Tóm tắt

Xương sông của Chatbot hiện nay - các mô hình ngôn ngữ lớn (Large Language Models - LLMs) - gần đây đã chứng minh khả năng hiểu ngôn ngữ đáng kể và hội thoại gần giống con người. Tuy nhiên, việc chỉ dựa vào LLMs cho các tác vụ hỏi-đáp là không đủ vì chúng bị giới hạn bởi dữ liệu huấn luyện và dễ tạo ra thông tin không chính xác, đặc biệt là trong các lĩnh vực cụ thể. Các kỹ thuật tạo nội dung tăng cường truy xuất (Retrieval-Augmented Generation - RAG) nhằm giải quyết những hạn chế đó bằng cách đưa vào các nguồn tri thức bên ngoài, cải thiện độ chính xác của câu trả lời do LLM tạo ra.

Bài khóa luận này phân tích sơ bộ điểm mạnh và điểm yếu của các hệ thống RAG sử dụng cơ sở dữ liệu vector so với cơ sở dữ liệu đồ thị. Kết quả cho thấy, các hệ thống RAG dựa trên vector vượt trội trong việc trả lời các câu hỏi chung và phù hợp hơn với các LLM có khả năng xử lý quy tắc phức tạp hạn chế. Ngược lại, các hệ thống dựa trên đồ thị hiệu quả hơn với các câu hỏi phức tạp đòi hỏi dữ liệu từ nhiều tài liệu. Từ đó đề xuất phương pháp kết hợp cả hai cơ sở dữ liệu để cải thiện hiệu suất của hệ thống RAG.

Table of Contents

Bảng viết tắt thuật ngữ

ACL	Association for Computational Linguistics
BART	Bidirectional Auto Regressive Transformer
BERT	Bidirectional Encoder Representations from Transformers
GB	Gradient Boosting
GPT	Generative Pre-training Transformer
LCS	Longest Common Subsequence
LLM	Large Language Model
LSTM	Long Short-Term Memory
MAE	Mean Absolute Error
MAS	Multi-Answer Summarization
MEDIQA	Medical Question-Answering
MLP	Multi-Layer Perception
MMR	Maximal Marginal Relevance
MSE	Mean Square Error
NER	Named-Entities Recognition
NLP	Natural Language Processing - Xử lý ngôn ngữ tự nhiên
PCA	Principle Component Analysis

PEGASUS	Pre-training with Extracted Gap-sentences for Abstractive Summarization
POS	Part-Of-Speech
QA	Question-Answering
RoBERTa	Robustly optimized BERT pre-training approach
ROUGE	Recall-Oriented Understudy for Gisting Evaluation

Chương 1

Đặt vấn đề

1.1 Bối cảnh

Mặc dù việc tìm cách tạo ra một thứ có thể hiểu và giao tiếp với người tạo ra nó đã ăn sâu vào lịch sử loài người, Alan Turing được cho là người đầu tiên hình thành ý tưởng về chatbot vào năm 1950, khi ông đặt câu hỏi: "Máy móc có thể suy nghĩ không?". Mô tả của Turing về hành vi của một cỗ máy thông minh gợi lên khái niệm chatbot mà chúng ta thường hiểu ngày nay.

Chatbot đã phát triển cùng với sự gia tăng dần khả năng tính toán và những tiến bộ trong các công cụ và kỹ thuật xử lý ngôn ngữ tự nhiên (NLP). Việc triển khai chatbot đầu tiên, dựa nhiều vào các quy tắc ngôn ngữ và kỹ thuật khớp mẫu, đã được thực hiện vào năm 1966 với sự ra đời của ELIZA. Chatbot này có thể giao tiếp với người dùng thông qua chương trình khớp từ khóa, tìm kiếm các quy tắc chuyển đổi thích hợp để tái cấu trúc đầu vào và đưa ra phản hồi, tức là câu trả lời cho người dùng. ELIZA là một hệ thống mang tính bước ngoặt, khuyến khích nghiên cứu sâu hơn trong lĩnh vực này. Tuy nhiên, phạm vi hiểu biết của ELIZA bị giới hạn vì nó phụ thuộc rất ít vào việc nhận diện ngữ cảnh và các quy tắc khớp mẫu thường không linh hoạt để triển khai trong các lĩnh vực mới.

Một bước tiến quan trọng trong sự phát triển của chatbot vào những năm 1980 là việc sử dụng trí tuệ nhân tạo (AI). A.L.I.C.E. (Artificial Intelligent Internet Computer Entity) dựa trên Ngôn ngữ Đánh dấu Trí tuệ Nhân tạo (AIML), một phần mở rộng của XML. AIML được phát triển đặc biệt để cho phép thêm kiến thức về mẫu hội thoại vào phần mềm của A.L.I.C.E., giúp mở rộng cơ sở dữ liệu kiến thức. Các đối tượng dữ liệu trong AIML bao gồm các chủ đề (topics) và danh mục (categories). Danh mục là đơn vị kiến thức cơ bản, bao gồm các quy tắc để khớp đầu vào của người dùng với đầu ra của chatbot. Đầu vào của người dùng được biểu diễn dưới dạng mẫu quy tắc, trong khi đầu ra của chatbot được xác định bằng mẫu quy tắc trong cơ sở kiến thức của A.L.I.C.E. Việc bổ sung các đối tượng dữ liệu mới vào AIML đại diện cho một cải tiến đáng kể so với các hệ thống khớp mẫu trước đây vì cơ sở dữ liệu kiến thức dễ dàng mở

rộng. Hơn nữa, ChatScript, kế thừa từ AIML, cũng là công nghệ nền tảng đằng sau các chatbot đoạt giải Loebner. Ý tưởng chính của công nghệ này là khớp các đầu vào văn bản từ người dùng với một chủ đề, và mỗi chủ đề sẽ có các quy tắc cụ thể để tạo ra phản hồi. ChatScript đã mở ra một kỷ nguyên mới trong sự phát triển công nghệ chatbot, bắt đầu chuyển trọng tâm sang phân tích ngữ nghĩa và hiểu biết.

Hạn chế chính của việc dựa vào các quy tắc và khớp mẫu trong chatbot là chúng phụ thuộc vào lĩnh vực, khiến chúng trở nên kém linh hoạt vì phải dựa vào các quy tắc được viết thủ công cho các lĩnh vực cụ thể. Với những tiến bộ gần đây trong các kỹ thuật học máy và công cụ xử lý ngôn ngữ tự nhiên, kết hợp với khả năng tính toán mạnh mẽ, các khung công việc và thuật toán mới đã được tạo ra để triển khai các chatbot "nâng cao" mà không phụ thuộc vào quy tắc và kỹ thuật khớp mẫu, đồng thời khuyến khích việc sử dụng chatbot trong thương mại. Việc áp dụng các thuật toán học máy vào chatbot đã được nghiên cứu, và những kiến trúc chatbot mới đã xuất hiện.

Ứng dụng của chatbot đã mở rộng với sự xuất hiện của các thuật toán học sâu (Deep Learning). Một trong những ứng dụng mới và thú vị nhất là sự phát triển của các trợ lý cá nhân thông minh (như Alexa của Amazon, Siri của Apple, Google Assistant của Google, Cortana của Microsoft, và Watson của IBM). Các trợ lý cá nhân thông minh hoặc tác nhân hội thoại này thường có thể giao tiếp với người dùng thông qua giọng nói và thường được tích hợp trong điện thoại thông minh, đồng hồ thông minh, loa và màn hình gia đình chuyên dụng, thậm chí cả xe hơi. Ví dụ, khi người dùng nói một từ hoặc cụm từ đánh thức, thiết bị sẽ kích hoạt và trợ lý cá nhân thông minh bắt đầu lắng nghe. Thông qua việc Hiểu ngôn ngữ tự nhiên (NLU), trợ lý có thể hiểu các lệnh và trả lời yêu cầu của người dùng, thường bằng cách cung cấp thông tin (ví dụ: "Alexa, thời tiết hôm nay ở Los Angeles thế nào?" – "Ở Los Angeles, trời nắng và nhiệt độ là 75°F"), hoặc thực hiện các nhiệm vụ (ví dụ: "Ok Google, phát danh sách nhạc buổi sáng của tôi trên Spotify"). Tuy nhiên, nhiệm vụ hiểu ngôn ngữ của con người vẫn là một thách thức lớn vì sự đa dạng về giọng điệu, vùng miền, địa phương, và thậm chí là cách nói cá nhân.

Tất cả các trợ lý cá nhân thông minh đều có các đặc điểm cốt lõi giống nhau về công nghệ sử dụng, giao diện người dùng và chức năng. Tuy nhiên, một số chatbot có tính cách phát triển hơn, và những chatbot phát triển nhất có thể cung cấp giải trí chứ không chỉ hỗ trợ các công việc hàng ngày; những chatbot này được gọi là chatbot xã hội. Một ví dụ thú vị về chatbot xã hội là XiaoIce của Microsoft. XiaoIce được thiết kế để trở thành một người bạn đồng hành lâu dài với người dùng, và để đạt được mức độ gắn kết cao, nó được xây dựng với tính cách, Chỉ số thông minh (IQ) và Chỉ số cảm xúc (EQ). Các khả năng IQ bao gồm mô hình hóa kiến thức và trí nhớ, hiểu hình ảnh và ngôn ngữ tự nhiên, lý luận, sáng tạo, và dự đoán. Đây là những thành phần quan trọng trong việc phát triển khả năng hội thoại, đáp ứng các nhu cầu cụ thể của người dùng và hỗ trợ họ. Khả năng quan trọng và phức tạp nhất là Core Chat, cho phép trò chuyện dài và trong các lĩnh vực mở với người dùng. Đồng cảm và kỹ năng xã hội là hai thành phần quan

trọng của EQ. Công cụ hội thoại của XiaoIce sử dụng một trình quản lý hội thoại để theo dõi trạng thái của cuộc hội thoại và lựa chọn giữa Core Chat (thành phần tạo hội thoại mở) hoặc kỹ năng hội thoại để tạo phản hồi. Do đó, mô hình tích hợp cả khả năng truy xuất thông tin và khả năng tạo hội thoại.

