**Truy xuất và đọc: Một cuộc khảo sát toàn diện trên trả lời miền mở**

Fengbin Zhu, Wenqiang Lei\*, Chao Wang, Jianming Zheng, Soujanya Poria, Tat-Seng Chua

**Tóm tắt**- Trả lời câu hỏi miền mở (OpenQA) là một nhiệm vụ quan trọng trong Xử lý ngôn ngữ tự nhiên (NLP), nhằm mục đích trả lời một câu hỏi dưới dạng ngôn ngữ tự nhiên dựa trên các tài liệu phi cấu trúc quy mô lớn. Gần đây, số lượng tài liệu nghiên cứu về OpenQA đã tăng vọt, đặc biệt là về các kỹ thuật tích hợp với tính năng Đọc hiểu máy thần kinh (MRC). Mặc dù các công trình nghiên cứu này đã nâng cao hiệu suất lên tầm cao mới trên bộ dữ liệu điểm chuẩn, chúng hiếm khi được đề cập trong các cuộc khảo sát hiện có về hệ thống QA. Trong công việc này, chúng tôi xem xét các xu hướng nghiên cứu mới nhất trong OpenQA, đặc biệt chú ý đến các hệ thống kết hợp các kỹ thuật MRC thần kinh. Cụ thể, chúng tôi bắt đầu với việc xem lại nguồn gốc và sự phát triển của các hệ thống OpenQA.

Sau đó, chúng tôi giới thiệu kiến trúc OpenQA hiện đại có tên là “Retriever-Reader” và phân tích các hệ thống khác nhau tuân theo kiến trúc này cũng như các kỹ thuật cụ thể được áp dụng trong từng thành phần. Sau đó, chúng tôi thảo luận về những thách thức chính đối với việc phát triển hệ thống OpenQA và đưa ra phân tích về các điểm chuẩn thường được sử dụng. Chúng tôi hy vọng công việc của chúng tôi sẽ cho phép các nhà nghiên cứu được thông báo về những tiến bộ gần đây và cũng như những thách thức mở trong nghiên cứu OpenQA, để kích thích sự tiến bộ hơn nữa trong lĩnh vực này.

Điều khoản lập chỉ mục — Trả lời câu hỏi dạng văn bản, Trả lời câu hỏi trên miền mở, Đọc hiểu máy, Thông tin.

1. **GIỚI THIỆU**

Trả lời câu hỏi (QA) nhằm mục đích cung cấp câu trả lời chính xác để trả lời câu hỏi của người dùng bằng ngôn ngữ tự nhiên. Đó là một nhiệm vụ lâu đời có từ những năm 1960 [1]. So với công cụ tìm kiếm, hệ thống QA nhằm mục đích trình bày câu trả lời cuối cùng cho một câu hỏi trực tiếp thay vì trả về danh sách các đoạn trích hoặc siêu liên kết có liên quan, do đó mang lại hiệu quả và thân thiện với người dùng hơn. Ngày nay, nhiều công cụ tìm kiếm web như Google và Bing đã và đang phát triển theo hướng thông minh hơn bằng cách kết hợp các kỹ thuật QA vào các chức năng tìm kiếm của chúng [2]. Được trao quyền với những kỹ thuật này, các công cụ tìm kiếm hiện có khả năng trả lời chính xác một số loại câu hỏi như:

- Hỏi: "Barack Obama sinh ra khi nào?"

- Đáp: “Ngày 4 tháng 8 năm 1961”

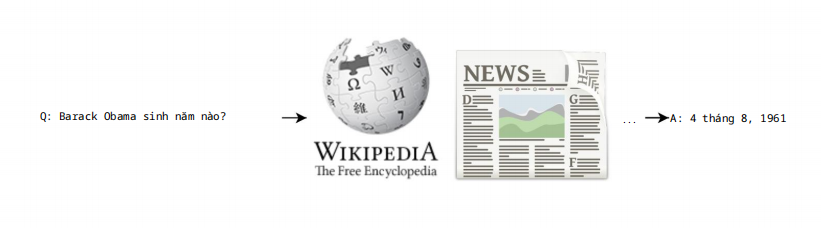
Toàn bộ bối cảnh QA đại khái có thể được chia thành hai phần: QA dạng văn bản và Cơ sở kiến thức (KB) -QA, xác định loại nguồn thông tin mà câu trả lời được bắt nguồn từ đó. QA văn bản khai thác câu trả lời từ các tài liệu văn bản không có cấu trúc trong khi KB-QA từ KB có cấu trúc được xác định trước thường được xây dựng theo cách thủ công. QA dạng văn bản có khả năng mở rộng đồng minh hơn so với loại sau, vì hầu hết các tài nguyên văn bản không có cấu trúc mà nó khai thác để có được câu trả lời là khá phổ biến và dễ dàng truy cập, chẳng hạn như Wikipedia [3], tin tức các bài báo [4] và sách khoa học [5], v.v ... Cụ thể, QA dạng văn bản được nghiên cứutheo hai cài đặt nhiệm vụ dựa trên sự sẵn có của thông tin ngữ cảnh, tức là kích thước của Machine Reading Compre (MRC) và Open-domain QA (OpenQA). MRC, ban đầu lấy cảm hứng từ các kỳ thi thông thạo ngôn ngữ, nhằm mục đích cho phép máy móc đọc và hiểu (các) đoạn ngữ cảnh cụ thể để trả lời một câu hỏi nhất định.

Trong khi đó, OpenQA cố gắng trả lời một câu hỏi nhất định mà không có bất kỳ ngữ cảnh cụ thể nào. Nó thường yêu cầu hệ thống trước tiên phải tìm kiếm các tài liệu có liên quan khi ngữ cảnh đưa ra một câu hỏi nhất định từ kho tài liệu cục bộ hoặc World Wide Web (WWW), và sau đó tạo ra câu trả lời, như được minh họa

trong Hình 1. Do đó OpenQA được hưởng phạm vi ứng dụng rộng hơn và phù hợp hơn với hành vi QA trong thế giới thực của con người trong khi MRC có thể được coi là một bước tiến của OpenQA [6]. Trên thực tế, việc xây dựng một hệ thống OpenQA có khả năng trả lời bất kỳ câu hỏi đầu vào nào được coi là mục tiêu cuối cùng

của nghiên cứu QA.

Trong tài liệu, OpenQA đã được nghiên cứu chặt chẽ với việc tìm kiếm lại trong Xử lý ngôn ngữ tự nhiên (NLP), Truy xuất thông tin (IR) và Trích xuất thông tin (IE) [7], [8], [9], [10]. Các hệ thống OpenQA truyền thống hầu hết tuân theo một đường ống bao gồm ba giai đoạn, tức là Phân tích câu hỏi, Truy xuất tài liệu và Trích xuất câu trả lời [6], [9], [11]. Đưa ra một câu hỏi đầu vào bằng ngôn ngữ tự nhiên. Phân tích Câu hỏi nhằm mục đích định dạng lại câu hỏi để tạo ra các truy vấn tìm kiếm nhằm hỗ trợ việc Truy xuất Tài liệu tiếp theo và phân loại các câu hỏi để có được (các) loại câu trả lời mong đợi sẽ hướng dẫn Trích xuất Câu trả lời. Trong giai đoạn Truy xuất Tài liệu, hệ thống tìm kiếm các tài liệu hoặc đoạn văn có liên quan đến câu hỏi bằng các truy vấn tìm kiếm được tạo, thường sử dụng các kỹ thuật IR hiện có như TF-IDF và BM25 hoặc các kỹ thuật cụ thể được phát triển cho các công cụ tìm kiếm Web như Google.com và Bing. com. Sau đó, trong giai đoạn Trích xuất câu trả lời, câu trả lời cuối cùng được trích xuất từ các tài liệu liên quan nhận được từ giai đoạn trước.



Hình 1: Một minh họa của OpenQA. Đưa ra một câu hỏi ngôn ngữ tự nhiên, hệ thống đưa ra câu trả lời từ một bộ sưu tập các tài liệu văn bản phi cấu trúc.

Các kỹ thuật học sâu, đã thúc đẩy sự tiến bộ đáng kể trong nhiều lĩnh vực, cũng đã được áp dụng thành công cho hầu hết các giai đoạn của hệ thống OpenQA [12]. Ví dụ: [13] và [14] phát triển bộ phân loại câu hỏi bằng cách sử dụng mô hình cơ sở CNN và mô hình dựa trên LSTM tương ứng. Trong [15], [16], [17], họ đề xuất các mô hình truy xuất thần kinh để tìm kiếm các tài liệu có liên quan trong một không gian tiềm ẩn. Trong những năm gần đây, với sự xuất hiện của một số bộ dữ liệu QA quy mô lớn [18], [19], [20], [21], [22], [23], kỹ thuật MRC thần kinh đã được nâng cao rất nhiều [18], [ 24], [25], [26], [27]. Bằng cách áp dụng các phương pháp MRC thần kinh phổ biến để trích xuất câu trả lời cho một câu hỏi nhất định từ (các) tài liệu liên quan, các hệ thống OpenQA truyền thống đã được cách mạng hóa [3], [28], [29], [30] và phát triển thành hiện đại “ Kiến trúc Retriever-Reader ”. Retriever chịu trách nhiệm truy xuất các tài liệu liên quan từ một câu hỏi nhất định, có thể được coi là một hệ thống IR, trong khi Reader nhằm mục đích suy ra câu trả lời cuối cùng từ các tài liệu đã nhận, thường là mô hình MRC thần kinh. Một số tác phẩm [3], [31], [32] thậm chí còn đổi tên OpenQA là Đọc hiểu Máy ở Quy mô (MRS). Theo kiến trúc này, nghiên cứu mở rộng đã được thực hiện theo nhiều hướng khác nhau, chẳng hạn như xếp hạng lại các tài liệu đã truy xuất trước khi đưa chúng vào mô hình MRC thần kinh [28], [33], [34], truy xuất các tài liệu liên quan lặp đi lặp lại với một câu hỏi [ 29], [35], [36], và đào tạo toàn bộ hệ thống OpenQA theo cách endto-end [15], [30], [37], [38], v.v.

Dựa trên những quan sát và hiểu biết ở trên, chúng tôi tin rằng đã đến lúc cung cấp một đánh giá tài liệu toàn diện về hệ thống OpenQA, đặc biệt chú ý đến các kỹ thuật kết hợp mô hình MRC thần kinh. Đánh giá của chúng tôi được thực hiện để ghi nhận những tiến bộ đã đạt được cho đến nay và tóm tắt những thách thức hiện tại để kích thích tiến bộ hơn nữa trong lĩnh vực này. Trong phần còn lại của khảo sát này, chúng tôi sẽ trình bày những nội dung sau. Trong Phần 2, chúng tôi xem xét sự phát triển của các hệ thống OpenQA, bao gồm nguồn gốc, kiến trúc truyền thống và tiến bộ gần đây trong việc sử dụng mạng nơ-ron sâu. Trong Phần 3, chúng tôi tóm tắt và đánh giá chi tiết kiến trúc “Retriever-Reader” cho OpenQA, sau đó là phân tích chi tiết về các kỹ thuật khác nhau được áp dụng. Trong Phần 4, trước tiên chúng ta thảo luận về một số thách thức nổi bật đối với các phường OpenQA, xác định các lỗ hổng nghiên cứu và hy vọng sẽ tăng cường nghiên cứu sâu hơn trong lĩnh vực này, và sau đó cung cấp bản tóm tắt và phân tích các điểm chuẩn QA có thể áp dụng cho MRC hoặc OpenQA. Cuối cùng, chúng tôi đưa ra kết luận dựa trên các nội dung đã trình bày ở trên trong Phần 5.

1. **SỰ PHÁT TRIỂN CỦA OPENQA**

Trong phần này, trước tiên, chúng tôi giới thiệu ngắn gọn về nguồn gốc của tính năng Trả lời câu hỏi trên miền mở (OpenQA) và sau đó xem xét tuần tự các phương pháp tiếp cận học sâu và truyền thống đối với OpenQA để mô tả sự tiến bộ vượt bậc của nó trong hai thập kỷ qua.

**2.1. Nguồn gốc của OpenQA**

Nghiên cứu tiên phong về hệ thống Trả lời Câu hỏi (QA) được tiến hành trong phạm vi Truy xuất Thông tin (IR), tập trung vào cài đặt miền hạn chế hoặc miền đóng. Hệ thống QA sớm nhất, như đã được công nhận rộng rãi, là Bóng chày [1], được thiết kế vào năm 1961 để trả lời các câu hỏi về các trận bóng chày của Mỹ, chẳng hạn như thời gian thi đấu, địa điểm và tên đội. Trong hệ thống này, tất cả thông tin liên quan được lưu trữ trong một từ điển được xác định rõ ràng và các câu hỏi của người dùng được dịch thành các câu lệnh truy vấn bằng cách sử dụng các phương pháp ngôn ngữ để trích xuất câu trả lời cuối cùng từ từ điển. Năm 1973, một hệ thống QA nổi tiếng khác LUNAR [39] được phát triển như một công cụ mạnh mẽ để hỗ trợ công việc nghiên cứu của các nhà địa chất mặt trăng, nơi dữ liệu phân tích hóa học về đất và đá mặt trăng thu được từ các sứ mệnh trên mặt trăng của Apollo được lưu trữ trong một tệp cơ sở dữ liệu do NASA cung cấp. MSC để mỗi nhà khoa học tiện xem và phân tích. Năm 1993, MURAX [40] được thiết kế để trả lời các câu hỏi học thuật đơn giản dựa trên một bách khoa toàn thư học thuật tiếng Anh chủ yếu sử dụng kỹ thuật phân tích ngôn ngữ và đối sánh mẫu cú pháp.

Năm 1999, OpenQA lần đầu tiên được định nghĩa là trích xuất 5 đoạn mã có thể xảy ra hàng đầu chứa câu trả lời chính xác từ tập hợp các bài báo trong bài QA do Hội nghị duyệt lại văn bản (TREC) đưa ra [4]. So với nghiên cứu trước đây về QA, trong cài đặt miền mở, một số lượng lớn tài liệu không có cấu trúc được sử dụng làm nguồn thông tin mà từ đó câu trả lời chính xác cho một câu hỏi nhất định sẽ được trích xuất. Trong những năm sau đó, một loạt TREC QA Tracks đã nâng cao đáng kể tiến độ nghiên cứu về OpenQA [41], [42], [43]. Cần lưu ý rằng hệ thống được yêu cầu trả lại câu trả lời ngắn chính xác cho các câu hỏi nhất định bắt đầu từ TREC-11 được tổ chức vào năm 2002 [42].

