phép thử khác nhau và độ đo khác nhau ở chương 4 để chứng minh hiệu năng của hệ thống.



*Hình 3-1: Đại diện các mẫu dữ liệu đã được thu thập từ 3 bài báo*

Trong đó, mỗi bài báo nhóm đặt ra 20 mẫu dữ liệu gồm 3 thuộc tính question, context và grouth truth được rút trích từ bài báo này. Trong 20 câu này, các câu hỏi đều được trải đều vào các mục Abstract, Introduction, Approach, Experiments, Conclusion. Thống kê trên bộ dữ liệu, ta có số lượng câu dạng “what” có 32 câu, “Explain” có 4, “How” có 3, “Let me know” có 8, “Hightlight” có 1, “Abstract” có 1, “Summarize” có 2, “Limitations” có 1, “Conclude” có 2, “Effects of” có 2, “Compare” có 2, loại khác 2 câu.

* 1. **Nhận xét bộ dữ liệu**:

Bộ dữ liệu phân bố đều ở các chủ đề, hơi lệch các câu hỏi dạng “What”. Đa phần khai thác các câu hỏi dạng hỏi tường minh dạng “là gì” và câu trả lời nằm cục bộ tại 1 điểm nhất định trong bộ dữ liệu, chưa nhiều câu hỏi đến việc “tóm tắt” toàn bài báo. Bên cạnh đó, số lượng tài liệu PDF nhỏ và số lượng dữ liệu thu được nhỏ.

Lý giải cho số lượng điểm dữ liệu nhỏ: Do các độ đo đánh giá cho hệ thống (trình bày ở chương 4) sử dụng thang đo của mô hình RAG được gọi thông qua API của OPENAI, nên quá trình đánh giá hệ thống khá lâu cho từng mẫu dữ liệu. Theo ghi nhận, một số dữ liệu mất khoảng hơn 1 phút để thu kết quả, có 1 số thì thời gian phản hồi lại khá lâu, gần 4 phút để thu được 4 chỉ số của một điểm dữ liệu. Do đó, việc đánh giá cho từng điểm khá lâu. Thêm nữa, đây là dịch vụ chỉ miễn phí ở 1 ngưỡng request nhất định và khi vượt ngưỡng này sẽ bị tính phí. Nên nhóm đã giới hạn số lượng dữ liệu ở mức 60, đúng với yêu cầu đặt ra của đồ án cũng như phù hợp với tài nguyên nhóm có.

## Chương 4: ĐỘ ĐO ĐÁNH GIÁ – KẾT QUẢ

Chương 4 gồm các đề mục:

* Độ đo đánh giá
* Kết quả thử nghiệm

Nội dung của chương 4 trình bày về khái niệm các độ đo dùng để đánh giá mô hình mà nhóm đã xây dựng. Bên cạnh đó, chương cũng trình bày kết quả thử nghiệm mà nhóm đã thử nghiệm với mô hình nhóm đã xây dựng qua bộ dữ liệu đã xây dựng.

### Độ đo đánh giá

Sử dụng [RAGAS](https://arxiv.org/pdf/2309.15217.pdf): một framework hỗ trợ đánh giá hiệu quả của mô hình RAG dựa trên GPT 3.5 turbo. Các chỉ số đánh giá được xây dựng dựa trên các prompt hướng dẫn GPT tạo các câu trả lời, câu hỏi, phân tích mức độ liên quan một cách chi tiết. Dưới đây là một số prompt của GPT được dùng trong quá trình đánh giá:

- Prompt faithfulness:

*“Given a question and answer, create one or more statements from each sentence in the given answer.*

*question: [question] answer: [answer]*

*Consider the given context and following statements, then determine whether they are supported by the information present in the context. Provide a brief explanation for each statement before arriving at the verdict (Yes/No). Provide a final verdict for each statement in order at the end in the given format. Do not deviate from the specified format.*

*statement: [statement 1] ... statement: [statement n]”*

* Prompt context:

*“Please extract relevant sentences from the provided context that can potentially help answer the following question. If no relevant sentences are found, or if you believe the question cannot be answered from the given context, return the phrase "Insufficient Information". While extracting candidate sentences you’re not allowed to make any changes to sentences from given context.”*

* Prompt answer:

*“Generate a question for the given answer. answer: [answer]”*

### Context Precision

Độ đo thể hiện mức độ thông tin context thực sự liên quan đến câu hỏi. Công thức tính lần lượt thể hiện ở bên dưới:

* + - Context\_precision@k = ∑ precision@k

Tổng số context liên quan đến câu hỏi trong top k kết quả trả về

* + - Precision@k = true positive@k

(true positive@k+false positive@k)

### Context Recall

Độ đo có ý nghĩa thể hiện mức độ đóng góp thông tin của câu trả lời chuẩn vào context. Công thức tính Context\_Recall thể hiện ở bên dưới:

* + - Context\_recall = Số câu trong câu trả lời chuẩn có nội dung đóng góp vào context

Số câu trong câu trả lời chuẩn

### Faithfulness

Độ đo có ý nghĩa thể hiện mức độ câu trả lời dựa trên context. Công thức tính Faithfulness thể hiện ở bên dưới:

* + - Faithfulness = Số nhận định được suy ra dựa trên context

Tổng số nhận định trong câu trả lời chuẩn

### Answer Correctness

Độ đo phản ánh sự tương đồng về mặt hình thức và ngữ nghĩa giữa phản hồi của mô hình và câu trả lời chuẩn với cách tính như sau:

Answer correctness = 0.4 \* độ tương đồng hình thức + 0.6 \* độ tương đồng ngữ nghĩa