Trong bối cảnh các mô hình ngôn ngữ lớn (LLMs) ngày càng phát triển, việc áp dụng chúng vào trong giáo dục càng trở nên quan trọng và trở thành một trong các lĩnh vực quan trọng để nghiên cứu. Mặc dù khả năng của các Chatbot sử dụng LLMs như GPT, Gemini, Claude chứng tỏ được năng lực mạnh mẽ trong nhiệm vụ hỏi đáp (Q&A downstream tasks) nhưng vẫn còn một số vấn đề nổi cộm còn tồn đọng khi vẫn phải đối mặt với những hạn chế cố hữu, chẳng hạn như ảo giác và các kiến thức được học đã lỗi thời cũng như chưa tập trung vào trong các miền tri thức cụ thể. Với khả năng mạnh mẽ của RAG trong việc cung cấp thông tin bổ trợ mới nhất và hữu ích, các Chatbot sử dụng Mô hình ngôn ngữ lớn tăng cường truy xuất (RA-LLM) đã xuất hiện để khai thác các cơ sở kiến thức bên ngoài và có thẩm quyền, thay vì chỉ dựa vào kiến thức bên trong của mô hình, nhằm tăng cường chất lượng tạo ra LLM.

Mặc dù vậy RA-LLM vẫn còn có một số hạn chế như phụ thuộc vào các mô hình nhúng văn bản (Embedding models), độ chính xác của việc trích xuất thông tin, và khả năng trả lời các câu hỏi cần suy luận. Bởi vậy trong bài khóa luận này, tôi đề xuất một phương pháp cải tiến RAG trong cho nhiệm vụ hỏi đáp trong một lĩnh vực cụ thể với các tài liệu đã được chuẩn hóa bằng cách kết hợp truy xuất vector (vector retrieval) với truy xuất đồ thị tri thức (KGs retrieval) để nâng cao khả năng trả lời chính xác của Chatbot.

1.2 Mục đích và câu hỏi nghiên cứu

Bài khóa luận này nhằm mục tiêu trả lời một số câu hỏi nghiên cứu liên quan đến các phương pháp RAG (Retrieval-Augmented Generation) khác nhau, tập trung đặc biệt vào sự khác biệt trong cấu trúc cơ sở dữ liệu và ảnh hưởng của chúng đến việc tăng cường các mô hình ngôn ngữ lớn (LLMs) trong các nhiệm vụ trả lời câu hỏi. Bằng cách trả lời bốn câu hỏi nghiên cứu mà chúng tôi đã xây dựng, chúng tôi tìm cách xác định và so sánh những lợi thế của các hệ thống RAG dựa trên embedding và dựa trên đồ thị trong việc hỗ trợ các mô hình LLM hiện đại, đặc biệt trong lĩnh vực giáo dục, cụ thể trong bài khóa luận này này là môn Lịch Sử.

Kết quả của nghiên cứu này cung cấp thêm những hiểu biết về các hệ thống RAG khác nhau và đưa ra hướng dẫn về cách xây dựng và đánh giá các hệ thống này cho lĩnh vực giáo dục trên các tập dữ liệu khác nhau. Các câu hỏi nghiên cứu bao gồm:

1. Sự khác nhau giữa khả năng của cơ sở dữ liệu vector (Vector database) và đồ thị tri thức (Knowledge Graph) trong việc truy xuất, cung cấp thông tin cho mô hình ngôn ngữ lớn?

   Câu hỏi này nhằm khám phá những ưu và nhược điểm của từng cấu trúc cơ sở dữ liệu

trong việc tăng cường các mô hình ngôn ngữ. Trong khi cơ sở dữ liệu vectơ có khả năng xử lý dữ liệu phi cấu trúc, chúng tôi giả định rằng chúng không tận dụng được hết tiềm năng của tập dữ liệu, thường cung cấp thông tin rộng cho các câu hỏi tổng quát. Ngoài ra, chúng tôi giả định rằng cấu trúc phong phú và logic của cơ sở dữ liệu đồ thị cải thiện khả năng của LLM trong việc trả lời các câu hỏi phức tạp, yêu cầu nhiều quy tắc.

2. Hiện nay có những phương pháp truy xuất nổi trội và hiệu quả nào trong việc sử dụng cơ sở dữ liệu vector và đồ thị tri thức?

   Bài khóa luận này cung cấp một cái nhìn tổng quan về các phương pháp truy xuất hiện có trong các hệ thống RAG sử dụng cơ sở dữ liệu vectơ hoặc đồ thị. Những thông tin thu được bao gồm các phương pháp cơ bản đóng vai trò làm tiêu chuẩn, từ đó có thể điều chỉnh dựa trên các tập dữ liệu và ứng dụng cụ thể.

3. Làm thế nào để kết hợp hai phương pháp cơ sở dữ liệu vector với cơ sở dữ liệu đồ thị để cải thiện độ chính xác của truy xuất dữ liệu?

   Câu hỏi này đề cập đến thách thức trong việc kết hợp các cấu trúc cơ sở dữ liệu khác nhau và nhấn mạnh sự cần thiết phải đảm bảo tính đồng nhất trong thông tin chứa trong cơ sở dữ liệu vector và đồ thị.

4. Làm thế nào để đánh giá chất lượng trả lời câu hỏi một cách có hệ thống trên các hệ thống RAG dựa trên LLM khác nhau?

   Mục tiêu là phát triển một framework đánh giá mạnh mẽ để đo lường hiệu suất của các triển khai RAG khác nhau, tập trung chủ yếu vào khả năng của chúng đối với các loại câu hỏi và mức độ phức tạp khác nhau. Hơn nữa, chúng tôi sẽ cung cấp các chỉ số hiệu quả để so sánh các câu trả lời được tạo bởi LLM với dữ liệu thực tế.

1.3 Đề xuất và phương pháp

Phương án đề xuất tập trung vào việc kết hợp KGs và Vector database cho việc suy luận và trả lời các câu hỏi để nâng cao RAG nhằm giúp cho Chatbot trả lời chính xác hơn. Trong khóa luận này sẽ tập trung vào nâng cao khả năng trả lời các câu hỏi trắc nghiệm cũng như các câu hỏi tìm kiếm thông tin.

Quy trình thực hiện khóa luận tốt nghiệp của tôi như sau:

Dựa trên các nghiên cứu về phương pháp hiện có, một quá trình tìm hiểu và khám phá kiến thức lặp đi lặp lại sẽ được bắt đầu để trả lời các câu hỏi nghiên cứu được đưa ra. Quy trình này bao gồm việc xác định các tiêu chí đánh giá chất lượng của việc trích xuất các tài liệu liên quan, việc kết hợp sử dụng đồ thị tri thức với cơ sở dữ liệu vector., triển khai các thuật toán và chỉ số cần thiết, cũng như tối ưu các phương pháp trích xuất để giải quyết nhiệm vụ được giao một cách tối ưu.

Điều quan trọng cần lưu ý là quy trình này mang tính chất lặp lại, vì các tiêu chí và thuộc tính, cũng như tác động của chúng đến chất lượng trích xuất, tìm kiếm văn bản và khả năng suy luận, tốc độ xử lý, sẽ được xác định thông qua việc đánh giá kết quả bằng cách sử dụng vào việc trả lời các câu hỏi, tìm kiếm thông tin thực tế. Bộ dữ liệu sử dụng sẽ bao gồm bộ sách giáo khoa, các đề thi tốt nghiệp THPT quốc gia.

Hơn nữa, trong suốt quá trình lặp lại này, các phương pháp và thuật toán sử dụng sẽ được liên tục thay đổi và tối ưu hóa để đạt được kết quả tốt nhất có thể. Cuối cùng, quy trình và kết quả sẽ được xem xét, đánh giá một cách kỹ lưỡng và so sánh với nhau.

1.4 Cấu trúc của khóa luận

Chương 2 tiếp theo sẽ đề cập đến nền tảng lý thuyết liên quan đến công việc này, tập trung vào quy trình khám phá tri thức, các phương pháp học máy, trích xuất thông tin, các mô hình ngôn ngữ lớn. Sau khi mô tả phương pháp đã được áp dụng trong chương 3, dữ liệu thực nghiệm và kết quả tương ứng của các thí nghiệm sẽ được trình bày trong chương 4. Chương 5 sẽ thảo luận về kết quả, đánh giá một cách phê phán các phương pháp và cách tiếp cận đã thực hiện, cũng như xem xét tính hợp lệ và độ tin cậy của các kết quả được trình bày. Cuối cùng, chương 6 sẽ tóm tắt toàn bộ công việc và đưa ra triển vọng cho các nghiên cứu và mở rộng trong tương lai về chủ đề này.

Chương 2

Cơ sở lý thuyết

2.1 Chatbot

2.1.1 Định nghĩa

Trong tài liệu khoa học, chatbot thường được gọi chính thức là tác nhân hội thoại (conversational agents). Trong ngữ cảnh của bài khóa luận này, các thuật ngữ chatbot/tác nhân hội thoại sẽ được sử dụng thay thế cho nhau.

Nguyên tắc cơ bản của mọi chatbot là tương tác với người dùng (trong hầu hết các trường hợp) thông qua tin nhắn văn bản và hành xử như thể nó có khả năng hiểu cuộc trò chuyện và trả lời người dùng một cách phù hợp. Nguồn gốc của việc máy tính giao tiếp với con người lâu đời như chính lĩnh vực Khoa học Máy tính. Thật vậy, Alan Turing đã định nghĩa một bài kiểm tra đơn giản, hiện được gọi là bài kiểm tra Turing, vào năm 1950, trong đó một giám khảo là con người sẽ dự đoán xem thực thể mà họ đang giao tiếp qua tin nhắn có phải là một chương trình máy tính hay không [1]. Tuy nhiên, tham vọng của bài kiểm tra này lớn hơn nhiều so với trường hợp sử dụng thông thường của chatbot; điểm khác biệt chính là kiến thức chuyên môn của chatbot thường hẹp, trong khi bài kiểm tra Turing giả định rằng một người có thể trò chuyện về bất kỳ chủ đề nào với tác nhân. Điều này giúp ích trong việc thiết kế các tác nhân hội thoại vì chúng không cần phải có một kiến thức chuyên môn (có thể là) vô hạn mà có thể tập trung vào các chủ đề rất cụ thể, chẳng hạn như giúp người dùng đặt bàn tại một nhà hàng.

Hơn nữa, một giả định chung khác mà các nhà thiết kế chatbot thường lưu ý là người dùng thường có một mục tiêu mà họ muốn đạt được vào cuối cuộc trò chuyện khi họ bắt đầu tương tác với chatbot. Điều này sau đó ảnh hưởng đến luồng và chủ đề của cuộc trò chuyện để đạt được mục tiêu đã chọn. Các nhà phát triển có thể khai thác điều này vì các mô hình hành vi nhất định có xu hướng xuất hiện như một kết quả.