Chiến dịch TREC cung cấp một bộ sưu tập tài liệu cục bộ làm nguồn thông tin để tạo câu trả lời, nhưng sự phổ biến của World Wide Web (WWW), đặc biệt là sự phát triển ngày càng tăng của các công cụ tìm kiếm, đã truyền cảm hứng cho những người tìm kiếm xây dựng hệ thống OpenQA dựa trên Web [40 ], [44], [45], [46] nhận câu trả lời từ các nguồn trực tuyến như Google.com và Ask.com, sử dụng các kỹ thuật IR. Các công cụ tìm kiếm web có thể thu thập các trang web mas sive một cách nhất quán và hiệu quả, do đó có khả năng cung cấp nhiều thông tin hơn để giúp tìm ra câu trả lời theo yêu cầu của người dùng. Năm 2001, một hệ thống QA được gọi là MVERER [44] đã được ký hợp đồng để tự động trả lời các câu hỏi thừa số miền mở bằng một công cụ tìm kiếm (ví dụ: Google.com). Đầu tiên, nó chuyển các câu hỏi của người dùng thành nhiều truy vấn tìm kiếm với một số trình phân tích cú pháp ngôn ngữ tự nhiên và gửi chúng tới search en gine để tìm kiếm các tài liệu có liên quan, sau đó sử dụng thành phần trích xuất câu trả lời để trích xuất câu trả lời từ các kết quả trả về. Sau đường ống này, một hệ thống QA nổi tiếng AskMSR [45] đã được phát triển, hệ thống này chủ yếu phụ thuộc vào dự phòng dữ liệu hơn là phân tích ngôn ngữ phức tạp của câu hỏi hoặc câu trả lời của ứng viên. Đầu tiên, nó chuyển câu hỏi của người dùng thành các truy vấn dựa trên một tập hợp các quy tắc viết lại được xác định trước để thu thập các tài liệu có liên quan từ các công cụ tìm kiếm và sau đó áp dụng một loạt các thuật toán dựa trên n-gram để khai thác, lọc và chọn câu trả lời tốt nhất.

Đối với các hệ thống OpenQA như vậy, các công cụ tìm kiếm có thể cung cấp quyền truy cập vào một đại dương thông tin, mở rộng đáng kể khả năng tìm kiếm câu trả lời chính xác cho các câu hỏi của người dùng. Tuy nhiên, nguồn thông tin dồi dào như vậy cũng mang đến những nội dung nhiễu đáng kể, thách thức hệ thống QA lọc ra.

**2.2. Kiến trúc truyền thống của OpenQA**

Kiến trúc truyền thống của hệ thống OpenQA được minh họa trong Hình 2, chủ yếu bao gồm ba giai đoạn: Phân tích truy vấn, Truy xuất tài liệu và Trích xuất câu trả lời [6], [11]. Với một câu hỏi ngôn ngữ tự nhiên, Phân tích Câu hỏi nhằm mục đích hiểu câu hỏi trước để tạo điều kiện thuận lợi cho việc truy xuất tài liệu và trích xuất câu trả lời trong các giai đoạn sau.

Hiệu suất của giai đoạn này được phát hiện có ảnh hưởng đáng chú ý so với các giai đoạn sau, và do đó quan trọng đối với đầu ra cuối cùng của hệ thống [47]. Sau đó, giai đoạn đánh giá lại tài liệu tìm kiếm các tài liệu liên quan đến câu hỏi dựa trên hệ thống IR tự xây dựng [4] hoặc công cụ tìm kiếm Web [44], [45] bằng cách sử dụng các truy vấn tìm kiếm do Phân tích câu hỏi tạo ra.

Cuối cùng, Answer Extraction chịu trách nhiệm trích xuất câu trả lời cuối cùng cho các câu hỏi của người dùng từ các tài liệu liên quan đã nhận được ở bước trước. Sau đây, chúng tôi sẽ phân tích từng giai đoạn một.

**2.2.1. Phân tích câu hỏi**

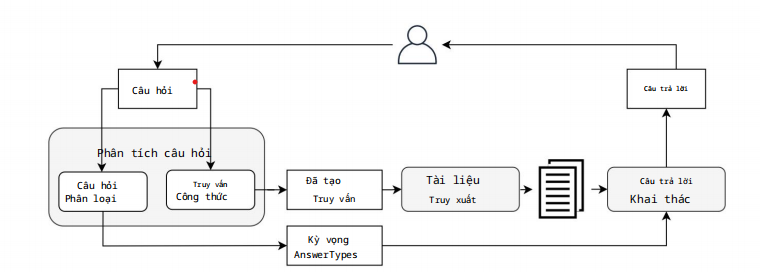
Mục tiêu của giai đoạn Phân tích câu hỏi gồm hai phần. Một mặt, nó nhằm mục đích tạo điều kiện thuận lợi cho việc truy xuất các tài liệu liên quan đến câu hỏi, mà mô-đun Công thức truy vấn thường được sử dụng để tạo các truy vấn tìm kiếm. Mặt khác, dự kiến sẽ nâng cao hiệu suất của giai đoạn Trích xuất câu trả lời bằng cách sử dụng mô-đun Phân loại câu hỏi để dự đoán loại câu hỏi đã cho, dẫn đến một bộ trong số các loại câu trả lời mong đợi. Một minh họa đơn giản của giai đoạn này được đưa ra trong hộp màu xám ngoài cùng bên trái của Hình 2.

Trong Công thức truy vấn, các kỹ thuật ngôn ngữ như gắn thẻ POS [40], [44], tạo gốc [40], phân tích cú pháp [44] và dừng loại bỏ từ [45], [48] thường được sử dụng để trích xuất các từ khóa để truy xuất. Tuy nhiên, các thuật ngữ được sử dụng trong các câu hỏi thường không giống với các thuật ngữ xuất hiện trong các tài liệu có chứa câu trả lời chính xác. Vấn đề này được gọi là "sự không phù hợp về thuật ngữ" và là một vấn đề lâu dài và quan trọng trong IR. Để giải quyết vấn đề này, các kỹ thuật mở rộng truy vấn [49], [50] và diễn giải [51], [52], [53], [54] thường được sử dụng để tạo ra các từ hoặc cụm từ tìm kiếm bổ sung để truy xuất các tài liệu liên quan hơn.

Phân loại câu hỏi, một mô-đun khác thường được sử dụng cho giai đoạn: Phân tích câu hỏi, nhằm xác định loại câu hỏi đã cho dựa trên một tập hợp các loại câu hỏi (ví dụ: ở đâu, khi nào, ai, cái gì) hoặc phân loại [55], [ 56] do các chuyên gia ngôn ngữ học xác định thủ công. Sau khi có được loại câu hỏi, các loại câu trả lời mong đợi có thể dễ dàng xác định bằng cách sử dụng các phương pháp ánh xạ dựa trên quy tắc [9]. Ví dụ: với một câu hỏi “Barack Obama sinh khi nào?”, Loại câu trả lời sẽ được suy ra là “Ngày tháng” khi biết loại câu hỏi là “Khi nào”. Việc xác định loại câu hỏi có thể hạn chế việc trích xuất câu trả lời và giảm đáng kể khó khăn trong việc tìm câu trả lời đúng. Phân loại câu hỏi đã thu hút nhiều sự quan tâm trong văn học [44], [55], [57], [58], [59]. Ví dụ, [59] đề xuất trích xuất các từ có liên quan từ một câu hỏi nhất định và sau đó phân loại câu hỏi dựa trên các quy tắc liên kết những từ này với các khái niệm; [57] đã đào tạo một danh sách các bộ phân loại câu hỏi bằng cách sử dụng các kỹ thuật học ma chine khác nhau như Máy vectơ hỗ trợ (SVM), Hàng xóm gần nhất và Cây quyết định trên đầu bảng phân loại thứ bậcđược đề xuất bởi [55].

**2.2.2. Truy xuất tài liệu**

Giai đoạn này nhằm mục đích thu được một số lượng nhỏ các tài liệu có liên quan có thể chứa câu trả lời chính xác cho một câu hỏi nhất định từ tập hợp các tài liệu không có cấu trúc, thường dựa vào công cụ IR. Nó có thể làm giảm đáng kể không gian tìm kiếm để đi đến câu trả lời cuối cùng.



Hình 2: Một minh họa về kiến trúc truyền thống của hệ thống OpenQA

Trong những thập kỷ qua, nhiều mô hình truy xuất khác nhau đã được phát triển để Truy xuất tài liệu, trong đó một số mô hình phổ biến là mô hình Boolean, Mô hình không gian vectơ, Mô hình xác suất, Mô hình ngôn ngữ [60], v.v., được xem xét lại một cách ngắn gọn như sau.

* Mô hình Boolean: Mô hình Boolean là một trong những mô hình truy xuất sim nhiều nhất. Câu hỏi được chuyển thành dạng biểu thức Boolean của các thuật ngữ, được kết hợp với các toán tử như “AND”, ”OR” và “NOT” để khớp chính xác với các tài liệu, với mỗi tài liệu được xem như một tập hợp các từ.
* Mô hình không gian vectơ: Mô hình không gian vectơ biểu thị câu hỏi và mỗi tài liệu dưới dạng vectơ từ trong không gian từ d chiều, trong đó d là số từ trong từ vựng. Khi tìm kiếm các tài liệu liên quan đến một câu hỏi nhất định, điểm phù hợp của mỗi tài liệu được tính bằng cách tính độ tương tự (ví dụ: độ tương tự cosin) hoặc khoảng cách (ví dụ: khoảng cách euclide) giữa vectơ của nó và vectơ câu hỏi. So với mô hình Boolean, cách tiếp cận này trả về tài liệu cho câu hỏi ngay cả khi các hạn chế do câu hỏi đặt ra chỉ được đáp ứng một phần, với độ chính xác bị hy sinh.
* Mô hình xác suất: Mô hình xác suất cung cấp một cách tích hợp các mối quan hệ xác suất giữa các từ vào một mô hình. Okapi BM25 [61] là mô hình xác suất nhạy cảm với tần suất thuật ngữ và độ dài tài liệu, là một trong những mô hình truy xuất thành công theo kinh nghiệm và được sử dụng rộng rãi trong các công cụ tìm kiếm hiện nay.
* Mô hình Ngôn ngữ: Các Mô hình Ngôn ngữ [62] cũng rất phổ biến, trong đó Mô hình Khả năng Truy vấn [60] được áp dụng rộng rãi nhất. Nó xây dựng mô hình ngôn ngữ xác suất LMd cho mỗi tài liệu d và xếp hạng tài liệu theo xác suất P (q | LMd) của mô hình ngôn ngữ tạo ra câu hỏi q đã cho.

Trong thực tế, các tài liệu nhận được thường chứa những tài liệu không quan trọng hoặc số lượng tài liệu quá lớn khiến dung lượng của mô hình Giải nén Trả lời bị quá tải. Để giải quyết các vấn đề trên, việc xử lý hậu kỳ đối với các tài liệu được truy xuất là rất cần thiết. Các phương pháp tiếp cận được sử dụng rộng rãi để xử lý các tài liệu đã truy xuất bao gồm lọc tài liệu, xếp hạng lại tài liệu và lựa chọn tài liệu [9], v.v. Lọc tài liệu được sử dụng để xác định và loại bỏ nhiễu trong một câu hỏi nhất định; phân loại lại tài liệu được phát triển để sắp xếp lại tài liệu theo mức độ hợp lý có chứa câu trả lời đúng theo thứ tự giảm dần; lựa chọn tài liệu là chọn những tài liệu có liên quan hàng đầu. Sau khi xử lý hậu kỳ, chỉ những tài liệu liên quan nhất sẽ được giữ lại và chuyển sang giai đoạn tiếp theo để rút ra câu trả lời cuối cùng.

**2.2.3. Trích xuất câu trả lời**

Mục tiêu cuối cùng của hệ thống OpenQA là trả lời thành công các câu hỏi nhất định và giai đoạn Trích xuất câu trả lời chịu trách nhiệm trả lại cho người dùng câu trả lời chính xác nhất cho một câu hỏi. Hiệu suất của giai đoạn này được quyết định bởi mức độ phức tạp của câu hỏi, các loại câu trả lời dự kiến từ giai đoạn Phân tích câu hỏi, các tài liệu được truy xuất từ giai đoạn Truy xuất Tài liệu cũng như phương pháp trích xuất được áp dụng, v.v. Với rất nhiều yếu tố ảnh hưởng, các nhà nghiên cứucần phải chăm chút nhiều và đặc biệt coi trọng khâu này.

Trong các hệ thống OpenQA truyền thống câu hỏi dạng thừa và câu hỏi liệt kê [63] đã được nghiên cứu rộng rãi trong một thời gian dài, Câu hỏi thực tế (ví dụ: Khi nào, Ở đâu, Ai ...) mà câu trả lời thường là một đoạn văn bản duy nhất trong tài liệu, chẳng hạn như tên thực thể, mã thông báo từ hoặc cụm danh từ. Trong khi liệt kê các câu hỏi có câu trả lời là một tập hợp các dữ kiện xuất hiện trong cùng một tài liệu hoặc được tổng hợp từ các tài liệu khác nhau. Loại câu trả lời nhận được từ giai đoạn Phân tích câu hỏi đóng một vai trò quan trọng, đặc biệt là đối với câu hỏi đã cho có câu trả lời được đặt tên là các thực thể. Do đó, các hệ thống ban đầu chủ yếu dựa vào kỹ thuật Nhận dạng thực thể được đặt tên (NER) [40], [46], [64] vì so sánh các thực thể được nhận dạng và loại câu trả lời có thể dễ dàng đưa ra câu trả lời cuối cùng. Trong [65], việc trích xuất câu trả lời được mô tả như một quá trình thống nhất, trước tiên phát hiện thông tin tiềm ẩn hoặc ẩn từ câu hỏi và câu trả lời tương ứng, sau đó sử dụng một số phương pháp đối sánh để phát hiện câu trả lời, chẳng hạn như đối sánh mẫu văn bản bề mặt [66], [67], đối sánh từ hoặc cụm từ [44], và đối sánh cấu trúc cú pháp [40], [48], [68].

Trong thực tế, đôi khi câu trả lời được trích xuất cần được xác nhận khi nó không đủ tự tin trước khi trình bày với người dùng cuối. Hơn nữa, trong một số trường hợp, nhiều ứng viên trả lời có thể được đưa ra cho một câu hỏi và chúng tôi phải chọn một trong số họ. Xác thực câu trả lời được áp dụng để giải quyết các vấn đề như vậy. Một phương pháp xác nhận được áp dụng rộng rãi là sử dụng một nguồn thông tin bổ sung như công cụ tìm kiếm trên Web để xác nhận độ tin cậy của từng câu trả lời ứng viên.Nguyên tắc là hệ thống phải trả về một số lượng đủ lớn các tài liệu có chứa cả điều khoản câu hỏi và câu trả lời. Số lượng tài liệu trả về càng lớn thì càng có nhiều khả năng là câu trả lời chính xác.Nguyên tắc này đã được nghiên cứu và chứng minh khá hiệu quả, mặc dù đơn giản [9].