### Kết quả thử nghiệm

### Kết quả

Nhóm đã thực hiện thử nghiệm kết hợp trên các bước kết hợp khác nhau, ở đây là thử nghiệm trên các cách truy vấn khác nhau, trên hệ thống hỏi đáp nhóm xây dựng, cụ thể gồm các cách truy vấn:

* + - Based retriever.
    - Ensemble retriever: Thực hiện kết hợp Chroma\_retriever và BM25\_retriever
    - Parent\_child retriever (450): Thực hiện Retrieval dựa trên child chứa các vector embedding và parent chứa context với chunk size là 450
    - Parent\_child retriever (600): Thực hiện Retrieval dựa trên child chứa các vector embedding và parent chứa context với chunk size là 600

*Bảng 4-1: Kết quả thử nghiệm của hệ thống truy vấn dựa trên bộ dữ liệu*

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| METRIC | Context recall | Context precision | Faithfulness | Answer correctness |
| Based retriever | 0.806 | 0.314 | 0.807 | 0.685 |
| Ensemble retriever | 0.881 | 0.351 | 0.888 | 0.722 |
| Parent\_child retriever (450) | 0.839 | 0.398 | 0.893 | 0.725 |
| Parent\_child retriever (600) | 0.866 | 0.345 | 0.876 | 0.741 |

### Nhận xét

* + - Sử dụng các bước cải tiến như phân cấp văn bản, kết hợp retriever giúp cải thiện hiệu quả của mô hình.
    - Xem xét phân chia văn bản đầu vào với độ dài phù hợp, không quá lớn giúp đem lại kết quả cao hơn.
    - Context precision còn thấp chứng tỏ quá trình truy vấn các context còn cần phải cải thiện.
    - Kết quả còn thấp ở các trường hợp câu hỏi dạng abstract, summarize do thông tin cần thiết để trả lời khá dài và nằm rải rác, khó truy vấn, tổng hợp.

## Chương 5: KẾT LUẬN

Đề mục chương này bao gồm:

* + - * Ưu điểm
      * Nhược điểm
      * Hướng phát triển

Chương này thực hiện đưa ra các điểm đã làm được của đồ án qua việc đưa ra các ưu điểm, nhược điểm của hệ thống. Từ đó đưa ra hướng phát triển tiếp theo cho đồ án này.

### Ưu điểm

* + Đáp ứng yêu cầu cơ bản, đưa ra tương đối các câu trả lời một số tri thức của các bài báo khoa học.
  + Context Recall và Faithfulness tương đối cao cho thấy các câu trả lời mà hệ thống trả về có kết quả khá tốt. Answer Correctness có chỉ số khoảng 0.7, chỉ số ở mức khá với

hệ thống truy vấn.

### Nhược điểm

* + Thời gian phản hồi trung bình khá lâu, khi sử dụng ứng dụng tầm 20s trở lên mới có 1 câu trả lời cho câu truy vấn.
  + Các câu trả lời của phần Abstract, Summarize do thông tin cần thiết để trả lời khá dài và nằm rải rác, khó truy vấn, tổng hợp.
  + Context Precision còn thấp cho thấy các bước truy vấn cần thêm cải thiện và thử nghiệm mới từ các mô hình khác.
  + Cần xem xét thêm về độ dài phân chia văn bản đầu vào để mang lại kết quả truy vấn tốt hơn.

### Hướng phát triển

* + Sử dụng các bước cải tiến như phân tích và chia nhỏ câu hỏi, kết hợp thêm nhiều

retriever, tạo thêm meta-data, thay đổi prompt cho LLMs generation giúp cải thiện hiệu quả của mô hình.

* + Xây dựng giao diện riêng cho hệ thống nhằm đảm bảo tính tiện dụng và tính kế thừa, có khả năng phát triển thêm tính năng mới cho hệ thống.

## Chương 6: TÀI LIỆU THAM KHẢO

[[2310.06825] Mistral 7B (arxiv.org)](https://arxiv.org/abs/2310.06825)

[[2310.16944] Zephyr: Direct Distillation of LM Alignment (arxiv.org)](https://arxiv.org/abs/2310.16944)

[[2309.07597] C-Pack: Packaged Resources To Advance General Chinese Embedding](https://arxiv.org/abs/2309.07597) [(arxiv.org)](https://arxiv.org/abs/2309.07597)

[[2205.12035] RetroMAE: Pre-Training Retrieval-oriented Language Models Via Masked](https://arxiv.org/abs/2205.12035) [Auto-Encoder (arxiv.org)](https://arxiv.org/abs/2205.12035)

[BAAI/bge-large-en · Hugging Face](https://huggingface.co/BAAI/bge-large-en) [HuggingFaceH4/zephyr-7b-beta · Hugging Face](https://huggingface.co/HuggingFaceH4/zephyr-7b-beta)

[GitHub - abetlen/llama-cpp-python: Python bindings for llama.cpp](https://github.com/abetlen/llama-cpp-python)

[GitHub - explodinggradients/ragas: Evaluation framework for your Retrieval Augmented](https://github.com/explodinggradients/ragas) [Generation (RAG) pipelines](https://github.com/explodinggradients/ragas)

## Phụ lục

1. Bảng phân chia công việc

*Bảng phụ lục 1: Bảng phân công công việc*

|  |  |  |  |
| --- | --- | --- | --- |
| Nhiệm vụ Thành viên | Châu Thiên Long | Nguyễn Thái Thành Long | Tăng Minh Hiển |
| Xác định đề tài | x | x | x |
| Tìm hiểu lí thuyết | x | x | x |
| Cài đặt | x |  |  |
| Xây dựng dữ liệu (ngữ  liệu) | x | x | x |
| Kiểm tra, đánh giá | x | x | x |
| Cải tiến | x |  |  |
| Viết báo cáo | x | x | x |