Do đó, định nghĩa về chatbot được sử dụng trong tài liệu này là một chương trình máy tính

giao tiếp bằng văn bản theo cách giống con người và cung cấp dịch vụ cho người dùng nhằm hoàn thành một mục tiêu được xác định rõ ràng.

2.1.2 Góc nhìn tổng quan về tác nhân đối thoại

Về mặt khái niệm, một chatbot được cấu thành từ nhiều thành phần hoạt động đồng bộ nhằm đạt được một mục tiêu chung. Hình 2.1 tóm tắt mối quan hệ giữa các phần của một tác nhân hội thoại dưới dạng trực quan.

Khi nhận được một tin nhắn mới, bước đầu tiên là xử lý nó thông qua mô-đun nhận diện ngôn ngữ. Quá trình này có thể đơn giản như việc truy xuất một thẻ (tag) hoặc phức tạp hơn với các phương pháp thống kê. Tin nhắn mới, cùng với thông tin ngôn ngữ và các tin nhắn trước đó được lấy từ hệ thống backend, sẽ được đưa vào mô-đun phân loại ý định. Vai trò của mô-đun này là suy luận ý định mà người dùng muốn truyền đạt.

Tiếp theo, metadata của tin nhắn, ý định suy luận được, và các thông tin khác từ backend sẽ được sử dụng để xác định một hành động hoặc chuỗi hành động phù hợp. Ví dụ, chatbot có thể quyết định trả lời bằng một câu hỏi nếu ý định của người dùng chưa rõ ràng, hoặc kích hoạt lại tài khoản người dùng nếu ý định của họ là yêu cầu khôi phục tài khoản.

Cuối cùng, mô-đun xử lý hành động nhận đầu vào là hành động được xác định và thực hiện hành động đó một cách phù hợp. Việc thiết kế theo cách này là hữu ích vì một hành động có thể được thực thi theo nhiều cách khác nhau tùy thuộc vào môi trường hoạt động của chatbot. Cách thực hiện hành động có thể hoàn toàn khác biệt nếu chatbot hoạt động trên nền tảng Messenger so với trên website của một công ty.

2.1.3 Phân loại ý định

Khi nhận được một tin nhắn mới, tác nhân hội thoại cần có khả năng xác định mục tiêu mà người dùng đang cố gắng đạt được. Điều này thường được mô hình hóa như một bài toán phân loại đa lớp, trong đó các nhãn đại diện cho tên của các ý định khả thi từ phía người dùng. Các kỹ thuật để giải quyết vấn đề này dao động từ phương pháp trích xuất từ khóa đơn giản đến suy luận Bayes nhằm xác định yêu cầu của người dùng dựa trên nhiều tin nhắn.

Các mạng mô hình ngôn ngữ lớn (Large Language Model (LLM)) đã được chứng minh là hoạt động hiệu quả trong lĩnh vực này trước đây [2]. Chúng là nền tảng, xương sống của bài khóa luận này

2.1.4 Quản lý tri thức

Một tác nhân thông minh chỉ có thể hoạt động hiệu quả trong giới hạn nếu thiếu kiến thức. Lĩnh vực cho phép máy tính xử lý kiến thức đã có những tiến bộ đáng kể trong thập niên 1980, với

tên gọi kỹ thuật tri thức (knowledge engineering). Các kỹ thuật ban đầu thường sử dụng một bộ suy luận (inference engine) để xử lý các dữ kiện và suy ra kiến thức mới bằng cách sử dụng logic bậc nhất và bậc hai. Đây là một cách để suy diễn câu trả lời cho những câu hỏi không đầy đủ và thường dễ dàng được chuyển thành các lệnh gọi API.

Đối với các tác nhân đối thoại (conversational agents), kỹ thuật tri thức rất hữu ích, chẳng hạn để trả lời các câu hỏi cơ bản về các sự kiện tổng quát. Siri và Amazon Alexa sử dụng các phương pháp suy luận tri thức nội bộ để truy xuất thông tin từ web và các nguồn khác (ví dụ, khi hỏi Alexa về các chuyến tàu khởi hành từ Brussels hôm nay, hệ thống có thể kích hoạt một phép suy luận nội bộ dưới dạng train(brussels, D, today), trong đó D là một biến ẩn danh đại diện cho điểm đến).

Ngày nay, việc quản lý tri thức chủ yếu được thực hiện thông qua các lệnh gọi API và các truy vấn cơ sở dữ liệu tối ưu. Mặc dù vậy, các phương pháp đặc biệt hơn lấy cảm hứng từ các ontology có cấu trúc đồ thị đôi khi vẫn được sử dụng trong các cơ sở tri thức [3].

2.1.5 Đánh giá hiệu suất

Một lĩnh vực cần cải thiện trong lĩnh vực tác nhân hội thoại là việc đánh giá hiệu suất và các chỉ số được sử dụng để định lượng chất lượng hành vi của chatbot. Trong nghiên cứu "How NOT To Evaluate Your Dialogue System: An Empirical Study of Unsupervised Evaluation Metrics for Dialogue Response Generation", Liu và cộng sự chỉ ra rằng các chỉ số chuẩn trong lĩnh vực này, chẳng hạn như điểm số BLEU [4] và ROUGE [5], thường không tương quan với đánh giá trực quan của con người [6].

Vấn đề cốt lõi nằm ở chỗ ngôn ngữ và sự trôi chảy vốn mang tính chủ quan, do đó rất khó để định lượng chính xác, tương tự như bất kỳ thuộc tính mang tính chủ quan nào khác của một hệ thống.

2.2 Mô hình ngôn ngữ lớn

Các mô hình ngôn ngữ tìm cách giải quyết nhiệm vụ dự đoán từ tiếp theo dựa trên một chuỗi các từ đứng trước đó [7]. Nhiệm vụ này tạo thành cốt lõi của việc hiểu ngôn ngữ và đặt nền tảng cho các nhiệm vụ NLP cụ thể hơn. Nghiên cứu về các mô hình ngôn ngữ bắt đầu từ những năm 1980 với sự phát triển của các Mô hình Ngôn ngữ Thống kê (Statistical Language Models - SLM) [8]. Những mô hình này dựa trên các phương pháp xác suất n-gram, được thiết kế để tính toán khả năng xuất hiện của từ tiếp theo trong một câu chưa hoàn chỉnh.

Bước tiến đáng kể tiếp theo trong các mô hình ngôn ngữ đến từ việc giới thiệu mạng Nơ-ron nhân tạo (Neural Networks) và sự phát triển của các Mô hình Ngôn ngữ Dựa trên mạng Nơ-ron (Neural-based Language Models - NLM). NLM mở ra cơ hội cho các nghiên cứu NLP

sâu hơn bằng cách chuyển từ các biểu diễn n-gram thưa thớt sang các biểu diễn vectơ dày đặc có kích thước thấp hơn của văn bản [9]. Những đổi mới như Word2Vec của Mikolov, K. Chen, Corrado, và Dean [10] và GloVe của Pennington, Socher, và Manning [11] đã giúp thúc đẩy lĩnh vực này bằng cách giới thiệu các kỹ thuật hiệu quả để nắm bắt các biểu diễn ngữ nghĩa của từ, cải thiện khả năng hiểu sự tương đồng và ngữ cảnh của từ. Khi sức mạnh tính toán trở nên mạnh mẽ và dễ tiếp cận hơn, nghiên cứu về Mạng Nơ-ron tiếp tục phát triển.

Các Mô hình Ngôn ngữ Lớn (Large Language Models - LLM), thường ám chỉ các mô hình dựa trên kiến trúc Transformer được giới thiệu bởi Vaswani, Shazeer, Parmar, và cộng sự [12], đại diện cho một cột mốc khác trong NLP. Các mô hình dựa trên bộ mã hóa (Encoder) như BERT [13] sử dụng Transformers để tạo ra các biểu diễn từ nhạy ngữ cảnh bằng cách xem xét toàn bộ ngữ cảnh của câu. Mặt khác, các mô hình dựa trên bộ giải mã (Decoder) như GPT [1], Llama [14], và PaLM [15] xử lý cốt lõi của các mô hình ngôn ngữ bằng cách cải thiện nhiệm vụ dự đoán từ tiếp theo dựa trên một tập hợp các từ trước đó. Những mô hình này được tinh chỉnh thêm để tạo ra các chuỗi văn bản mạch lạc khi nhận được một gợi ý.

Các mô hình Seq2Seq như BART [16] và T5 [17] bao gồm cả thành phần mã hóa và giải mã. Khác với các mô hình chỉ sử dụng giải mã, vốn xử lý đầu vào theo hướng đơn chiều, các mô hình mã hóa-giải mã mã hóa đầu vào bằng các cơ chế chú ý hai chiều để hiểu rõ ngữ cảnh. Sau đó, biểu diễn vectơ được đưa vào bộ giải mã để tạo văn bản, xử lý các nhiệm vụ như tóm tắt, dịch thuật, và nhiều nhiệm vụ tạo ngôn ngữ phức tạp khác.

Mặc dù đã đạt được nhiều tiến bộ, các LLM tiêu tốn nhiều tài nguyên và đòi hỏi sức mạnh tính toán lớn để huấn luyện, điều này làm cho việc tích hợp và cập nhật dữ liệu thời gian thực trở nên không khả thi. Hơn nữa, vì thông tin được chứa trong các tham số tĩnh, các LLM không đáng tin cậy trong việc duy trì độ chính xác về mặt thông tin. Những hạn chế này đã dẫn đến các nghiên cứu sâu hơn, bao gồm RAG, vốn kết hợp thông tin liên quan một cách hiệu quả trong quá trình tạo nội dung [2]. Mục tiêu chính của RAG là cải thiện phản hồi của LLM, đảm bảo chúng vừa chính xác về mặt thông tin vừa có căn cứ bằng cách tận dụng dữ liệu được chọn lọc.

2.3 Truy xuất thông tin

Trước khi được sử dụng để tăng cường LLMs, Information Retrieval (IR) đã được giới thiệu nhằm đơn giản hóa việc điều hướng các cơ sở dữ liệu lớn bằng cách cho phép người dùng truy cập thông tin thông qua các truy vấn văn bản. Nhiệm vụ NLP này đã được sử dụng rộng rãi trong các chương trình như công cụ tìm kiếm và hệ thống gợi ý [3]. Các tiến bộ trong IR song hành với sự phát triển của các mô hình ngôn ngữ, với các kỹ thuật ban đầu dựa vào đếm từ thưa thớt và n-grams. Các phương pháp nổi bật bao gồm Boolean Retrieval [18], Term Frequency-Inverse Document Frequency (TF-IDF) [19], và BM25 [6], những phương pháp này cố gắng trích xuất các tài liệu liên quan bằng cách khớp các từ trong truy vấn.