**2.3. Ứng dụng của Deep Neural Networks trong OpenQA**

Trong thập kỷ gần đây, các kỹ thuật học sâu cũng đã được áp dụng thành công cho OpenQA. Đặc biệt, học sâu đã được sử dụng trong hầu hết mọi giai đoạn trong hệ thống OpenQA và hơn thế nữa, nó cho phép các hệ thống OpenQA end-to-end trainable. Đối với Phân tích câu hỏi, một số công trình phát triển bộ phân loại thần kinh để xác định loại câu hỏi.Ví dụ: [13] và [14] lần lượt áp dụng mô hình dựa trên CNN và dựa trên LSTM để phân loại các câu hỏi đã cho, cả hai đều đạt được kết quả cạnh tranh. Đối với Truy xuất tài liệu, các phương pháp dựa trên đại diện dày đặc [16], [29], [30], [35] đã được đề xuất để giải quyết “sự không khớp về thời hạn”, đây là một vấn đề lâu dài gây hại cho việc truy xuất. Không giống như các phương pháp truyền thống như TF-IDF và BM25 sử dụng các biểu diễn thưa thớt, các phương pháp truy xuất sâu học cách mã hóa các câu hỏi và tài liệu vào một không gian vectơ tiềm ẩn nơi có thể đo lường ngữ nghĩa văn bản ngoài đối sánh thuật ngữ. Ví dụ: [29] và [35] đào tạo các bộ mã hóa của riêng họ để mã hóa từng tài liệu và câu hỏi độc lập thành các vectơ dày đặc, và điểm giống nhau giữa chúng được tính bằng cách sử dụng tích bên trong của các vectơ của chúng. Thuật toán Tìm kiếm sản phẩm bên trong tối đa tuyến tính (MIPS) [69], [70], [71] được sử dụng để cải thiện hiệu quả truy xuất cho một câu hỏi, đặc biệt khi kho tài liệu có quy mô lớn. Đối với Giải nén câu trả lời, là giai đoạn quyết định để hệ thống OpenQA đi đến câu trả lời cuối cùng, các mô hình thần kinh cũng có thể được áp dụng. Việc trích xuất câu trả lời từ một số tài liệu có liên quan cho một câu hỏi nhất định về cơ bản thực hiện nhiệm vụ của Machine Reading Comprehen (MRC). Trong vài năm qua, với sự xuất hiện của một số bộ dữ liệu quy mô lớn như CNN / Daily Mail [18], MS MARCO [20], RACE [21] và SQuAD 2.0 [22], nghiên cứu về MRC thần kinh đã đạt được những tiến bộ đáng kể. [24], [25], [26], [27]. Ví dụ: BiDAF [24] đại diện cho tài liệu đã cho ở các mức độ chi tiết khác nhau thông qua cấu trúc phân cấp nhiều giai đoạn bao gồm lớp nhúng ký tự, lớp nhúng từ và lớp nhúng theo ngữ cảnh và tận dụng cơ chế luồng chú ý hai chiều để có được bản trình bày tài liệu nhận biết câu hỏi mà không cần tóm tắt sớm. QANet [26] sử dụng CNN và cơ chế tự chú ý [72] để lập mô hình tương ứng các hành động nội bộ và tương tác toàn cầu, hoạt động nhanh hơn đáng kể so với các mô hình lặp lại thông thường.

Hơn nữa, việc áp dụng học sâu cho phép các hệ thống OpenQA có thể đào tạo từ đầu đến cuối [15], [30], [37]. Ví dụ: [37] cho rằng việc kết hợp một hệ thống IR độc lập trong một hệ thống OpenQA là không tối ưu và họ phát triển một hệ thống ORQA coi việc truy xuất tài liệu từ nguồn thông tin là một biến tiềm ẩn và chỉ đào tạo toàn bộ hệ thống từ câu hỏi. -answer cặp chuỗi dựa trên BERT [27]. REALM [30] là một mô hình ngôn ngữ được đào tạo trước có chứa bộ truy xuất kiến thức và bộ mã hóa tăng cường kiến thức. Cả bộ truy xuất và bộ mã hóa của nó đều là các mạng nơ-ron đáng tin cậy khác nhau, có thể tính toán gradient wrt các tham số mô hình để được truyền ngược lại trong toàn bộ mạng. Tương tự như các mô hình ngôn ngữ đào tạo trước khác, nó cũng có hai giai đoạn, tức là đào tạo trước và tinh chỉnh. Trong giai đoạn đào tạo trước, mô hình được đào tạo theo cách không giám sát, sử dụng mô hình ngôn ngữ được che mặt làm tín hiệu học tập trong khi các thông số được tinh chỉnh bằng cách sử dụng các ví dụ được giám sát trong giai đoạn tinh chỉnh.

Trong các hệ thống OpenQA đầu tiên, sự thành công của việc trả lời một câu hỏi phụ thuộc nhiều vào hiệu suất của Phân tích câu hỏi, đặc biệt là Phân loại câu hỏi, chuyên cung cấp các loại câu trả lời mong đợi [47]. Tuy nhiên, một trong hai loại hoặc phân loại câu hỏi được các nhà ngôn ngữ học làm thủ công, không phải là tối ưu vì không thể bao quát được tất cả các dạng câu hỏi trong thực tế, đặc biệt là những dạng phức tạp. Hơn nữa, các lỗi phân loại sẽ dễ dẫn đến việc trích xuất câu trả lời không thành công, do đó ảnh hưởng nghiêm trọng đến hiệu suất chung của hệ thống. Theo các thí nghiệm trong [47], khoảng 36,4% lỗi trong các hệ thống OpenQA ban đầu là do không phân loại được các loại câu hỏi. Các mô hình thần kinh có thể tự động chuyển đổi các nhiệm vụ từ ngôn ngữ tự nhiên thành các biểu diễn dễ nhận biết hơn đối với máy móc. Hơn nữa, các mô hình MRC thần kinh cung cấp một giải pháp mạnh mẽ chưa từng có cho lực kéo Answer Ex trong OpenQA, phần lớn bù đắp sự cần thiết của việc áp dụng các kỹ

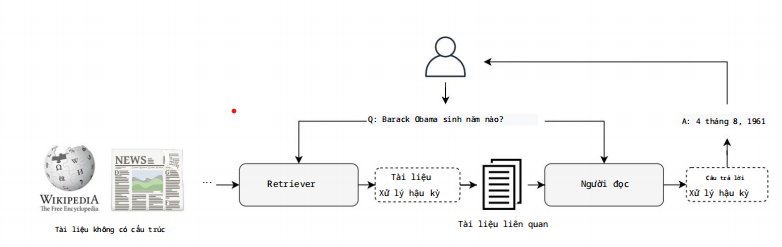
thuật phân tích ngôn ngữ truyền thống cho các câu hỏi và mang lại cuộc cách mạng cho các hệ thống OpenQA [3], [28], [29], [ 37]. Công trình đầu tiên kết hợp các mô hình MRC thần kinh vào hệ thống OpenQA là DrQA do [3] đề xuất, phát triển thành kiến trúc “Retriever-Reader”. Nó kết hợp kỹ thuật IR dựa trên TF-IDF và mô hình MRC thần kinh để trả lời các câu hỏi dạng thừa miền mở trên Wikipedia và đạt được hiệu suất ấn tượng. Sau [3], rất nhiều tác phẩm đã được phát hành [28], [30], [33], [34], [37], [73], [74], [75]. Ngày nay, để xây dựng các hệ thống OpenQA theo kiến trúc “Retriever-Reader” đã được nhiều người thừa nhận là cách hiệu quả nhất và đầy hứa hẹn, đây cũng là trọng tâm chính của bài báo này.

1. **OPENQA HIỆN ĐẠI: QUAY LẠI VÀ ĐỌC**

Trong phần này, chúng tôi giới thiệu tài liệu lưu trữ “Retriever-Reader” của hệ thống OpenQA, như được minh họa trong Hình 3. Retriever nhằm mục đích truy xuất các tài liệu có liên quan từ một câu hỏi nhất định, có thể được coi là hệ thống IR, trong khi Reader nhằm mục đích suy ra câu trả lời cuối cùng từ các tài liệu đã nhận,thường là mô hình MRC thần kinh. Chúng là hai thành phần chính của hệ thống OpenQA hiện đại. Ngoài ra, một số mô-đun phụ trợ khác, được đánh dấu bằng các đường gạch ngang trong Hình 3, cũng có thể được kết hợp vào hệ thống OpenQA, bao gồm Xử lý sau tài liệu để lọc và xếp hạng lại các tài liệu đã truy xuất theo cách chi tiết để chọn các câu trả lời phù hợp nhất và xử lý Bài trả lời nhằm xác định câu trả lời cuối cùng trong số nhiều ứng viên trả lời. Các hệ thống tuân theo kiến trúc này có thể được phân thành hai nhóm, tức là hệ thống đường ống và hệ thống đầu cuối. Trong phần sau, chúng tôi sẽ giới thiệu từng thành phần với các cách tiếp cận tương ứng trong hệ thống đường ống, sau đó là các cách tiếp cận đầu cuối có thể đào tạo. Trong Hình 4, chúng tôi cung cấp phân loại của hệ thống OpenQA hiện đại để làm cho các mô tả của chúng tôi dễ hiểu hơn.

**3.1. Retriever ( Người truy xuất)**

Retriever thường được coi là một hệ thống IR, với mục tiêu truy xuất các tài liệu hoặc đoạn văn liên quan có thể chứa câu trả lời chính xác theo ngôn ngữ tự nhiên cũng như xếp hạng chúng theo thứ tự giảm dần theo mức độ liên quan của chúng. Nói chung, các phương pháp tiếp cận hiện tại đối với Retriever có thể được phân loại thành ba loại, tức là Sprint Retriever, Dense Retriever và Iterative Retriever, sẽ được trình bày chi tiết trong phần sau.



Hình 3: Một minh họa về kiến trúc “Retriever-Reader” của hệ thống OpenQA. Các mô-đun được đánh dấu bằng đường gạch ngang là phụ trợ.

**3.1.1. Sparse Retriever (Bộ truy xuất thưa thớt)**

Nó đề cập đến các hệ thống tìm kiếm các tài liệu liên quan bằng cách áp dụng các phương pháp IR cổ điển như được giới thiệu trong Phần 2.2.2, chẳng hạn như TF-IDF [3], [34], [76], [77] và BM25 [78], [79]. DrQA [3] là cách tiếp cận đầu tiên đối với các hệ thống OpenQA hiện đại và được phát triển bằng cách kếthợp các kỹ thuật IR cổ điển và mô hình MRC thần kinh để trả lời các câu hỏi dạng nhân tố miền mở. Đặc biệt, bộ truy xuất trong DrQA sử dụng phép băm bi-gram [80] và khớp TF-IDF để tìm kiếm trên Wikipedia, với một câu hỏi ngôn ngữ tự nhiên. BERTserini [78] sử dụng Anserini [81] làm thành viên chính của mình, đây là một bộ công cụ IR mã nguồn mở dựa trên Lucene. Trong [78], các chi tiết khác nhau của văn bản bao gồm cấp độ tài liệu, cấp độ đoạn văn và cấp độ câu được điều tra bằng thực nghiệm và kết quả cho thấy chỉ số cấp độ đoạn văn đạt được hiệu suất tốt nhất. Các phương pháp truy xuất truyền thống như TF-IDF và BM25 sử dụng các biểu diễn thưa thớt để kết hợp các điều khoản chắc chắn. Tuy nhiên, các thuật ngữ được sử dụng trong các câu hỏi của người dùng thường không giống với các thuật ngữ xuất hiện trong tài liệu.[29],[30],[35] đã được phát triển trong những năm gần đây, giúp hóa các câu hỏi và tài liệu thành một không gian vectơ tiềm ẩn nơi có thể đo lường ngữ nghĩa của văn bản ngoài đối sánh thuật ngữ.

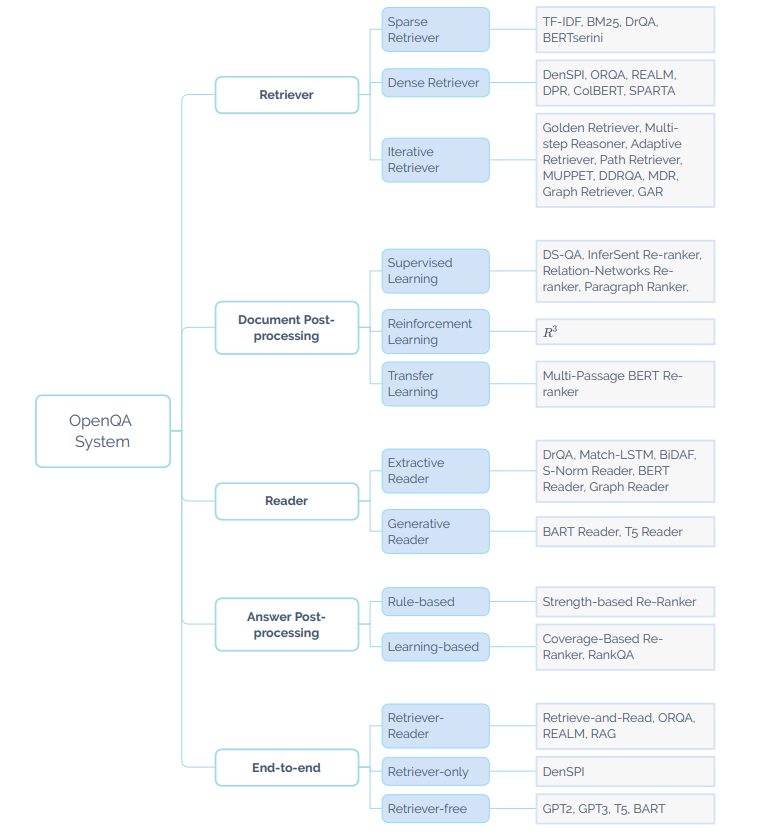
**3.1.2. Dense Retriever (Máy thu thập mật độ)**

Cùng với sự thành công của học sâu cung cấp biểu diễn ngữ nghĩa có thể đánh dấu lại, các mô hình truy xuất sâu khác nhau đã được phát triển trong vài năm qua, nâng cao đáng kể hiệu quả truy xuất và do đó nâng cao hiệu suất QA cuối cùng. Theo các cách khác nhau để mã hóa câu hỏi và tài liệu cũng như tính điểm tương đồng của chúng, các bộ truy xuất dày đặc trong các hệ thống OpenQA hiện tại có thể được chia thành ba loại: Bộ truy xuất dựa trên đại diện [16], [30], [37], [ 73], Trình truy xuất dựa trên tương tác [15], [32] và Trình truy xuất tương tác đại diện [17], [82], [83], như được minh họa trong Hình 5.