Với sự trỗi dậy của mạng nơ-ron, IR đã phát triển song song với các mô hình ngôn ngữ và chuyển hướng sang các biểu diễn vectơ dày đặc. Hiện nay, kỹ thuật truy xuất phổ biến nhất cho các nhiệm vụ NLP khác nhau là Dense Passage Retrieval (DPR) được giới thiệu bởi Karpukhin, O˘guz, S. Min, và cộng sự [4], sử dụng các biểu diễn vectơ dày đặc để truy xuất dữ liệu. DPR là một mô hình sử dụng hai bộ mã hóa: một bộ mã hóa các tài liệu để lưu trữ, và một bộ mã hóa các câu hỏi để liên kết với các vectơ tài liệu. Cách tiếp cận này đã chứng minh sự cải thiện đáng kể so với các biểu diễn vectơ thưa truyền thống.

Nhiều nghiên cứu đã tập trung vào việc tinh chỉnh kết quả truy xuất bằng cách xếp hạng dữ liệu thu thập được để mang lại kết quả chính xác hơn. RankNet của Burges, Shaked, Renshaw, và cộng sự [5] và ListNet của Cao, Qin, T.-Y. Liu, và cộng sự [20] là hai kỹ thuật nền tảng được thiết kế để sắp xếp lại danh sách thông tin đã truy xuất nhằm cải thiện mức độ liên quan. RankNet sử dụng phương pháp pairwise, trong đó một truy vấn và hai tài liệu được đưa vào để xác định tài liệu nào liên quan hơn trong cặp. Ngược lại, ListNet sử dụng phương pháp listwise và xử lý toàn bộ danh sách các tài liệu đã truy xuất cùng với truy vấn để đánh giá mức độ liên quan của từng tài liệu và sắp xếp dữ liệu tương ứng.

Những tiến bộ trong Mạng Nơ-ron Sâu (Deep Neural Network) và LLMs đã cho phép triển khai các kỹ thuật cross-encoding để tái xếp hạng tài liệu, như được chứng minh bởi Nogueira và Cho [21]. Trong phương pháp này, một mô hình tái xếp hạng với cơ chế cross-encoding xử lý đồng thời truy vấn và các tài liệu để đánh giá chính xác hơn, so với việc chỉ sử dụng điểm tương đồng của các vectơ dày đặc độc lập. Phương pháp này đã được chứng minh là cải thiện đáng kể độ chính xác của việc truy xuất nhờ khả năng phân tích kỹ lưỡng mối quan hệ giữa truy vấn và tài liệu. Tuy nhiên, các mô hình tái xếp hạng đòi hỏi nhiều tài nguyên tính toán, đặc biệt khi phải đánh giá tất cả các tài liệu trong cơ sở dữ liệu lớn. Do đó, các mô hình tái xếp hạng thường được sử dụng để tinh chỉnh kết quả truy xuất sau khi sử dụng DPR hoặc các phương pháp khác để lọc ban đầu.

2.4 Tạo tăng cường truy xuất

Retrieval-Augmented Generation (RAG) đã nổi lên như một phương pháp quan trọng để cải thiện khả năng của các mô hình ngôn ngữ bằng cách kết hợp thông tin bên ngoài vào quá trình mô hình hóa sinh văn bản. Phương pháp này gồm 3 giai đoạn chính: Truy xuất (Retrieval), kết hợp truy xuất (Fusion Retrieval), Tạo sinh (Generation). Nhưng trong phạm vi bài nghiên cứu này sẽ chỉ tập trung vào việc cải thiện khả năng truy xuất, suy luận.

2.4.1 RAG truyền thống dựa trên truy xuất cơ sở dữ liệu vector

Nhiều kỹ thuật đã được giới thiệu nhằm tích hợp dữ liệu văn bản được truy xuất và các mô hình ngôn ngữ, một số kỹ thuật trong đó xuất hiện trước khi thuật ngữ "Tạo sinh tăng cường

truy xuất"(Retrieval-Augmented Generation - RAG) được giới thiệu. Ví dụ, bài báo Retrieval Augmented Language Model Pre-Training (REALM) của Guu, Lee, Tung, và cộng sự [22] đánh dấu một trong những kỹ thuật sớm nhất tích hợp việc truy xuất tri thức trong quá trình huấn luyện một mô hình ngôn ngữ. Cách tiếp cận này chia nhiệm vụ thành hai phần: một bộ truy xuất tri thức nơ-ron để truy xuất tài liệu và một bộ mã hóa tăng cường tri thức để diễn giải các tài liệu được truy xuất. Dựa trên kiến trúc BERT, bộ truy xuất tri thức nơ-ron mã hóa các truy vấn và tài liệu để xác định và truy xuất tài liệu phù hợp nhất cho từng truy vấn. Sau khi truy xuất, bộ mã hóa tăng cường tri thức sẽ lấy đầu vào là sự kết hợp giữa truy vấn gốc và tài liệu đã truy xuất. Đầu vào kết hợp này được xử lý thông qua kiến trúc Transformer, cho phép sự tương tác chéo phong phú giữa truy vấn và tài liệu. Không giống như nhiều mô hình sinh, REALM tập trung vào các nhiệm vụ trích xuất, huấn luyện hệ thống một cách từ đầu tới cuối để tận dụng tri thức truy xuất và khả năng hiểu ngôn ngữ một cách liền mạch.

Trái ngược với bản chất trích xuất của REALM, mục tiêu của bài khóa luận này khám phá việc sử dụng các mô hình sinh trong các hệ thống RAG, tách biệt quy trình truy xuất khỏi quy trình sinh để đánh giá hiệu quả của chúng trong việc trả lời các câu hỏi phức tạp. Chúng tôi tìm cách khám phá tiềm năng của các mô hình sinh trong việc sử dụng tri thức bên ngoài một cách hiệu quả và tạo ra các phản hồi chi tiết và sâu sắc hơn, từ đó mở rộng khả năng của hệ thống RAG vượt xa những gì có thể đạt được thông qua các phương pháp trích xuất đơn thuần.

Thuật ngữ RAG lần đầu được giới thiệu bởi P. Lewis, Perez, Piktus, và cộng sự [23]. RAG đánh dấu một bước tiến đáng kể trong NLP bằng cách tích hợp bộ truy xuất thần kinh và bộ sinh Seq2Seq để cải thiện các nhiệm vụ đòi hỏi nhiều tri thức. Không giống như REALM, tập trung vào hỏi đáp trích xuất (extractive QA) sử dụng một mô hình duy nhất cho việc truy xuất và mã hóa, RAG giới thiệu một kiến trúc khác biệt, kết hợp một DPR để truy xuất tài liệu và một BART để tạo sinh chuỗi. Ngoài ra, các mô hình truy xuất và sinh được huấn luyện đầu-cuối trong khi giữ nguyên các vector embedding của mỗi tài liệu, bởi việc cập nhật embedding trong quá trình huấn luyện vừa tốn kém về mặt tính toán vừa không cần thiết.

Bài khóa luận này tập trung vào các mô hình sinh trong các hệ thống RAG, trong đó tôi tách biệt rõ ràng quy trình truy xuất khỏi quy trình sinh. Cách tiếp cận này nhằm điều tra khả năng của mô hình sinh trong việc tổng hợp thông tin mới dựa trên các tài liệu được truy xuất, có khả năng cung cấp các phản hồi sâu sắc hơn cho các câu hỏi phức tạp.

Sau đó, nhiều thuật toán khác đã xuất hiện và tích hợp các phương pháp truy xuất hoặc mô hình sinh khác nhau để cải thiện kỹ thuật RAG. Ví dụ, trong một nghiên cứu của Izacard và Grave [24], cách xử lý dữ liệu truy xuất khác biệt. Mô hình tạo sinh câu trả lời của thuật toán được thiết kế bằng mô hình Fusion-in-Decoder (FiD), có khả năng mở rộng để xử lý số lượng dữ liệu văn bản lớn hơn. Mô hình này mã hóa sự kết hợp giữa câu hỏi và từng tài liệu riêng biệt, sau đó tổng hợp các mã hóa vector này trong bộ giải mã. Hệ thống tạo ra các câu trả lời phong phú về ngữ cảnh thông qua các cơ chế attention được áp dụng trên các mã hóa kết hợp.

Retrieval-Enhanced Transformer (RETRO) của Borgeaud, Mensch, Hoffmann, và cộng sự [25] là một mô hình ngôn ngữ khác được đề xuất nhằm cải thiện các kỹ thuật RAG và DPR. RETRO cho phép việc truy xuất lặp lại trong suốt quá trình sinh, không chỉ dựa trên gợi ý ban đầu như RAG hay FiD mà liên tục khi dữ liệu truy xuất được mở rộng. Khả năng truy xuất liên tục này cho phép RETRO thích ứng động với ngữ cảnh được thu thập từ văn bản. RETRO cũng được huấn luyện từ đầu, sử dụng kiến trúc Transformer và một cơ chế cross-attention theo từng đoạn; tuy nhiên, embedding được tính toán bằng cách sử dụng mô hình BERT cố định. Các tài liệu được chia thành nhiều phân đoạn, mã hóa và lưu trữ trong cơ sở dữ liệu vector để truy xuất nhanh hơn. RETRO hoạt động tương đương với GPT-3 và Jurassic-1 nhưng sử dụng ít hơn 25 lần số lượng tham số.

Nhiều bài báo khác đã được xuất bản để nghiên cứu sự phù hợp của việc tích hợp nhiều nguồn và loại dữ liệu vào hệ thống RAG. Bài báo "Unified representations of structured and unstructured knowledge for open-domain question answering"(UniK-QA) của Oguz, X. Chen, Karpukhin, và cộng sự [26] phân tích sự liên quan của việc kết hợp các loại dữ liệu khác nhau để tăng cường các mô hình ngôn ngữ. Mô hình FiD, được giới thiệu bởi Izacard và Grave [24], đã được sử dụng cho RAG với một cơ sở dữ liệu kết hợp dữ liệu có cấu trúc, không cấu trúc và bán cấu trúc, kết hợp dữ liệu văn bản và cơ sở tri thức.