Trình truy xuất dựa trên đại diện: Trình truy xuất dựa trên đại diện, còn được gọi là Bộ mã hóa kép hoặc Trình truy xuất hai tháp, sử dụng hai bộ mã hóa độc lập như BERT [27] để mã hóa câu hỏi và tài liệu tương ứng và ước tính mức độ liên quan của chúng bằng cách tính một điểm tương tự duy nhất. tween hai đại diện. Ví dụ: ORQA [37] thông qua hai bộ mã hóa dựa trên BERT độc lập để mã hóa một câu hỏi và một tài liệu tương ứng và điểm phù hợp giữa chúng được tính bằng sản phẩm bên trong của các mã vec của chúng. Để có được một con chó tha mồi đủ mạnh, họ đào tạo trước cho con chó săn mồi bằng cách sử dụng Inverse Cloze Task (ICT), tức là, để dự đoán ngữ cảnh của nó cho một câu. DPR [16] cũng sử dụng hai bộ mã hóa BERT độc lập như ORQA nhưng phủ nhận sự cần thiết của giai đoạn đào tạo trước tốn kém. Thay vào đó, nó tập trung vào việc học một loài chó săn khỏe mạnh bằng cách sử dụng các câu hỏi và câu trả lời theo từng cặp. DPR cẩn thận thiết kế các cách để chọn các mẫu phủ định cho một câu hỏi, bao gồm bất kỳ tài liệu ngẫu nhiên nào từ kho tài liệu, các tài liệu hàng đầu do BM25 trả về không chứa câu trả lời chính xác và phủ định hàng loạt là các tài liệu vàng được ghép nối với các câu hỏi khác trong cùng một đợt. Điều đáng nói là các thử nghiệm của họ cho thấy hàm sản phẩm bên trong là tối ưu để tính điểm tương tự cho bộ truy xuất mã hóa kép. Phương pháp dựa trên biểu diễn [16], [16], [30], [37] có thể rất nhanh vì các biểu diễn của tài liệu có thể được đưa vào và lập chỉ mục ngoại tuyến trước. Nhưng nó có thể hy sinh hiệu quả truy xuất vì các đại diện của câu hỏi và tài liệu được thu thập một cách độc lập, dẫn đến chỉ có các tương tác nông được ghi lại giữa chúng.

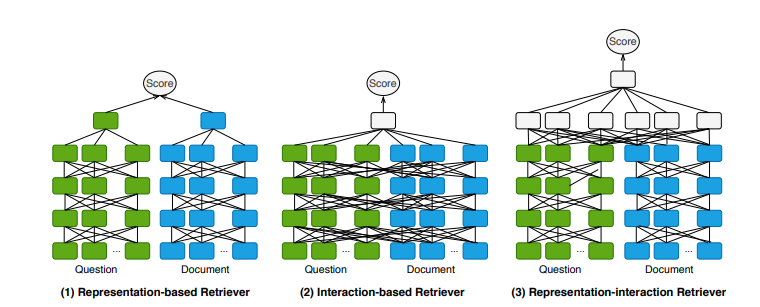
Trình truy xuất dựa trên tương tác: Loại trình truy xuất như

vậy đưa một câu hỏi cùng với một tài liệu cùng lúc với đầu vào và rất mạnh mẽ bằng cách thường mô hình hóa các tương tác cấp mã thông báo giữa chúng, chẳng hạn như bộ mã hóa dựa trên máy biến áp [27], [72] . [15] đề xuất cùng đào tạo Retriever và Reader bằng cách sử dụng phương pháp học đa tác vụ có giám sát [24]. Dựa trên BiDAF [24], một lớp truy xuất được thêm vào để tính điểm mức độ liên quan giữa câu hỏi và tài liệu trong khi lớp hiểu được sử dụng để dự đoán vị trí bắt đầu và kết thúc của khoảng câu trả lời trong [15]. [32] phát triển Retriever dày đặc cấp độ đoạn văn và Retriever mật độ cấp độ câu, cả hai đều dựa trên BERT [27]. Họ coi quá trình truy xuất dày đặc như một vấn đề phân loại nhị phân. Đặc biệt, họ lấy từng cặp câu hỏi và tài liệu làm đầu vào và sử dụng việc nhúng mã thông báo [CLS] để xác định xem chúng có liên quan hay không. Các thí nghiệm của họ cho thấy rằng cả truy xuất cấp độ đoạn văn và cấp độ câu đều cần thiết để đạt được hiệu suất tốt của hệ thống. Phương pháp dựa trên tương tác [15], [32] rất mạnh mẽ vì nó cho phép tạo ra các tương tác rất phong phú giữa câu hỏi và tài liệu. Tuy nhiên, một phương pháp như vậy thường đòi hỏi nhiều tính toán, đôi khi rất tốn kém, khiến nó khó có thể phù hợp với các tài liệu quy mô lớn.



Hình 4: Một phân loại của hệ thống OpenQA “Retriever-Reader”.

Trình truy xuất tương tác-đại diện: Để hệ thống [17], [82], [83] kết hợp dựa trên đại diện và các phương pháp dựa trên tương tác. Ví dụ: ColBERT-QA [17] phát triển trình phục hồi của nó dựa trên ColBERT [84], mở rộng kiến trúc bộ mã hóa kép bằng cách thực hiện một mã thông báo đơn giản tương tác mức độ qua các lần gửi câu hỏi và trả lời tài liệu để tính điểm tương tự. Tương tự như DPR [16], ColBERT-QA đầu tiên mã hóa câu hỏi và tài liệu một cách phụ thuộc với hai bộ mã hóa BERT. Về mặt chính thức, đã cho một câu hỏi q và một tài liệu d, với các vectơ tương ứng được ký hiệu là Eq (độ dài n) và Ed (độ dài m),



Hình 5: Ba loại của dense retrievers

mức độ liên quan điểm giữa chúng được tính như sau:



Sau đó, ColBERT tính toán điểm của mỗi mã thông báo nhúng ding của câu hỏi trên tất cả các điểm của tài liệu trước tiên, và sau đó tổng tất cả các điểm này thành điểm số liên quan cuối cùng là mười bảy q và d. Như một ví dụ khác, SPARTA [82] phát triển một trình xếp hạng thần kinh để tính điểm phù hợp ở cấp độ mã thông báo bằng cách sử dụng sản phẩm chấm giữa câu hỏi được mã hóa không theo ngữ cảnh (ví dụ: nhúng từ BERT) và tài liệu được mã hóa theo ngữ cảnh (ví dụ: bộ mã hóa BERT). Cụ thể, với các đại diện của câu hỏi và tài liệu, trọng lượng của mỗi mã thông báo câu hỏi được tính toán theo tuần tự max-pooling, ReLU và nhật ký; điểm mức độ liên quan cuối cùng là tổng trọng lượng của mỗi mã thông báo câu hỏi. Phương pháp đại diện-tương tác là một cách tiếp cận đầy hứa hẹn để truy xuất dày đặc, do sự cân bằng tốt giữa hiệu quả và hiệu quả. Nhưng nó vẫn cần được khám phá thêm.

Mặc dù hiệu quả, Dense Retriever thường phải chịu gánh nặng

tính toán nặng nề khi áp dụng cho các tài liệu quy mô lớn. Để tăng tốc độ tính toán, một số công trình đề xuất tính toán và lưu trữ các biểu diễn của tất cả các tài liệu ngoại tuyến trước[16], [29], [30], [35], [37]. Bằng cách này, các biểu diễn này sẽ không bị thay đổi sau khi được tính toán, có nghĩa là các tài liệu được mã hóa phụ thuộc vào câu hỏi, ở một mức độ nào đó sẽ hy sinh hiệu quả của việc truy xuất.

**3.1.3. Interative Retriever (Trình truy xuất lặp lại)**

Iterative Retriever nhằm mục đích tìm kiếm các tài liệu có liên

quan từ một bộ sưu tập lớn theo nhiều bước cho một câu hỏi, còn được gọi là Multi-step Retriever. Nó đã được khuyến khích rộng rãi trong vài năm qua [29], [35], [36], [85], [86], [87], [88], [89], đặc biệt là khi trả lời những câu hỏi phức tạp như thế đòi

hỏi suy luận nhiều bước [90], [91]. Để có đủ số lượng tài liệu liên quan, các truy vấn tìm kiếm cần thay đổi theo các bước khác nhau và được định dạng lại dựa trên thông tin ngữ cảnh trong bước previ ous. Trong phần sau, chúng tôi sẽ giải thích kỹ hơn về Lặp lại. Retriever dựa trên quy trình làm việc của nó: 1) Truy xuất Tài liệu: các kỹ thuật IR được sử dụng để truy xuất tài liệu trong mỗi bước truy xuất; 2) Định dạng lại truy vấn: cơ chế được sử dụng để tạo truy vấn cho mỗi lần truy xuất; 3) Cơ chế dừng truy xuất: phương pháp để quyết định thời điểm kết thúc quá trình truy xuất.

Truy xuất tài liệu: Trước tiên, chúng tôi xem lại các kỹ thuật IR được sử dụng để truy xuất tài liệu trong mỗi bước truy xuất được đưa ra một truy vấn. Một số tác phẩm [36], [86], [89] áp dụng lặp lại Sparse Retriever, và một số [29], [35], [85], [88] sử dụng Dense Retriever một cách tương tác. Trong số các tác phẩm sử dụng Sparse Retriever, GOLDEN Retriever [36] sử dụng tính năng truy xuất BM25, trong khi Graph Retriever [89] và DDRQA [86] truy xuất các tài liệu K hàng đầu bằng cách sử dụng TF-IDF. Đối với những người có Dense Retriever, hầu hết các hệ thống trước đó bao gồm Multi-step Reasoner [29], MUPPET [35] và MDR [85] sử dụng tính năng truy xuất MIPS để có được các tài liệu tương tự về mặt cơ bản nhất đưa ra đại diện cho câu hỏi; Path Retriever [88] phát triển truy xuất Mạng Nơron Tái tạo (RNN) để tìm hiểu cách truy xuất các đường lập luận cho một câu hỏi trên biểu đồ Wikipedia, được xây dựng để mô hình hóa mối quan hệ giữa các đoạn văn dựa trên các siêu liên kết Wikipedia và cấu trúc bài viết.

Định dạng lại truy vấn: Để có đủ số lượng tài liệu liên quan, các truy vấn tìm kiếm được sử dụng cho mỗi bước truy xuất thường rất đa dạng và được tạo dựa trên truy vấn trước đó và các tài liệu được truy xuất. Các truy vấn được tạo ra lấy mỗi từ hai dạng: 1) dạng rõ ràng, tức là một truy vấn ngôn ngữ tự nhiên [36], [86], [87]; và 2) dạng ẩn, tức là dạng biểu diễn dày đặc [29], [35], [85]

Một số tác phẩm tạo ra một truy vấn mới ở dạng ngôn ngữ tự nhiên. Ví dụ: GOLDEN Retriever [36] sử dụng tác vụ định dạng lại truy vấn như một tác vụ MRC vì cả hai đều lấy một câu hỏi và một số tài liệu ngữ cảnh làm đầu vào và nhằm mục đích tạo một chuỗi bằng ngôn ngữ tự nhiên. Thay vì theo đuổi một câu trả lời trong MRC, mục tiêu để điều chế bộ lọc truy vấn là một truy vấn mới giúp thu được nhiều tài liệu hỗ trợ hơn trong bước truy xuất tiếp theo. GAR [87] phát triển mô-đun mở rộng truy vấn bằng cách sử dụng mô hình Seq2Seq BART [92] được đào tạo trước, lấy câu hỏi ban đầu làm đầu vào và tạo ra các truy vấn mới. Nó được đào tạo bằng cách lấy nhiều generation nhắm mục tiêu là đầu ra bao gồm câu trả lời, câu thuộc về câu trả lời và tiêu đề của đoạn văn có chứa câu trả lời.

Một số tác phẩm khác tạo ra các đại diện dày đặc được sử dụng để tìm kiếm trong một không gian tiềm ẩn. Ví dụ: Multi-step Reasoner [29] sử dụng Đơn vị lặp lại Gated (GRU) [93] lấy biểu diễn ẩn cấp mã thông báo từ Reader và câu hỏi làm đầu vào để tạo vectơ truy vấn mới, được đào tạo bằng cách sử dụng Học tăng cường (RL) bằng cách đánh giá xem câu trả lời do Reader trích xuất phù hợp với sự thật cơ bản như thế nào sau khi đọc nhóm đoạn văn mới đã đạt được câu hỏi mới. MUPPET [35] áp dụng lớp chú ý lặp lại bidi được điều chỉnh từ [94] sang biểu diễn truy vấn mới q˜, lấy mỗi đoạn P thu được và câu hỏi ban đầu Q làm đầu vào. MDR [85] sử dụng mô hình ngôn ngữ có mặt nạ được đào tạo trước (chẳng hạn như RoBert) làm bộ mã hóa của nó, mã hóa phần nối của phần biểu diễn câu hỏi cũng như tất cả các đoạn trước đó dưới dạng một truy vấn dày đặc mới.

So sánh, truy vấn rõ ràng dễ hiểu và có thể kiểm soát được đối với con người nhưng bị hạn chế bởi các thuật ngữ trong từ vựng, trong khi truy vấn ngầm được tạo ra trong một không gian ngữ nghĩa, có thể thoát khỏi giới hạn của từ vựng nhưng thiếu khả năng diễn giải.

Cơ chế dừng truy xuất: Cách thức truy xuất lặp lại mang lại nhiều khả năng hơn để thu thập nhiều đoạn có liên quan hơn, nhưng hiệu quả truy xuất sẽ giảm nghiêm trọng cùng với số lần lặp lại ngày càng tăng. Đề phòng cơ chế dừng truy xuất lặp lại, hầu hết các hệ thống hiện có đều chọn chỉ định số lần lặp cố định [29], [36], [85], [86], [89] hoặc số lượng tài liệu được truy xuất tối đa [35] , [87], khó có thể đảm bảo hiệu quả truy xuất. [77] cho rằng việc thiết lập một số lượng tài liệu cố định cần lấy cho tất cả các câu hỏi đầu vào là không tối ưu và thay vào đó, họ phát triển một biện pháp Thích ứng Re dựa trên Trình thu thập tài liệu trong DrQA [3]. Họ đề xuất hai phương pháp để đặt động số lượng tài liệu đã đạt được cho mỗi câu hỏi, tức là một phương pháp heuristic dựa trên ngưỡng đơn giản cũng như một bộ phân loại có thể đào tạo bằng cách sử dụng hồi quy sườn núi theo thứ tự. Vì đối với các câu hỏi yêu cầu các bước suy luận tùy ý, rất khó để chỉ định số lần lặp lại, Path Retriever [88] chỉ chấm dứt việc truy xuất khi mã thông báo kết thúc bằng chứng (ví dụ: [EOE]) được phát hiện bởi Recurrent Retriever của nó. . Điều này cho phép nó thực hiện các bước truy xuất thích ứng nhưng chỉ thu được một tài liệu ở mỗi bước. Theo hiểu biết tốt nhất của chúng tôi, vẫn là một thách thức quan trọng để phát triển một bộ truy xuất lặp đi lặp lại hiệu quả trong khi không hy sinh độ chính xác.

Ngoài ra, các hệ thống IR điển hình theo đuổi hai mục tiêu tối ưu hóa, tức là độ chính xác và khả năng thu hồi. Hệ số tính toán tỷ lệ tài liệu liên quan được trả lại trên tổng số tài liệu được trả lại trong khi tính toán thứ hai là số lượng tài liệu liên quan được trả lại trong tổng số tài liệu có liên quan trong kho lưu trữ bên dưới. Tuy nhiên, đối với hệ thống OpenQA, việc thu hồi quan trọng hơn nhiều so với độ chính xác do quá trình xử lý sau thường được áp dụng cho các tài liệu bị trả lại [95], như được mô tả bên dưới.