Tất cả các loại dữ liệu đều được biến đổi theo cách heuristic thành văn bản, và quá trình truy xuất được thực hiện bằng phương pháp DPR. Y. Li, Peng, Shen, và cộng sự [27] cũng tích hợp các loại dữ liệu và nguồn khác nhau để truy xuất bằng cách giới thiệu mô hình ngôn ngữ PLUG. Mô hình này được huấn luyện sử dụng nhiều nguồn tri thức khác nhau, từ văn bản đến dữ liệu từ đồ thị tri thức (KG). Quá trình truy xuất khá đơn giản, dựa trên các truy vấn tìm kiếm và khớp từ khóa để tìm dữ liệu liên quan. Sau đó, dữ liệu được biến đổi thành văn bản và lọc qua hai giai đoạn: xếp hạng thống kê và xếp hạng ngữ nghĩa. Xếp hạng thống kê sử dụng TF-IDF để tạo một tập dữ liệu ban đầu. Sau đó, tập này được lọc thêm bằng xếp hạng ngữ nghĩa dựa trên điểm tương đồng của sentence-BERT và một ngưỡng được đặt trước.

Không giống như các nghiên cứu khác tinh chỉnh các mô hình để tạo câu trả lời từ các tài liệu được truy xuất và sử dụng cơ chế cross-attention giữa các tài liệu và đầu vào, Re-plug của Weijia, Sewon, Michihiro, và cộng sự [28] coi mô hình ngôn ngữ lớn sinh (LLM) như một hộp đen. Mỗi tài liệu được truy xuất từ hệ thống RAG được sử dụng riêng để tăng cường LLM và tạo ra xác suất cho các token đầu ra, sau đó các xác suất này được tổng hợp trên các tài liệu để tính toán xác suất token cuối cùng. Kỹ thuật này ngăn chặn việc cắt bớt các tài liệu được thêm vào nếu chúng vượt quá kích thước ngữ cảnh của LLM.

2.4.2 Tạo sinh tăng cường truy xuất dựa trên đồ thị

GraphRAG (Tạo sinh tăng cường truy xuất dựa trên đồ thị) [20, 21, 22, 24, 25, 26] là một phương pháp mới tận dụng đồ thị tri thức (KGs) để cải thiện hiệu suất của các tác vụ NLP như

hệ thống hỏi đáp (Q&A). Bằng cách tích hợp KGs với các kỹ thuật RAG, GraphRAG cho phép tạo ra các phản hồi chính xác và có ngữ cảnh hơn dựa trên thông tin có cấu trúc được trích xuất từ các tài liệu tài chính. Tuy nhiên, GraphRAG thường hoạt động kém hiệu quả trong các tác vụ hỏi đáp mang tính trừu tượng hoặc khi câu hỏi không đề cập rõ ràng đến thực thể cụ thể nào.

Nhiều nghiên cứu đã khám phá các cách để trích xuất thông tin từ Đồ thị Tri thức (Knowledge Graphs - KGs) và cơ sở dữ liệu đồ thị, tận dụng định dạng có cấu trúc của chúng. Một trong những hệ thống hỏi đáp dựa trên Đồ thị Tri thức sớm nhất thực hiện một quá trình nhiều giai đoạn để liên kết các câu hỏi của người dùng với các thực thể trong đồ thị và trích xuất câu trả lời. Đầu tiên, các thực thể trong câu hỏi của người dùng được trích xuất và liên kết với các nút tương ứng trong Đồ thị Tri thức. Sau đó, việc phát hiện mối quan hệ được thực hiện để tìm ra cạnh thích hợp trong đồ thị dẫn đến câu trả lời chính xác. Ví dụ, Berant, Chou, Frostig và Liang [29] đã đề xuất một phương pháp phân tích ngữ nghĩa để chuyển đổi các câu hỏi văn bản thành các dạng logic hoặc truy vấn có thể thực thi, từ đó trích xuất câu trả lời từ cơ sở dữ liệu đồ thị. Các kỹ thuật khác, chẳng hạn như những kỹ thuật được giới thiệu bởi Z. Wang, Ng, Nallapati và Xiang [30], sử dụng phương pháp truy xuất, tái xếp hạng và học đa nhiệm để cải thiện độ chính xác trong việc phát hiện thực thể, liên kết và xếp hạng.

Các tiến bộ gần đây nhằm loại bỏ các quy trình nhiều bước của việc truy xuất dữ liệu dựa trên đồ thị truyền thống để tránh sự tích lũy lỗi. Một trong những cách tiếp cận như vậy là phương pháp Direct Fact Retrieval - DiFaR do Baek, Aji, Lehmann và Hwang [31] đề xuất. Phương pháp này trích xuất tất cả các bộ ba từ cơ sở dữ liệu và mã hóa chúng thành một embedding vector dày đặc tương tự như kỹ thuật DPR. Phương pháp này vì vậy lưu trữ các mối quan hệ logic trong cơ sở dữ liệu vector thay vì chỉ lưu trữ dữ liệu văn bản. Nghiên cứu cũng sử dụng một mô hình tái xếp hạng (re-ranker) nhận đầu vào là câu hỏi và tài liệu, sau đó đưa ra một điểm xếp hạng chỉ ra tính hữu ích của tài liệu trong việc trả lời câu hỏi. Các bộ ba hàng đầu K được truy xuất từ cơ sở dữ liệu vector, và chỉ có các bộ ba này được đưa qua mô hình tái xếp hạng để sắp xếp lại mức độ liên quan của chúng với câu hỏi.

Y. Wang, Lipka, Rossi và cộng sự [32] cũng tận dụng tính có cấu trúc của cơ sở dữ liệu đồ thị trong quá trình truy xuất của một hệ thống RAG. Bài báo này xây dựng một Đồ thị Tri thức sử dụng một tập hợp các tài liệu và cấu trúc dữ liệu khác. Sau đó, một phương pháp duyệt đồ thị dựa trên mô hình ngôn ngữ lớn (LLM) được thiết kế để truy xuất thông tin liên quan từ cơ sở dữ liệu đồ thị. Ban đầu, TF-IDF được sử dụng để so sánh nội dung của các nút với truy vấn đã cho. Nút có điểm tương đồng cao nhất được trích xuất và coi là điểm khởi đầu cho việc duyệt đồ thị. Sau đó, chức năng này xếp hạng và điều hướng qua các nút lân cận để thu thập thông tin quý giá cho việc gợi ý LLM. Một mô hình LLM đã được tinh chỉnh được sử dụng để xếp hạng các nút lân cận, trích xuất nút có dữ liệu liên quan nhất và xác định các bước duyệt tiếp theo. LLM được huấn luyện để xem xét cả câu hỏi và các nút đã duyệt trước đó để chọn nút hứa hẹn nhất tiếp theo, đảm bảo rằng việc hiểu biết dữ liệu đã được truy xuất tích lũy sẽ thông báo

cho từng bước duyệt. Kỹ thuật này làm tăng khả năng chọn các nút với ít sự lặp lại thông tin và chất lượng cao hơn như một bổ sung vào thông tin đã thu thập. So với các kỹ thuật embedding vector, thường dựa vào dữ liệu không liên kết với các biểu diễn vector cô đặc, duyệt đồ thị nắm bắt các kết nối logic tiềm ẩn trong KGs, tận dụng các thuộc tính cấu trúc thường bị mất trong không gian vector. Điều này làm cho các phương pháp dựa trên đồ thị đặc biệt mạnh mẽ trong việc trả lời câu hỏi, đặc biệt là những câu hỏi cần đến lý luận quan hệ và nhận thức về ngữ cảnh.

Tiếp đó Darren Edge, Ha Trinh và cộng sự [33] cũng đã tận dụng tính có cấu trúc của cơ sở dữ liệu đồ thị dựa trên việc tóm tắt toàn cục từ đồ thị tri thức được tạo bởi LLM. Bài báo này xây dựng một phương pháp sử dụng một LLM để xây dựng chỉ mục văn bản dựa trên đồ thị qua hai giai đoạn: giai đoạn đầu tiên là tạo ra một đồ thị tri thức thực thể từ các tài liệu nguồn, giai đoạn sau đó là tiền tạo các bản tóm tắt cộng đồng cho tất cả các nhóm thực thể có mối liên hệ chặt chẽ. Với một truy vấn sẽ được sử dụng để tìm kiếm trong đồ thị tri thức nhằm truy xuất các nút (thực thể) và cạnh (quan hệ) liên quan đến truy vấn. Một đồ thị con, bao gồm các nút và cạnh liên quan này, được trích xuất từ toàn bộ KG để cung cấp ngữ cảnh. Đồ thị con này sau đó được tích hợp với tri thức nội tại của mô hình ngôn ngữ bằng cách mã hóa cấu trúc đồ thị thành các biểu diễn nhúng (embeddings) mà mô hình có thể hiểu được. Các bản tóm tắt do LLM tạo ra từ các mô tả đồ thị con này cung cấp phạm vi bao phủ đầy đủ của chỉ mục đồ thị cơ bản và các tài liệu đầu vào mà nó đại diện. Việc tóm tắt tập trung vào truy vấn của toàn bộ tập dữ liệu sau đó được thực hiện bằng cách sử dụng cách tiếp cận map-reduce: đầu tiên sử dụng từng bản tóm tắt đồ thị con để trả lời truy vấn một cách độc lập và song song, sau đó tóm tắt tất cả các câu trả lời từng phần liên quan thành một câu trả lời toàn cục cuối cùng.

2.4.3 RAG kết hợp đồ thị tri thức và văn bản

Đây là một đề tài nghiên cứu được chú trọng trong thời gian gần đây, tập trung vào việc nâng cao khả năng của các nhiệm vụ NLP bằng cách tối ưu khả năng của truy xuất thông tin dựa trên cơ sở dữ liệu vector. Tuy nhiên khả năng khi áp dụng của phương pháp này đối với các văn bản dài vẫn còn nhiều hạn chế khi độ chính xác trong việc truy xuất các phản hồi liên quan vẫn là một thách thức. Trong khi đó, các phương pháp GraphRAG bằng các tích hợp đồ thị tri thức với các kỹ thuật RAG đã cho phép tạo ra các phản hồi chính xác hơn và có nhận thức về ngữ cảnh dựa trên thông tin có cấu trúc được trích xuất từ các tài liệu chính. Nhưng thường hoạt động kém trong các nhiệm vụ hỏi đáp trừu tượng hoặc khi không có thực thể rõ ràng nào được đề cập trong câu hỏi.

Từ những ưu điểm và nhược điểm của cả hai phương pháp truy xuất phía trên, nhiều nghiên cứu nổi về RAG gần đây đã đề xuất nhiều phương án kết hợp cả hai phương pháp nhằm tận dụng những khả năng và giảm bớt hạn chế của cả hai phương pháp. Các nghiên cứu tiêu biểu gần đây có thể kể đến HybridRAG được đề xuất bởi Bhaskarjit Sarmah, Benika Hall, và đồng nghiệp [34]. Cụ thể, phương pháp này giải quyết các hạn chế của VectorRAG (dựa trên cơ sở dữ

liệu vector) và GraphRAG (sử dụng đồ thị tri thức). Với VectorRAG có thế mạnh trong việc xử lý văn bản không cấu trúc với quy mô lớn thông qua biểu diễn vector. Tuy nhiên, nó thường gặp khó khăn trong việc hiểu mối quan hệ giữa các thực thể. Còn trong khi đó, GraphRAG mang lại hiệu quả trong việc lập luận dựa trên dữ liệu có cấu trúc nhưng hạn chế về khả năng mở rộng và yêu cầu mối quan hệ được xác định trước giữa các thực thể. HybridRAG việc tận dụng cả hai phương pháp bằng cách kết hợp các bộ mã hóa: bộ mã hóa đồ thị tri thức để lý giải mối quan hệ phức tạp giữa các thực thể và bộ mã hóa vector để trích xuất thông tin từ văn bản tự do. Kết quả sau đó được xử lý bởi một bộ giải mã lai, tạo ra các phản hồi chính xác và phù hợp với ngữ cảnh. Cách tiếp cận này đã cho thấy hiệu quả vượt trội trong các bài toán như trích xuất thực thể, phát hiện mối quan hệ, và hỏi đáp, đặc biệt trong các lĩnh vực như tài chính hoặc các tài liệu phức tạp.

Jiashuo Sun21, Chengjin Xu và cộng sự [35] đã đề xuất Think-on-Graph 2.0 mang đến một cải tiến đáng kể đối với các mô hình ngôn ngữ lớn (LLMs) bằng cách kết hợp khả năng suy luận với truy xuất có sự hướng dẫn từ tri thức. Mô hình này xây dựng trên phương pháp Think-on-Graph [36] ban đầu, kết hợp việc tích hợp sâu hơn với các nguồn tri thức bên ngoài, đặc biệt là đồ thị tri thức (KGs), nhằm nâng cao độ chính xác và độ tin cậy của các phản hồi. Điểm đổi mới chính trong khuôn khổ này là việc tăng cường khả năng suy luận của LLM với một hệ thống truy xuất hiệu quả. Bằng cách khai thác dữ liệu liên quan từ một đồ thị tri thức, mô hình không chỉ tận dụng các mẫu ngôn ngữ mà còn làm phong phú thêm khả năng suy luận bằng thông tin có cấu trúc và chính xác. Quá trình này hỗ trợ các nhiệm vụ suy luận nhiều bước và giải quyết các lỗ hổng tri thức trong thời gian thực bằng cách truy xuất thông tin giúp hình thành các phản hồi mạch lạc và chính xác về ngữ cảnh. Điều này đặc biệt hữu ích đối với các câu hỏi phức tạp, yêu cầu nhiều tri thức, nơi mà các mô hình ngôn ngữ truyền thống có thể gặp khó khăn nếu không có một nền tảng tri thức bên ngoài đáng tin cậy. Ngoài ra, Think-on-Graph 2.0 cũng chú trọng vào khả năng giải thích, cung cấp cái nhìn về cách thức ra quyết định của mô hình dựa trên tri thức đã được truy xuất. Sự minh bạch này rất quan trọng đối với các nhiệm vụ cần giải thích hoặc biện minh cho các câu trả lời, giúp phương pháp này trở nên mạnh mẽ hơn trong các ứng dụng như chăm sóc sức khỏe, phân tích pháp lý và nghiên cứu khoa học, nơi độ chính xác và khả năng giải thích là yếu tố quan trọng.

Chương 3

Đề xuất phương pháp cải thiện khả năng trả lời của Chatbot thông qua sử dụng mô hình RAG tích hợp với truy xuất đồ thị tri thức

Nội dung của chương này cung cấp chi tiết về phương pháp được đề xuất để cải thiện khả năng của Chatbot hỗ trợ hỏi đáp từ tài liệu. Cấu trúc của chương này gồm có hai phần chính, trước tiên là phần 3.1 thảo luận về phương pháp xây dựng đồ thị tri thức, cơ sở dữ liệu từ những tài liệu được cung cấp. Tiếp đó phần 3.2 mô tả cốt lõi khả năng của một Chatbot là phương pháp RAG tích hợp truy xuất, suy luận đồ thị tri thức với truy xuất cơ sở dữ liệu vector để nâng cao khả năng của mô hình ngôn ngữ lớn (LLMs). Phương pháp đề xuất kết hợp việc mở rộng chuỗi logic dựa trên các liên kết trong KG với thông tin ngữ cảnh liên kết với các thực thể liên quan bằng cách thực hiện lặp đi lặp lại việc truy xuất ngữ cảnh dựa tri thức và sử dụng ngữ cảnh tăng cường truy xuất đồ thị. từ đó tích hợp và sử dụng hiệu quả hơn kiến thức bên ngoài từ các dạng cấu trúc khác nhau.

3.1 Xây dựng đồ thị tri thức, cơ sở dữ liệu từ những tài liệu được cung cấp

Việc lưu trữ các tài liệu cũng như tri thức từ các tài liệu dưới dạng đồ thị tri thức (Knowledge Graph - KG) là một phần quan trọng trong việc xây dựng một hệ thống hỏi đáp thông minh. Trong phần này, phương pháp được sử dụng lưu trữ tài liệu và tri thức của chúng sẽ được trình bày một cách chi tiết.

3.1.1 Xây dựng đồ thị tri thức từ tài liệu

Một đồ thị tri thức (KG) là một biểu diễn có cấu trúc của các thực thể trong thế giới thực, các thuộc tính của chúng và mối quan hệ giữa chúng. Một KG thường được biểu diễn dưới dạng một tập hợp các bộ ba có hướng $(\text{thực thể}, \text{mối quan hệ}, \text{thực thể})$. Nói cách khác, một KG bao gồm các nút đại diện cho các thực thể và các cạnh đại diện cho các mối quan hệ, cùng với nhãn và thuộc tính cho cả hai. Một bộ ba đồ thị là một đơn vị thông tin cơ bản trong một KG, bao gồm một chủ ngữ, một vị ngữ và một tân ngữ. Trong phạm vi của bài khóa luận này, việc xây dựng KG từ tài liệu sẽ được thực hiện qua hai bước chính: trích xuất tri thức và cải thiện tri thức.

Tri thức trích xuất: Mục tiêu của bước này đến việc chuyển đổi thông tin, tri thức từ dữ liệu không có cấu trúc như văn bản thành một KG có cấu trúc có thể được truy vấn sau này. Quá trình này chủ yếu phụ thuộc vào việc trích xuất các bộ ba có định dạng là (thực thể) $-$ $\left[\text{mối quan hệ}\right]$ $\rightarrow$ (thực thể). Để hỗ trợ điều này, phương pháp few-shot prompt được áp dụng vào một LLM, yêu cầu nó trích xuất càng nhiều bộ ba càng tốt từ các đoạn văn bản, đồng thời đưa các ví dụ về chuyển đổi văn bản thành bộ ba vào trong câu lệnh. Các tác vụ chính trong bước này sẽ bao gồm nhận dạng thực thể (ER), trích xuất mối quan thông qua sử dụng LLM với các prompt tương ứng.

Cải thiện tri thức: Bước này nhằm nâng cao chất lượng và độ hoàn thiện của KG bằng cách loại bỏ các thông tin dư thừa và khắc phục những lỗ hổng trong dữ liệu đã trích xuất. Các nhiệm vụ chính trong bước này bao gồm hoàn thiện KG và hợp nhất tri thức. Kỹ thuật hoàn thiện KG tìm ra các thực thể, mối quan hệ còn thiếu trong đồ thị bằng phương pháp dự đoán liên kết và xác định thực thể. Dự đoán liên kết dự đoán sự tồn tại và loại mối quan hệ giữa hai thực thể dựa trên cấu trúc và đặc điểm của đồ thị. Trong khi đó xác định thực thể khớp và hợp nhất các thực thể cùng biểu diễn một thực thể, khái niệm trong thế giới thực. Quá trình hợp nhất tri thức kết hợp thông tin từ nhiều nguồn khác nhau để tạo ra một KG hoàn chỉnh và thống nhất. Các nguồn thông tin này có thể bao gồm các KG khác, dữ liệu từ các nguồn bên ngoài, ... Quá trình này bao gồm giải quyết xung đột và trùng lặp giữa các nguồn và tổng hợp hoặc điều hòa thông tin dựa trên các quy tắc, xác suất, hoặc sự tương đồng ngữ nghĩa.

Một chuỗi LLM được triển khai gồm 2 tầng để tinh chỉnh nội dung và trích xuất chi thức. Tầng thứ nhất sử dụng một LLM để tạo ra biểu diễn tóm tắt cho từng đoạn tài liệu. Quá trình tinh chỉnh này rất quan trọng vì nó chắt lọc thông tin cốt lõi đồng thời giữ nguyên ý nghĩa ban đầu và các mối quan hệ chính giữa các khái niệm. Điều này cung cấp một đầu vào tập trung hơn cho các bước xử lý tiếp theo, nâng cao hiệu quả và độ chính xác của quy trình trích xuất bộ ba. Tầng thứ hai là một LLM dành riêng cho việc trích xuất thực thể và xác định mối quan hệ. Tất cả các bước đều được thực hiện thông qua kỹ thuật gợi ý (prompt engineering) một cách cẩn thận.

3.1.2 Xây dựng cây tài liệu từ tài liệu

Trong phạm vi là các tài liệu liên quan đến một chủ đề cụ thể, thì các tài liệu này đều sẽ là dạng văn bản bán cấu trúc với các tiêu đề, đoạn văn, câu, từ, ... Để tận dụng được cấu trúc của các tài liệu này, một cây tài liệu sẽ được xây dựng từ các tài liệu. Cây tài liệu này sẽ được xây dựng dựa trên mục lục của tài liệu. Mỗi nút trong cây tài liệu sẽ biểu diễn một phần của tài liệu, từ đó giúp cho việc truy xuất thông tin từ tài liệu trở nên dễ dàng hơn. Với việc xây dựng cây tài liệu, việc truy xuất thông tin từ tài liệu sẽ trở nên dễ dàng hơn cũng như có thể phản ảnh được mối liên quan giữa các tài liệu với nhau thông qua khoảng cách giữa các nút trong cây tài liệu. Càng gần nhau thì mối liên quan về ngữ cảnh giữa các tài liệu càng cao.