**3.2 Xử lý sau tài liệu**

Xử lý hậu kỳ đối với các tài liệu được truy xuất từ Retriever thường là cần thiết vì các tài liệu được truy xuất chắc chắn chứa những tài liệu không liên quan, và đôi khi, số lượng tài liệu được lật lại rất lớn, lấn át khả năng của Reader. Tài liệu xử lý hậu kỳ trong kiến trúc OpenQA hiện đại tương tự như trong kiến trúc truyền thống, như đã giới thiệu trong Phần 2.2.2. Nó nhằm mục đích giảm số lượng tài liệu ứng viên và chỉ cho phép những tài liệu phù hợp nhất được chuyển sang giai đoạn tiếp theo.

Trong vài năm qua, mô-đun này đã được khám phá với nhiều sự quan tâm [28], [33], [34], [79], [96], [97], [98], [99]. Ví dụ: R3 [28] sử dụng Trình xếp hạng thông qua thần kinh và đào tạo nó cùng với Reader thông qua Học tăng cường (RL). DS-QA [33] thêm Bộ chọn đoạn để loại bỏ những đoạn nhiễu khỏi các tài liệu được truy xuất bằng cách đo xác suất của mỗi đoạn có chứa câu trả lời trong số tất cả các đoạn ứng viên. [96] khám phá hai công cụ xếp hạng đoạn văn khác nhau chỉ định điểm số cho các đoạn văn đã truy xuất dựa trên khả năng chứa câu trả lời cho một câu hỏi nhất định. Một là InferSent Ranker, một mạng nơ-ron chuyển tiếp sử dụng các biểu diễn câu InferSent [100], để xếp hạng các đoạn văn dựa trên sự tương đồng về ngữ nghĩa với câu hỏi.Công cụ còn lại là Relation-Networks Ranker sử dụng Mạng quan hệ [101], tập trung vào việc đo lường mức độ liên quan ở cấp độ từ giữa câu hỏi và các đoạn văn. Các thử nghiệm của họ cho thấy rằng đối sánh mức độ liên quan cấp độ từ cải thiện đáng kể hiệu suất truy xuất và sự tương đồng về ngữ nghĩa có lợi hơn cho hiệu suất tổng thể. [34] phát triển Trình xếp hạng đoạn bằng cách sử dụng hai RNN riêng biệt để hạ thấp kiến trúc bộ mã hóa kép. Mỗi cặp đoạn câu hỏi được đưa vào Trình xếp hạng để có được các đại diện của chúng một cách độc lập và sản phẩm bên trong được áp dụng để tính điểm mức độ phù hợp. [98] đề xuất một mô-đun xếp hạng lại nhận thức về thời gian kết hợp thông tin tạm thời từ các khía cạnh khác nhau để xếp hạng các tài liệu ứng viên trên bộ sưu tập tạm thời của các bài báo.

Trọng tâm của nghiên cứu về mô-đun này là học cách xếp hạng lại các tài liệu đã truy xuất [28], [33], [34], [79]. Tuy nhiên, với sự phát triển của Dense Retriever, các hệ thống OpenQA gần đây có xu hướng phát triển một bộ truy xuất có thể đào tạo có khả năng học cách xếp hạng và truy xuất các tài liệu liên quan nhất đồng thời, điều này sẽ dẫn đến việc không có mô-đun này.

**3.3. Reader (Đầu đọc)**

Đầu đọc là thành phần cốt lõi khác của hệ thống OpenQA hiện đại và cũng là tính năng chính giúp phân biệt hệ thống QA với các hệ thống IR hoặc IE khác, thường được triển khai dưới dạng mô hình MRC thần kinh. Nó nhằm mục đích suy ra câu trả lời cho câu hỏi từ một tập hợp các tài liệu có thứ tự và khó hơn so với nhiệm vụ MRC ban đầu khi chỉ đưa ra một đoạn văn trong hầu hết các trường hợp [18], [19], [90 ], [102], [103]. Nói chung, Trình đọc hiện có có thể được phân loại thành hai loại: Trình đọc chiết xuất dự đoán khoảng câu trả lời từ các tài liệu được truy xuất và Trình đọc tạo ra câu trả lời bằng ngôn ngữ tự nhiên bằng cách sử dụng mô hình trình tự-chuỗi (Seq2Seq). Hầu hết các hệ thống OpenQA trước đây đều được trang bị. Ví dụ: R3 [28] sử dụng Trình xếp hạng thông qua thần kinh và đào tạo nó cùng với Reader thông qua Học tăng cường (RL). DS-QA [33] thêm Bộ chọn đoạn để loại bỏ những đoạn nhiễu khỏi các tài liệu được truy xuất bằng cách đo xác suất của mỗi đoạn có chứa câu trả lời trong số tất cả các đoạn ứng viên. [96] khám phá hai công cụ xếp hạng đoạn văn khác nhau chỉ định điểm số cho các đoạn văn đã truy xuất dựa trên khả năng chứa câu trả lời cho một câu hỏi nhất định. Một là InferSent Ranker, một mạng nơ-ron chuyển tiếp sử dụng các biểu diễn câu InferSent [100], để xếp hạng các đoạn văn dựa trên sự tương đồng về ngữ nghĩa với câu hỏi. Đầu đọc là thành phần cốt lõi khác của hệ thống OpenQA hiện đại và cũng là tính năng chính giúp phân biệt hệ thống QA với các hệ thống IR hoặc IE khác, thường được triển khai dưới dạng mô hình MRC thần kinh. Nó nhằm mục đích đưa ra câu trả lời cho câu hỏi từ một tập hợp các tài liệu theo thứ tự và khó khăn hơn so với nhiệm vụ MRC ban đầu khi chỉ đưa ra một đoạn văn trong hầu hết các trường hợp [18], [19], [ 90], [102], [103]. Nói chung, các Trình đọc hiện có có thể được phân loại thành hai loại: Trình đọc Extrac tive dự đoán khoảng câu trả lời từ các tài liệu được truy xuất và Trình đọc tạo ra các câu trả lời bằng ngôn ngữ tự nhiên bằng cách sử dụng mô hình trình tự nối tiếp (Seq2Seq). Hầu hết các hệ thống OpenQA trước đây đều được trang bị Trình đọc trích xuất [3], [16], [28], [29], [30], [33], [35], [78], [89], trong khi một số hệ thống gần đây phát triển một Generative Reader [38], [85], [104].

**3.3.1. Extractive Reader (Đầu đọc giải nén)**

Extractive Reader dựa trên giả định rằng câu trả lời chính xác cho một câu hỏi nhất định chắc chắn tồn tại trong ngữ cảnh và thường tập trung vào việc học cách dự đoán vị trí bắt đầu và kết thúc của khoảng câu trả lời từ các tài liệu được truy xuất. Các phương pháp tiếp cận có thể được chia thành hai loại tùy theo việc các tài liệu truy xuất được xử lý độc lập hay chung để trích xuất câu trả lời.

Nhiều hệ thống trước đó [16], [33], [86], [88], [89] xếp hạng các tài liệu được truy xuất theo xác suất bao gồm câu trả lời và trích ra khoảng câu trả lời từ phần nối câu hỏi và tài liệu có khả năng xảy ra cao nhất ( S). Ví dụ: DS-QA [33] trích xuất một khoảng câu trả lời từ đoạn văn được chọn bởi một module Paragraph Selector chuyên dụng thông qua việc đo xác suất của mỗi đoạn có chứa câu trả lời trong số tất cả các đoạn ứng viên. DPR [16] tính xác suất của một đoạn văn có chứa câu trả lời và mã thông báo là vị trí bắt đầu và kết thúc của một khoảng câu trả lời bằng cách sử dụng BERT Reader, và chọn câu trả lời có xác suất cao nhất sau khi kết hợp chúng. Một số hệ thống phát triển Trình đọc dựa trên đồ thị [88], [89] để tìm hiểu cách trích xuất một khoảng câu trả lời từ một đồ thị được truy xuất. Ví dụ, Graph Reader [89] lấy biểu đồ làm đầu vào và học cách biểu diễn đoạn văn chủ yếu bằng cách sử dụng Graph Convolution Networks [105] trước, và sau đó trích xuất khoảng câu trả lời từ câu có thể xảy ra nhất. Path Retriever [88] thúc đẩy BERT Reader để đồng thời xếp hạng lại các đường dẫn soning và trích xuất câu trả lời từ khoảng có xác suất cao nhất chứa câu trả lời chính xác mà chúng ta sử dụng trong quá trình học đa tác vụ. Tuy nhiên, với các tài liệu truy xuất được xử lý độc lập, mô hình không tận dụng được các bằng chứng khác nhau từ tài liệu tường thuật dài hoặc nhiều tài liệu để trích xuất câu trả lời, gây hại cho hiệu suất, đặc biệt trong trường hợp các câu hỏi đưa ra yêu cầu suy luận nhiều bước.

Ngược lại, một số hệ thống [3], [78], [82], [94] trích xuất một khoảng câu trả lời dựa trên tất cả các tài liệu được truy xuất theo cách chung. Ví dụ: DrQA [3] phân tách các tài liệu được truy xuất thành các đoạn văn và trích xuất các tính năng khác nhau bao gồm Phần nói (POS), Thực thể được đặt tên (NE) và Tần suất thuật ngữ (TF), v.v. Sau đó, Trình đọc DrQA, là được triển khai với Bi-LSTM nhiều lớp, nhận làm đầu vào cho câu hỏi và các đoạn văn, đồng thời dự đoán khoảng câu trả lời. Trong quá trình này, để làm cho điểm câu trả lời có thể so sánh giữa các đoạn văn, nó sử dụng hàm mũ không chuẩn hóa và lấy argmax trên tất cả các khoảng câu trả lời để thu được kết quả cuối cùng. BERTserini [78] phát triển Trình đọc của mình dựa trên BERT bằng cách loại bỏ lớp softmax để cho phép so sánh và tổng hợp điểm câu trả lời giữa các đoạn văn khác nhau. [94] lập luận bằng cách sử dụng điểm chưa chuẩn hóa (ví dụ: điểm số cũ hoặc điểm logits) cho tất cả các nhịp câu trả lời là tối ưu nhất và đề xuất cơ chế Chia sẻ-Chuẩn hóa bằng cách sửa đổi hàm mục tiêu để chuẩn hóa điểm đầu và điểm cuối trên tất cả các đoạn văn, đạt được đạt được hiệu suất nhất quán. Sau đó, nhiều hệ thống OpenQA [17], [29], [35], [36], [79], [82] phát triển trình đọc của họ bằng cách áp dụng cơ chế này dựa trên các mô hình MRC ban đầu như BiDAF [24], BERT [ 27] và SpanBERT [106].

**3.3.2 Generative Reader**

Generative Reader nhằm tạo ra các câu trả lời tự nhiên nhất có thể thay vì trích xuất các khoảng câu trả lời, thường dựa trên các mô hình Seq2Seq. Ví dụ, S-Net [107] được phát triển bằng cách kết hợp các phương pháp khai thác và tạo để hoàn thiện lẫn nhau. Nó sử dụng mô hình trích xuất bằng chứng để dự đoán ranh giới của khoảng văn bản làm bằng chứng cho câu trả lời trước và sau đó đưa nó vào mô hình tổng hợp câu trả lời Seq2Seq để tạo ra câu trả lời cuối cùng. Gần đây, một số hệ thống OpenQA [38], [85], [104] áp dụng các mô hình ngôn ngữ Seq2Seq được đào tạo trước để phát triển Trình đọc của họ, như BART [92] và T5 [108]. Ví dụ: RAG [38] sử dụng mô hình BART được đào tạo trước làm trình đọc của nó để tạo câu trả lời bằng cách lấy câu hỏi đầu vào cũng như các tài liệu được DPR truy xuất [16]. FID [104] đầu tiên mã hóa từng tài liệu được truy xuất một cách độc lập bằng cách sử dụng bộ mã hóa T5 hoặc BART, sau đó mỗi lần tạo sự chú ý trên tất cả các biểu diễn đầu ra bằng cách sử dụng bộ giải mã để tạo ra câu trả lời cuối cùng. Tuy nhiên, kết quả tạo hiện tại thường bị lỗi cú pháp, không mạch lạc hoặc không logic [109]. Generative Reader cần được khám phá và nâng cao hơn nữa.

**3.4 Trả lời Các kỹ thuật MRC**

Thần kinh xử lý hậu kỳ đã được tiến bộ nhanh chóng trong những năm gần đây, nhưng hầu hết các mô hình MRC hiện tại vẫn chỉ chuyên về việc trích xuất câu trả lời từ một hoặc một số đoạn văn ngắn và có xu hướng thất bại trong trường hợp câu trả lời đúng đến từ các bằng chứng khác nhau trong một văn bản tường thuật hoặc nhiều văn bản [110]. Mô-đun Xử lý sau câu trả lời được phát triển để giúp phát hiện câu trả lời cuối cùng từ một tập hợp các câu trả lời được Người đọc trích xuất, có tính đến các dữ kiện hỗ trợ tương ứng của họ. Các phương pháp được áp dụng trong các hệ thống hiện có có thể được phân loại thành hai phương pháp phân loại, tức là phương pháp dựa trên quy tắc [34], [110] và phương pháp dựa trên học tập [76], [110] Ví dụ, [110] đề xuất hai trình xếp hạng lại câu trả lời , "người xếp hạng lại dựa trên sức mạnh" và "người xếp hạng lại dựa trên mức độ phù hợp", để tổng hợp bằng chứng từ các đoạn khác nhau để quyết định câu trả lời cuối cùng. “Người xếp hạng lại dựa trên sức mạnh” là một phương pháp dựa trên quy tắc chỉ đơn giản là thực hiện tính toán xác suất hoặc đếm dựa trên các lần chọn trước của ứng viên và không yêu cầu bất kỳ đào tạo nào. “Người xếp hạng lại dựa trên phạm vi bảo hiểm” được phát triển bằng cách sử dụng mô hình LSTM đối sánh dựa trên sự chú ý [111]. Đầu tiên nó nối các đoạn văn chứa cùng một câu trả lời thành một đoạn văn giả và sau đó đo lường mức độ hiệu quả của đoạn văn giả này mang lại câu trả lời cho câu hỏi đã cho. Các thí nghiệm trong [110] cho thấy rằng sự kết hợp có trọng số của các kết quả đầu ra của các trình xếp hạng lại khác nhau ở trên sẽ đạt được hiệu suất tốt nhất trên một số điểm chuẩn. RankQA [76] phát triển một mô-đun xếp hạng lại câu trả lời bao gồm ba bước: trích xuất tính năng, tổng hợp câu trả lời và xếp hạng lại. Thứ nhất, lấy k câu trả lời hàng đầu từ Reader làm đầu vào, mô-đun trích xuất một tập hợp các tính năng từ cả Retriever như độ giống nhau của tài liệu-câu hỏi, độ dài câu hỏi và độ dài đoạn văn, và Reader chẳng hạn như điểm ban đầu của ứng viên trả lời, phần- thẻ lời nói của câu trả lời và thực thể được đặt tên của câu trả lời. Thứ hai, các nhóm mô-đun đều trả lời các ứng viên có khoảng trả lời giống hệt nhau để tạo ra một tập hợp các tính năng tổng hợp như tổng, trung bình, tối thiểu và tối đa của điểm khoảng, v.v. Dựa trên các tính năng này, mạng chuyển tiếp hoặc RNN được sử dụng để tìm hiểu cách xếp hạng lại các câu trả lời để chọn câu trả lời đúng nhất làm câu trả lời cuối cùng.

**3.5 Các phương pháp end-to-end**

Trong những năm gần đây, nhiều hệ thống OpenQA khác nhau [15], [30], [37] đã được phát triển, trong đó Retriever và Reader có thể được đào tạo một cách toàn diện. Ngoài ra, có một số hệ thống chỉ có Retriever [73] và cũng có một số hệ thống có thể trả lời các câu hỏi mở mà không cần giai đoạn truy xuất, hầu hết là các mô hình

ngôn ngữ Seq2Seq đã được đào tạo trước [92], [108], [112] , [113]. Sau đây, chúng tôi sẽ giới thiệu ba loại hệ thống này, đó là Retriever-Reader, Retriever-only và Retriever-free.

**3.5.1 Retriever-Reader**

Các kỹ thuật học sâu cho phép Retriever và Reader trong hệ thống OpenQA có thể đào tạo từ đầu đến cuối [15], [30], [37], [38]. Ví dụ, [15] đề xuất cùng đào tạo Retriever và Reader bằng cách sử dụng phương pháp học đa tác vụ dựa trên mô hình BiDAF [24], đồng thời tính toán độ tương tự của một đoạn văn với câu hỏi đã cho và dự đoán vị trí bắt đầu và kết thúc của khoảng câu trả lời. [37] cho rằng việc kết hợp một hệ thống IR độc lập trong một hệ thống OpenQA và phát triển ORQA để đào tạo cùng Retriever và Reader từ các cặp câu hỏi-trả lời là không tối ưu, cả hai đều được phát triển bằng cách sử dụng BERT [27]. REALM [30] là một mô hình ngôn ngữ có mặt nạ được đào tạo trước bao gồm Bộ truy xuất thần kinh và Bộ đọc thần kinh, có thể tính toán gradient wrt các tham số mô hình và sao chép ngược gradient trong toàn bộ mạng. Vì cả hai mô-đun đều được phát triển bằng cách sử dụng mạng nơ-ron, nên tốc độ phản hồi cho một câu hỏi là vấn đề quan trọng nhất trong quá trình suy luận, đặc biệt là trên một bộ sưu tập lớn tài liệu.

**3.5.2 Chỉ sử dụng trình truy xuất**

Để nâng cao hiệu quả trả lời câu hỏi, một số hệ thống được phát triển bằng cách chỉ sử dụng Trình truy xuất trong khi bỏ qua Trình đọc vốn thường là giai đoạn tốn thời gian nhất trong các hệ thống OpenQA hiện đại khác. DenSPI [73] xây dựng chỉ mục nhúng cấp độ cụm từ câu hỏi bất khả tri ngoại tuyến với một bộ sưu tập các tài liệu như các bài viết trên Wikipedia. Theo suy luận, câu hỏi đã cho được mã hóa theo cùng một cách, và FAISS [114] được sử dụng để tìm kiếm cụm từ tương tự nhất làm câu trả lời cuối cùng. Các thử nghiệm cho thấy rằng nó thu được hiệu quả đáng kể và giảm đáng kể chi phí đặt hàng trong khi vẫn duy trì độ chính xác.Tuy nhiên, hệ thống tính toán sự giống nhau giữa từng cụm từ và câu hỏi một cách độc lập, hệ thống này bỏ qua thông tin ngữ cảnh thường rất quan trọng để trả lời câu hỏi.

**3.5.3 Không có trình truy xuất**

Những tiến bộ gần đây trong mod ngôn ngữ Seq2Seq đào tạo trước như GPT-2 [112], GPT-3 [113], BART [92] và T5 [108] mang lại nhiều cải tiến cho các tác vụ NLG hạ lưu, hầu hết trong số đó đã được xây dựng bằng cách sử dụng các biện pháp lưu trữ dựa trên Biến áp. Đặc biệt, GPT-2 và GPT-3 sử dụng Transformer bộ giải mã từ trái sang phải trong khi BART và T5 sử dụng bộ mã hóagiải mã Transformer theo sát hình thức ban đầu của nó [72]. Các nghiên cứu trước đây [115], [116] cho thấy rằng một lượng lớn kiến thức học được từ dữ liệu văn bản quy mô lớn có thể được lưu trữ trong các tham số cơ bản, và do đó các mô hình này chỉ có thể trả lời các câu hỏi mà không cần truy cập vào bất kỳ kiến thức bên ngoài nào. Ví dụ: GPT-2 [112] có thể tạo chính xác câu trả lời chỉ đưa ra một câu hỏi ngôn ngữ tự nhiên mà không cần tinh chỉnh. Sau đó, GPT-3 [113] đạt được hiệu suất cạnh tranh với việc học ít lần so với các phương pháp tinh chỉnh hiện đại nhất trước đây, trong đó một số minh chứng được đưa ra với suy luận là điều kiện trong [112] trong khi cập nhật trọng lượng thì không. cho phép. Gần đây, [116] đánh giá toàn diện khả năng của các mô hình ngôn ngữ để trả lời các câu hỏi mà không cần tiếp cận với bất kỳ kiến thức bên ngoài nào. Các thử nghiệm của họ chứng minh rằng các mô hình ngôn ngữ được đào tạo trước có thể đạt được hiệu suất ấn tượng trên các điểm chuẩn khác nhau và các phương pháp Truy xuất miễn phí như vậy tạo nên một cách tiếp cận khác về cơ bản để xây dựng hệ thống OpenQA. Trong Bảng 1, chúng tôi tóm tắt các hệ thống OpenQA hiện đại hiện có cũng như các phương pháp tiếp cận được áp dụng cho các thành phần khác nhau.

1. **THÁCH THỨC VÀ BENCHMARKS**

Trong phần này, trước tiên chúng ta thảo luận về những thách thức chính đối với việc xây dựng hệ thống OpenQA, sau đó là phân tích các điểm chuẩn QA hiện có thường được sử dụng không chỉ cho OpenQA mà còn cho MRC.

**4.1 Những thách thức đối với OpenQA**

Để xây dựng một hệ thống OpenQA có khả năng trả lời bất kỳ câu hỏi đầu vào nào được coi là mục tiêu cuối cùng của nghiên cứu QA. Tuy nhiên, giới nghiên cứu vẫn còn một chặng đường dài phía trước. Ở đây chúng tôi thảo luận về một số thách thức nổi bật cần được giải quyết trên đường đi. Bằng cách này, chúng tôi hy vọng những khoảng trống trong nghiên cứu có thể được làm rõ hơn để đẩy nhanh tiến độ trong lĩnh vực này.

**4.1.1 Giám sát từ xa**

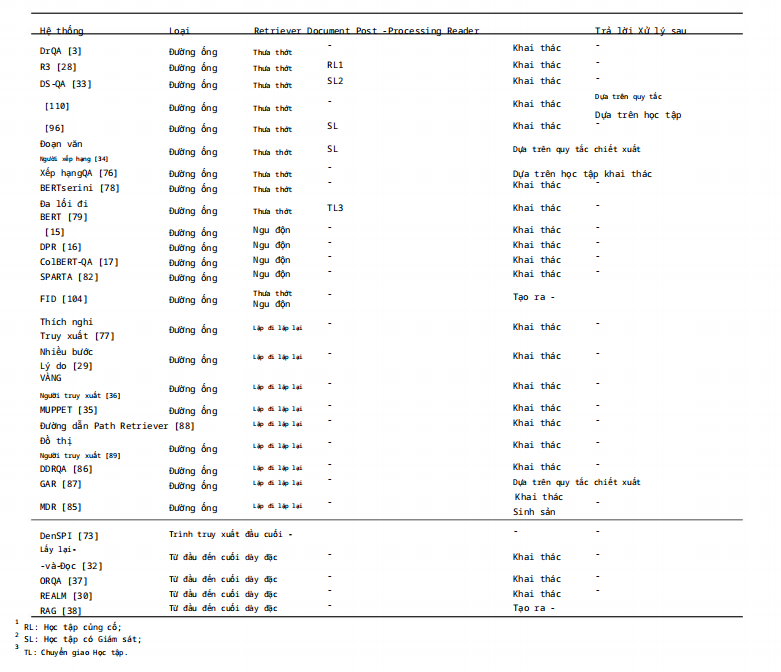
Trong cài đặt OpenQA, hầu như không thể tạo bộ sưu tập chứa dữ liệu đào tạo chất lượng cao “đủ” để phát triển hệ thống OpenQA trước. Do đó, tính năng giám sát từ xa được sử dụng phổ biến, có thể tự động gắn nhãn dữ liệu dựa trên kho dữ liệu hiện có, chẳng hạn như Wikipedia. Tuy nhiên, việc giám sát từ xa chắc chắn gặp phải vấn đề nhãn sai và thường dẫn đến một lượng dữ liệu nhiễu đáng kể, làm tăng đáng kể độ khó của việc lập mô hình và đào tạo. Vì vậy, các hệ thống có thể chịu được tiếng ồn như vậy luôn được yêu cầu.

**4.1.2 Hiệu quả truy xuất và hiệu quả**

Hiệu quả truy xuất có nghĩa là khả năng của hệ thống để tách các tài liệu có liên quan khỏi những tài liệu không liên quan cho một câu hỏi nhất định. Hệ thống thường gặp phải tình trạng "không khớp thuật ngữ", dẫn đến không thể truy xuất các tài liệu liên quan; mặt khác, hệ thống có thể nhận được các tài liệu nhiễu có chứa các thuật ngữ chính xác trong câu hỏi hoặc thậm chí khoảng câu trả lời chính xác, nhưng không liên quan đến câu hỏi. Cả hai vấn đề đều làm tăng độ khó chính xác hiểu ngữ cảnh trong quá trình suy luận câu trả lời. Một số các phương pháp truy xuất thần kinh [15], [16], [30], [37], [73], [117] là được đề xuất gần đây để cải thiện hiệu quả truy xuất. Vì ví dụ, [37] và [30] cùng huấn luyện việc truy xuất và người đọc mô-đun, tận dụng lợi thế của ngôn ngữ được đào tạo trước mô hình và coi mô hình truy xuất như một biến tiềm ẩn. Tuy nhiên, các phương pháp truy xuất thần kinh này thường bị hiệu quả thấp. Một số công trình [15], [16], [37], [117] đề xuất để tính toán trước việc nhúng không phụ thuộc vào câu hỏi cho mỗi tài liệu hoặc cụm từ và xây dựng nhúng chỉ mục một lần. Các thuật toán Tìm kiếm nội bộ sản phẩm tối đa (MIPS) tuyến tính phụ nâng cao [69], [70], [71] thường là được sử dụng để có được K tài liệu liên quan hàng đầu được đưa ra câu hỏi. Tuy nhiên, tốc độ phản hồi vẫn còn một khoảng cách rất lớn từ kỹ thuật IR điển hình khi hệ thống phải đối mặt với bộ tài liệu đồ sộ.

Hiệu quả và hiệu quả truy xuất đều là những yếu tố quan trọng đối với việc triển khai hệ thống OpenQA trong thực tế, đặc biệt là khi nói đến các kịch bản thời gian thực. Làm thế nào để tăng cường liên tục cả hai khía cạnh (cũng với sự đánh đổi tốt) sẽ là một thách thức lâu dài trong quá trình phát triển OpenQA.

BẢNG 1: Các cách tiếp cận được áp dụng cho các thành phần khác nhau của các hệ thống OpenQA hiện đại hiện có.



**4.1.3 Kết hợp tri thức**

Để kết hợp kiến thức ngoài các tài liệu ngữ cảnh và các câu hỏi đưa ra là một cải tiến quan trọng đối với hệ thống OpenQA [7], ví dụ như kiến thức thế giới, kiến thức chung hoặc kiến thức miền cụ thể. Trước khi sử dụng những kiến thức đó,trước tiên chúng ta cần xem xét cách đại diện cho chúng. Có nói chung là hai cách: rõ ràng và ngầm hiểu.

Nói một cách rõ ràng, kiến thức thường được chuyển đổi thành dạng bộ ba và được lưu trữ trong các KB cổ điển chẳng hạn như DBPedia [118], Freebase [119] và Yago2 [120], mà con người dễ dàng hiểu được. Một số QA sớm hệ thống cố gắng kết hợp kiến thức để giúp tìm ratrả lời theo cách này. Ví dụ: IBM Watson DeepQA [58] kết hợp công cụ tìm kiếm Web và KB để cạnh tranh vớinhà vô địch của con người trên chương trình truyền hình Mỹ "Jeopardy";QuASE [121] tìm kiếm danh sách các câu nổi bật nhất từ công cụ tìm kiếm Web (ví dụ: Google.com), và sau đó uti liên kết thực thể qua Freebase [119] để phát hiện câu trả lời chính xác từ các câu đã chọn. Trong những năm gần đây, với sự phổ biến của Mạng Neural Đồ thị (GNN), một số công trình [89], [122], [123] đề xuất thu thập thông tin liên quan không chỉ từ kho văn bản mà còn từ KB để tạo điều kiện cho việc truy xuất bằng chứng và câu hỏi. trả lời. Ví dụ: [122] xây dựng một biểu đồ con dành riêng cho câu hỏi có chứa các lớp từ ngữ liệu, các thực thể và quan hệ từ KB. Sau đó, các phương pháp dựa trên biểu đồ CNN [105], [124], [125] được sử dụng để suy ra câu trả lời cuối cùng qua biểu đồ phụ. Tuy nhiên, cũng có những vấn đề tồn tại đối với việc lưu trữ kiến thức một cách rõ ràng, chẳng hạn như kiến thức không đầy đủ và lỗi thời. Hơn nữa, để xây dựng một KB vừa tốn nhiều công sức và thời gian.

Mặt khác, với phương pháp tiếp cận ngầm, một lượng lớn kiến thức [115] có thể được lưu trữ trong các tham số cơ bản học được từ các văn bản khổng lồ bằng các mô hình ngôn ngữ được đào tạo trước như BERT [27], XLNet [126] và T5 [108] , có thể được áp dụng trơn tru trong các tác vụ hạ nguồn. Gần đây, các mô hình ngôn ngữ đào tạo trước đã được nghiên cứu và ứng dụng phổ biến để phát triển hệ thống OpenQA [16], [30], [32], [37], [78], [87], [88]. Ví dụ, [32], [78], [88] phát triển Reader của họ bằng BERT [27] trong khi [16], [37] sử dụng BERT để phát triển cả Retriever và Reader. Ngoài ra, các mô hình ngôn ngữ được đào tạo trước như GPT-2 [112] có thể tạo ra câu trả lời chỉ cho một câu hỏi ngôn ngữ tự nhiên. Tuy nhiên, những hệ thống như vậy hoạt động giống như một “hộp đen” và gần như không thể biết chính xác kiến thức nào đã được lưu trữ và sử dụng cho một câu trả lời cụ thể. Chúng thiếu khả năng diễn giải, đặc biệt là đối với các ứng dụng trong thế giới thực.