3.2 Mô hình RAG tích hợp truy xuất, suy luận đồ thị tri thức với truy xuất cơ sở dữ liệu vector

Phần này sẽ trình bày chi tiết từng bước của mô hình được đề xuất trong bài khóa luận này. Mô hình này sẽ bắt đầu với phần khởi tạo, trích xuất các thực thể từ câu hỏi đã cho làm các thực thể chủ đề ban đầu. Sau đó, nó thực hiện một quy trình lặp đi lặp lại việc khám phá tri thức và lý luận các tri thức được khám phá thông qua việc sử dụng LLM. Tại mỗi vòng lặp, bước "khám phá" sẽ truy xuất, tìm kiếm tri thức có chọn lọc bao gồm các quan hệ, thực thể liền kề với thực thể chủ đề hiện tại dựa trên KG, sử dụng các thực thể mới gặp phải để tinh chỉnh phạm vi truy xuất, từ đó nâng cao cả hiệu quả và độ chính xác. Rồi sau đó, sẽ xếp hạng và chọn lọc các thực thể dựa trên truy vấn và các ngữ cảnh thu thập được từ các tài liệu có liên quan để giảm thiểu sự mơ hồ từ đó tìm ra top-N reasoning paths. Tiếp đến tại bước "lý luận", sử dụng LLM đánh giá các thông tin, kiến thức có được từ các reasoning path, ngữ cảnh có đủ thông tin để trả lời câu hỏi. Quá trình này tiếp tục cho đến khi thu thập đủ thông tin thông qua top-N đường lý luận để trả lời câu hỏi (được đánh giá bởi LLM trong bước "lý luận") hoặc đạt đến độ sâu tìm kiếm tối đa được định trước.

3.2.1 Khởi tạo

Với đầu vào là câu hỏi q, bước đầu tiên là xác định thực thể xuất hiện trong q và liên kết chúng với các thực thể tương ứng trong đồ thị tri thức. Bước này có thể được hoàn thành dựa vào nhiều phương pháp liên kết thực thể (Entity Linking - EL) khác nhau, tiêu biểu có thể sử dụng LLMs hoặc các công cụ, mô hình chuyên về EL. Tiếp theo là bước đánh giá thực thể (Topic Evaluate - TE) để chọn ra những thực thể phù hợp nhất làm điểm bắt đầu cho việc khám phá trong một KG. Bước này sẽ sử dụng LLM để đánh giá câu hỏi q và các thực thể xuất hiện, từ đó chọn ra chủ đề $E_n(e_1, e_2, \ldots, e_n)$

Trước khi bước vào lần đầu tiên truy xuất đồ thị, mô hình truy xuất đặc trưng sẽ được sử

dụng để trích xuất $top-k$ đoạn văn từ các tài liệu liên kết với với các chủ đề ban đầu E_{topic}. Sau đó LLM sẽ đánh giá thông tin này có đủ để trả lời câu hỏi hay không dựa vào kiến thức đã được huấn luyện của nó. Nếu LLM kết luận rằng thông tin hiện có đủ để trả lời câu hỏi, các bước tiếp theo là không cần thiết và có thể trực tiếp trả về câu hỏi cho người dùng.

3.2.2 Khám phá tri thức

Phần này sẽ trình bày chi tiết cách mô hình được đề xuất lặp lại quy trình khám phá tri thức để thống nhất và gắn kết chặt chẽ các tri thức từ KG và các đoạn văn. Tại thời điểm bắt đầu của vòng lặp thứ i, mỗi path p_n bao gồm $n-1$ bộ 3, một bộ ba ở đây tức là: $p_n = (e^d_{s,n}, r^d_j, e^d_{o,n})^{D-1}_{d=1}$, trong đó $e^d_{s,n}$ và $e^d_{o,n}$ lần lượt biểu thị các thực thể chủ ngữ và thực thể tân ngữ, r^d_j là một quan hệ cụ thể giữa chúng. Các bộ ba $(e^d_{s,n}, r^d_j, e^d_{o,n})$ và $(e^{d+1}_{s,n}, r^{d+1}_j, e^{d+1}_{o,n})$ được kết nối với nhau. Tập hợp các thực thể đuôi và các quan hệ trong P lần lượt được ký hiệu là $E^{D-1} = e^{D-1}_1, e^{D-1}_2, ..., e^{D-1}_N$ và $R^{D-1} = r^{D-1}_1, r^{D-1}_2, ..., r^{D-1}_N$. Quá trình lặp lại này gồm có 2 phần chính: nâng cao truy xuất đồ thị thông qua ngữ cảnh và truy xuất ngữ cảnh thông qua hướng dẫn từ tri thức:

3.2.2.1 Nâng cao truy xuất đồ thị thông qua ngữ cảnh:

Bằng cách tận dụng sự kết nối phong phú có cấu trúc của kiến thức trên đồ thị tri thức (KG), việc tìm kiếm trên đồ thị nhằm khám phá và thiết lập các khái niệm cấp cao cũng như mối quan hệ giữa câu hỏi và thông tin mục tiêu, vốn dường như cách xa nhau trong không gian ngữ nghĩa. Mô hình được đề xuất trong bài khóa luận bao gồm các bước sau đây.

Tìm kiếm quan hệ liên quan (Relation Exploration): tại thời điểm bắt đầu của vòng lặp thứ i, mô hình sẽ tìm kiếm toàn bộ quan hệ được liên kết tới thực thể cuối cùng của mỗi reasoning path thông qua sử dụng hàm:

$$\text{Edge}(e^{D-1}_{s,n}, e^{D-1}_{o,n}) = \{(r^{D-1}_{s,n}, m, h_m) | h_m \in \{True, False\}\} \quad (1)$$

Hàm Edge() là hàm tìm kiếm các quan hệ (relationship) của thực thể. Trong đó h biểu thị chiều của quan hệ r đối với thực thể e.

Cắt giảm các quan hệ (Relation Prune): Thông qua phương trình 1 ta thu được các tập quan hệ $\{Edge(e^i_j)\}^W_{j=1}$, tại đây sẽ sử dụng phương pháp few-shot prompt để yêu cầu LLM đánh giá, lựa chọn và chấm điểm các mối quan hệ có khả năng tìm thấy thực thể chứa thông tin ngữ cảnh phù hợp hữu ích cho việc giải quyết câu hỏi q. Rồi sau đó chọn ra top-N mối quan hệ phù hợp nhất cho từng reasoning path. Tại đây, có 2 prompt được xây dựng để sử dụng:

$$PROMPT_{RP}(e^i_j, q, Edge(e^i_j)) \quad (2)$$

và

$$PROMPT_{RP\_cmb}(E^i_{topic}, q, Edge(e^i_j)^W_{j=1}) \tag{3}$$

Chi tiết của các prompt sẽ được trình bày tại phần phụ lục. phương trình 2 được xây dựng với mục tiêu là gọi LLM nhiều lần cho việc RP trên từng reasoning paths, trong khi đó phương trình 3 được xây dựng để xử lý việc chọn quan hệ phù hợp cho tất cả các reasoning paths trong 1 lần sử dụng LLM. Phương trình 2 đơn giản hóa nhiệm vụ của LLM, nhưng nó lại thiếu hiệu quả khi phải sử dụng LLM nhiều lần. Còn tại phương trình 3, việc xử lý tất cả trong 1 lần giúp cho việc sử dụng LLM giảm đi từ đó nâng cao tốc độ suy luận và cho phép xem xét một cách tổng quan các mối liên kết giữa nhiều reasoning path cùng lúc, tạo điều kiện cho việc đánh giá trở nên khách quan, chính xác hơn. Tuy nhiên, việc đưa toàn bộ các quan hệ với số lượng lớn đối với từng reasoning path là một thử thách cho hiệu quả xử lý văn bản dài của LLM.

Khám phá thực thể (Entity Discovery): Tại đây chúng ta có tập các quan hệ tương ứng liên kết với thực thể cuối của từng reasoning path từ bước RP. Cho một reasoning path p^i trong P với e^i_j là thực thể đuôi của p^i và các mối quan hệ tương ứng $r^i_{j,m}$ đã được chọn trong R_i, mô hình sử dụng hàm sau:

$$Tail(e^i_j, (r^i_{j,m}, h_m)) = c^i_{j,m} \tag{4}$$

từ đó xác định các thực thể $c^i_{j,m}$ kết nốt với e^i_j thông qua quan hệ $(r^i_{j,m}, h_m)$.

3.2.2.2 Truy xuất ngữ cảnh thông qua hướng dẫn từ tri thức:

Trong bước này, mô hình sẽ khai thác thông tin chi tiết dựa theo các tri thức trích xuất được từ KG. Sau khi xác định được tất cả các thực thể ứng viên thông qua thực thi hàm 4. Hệ thống sẽ thu thập các tài liệu được liên kết với các thực thể ứng viên, rồi sau đó đánh giá, xếp hạng những tài liệu này rồi chọn ra $top - k$ tài liệu có điểm cao nhất. Say đây là các bước chi tiết thực hiện:

Truy xuất ngữ cảnh, đoạn văn dựa trên các thực thể: để tìm kiếm những thông tin hữu dụng từ trong các tài liệu ngữ cảnh liên kết với các thực thể ứng viên, mô hình sẽ sử dụng mô hình DRM để tính điểm liên quan của các tài liệu. Thay vì tính toán trực tiếp điểm liên quan giữa tài liệu ngữ cảnh và câu hỏi - từ đó bỏ qua ngữ cảnh giữa tài liệu và thực thể tương ứng của nó - điểm sẽ được tính thông qua việc chuyển đổi reasoning paths p_i hiện tại của thực thể ứng viên thành một câu ngắn gọn và thêm nó vào ngữ cảnh rồi từ đó sẽ sử dụng mô hình DRM để đánh giá điểm liên quan giữa ngữ cảnh mới này và câu hỏi. Đây là phương trình tính điểm liên quan giữa tài liệu thứ z của thực thể ứng viên $c^i_{j,m}$ như sau:

$$srl^i_{j,m} = \text{DRM}(q, [brief_sentence(p^i), : chunk^i_{j,m,z}]) \tag{5}$$

Từ phương trình 5, ta thu được tập .Vì các tài liệu được lưu trữ dưới dạng cây tài liệu, nên giữa các tài liệu luôn có mối quan hệ về ngữ cảnh, vậy nên để tính điểm của mỗi tài liệu sẽ chịu tác

động từ các tài liệu khác. Dưới đây là phương trình tính điểm giữa tài liệu i và j:

$$srl^{i}_{j,m} = 1/Distance(i,j) \sum_{z=1}^{Z} srl^{i}_{j,m,z} \tag{6}$$

Cắt giảm thực thể (Entity Prune): sau khi đã xếp hạng và chọn được $top-k$ tài liệu có điểm cao nhất, mô hình sẽ sử dụng LLM với prompt:

$$PROMPT_{EP}(c^{i}_{j,m}, q, \{srl^{i}_{j,m}\}^{Z}_{z=1}) \tag{7}$$

để đánh giá, chấm điểm các reasoning paths với đuôi là các thực thể ứng viên để chọn ra nhưng reasoning paths có khả năng trả lời câu hỏi tốt nhất và để sử dụng cho các bước tiếp theo.