OpenQA nâng cao kiến thức được mong muốn không chỉ vì nó hữu ích trong việc tạo ra câu trả lời mà còn vì nó đóng vai trò là nguồn để diễn giải câu trả lời thu được. Làm thế nào để đại diện và sử dụng đầy đủ kiến thức cho OpenQA vẫn cần nhiều nỗ lực nghiên cứu hơn.

**4.1.4 Đối thoại OpenQA**

Không hội thoại OpenQA bị thách thức bởi một số vấn đề khó giải quyết, chẳng hạn như các từ dài cho một câu hỏi phức tạp (ví dụ: Ai là con trai thứ hai của Thủ tướng đầu tiên của Singapore?) dẫn đến phản hồi không chính xác (ví dụ: Michael Jordan sinh khi nào?) và không đủ kiến thức nền tảng từ người dùng dẫn đến kết quả không hợp lý (ví dụ: Tại sao hôm nay tôi lại bị đau đầu dữ dội?). Những vấn đề này sẽ được giải quyết tốt trong bối cảnh hội thoại.

Các hệ thống hội thoại [150], [151] được trang bị giao diện giống như hội thoại cho phép tương tác giữa người dùng của con người và hệ thống để trao đổi thông tin. Đối với ví dụ câu hỏi phức tạp được đưa ra ở trên, nó có thể được sắp xếp thành hai câu hỏi đơn giản tuần tự: "Thủ tướng đầu tiên của Singapore là ai?" tiếp theo là “Ai là con trai thứ của ông ấy?”. Khi phát hiện ra sự không rõ ràng trong câu hỏi, hệ thống OpenQA hội thoại dự kiến sẽ đưa ra một câu hỏi tiếp theo để làm rõ, chẳng hạn như “Ý bạn là cầu thủ bóng rổ?”. Nếu một câu hỏi không đủ kiến thức cơ bản được đưa ra, một câu hỏi tiếp theo cũng có thể được yêu cầu để thu thập thêm thông tin từ người dùng để đi đến câu trả lời cuối cùng. Để đạt được những mục tiêu này, cần giải quyết ba thách thức lớn.

Đầu tiên, OpenQA đàm thoại phải có khả năng xác định xem một câu hỏi có thể trả lời được hay không, chẳng hạn như phát hiện xem có tồn tại sự mơ hồ trong câu hỏi hay không hoặc liệu ngữ cảnh hiện tại có đủ để tạo ra câu trả lời hay không. Nghiên cứu về các câu hỏi không trả lời được đã thu hút rất nhiều sự chú ý trong sự phát triển của MRC trong vài năm qua [20], [22], [128], [144], [152], [153]. Tuy nhiên, các hệ thống OpenQA hiện tại hiếm khi kết hợp một cơ chế như vậy để xác định độ khó của các câu hỏi, điều này đặc biệt cần thiết cho các hệ thống OpenQA đàm thoại.

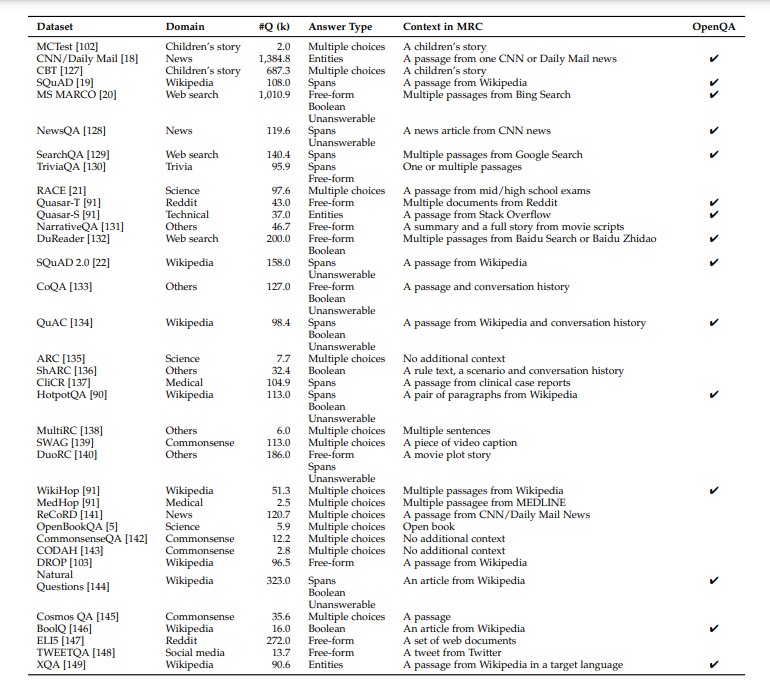
Thứ hai, khi câu hỏi được phân loại là không trả lời được do không rõ ràng hoặc không đủ kiến thức nền tảng, hệ thống OpenQA đàm thoại cần tạo ra một câu hỏi tiếp theo [154]. Tạo câu hỏi (QG) sau đó có thể được coi là một nhiệm vụ phụ của QA, là một mô- đun quan trọng của OpenQA đàm thoại. Trong vài năm trở lại đây, nghiên cứu về việc tạo câu hỏi tự động từ các đoạn văn bản ngày càng được chú ý [155], [156], [157], [158]. Tương đương với nhiệm vụ QG điển hình nhắm mục tiêu tạo câu hỏi dựa trên một đoạn văn nhất định mà câu trả lời cho câu hỏi đã tạo có thể được tìm thấy, câu hỏi được tạo trong OpenQA đàm thoại chỉ được trả lời bởi người dùng.

Thách thức thứ ba là làm thế nào để mô hình hóa lịch sử trò chuyện tốt hơn không chỉ trong Reader mà còn trong Retriever [159]. Các bộ dữ liệu MRC hội thoại được phát hành gần đây như CoQA [133] và QuAC [134] nhằm cho phép Người đọc trả lời câu hỏi mới nhất bằng cách hiểu không chỉ đoạn ngữ cảnh nhất định mà còn cả lịch sử hội thoại cho đến nay. Khi họ cung cấp các đoạn ngữ cảnh trong bộ nhiệm vụ của mình, họ bỏ qua giai đoạn truy xuất tài liệu cần thiết khi nói đến OpenQA. Gần đây, trong [159] tập dữ liệu QuAC được mở rộng thành tập dữ liệu OR-QuAC mới bằng cách thích ứng với cài đặt truy xuất mở và hệ thống trả lời câu hỏi hội thoại truy xuất mở (OpenConvQA) được phát triển, có thể truy xuất các đoạn có liên quan từ một bộ sưu tập lớn trước khi suy ra câu trả lời, có tính đến các cặp QA hội thoại. OpenConvQA cố gắng trả lời một câu hỏi nhất định mà không có bất kỳ ngữ cảnh cụ thể nào, do đó có phạm vi ứng dụng rộng hơn và phù hợp hơn với hành vi QA trong thế giới thực của con người. Tuy nhiên, hiệu suất tốt nhất (F1: 29.4) của hệ thống trên ORQuAC thấp hơn nhiều so với hiệu suất hiện đại (F1: 74.41) trên QuAC, cho thấy rằng đó là một thách thức lớn hơn khi mở -thiết lập hồi ký.

**4.2. Benchmarks**

Một số lượng lớn các tiêu chuẩn QA đã được phát hành trong thập kỷ qua, được tóm tắt trong Bảng 2. Dưới đây chúng tôi cung cấp phân tích ngắn gọn về chúng với trọng tâm là các đặc điểm tương ứng, phân phối tập dữ liệu wrt miền thông tin nền, số lượng câu hỏi, năm của phóng thích. Như đã đề cập ở trên trong bài báo này, sự thành công của nhiệm vụ MRC là một bước quan trọng để OpenQA nâng cao hơn và chúng tôi tin rằng sự tiến bộ trong tương lai của các phương pháp MRC sẽ thúc đẩy đáng kể hệ thống OpenQA. Vì vậy, chúng tôi

BẢNG 2: Dataset: Tên của tập dữ liệu. Miền: Miền của thông tin cơ bản trong tập dữ liệu. #Q (k): Cái số lượng câu hỏi có trong tập dữ liệu, với đơn vị (k) biểu thị “nghìn”. Loại câu trả lời: Các loại câu trả lời bao gồm trong tập dữ liệu. Ngữ cảnh trong MRC: Các tài liệu ngữ cảnh hoặc đoạn văn được đưa ra để tạo câu trả lời trong các nhiệm vụ MRC. OpenQA: Cột này cho biết liệu tập dữ liệu có thể áp dụng để phát triển hệ thống OpenQA hay không, với dấu tích biểu thị có.



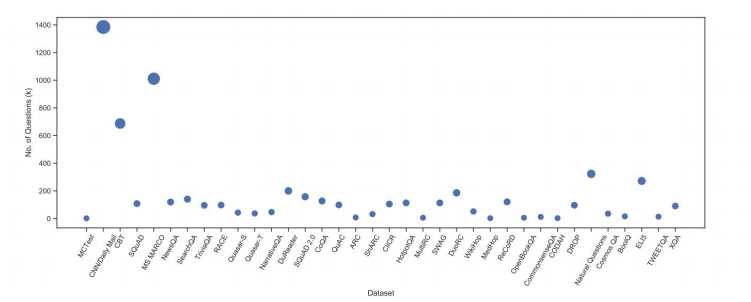
không chỉ bao gồm các tập dữ liệu cho OpenQA mà còn bao gồm những chỉ để MRC làm cho cuộc khảo sát của chúng tôi toàn diện hơn.

Tiêu chí chính để đánh giá khả năng áp dụng của Tập dữ liệu QA để phát triển hệ thống OpenQA là liệu nó có nằm trong tập một bộ tài liệu riêng biệt (thường là quy mô lớn) [90] hay không liệu nó có tương đối dễ dàng truy cập vào một thông tin như vậy không nguồn [18], [22] nơi có thể trả lời cho các câu hỏi suy ra. Ví dụ: HotpotQA [90] cung cấp một wiki đầy đủ tự thiết lập yêu cầu hệ thống tìm câu trả lời cho câu hỏi trong phạm vi của toàn bộ Wikipedia. [3] mở rộng SQuAD [19] sang SQuADopen bằng cách sử dụng toàn bộ Wikipedia như nguồn thông tin của nó. Chúng tôi tóm tắt và minh họa phân phối của các tập dữ liệu được liệt kê trong Bảng 2 wrt năm phát hành trong Hình 7, miền thông tin nền trong Hình 8 và số lượng câu hỏi trong Hình 6. Ngoài ra, chúng tôi tóm tắt loại nguồn thông tin của bộ dữ liệu có thể áp dụng để phát triển hệ thống OpenQA trong Bảng 3.

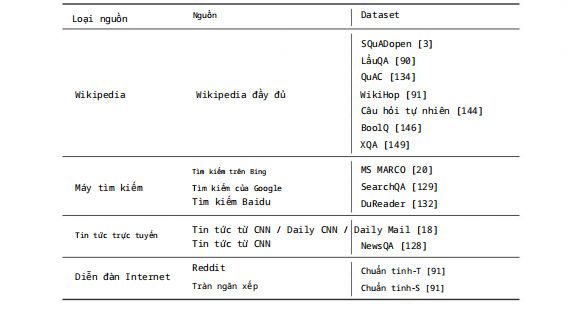
1. **KẾT LUẬN**

Trong công việc này, chúng tôi đã trình bày một cuộc khảo sát toàn diện về tiến bộ mới nhất của hệ thống QA miền mở (OpenQA). Trong đặc biệt, lần đầu tiên chúng tôi xem xét sự phát triển của OpenQA và minh họa kiến trúc “Retriever-Reader”. Hơn thế nữa, chúng tôi cũng đã xem xét nhiều hệ thống OpenQA hiện có như các cách tiếp cận khác nhau của họ. Cuối cùng, chúng tôi đã thảo luận về một số những thách thức nổi bật đối với OpenQA, theo sau là một bản tóm tắt các tiêu chuẩn QA khác nhau, hy vọng sẽ tiết lộ nghiên cứu những khoảng trống để thúc đẩy tiến bộ hơn nữa trong lĩnh vực này. Dựa trên đánh giá của chúng tôi về nghiên cứu trước, chúng tôi tuyên bố rằng OpenQA sẽ tiếp tục là một điểm nóng về nghiên cứu. Đặc biệt, đơn bộ truy xuất thần kinh bước và nhiều bước sẽ thu hút ngày càng tăng sự chú ý do nhu cầu truy xuất chính xác hơn tài liệu liên quan. Ngoài ra, nhiều hệ thống OpenQA end-to-end hơn sẽ được phát triển với sự tiến bộ của học sâu kỹ thuật. OpenQA nâng cao kiến thức rất nổi bật không chỉ vì nó hữu ích trong việc tạo ra câu trả lời mà còn vì nó đóng vai trò là nguồn để diễn giải câu trả lời thu được. Tuy nhiên, làm thế nào để đại diện và làm cho đầy đủ việc sử dụng kiến thức cho OpenQA vẫn cần nghiên cứu thêm nỗ lực. Hơn nữa, để trang bị cho OpenQA một cuộc đối thoại giống như giao diện cho phép tương tác giữa người dùng và hệ thống trao đổi thông tin dự kiến sẽ thu hút tăng sự chú ý, phù hợp với thế giới thực các kịch bản ứng dụng.

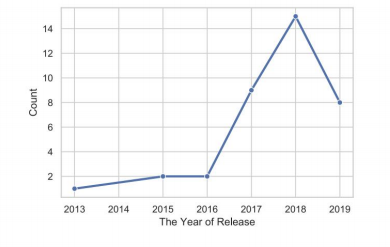
Hình 6: Số lượng câu hỏi trong mỗi tập dữ liệu



BẢNG 3: Nguồn thông tin của bộ dữ liệu có thể áp dụng cho việc phát triển hệ thống OpenQA. Loại nguồn : loại nguồn thông tin nền. Nguồn: Nguồn thông tin nền trong cài đặt OpenQA.



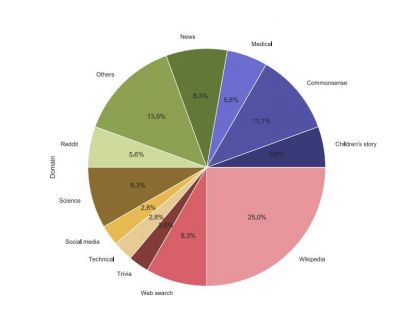
Hình 7: Phân phối các tập dữ liệu phổ biến trong năm phát hành wrt



1. **LỜI CẢM ƠN**

Nghiên cứu này được hỗ trợ bởi National Research Foun dation, Singapore thuộc Trung tâm Nghiên cứu Quốc tế của nó tại Singapore Funding Initiative và A \* STAR theo RIE của nó

Hình 8: Miền thông tin nền wrt phân phối tập dữ liệu



Tài trợ ngữ pháp chuyên nghiệp về Sản xuất và Kỹ thuật tiên tiến (AME) năm 2020, Giải thưởng số - A19E2b0098, Tên dự án - K-EMERGE: Khai thác kiến thức, Mô hình hóa và Lập luận Ex có thể giải được cho Chuyên môn Chung. Mọi ý kiến, phát hiện và kết luận hoặc khuyến nghị được trình bày trong tài liệu này là của (các) tác giả và không phản ánh quan điểm của National Research Foundation và A \* STAR, Singapore.

**NGƯỜI GIỚI THIỆU**

1. BF Green, Jr., AK Wolf, C. Chomsky và K. Laughery, “Bóng chày: Người trả lời câu hỏi tự động,” trong Bài báo được trình bày tại Hội nghị máy tính IRE-AIEE-ACM của Western Joint IRE-AIEE-ACM ngày 9-11 tháng 5 năm 1961. ACM, 1961, trang 219–224.
2. J. Falconer, “Google: Chiến lược tìm kiếm mới của chúng tôi là tính toán các câu trả lời chứ không phải các liên kết,” 2011. [Trực tuyến]. Có sẵn: https://thenextweb.com/google/2011/06/01/ google-our-new-searchstrategy-is-to-compute-answers-not-links / [24] MJ Seo, A. Kembhavi, A.
3. D. Chen, A. Fisch, J. Weston, và A. Bordes, “Đọc Wikipedia để trả lời các câu hỏi miền mở,” trong Kỷ yếu Cuộc họp thường niên lần thứ 55 của Hiệp hội Ngôn ngữ học Tính toán (Tập 1: Long Giấy tờ). Hiệp hội Ngôn ngữ học Tính toán, 2017, trang 1870–1879.
4. EM Voorhees, “Báo cáo theo dõi trả lời câu hỏi trec-8,” NIST, Công nghệ. Rep., 1999.
5. T. Mihaylov, P. Clark, T. Khot, và A. Sabharwal, “Một bộ áo giáp có thể dẫn điện không? Một tập dữ liệu mới để trả lời câu hỏi về cuốn sách mở, ”CoRR, vol. abs / 1809.02789, năm 2018.
6. SM Harabagiu, SJ Maiorano và MA Pellowefinedca, “Kỹ thuật trả lời câu hỏi dạng văn bản miền mở”, Nat Lang. Bản tiếng Anh, tập. 9, không. 3, tr. 231–267, 2003.
7. VC John Burger, Claire Cardie và cộng sự, “Các vấn đề, nhiệm vụ và cấu trúc chương trình để lập lộ trình nghiên cứu câu hỏi và trả lời (q & a,” NIST, Công nghệ. Rep., 2001.
8. O. Kolomiyets và M.-F. Moens, “Một cuộc khảo sát về công nghệ trả lời câu hỏi từ góc độ truy xuất thông tin,” Inf. Khoa học, tập. 181, không. 24, trang 5412–5434, 2011.
9. A. Allam và M. Haggag, “Các hệ thống trả lời câu hỏi: Một cuộc khảo sát,” Tạp chí Nghiên cứu và Đánh giá Quốc tế về Khoa học Thông tin, trang 211– 221, 2012.
10. A. Mishra và SK Jain, “Một cuộc khảo sát về hệ thống trả lời câu hỏi có phân loại,” J. King Saud Univ. Tính toán. Inf. Khoa học, tập. 28, không. 3, tr. 345–361, năm 2016.
11. M. Pas¸ca, “Trả lời câu hỏi miền mở từ bộ sưu tập văn bản lớn,” Ngôn ngữ học tính toán, tập. 29, không. 4, trang 665–667, 2003.
12. Z. Huang, S. Xu, M. Hu, X. Wang, J. Qiu, Y. Fu, Y. Zhao, Y. Peng, và C. Wang, “Các xu hướng gần đây trong học sâu dựa trên miền văn bản hệ thống trả lời câu hỏi, ”IEEE Access, vol. 8, trang 94 341–94 356, năm 2020.
13. T. Lei, Z. Shi, D. Liu, L. Yang và F. Zhu, “Một phương pháp mới dựa trên cnn để phân loại câu hỏi trong trả lời câu hỏi thông minh,” trong Kỷ yếu của Hội nghị Quốc tế 2018 về Thuật toán, Máy tính và Trí tuệ nhân tạo. Hiệp hội Máy tính, 2018.
14. W. Xia, W. Zhu, B. Liao, M. Chen, L. Cai và L. Huang, “Kiến trúc tiểu thuyết cho trí nhớ ngắn hạn dài hạn được sử dụng trong hư cấu phân loại câu hỏi,” Neurocomputing, vol. 299, trang 20–31, 2018.
15. K. Nishida, I. Saito, A. Otsuka, H. Asano, và J. Tomita, “Truy xuất và đọc: Học đa tác vụ về truy xuất thông tin và đọc hiểu,” trong Kỷ yếu của ACM International lần thứ 27 Hội nghị Quản lý Thông tin và Tri thức, ser. CIKM '18. Hiệp hội Máy tính, 2018, tr. 647–656.
16. V. Karpukhin, B. Oguz, S. Min, L. Wu, S. Edunov, D. Chen, và W.-t. Yih, "Truy xuất đoạn văn dày đặc để trả lời câu hỏi miền mở", arXiv preprint arXiv: 2004.04906, 2020
17. O. Khattab, C. Potts và M. Zaharia, “Giám sát hướng dẫn về mức độ liên quan cho OpenQA với ColBERT,” 2020. [Trực tuyến]. Có sẵn: http://arxiv.org/abs/2007.00814 [18] KM Hermann, T. Kocisk ˇ y, E. Grefenstette, L. Espeholt, W. Kay, M. Suleyman và P. Blunsom, “Máy dạy để đọc và hiểu, ”trong Kỷ yếu của Hội nghị Quốc tế lần thứ 28 về Hệ thống Xử lý Thông tin Thần kinh - Tập 1. MIT Press, 2015, trang 1693– 1701.
18. P. Rajpurkar, J. Zhang, K. Lopyrev và P. Liang, “SQuAD: 100.000+ câu hỏi để hiểu văn bản bằng máy,” trong Pro ceedings của Hội nghị năm 2016 về các phương pháp thực nghiệm trong xử lý ngôn ngữ tự nhiên. Hiệp hội Ngôn ngữ học Tính toán, 2016, trang 2383–2392.
19. T. Nguyen, M. Rosenberg, X. Song, J. Gao, S. Tiwary, R. Ma jumder và L. Deng, “MS MARCO: Bộ dữ liệu đọc hiểu do máy tạo ra,” 2016.
20. G. Lai, Q. Xie, H. Liu, Y. Yang, và EH Hovy, “RACE: bộ dữ liệu đọc hiểu quy mô lớn từ các kỳ thi,” CoRR, vol. abs / 1704.04683, năm 2017.
21. P. Rajpurkar, R. Jia và P. Liang, “Biết những gì bạn chưa biết: Những câu hỏi không thể trả lời cho SQuAD,” trong Kỷ yếu của Hội nghị thường niên lần thứ 56 của Hiệp hội Ngôn ngữ học Tính toán (Tập 2: Ngắn gọn Giấy tờ). Hiệp hội Ngôn ngữ học Tính toán, 2018, trang 784–789.
22. J. Li, M. Liu, M.-Y. Kan, Z. Zheng, Z. Wang, W. Lei, T. Liu và B. Qin, “Molweni: Tập dữ liệu đọc hiểu ma chine dựa trên đối thoại đa bên với cấu trúc diễn ngôn,”
23. W. Wang, N. Yang, F. Wei, B. Chang, và M. Zhou, “Các mạng tự kết hợp được kiểm soát để đọc hiểu và trả lời câu hỏi,” trong Kỷ yếu của Hội nghị thường niên lần thứ 55 của Hiệp hội Tính toán Ngôn ngữ học, ACL. Hiệp hội Ngôn ngữ học Tính toán, 2017, trang 189–198.
24. AW Yu, D. Dohan, M. Luong, R. Zhao, K. Chen, M. Norouzi, và QV Le, “Qanet: Kết hợp tích hợp cục bộ với sự chú ý toàn cầu để đọc hiểu,” trên International Confer ence về Biểu diễn Học tập, ICLR. OpenReview.net, 2018.
25. J. Devlin, M. Chang, K. Lee, và K. Toutanova, “BERT: đào tạo trước về máy biến áp hai chiều sâu cho ngôn ngữ đang đứng,” CoRR, vol. abs / 1810.04805, 2018.
26. S. Wang, M. Yu, X. Guo, Z. Wang, T. Klinger, W. Zhang, S. Chang, G. Tesauro, B. Zhou và J. Jiang, “R3: Người xếp hạng tăng cường- người đọc để trả lời câu hỏi trong miền mở, ”trong AAAI, 2018.
27. R. Das, S. Dhuliawala, M. Zaheer và A. McCallum, “Tương tác giữa người đọc và người truy xuất nhiều bước để trả lời câu hỏi miền mở có thể mở rộng,” trong Hội nghị Quốc tế về Đại diện Học tập, 2019.
28. K. Guu, K. Lee, Z. Tung, P. Pasupat, và M.-W. Chang, “Realm: Đào tạo trước mô hình ngôn ngữ tăng cường khả năng truy xuất”, CoRR, 2020.
29. M. Ding, C. Zhou, Q. Chen, H. Yang và J. Tang, “Biểu đồ nhận thức cho khả năng đọc hiểu nhiều bước trên quy mô lớn,” trong Kỷ yếu của Hội nghị thường niên lần thứ 57 của Hiệp hội Tính toán
30. Y. Nie, S. Wang và M. Bansal, “Tiết lộ tầm quan trọng của việc truy xuất ngữ nghĩa đối với việc đọc máy trên quy mô lớn,” trong Kỷ yếu của Hội nghị 2019 về các phương pháp thực nghiệm trong xử lý ngôn ngữ tự nhiên và Hội nghị chung quốc tế lần thứ 9 về Xử lý ngôn ngữ tự nhiên (EMNLPIJCNLP). Hiệp hội Ngôn ngữ học Tổng hợp, 2019, trang 2553–2566.
31. Y. Lin, H. Ji, Z. Liu và M. Sun, “Biểu thị câu trả lời câu hỏi miền mở được giám sát từ xa,” trong Kỷ yếu Cuộc họp thường niên lần thứ 56 của Hiệp hội Ngôn ngữ học Tính toán (Tập 1: Long Giấy tờ). Hiệp hội Ngôn ngữ học Tính toán, 2018, trang 1736–1745.
32. J. Lee, S. Yun, H. Kim, M. Ko và J. Kang, “Xếp hạng các đoạn văn để cải thiện khả năng nhớ lại câu trả lời trong câu trả lời câu hỏi miền mở”, trong Kỷ yếu của Hội nghị năm 2018 về các phương pháp thực nghiệm trong Xử lý ngôn ngữ tự nhiên. Hiệp hội Ngôn ngữ học Tính toán, 2018, trang 565–569
33. Y. Feldman và cộng sự, “Truy xuất đoạn văn nhiều bước nhảy để trả lời câu hỏi miền mở,” trong Kỷ yếu của Hội nghị thường niên lần thứ 57 của Hiệp hội Ngôn ngữ học Tính toán. Associa tion for Computational Linguistics, 2019, trang 2296–2309.
34. P. Qi, X. Lin, L. Mehr, Z. Wang và CD Manning, “Một câu hỏi miền mở phức tạp hấp dẫn thông qua việc tạo truy vấn lặp lại”, trong Kỷ yếu của Hội nghị năm 2019 về các phương pháp thực nghiệm trong xử lý ngôn ngữ tự nhiên và Hội nghị chung quốc tế về xử lý ngôn ngữ tự nhiên lần thứ 9 (EMNLP-IJCNLP).
35. K. Lee, M.-W. Chang, và K. Toutanova, “Truy xuất tiềm ẩn để trả lời câu hỏi miền mở được giám sát yếu”, trong Kỷ yếu của Hội nghị thường niên lần thứ 57 của Hiệp hội Ngôn ngữ học Tính toán. Hiệp hội Ngôn ngữ học Tính toán, 2019, trang 6086–6096.
36. P. Lewis, E. Perez, A. Piktus, F. Petroni, V. Karpukhin, N. Goyal, H. Kuttler, M. Lewis, W.-t. Yih, T. Rockt aschel, ¨ S. Riedel và D. Kiela, “Thế hệ truy xuất-tăng cường cho các nhiệm vụ NLP chuyên sâu về kiến thức,” năm 2020. [Trực tuyến]. Có sẵn: http://arxiv.org/abs/2005.11401
37. WA Woods, “Tiến bộ trong hiểu biết ngôn ngữ tự nhiên: Ứng dụng cho địa chất Mặt Trăng,” trong Kỷ yếu của Hội nghị và Triển lãm Máy tính Quốc gia ngày 4-8 tháng 6 năm 1973. ACM, 1973, trang 441–450.
38. J. Kupiec, “Murax: Một cách tiếp cận ngôn ngữ mạnh mẽ để trả lời câu hỏi bằng cách sử dụng một bách khoa toàn thư trực tuyến,” trong Kỷ yếu của Hội nghị ACM SIGIR quốc tế thường niên lần thứ 16 về Nghiên cứu và Phát triển trong Truy xuất Thông tin. Hiệp hội Máy tính, 1993, tr. 181–190.
39. EM Voorhees, “Tổng quan về phần trả lời câu hỏi trec 2001,” trong Kỷ yếu của TREC-10, 2001, trang 42–51. [42] ——, “Tổng quan về phần trả lời câu hỏi TREC 2002,” trong Kỷ yếu Hội nghị Duyệt lại Văn bản lần thứ 11, TREC 2002, Gaithersburg, Maryland, Hoa Kỳ, 19-22 tháng 11 năm 2002, ser. NIST
40. E. Voorhees, “Tổng quan về bản trả lời câu hỏi trec 2003,” NIST, Tech. Rep., 2003.
41. C. Kwok, O. Etzioni, O. Etzioni, và DS Weld, “Trả lời câu hỏi mở rộng quy mô trên web,” Giao dịch ACM trên Hệ thống Thông tin, tập. 19, không. 3, trang 242–262, 2001
42. E. Brill, S. Dumais và M. Banko, “Phân tích hệ thống trả lời câu hỏi AskMSR,” trong Kỷ yếu của Hội nghị năm 2002 về các phương pháp thực nghiệm trong xử lý ngôn ngữ tự nhiên (EMNLP 2002). Hiệp hội Ngôn ngữ học Tính toán, 2002, trang 257– 264
43. Z. Zheng, “Hệ thống trả lời câu hỏi Answerbus,” trong Kỷ yếu của Hội nghị Quốc tế lần thứ hai về Nghiên cứu Công nghệ Ngôn ngữ Con người, ser. HLT '02. Morgan Kaufmann Publishers Inc., 2002, tr. 399–404.
44. D. Moldova, M. Pasca, S. Harabagiu và M. Surdeanu, “Các vấn đề về dạng thức và phân tích lỗi trong hệ thống trả lời câu hỏi miền mở,” trong Kỷ yếu của Hội nghị thường niên lần thứ 40 của Hiệp hội Ngôn ngữ học Tính toán . Hiệp hội Ngôn ngữ học Tính toán, 2002, trang 33–40