3.2.3 Suy luận từ tri thức kết hợp

Cuối vòng lặp thứ i, mô hình sẽ sử dụng LLM với prompt với toàn bộ tri thức tìm được từ các bước phía trên, bao gồm reasoning path,$Clue^{i} - 1$, và các đoạn văn ngữ cảnh tương ứng, để đánh giá liệu rằng các thông tin đó có đủ để trả lời câu hỏi, với$Clue^{i} - 1$ là phản hồi truy xuất từ lần lặp phía trước với mục tiêu là để duy trì tri thức hữu ích từ bối cảnh lịch sử. Nếu LLM đánh giá rằng kiến thức được cung cấp là đủ để đưa ra câu trả lời, sẽ trực tiếp trả về câu trả lời. Ngược lại, thì sẽ tiếp tục vòng lặp kế tiếp.

Chương 4

Thực nghiệm

4.1 Dữ liệu thực nghiệm

4.2 Thiết lập thực nghiệm

4.3 Số liệu đánh giá

4.4 Kết quả thực nghiệm

References

[1] A. Radford, K. Narasimhan, T. Salimans, I. Sutskever, and et al., "Improving language understanding by generative pre-training," *arXiv preprint*, 2018. arXiv:1801.06146.

[2] Y. Gao, Y. Xiong, X. Gao, K. Jia, J. Pan, Y. Bi, Y. Dai, J. Sun, and H. Wang, "Retrieval-augmented generation for large language models: A survey," *arXiv preprint arXiv:2312.10997*, 2023.

[3] X. Li, J. Jin, Y. Zhou, Y. Zhang, P. Zhang, Y. Zhu, and Z. Dou, "From matching to generation: A survey on generative information retrieval," *arXiv preprint arXiv:2404.14851*, 2024.

[4] V. Karpukhin, B. O˘guz, S. Min, P. Lewis, L. Wu, S. Edunov, D. Chen, and W. t. Yih, "Dense passage retrieval for open-domain question answering," *arXiv preprint arXiv:2004.04906*, 2020.

[5] C. Burges, T. Shaked, E. Renshaw, A. Lazier, M. Deeds, N. Hamilton, and G. Hullender, "Learning to rank using gradient descent," in *Proceedings of the 22nd international conference on Machine learning*, pp. 89–96, 2005.

[6] S. Robertson and H. Zaragoza, "The probabilistic relevance framework: Bm25 and beyond," *Foundations and Trends® in Information Retrieval*, vol. 3, no. 4, pp. 333–389, 2009.

[7] W. X. Zhao, K. Zhou, J. Li, T. Tang, X. Wang, Y. Hou, Y. Min, B. Zhang, J. Zhang, Z. Dong, and et al., "A survey of large language models," *arXiv preprint arXiv:2303.18223*, 2023.

[8] R. Rosenfeld, "Two decades of statistical language modeling: Where do we go from here?," *Proceedings of the IEEE*, vol. 88, no. 8, pp. 1270–1278, 2000.

[9] Y. Bengio, R. Ducharme, and P. Vincent, "A neural probabilistic language model," in *Advances in neural information processing systems 13*, 2000.

[10] T. Mikolov, K. Chen, G. Corrado, and J. Dean, "Efficient estimation of word representations in vector space," *arXiv preprint arXiv:1301.3781*, 2013.

[11] J. Pennington, R. Socher, and C. D. Manning, "Glove: Global vectors for word representation," in *Proceedings of the 2014 conference on empirical methods in natural language processing (EMNLP)*, pp. 1532–1543, 2014.

[12] A. Vaswani, N. Shazeer, N. Parmar, J. Uszkoreit, L. Jones, A. N. Gomez, Kaiser, and I. Polosukhin, "Attention is all you need," in *Advances in neural information processing systems 30*, 2017.

[13] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova, "Bert: Pre-training of deep bidirectional transformers for language understanding," *arXiv preprint arXiv:1810.04805*, 2018.

[14] H. Touvron, T. Lavril, G. Izacard, X. Martinet, M.-A. Lachaux, T. Lacroix, B. Rozière, N. Goyal, E. Hambro, F. Azhar, and et al., "Llama: Open and efficient foundation language models," *arXiv preprint arXiv:2302.13971*, 2023.

[15] A. Chowdhery, S. Narang, J. Devlin, M. Bosma, G. Mishra, A. Roberts, P. Barham, H. W. Chung, C. Sutton, S. Gehrmann, and et al., "Palm: Scaling language modeling with pathways," *Journal of Machine Learning Research*, vol. 24, no. 240, pp. 1–113, 2023.

[16] M. Lewis, Y. Liu, N. Goyal, M. Ghazvininejad, A. Mohamed, O. Levy, V. Stoyanov, and L. Zettlemoyer, "Bart: Denoising sequence-to-sequence pre-training for natural language generation, translation, and comprehension," *arXiv preprint arXiv:1910.13461*, 2019.

[17] C. Raffel, N. Shazeer, A. Roberts, K. Lee, S. Narang, M. Matena, Y. Zhou, W. Li, and P. J. Liu, "Exploring the limits of transfer learning with a unified text-to-text transformer," *Journal of machine learning research*, vol. 21, no. 140, pp. 1–67, 2020.

[18] G. Salton, E. A. Fox, and H. Wu, "Extended boolean information retrieval," *Communications of the ACM*, vol. 26, no. 11, pp. 1022–1036, 1983.

[19] H. P. Luhn, "A statistical approach to mechanized encoding and searching of literary information," *IBM Journal of Research and Development*, vol. 1, no. 4, pp. 309–317, 1957.

[20] Z. Cao, T. Qin, T.-Y. Liu, M.-F. Tsai, and H. Li, "Learning to rank: from pairwise approach to listwise approach," in *Proceedings of the 24th international conference on Machine learning*, pp. 129–136, 2007.

[21] R. Nogueira and K. Cho, "Passage re-ranking with bert," *arXiv preprint arXiv:1901.04085*, 2019.

[22] K. Guu, K. Lee, Z. Tung, P. Pasupat, and M. Chang, "Realm: Retrieval augmented language model pre-training," in *International conference on machine learning*, pp. 3929–3938, 2020.

[23] P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Küttler, M. Lewis, W. t. Yih, T. Rocktäschel, and et al., "Retrieval-augmented generation for knowledge-intensive nlp tasks," in *Advances in Neural Information Processing Systems 33*, pp. 9459–9474, 2020.

[24] G. Izacard and E. Grave, "Leveraging passage retrieval with generative models for open domain question answering," *arXiv preprint arXiv:2007.01282*, 2020.

[25] S. Borgeaud, A. Mensch, J. Hoffmann, T. Cai, E. Rutherford, K. Millican, G. B. V. D. Driessche, J.-B. Lespiau, B. Damoc, A. Clark, and et al., "Improving language models by retrieving from trillions of tokens," in *International conference on machine learning*, pp. 2206–224, 2022.

[26] B. Oguz, X. Chen, V. Karpukhin, S. Peshterliev, D. Okhonko, M. Schlichtkrull, S. Gupta, Y. Mehdad, and S. Yih, "Unik-qa: Unified representations of structured and unstructured knowledge for open-domain question answering," *arXiv preprint arXiv:2012.14610*, 2020.

[27] Y. Li, B. Peng, Y. Shen, Y. Mao, L. Liden, Z. Yu, and J. Gao, "Knowledge-grounded dialogue generation with a unified knowledge representation," *arXiv preprint arXiv:2112.07924*, 2021.

[28] S. Weijia, M. Sewon, Y. Michihiro, S. Minjoon, J. Rich, L. Mike, and et al., "Replug: Retrieval-augmented black-box language models," *arXiv preprint arXiv:2301.12652*, 2023.

[29] J. Berant, A. Chou, R. Frostig, and P. Liang, "Semantic parsing on freebase from question-answer pairs," in *Proceedings of the 2013 conference on empirical methods in natural language processing*, pp. 1533–1544, 2013.

[30] Z. Wang, P. Ng, R. Nallapati, and B. Xiang, "Retrieval, re-ranking and multi-task learning for knowledge-base question answering," in *Proceedings of the 16th Conference of the European Chapter of the Association for Computational Linguistics: Main Volume*, pp. 347–357, 2021.

[31] J. Baek, A. F. Aji, J. Lehmann, and S. J. Hwang, "Direct fact retrieval from knowledge graphs without entity linking," *arXiv preprint arXiv:2305.12416*, 2023.

[32] Y. Wang, N. Lipka, R. A. Rossi, A. Siu, R. Zhang, and T. Derr, "Knowledge graph prompting for multi-document question answering," in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 38, pp. 19206–19214, 2024.

[33] D. Edge, H. Trinh, N. Cheng, J. Bradley, A. Chao, A. Mody, S. Truitt, and J. Larson, "From local to global: A graph rag approach to query-focused summarization," *Microsoft Research*, 2024.

[34] B. Sarmah, B. Hall, R. Rao, S. Patel, S. Pasquali, and D. Mehta, "Hybridrag: Integrating knowledge graphs and vector retrieval augmented generation for efficient information extraction," *arXiv preprint arXiv:2408.04948*, 2024.

[35] S. Ma, C. Xu, X. Jiang, M. Li, H. Qu, C. Yang, J. Mao, and J. Guo, "Think-on-graph 2.0: Deep and faithful large language model reasoning with knowledge-guided retrieval augmented generation," *IDEA Research, International Digital Economy Academy*, 2024.

[36] J. Sun, C. Xu, L. Tang, S. Wang, C. Lin, Y. Gong, L. M. Ni, H.-Y. Shum, and J. Guo, "Think-on-graph: Deep and responsible reasoning of large language model on knowledge graph," *arXiv preprint arXiv:2407.10805*, 2024.