BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Đình Quý

XÂY DỰNG MÔ HÌNH HỎI ĐÁP HỖ TRỢ SINH VIÊN TRƯỜNG ĐẠI HỌC XÂY DỰNG

LUẬN VĂN THẠC SĨ KỸ THUẬT

(***Theo định hướng ứng dụng***)

HÀ NỘI - NĂM 2021

BỘ THÔNG TIN VÀ TRUYỀN THÔNG

HỌC VIỆN CÔNG NGHỆ BƯU CHÍNH VIỄN THÔNG

Nguyễn Đình Quý

XÂY DỰNG MÔ HÌNH HỎI ĐÁP HỖ TRỢ SINH VIÊN TRƯỜNG ĐẠI HỌC XÂY DỰNG

Chuyên ngành: Khoa học máy tính

Mã số: 8.48.01.01

LUẬN VĂN THẠC SĨ KỸ THUẬT

(*Theo định hướng ứng dụng*)

Người hướng dẫn: GS.TS Từ Minh Phương

HÀ NỘI - NĂM 2021

MỤC LỤC

LỜI CAM ĐOAN

Tôi cam đoan đây là công trình nghiên cứu của riêng tôi được GS.TS. Từ Minh Phương - giảng viên khoa Công nghệ thông tin 1 trường Học viện Công nghệ Bưu chính Viễn thông hướng dẫn khoa học. Nguồn tài liệu của các tác giả, cơ quan, tổ chức nếu sử dụng thì tôi đều ghi rõ trong phần tài liệu tham khảo.

Tôi xin hoàn toàn chịu trách nhiệm về nội dung luận văn của mình.

Hà nội, ngày tháng năm 2021.

Học viên Cao học.

Nguyễn Đình Qúy.

LỜI CẢM ƠN

Lời đầu tiên, tôi xin bày tỏ sự biết ơn chân thành và sâu sắc nhất tới GS.TS. Từ Minh Phương - Giáo viên hướng dẫn khoa học, người đã tận tình hướng dẫn, hỗ trợ và giúp đỡ tôi trong quá trình nghiên cứu và hoàn thiện luận văn của mình.

Tôi xin gửi lời cảm ơn chân thành tới các thầy, các cô là giảng viên khoa Công nghệ thông tin 1 của trường Học viện công nghệ bưu chính viễn thông đã tận tình truyền đạt kiến thức và hướng dẫn cho tôi trong suốt quá trình học tập tại trường.

Tôi xin gửi lời cảm ơn tới những người thân trong gia đình tôi đã chăm lo cho tôi, động viên tôi, cảm ơn cơ quan nơi tôi đang công tác trường Đại học Xây dựng đã hết sức tạo điều kiện để tôi hoàn thành khóa học này. Cám ơn các bạn sinh viên khoa Công nghệ Thông tin trường Đại học Xây dựng đã giúp đỡ tôi trong việc thu thập dữ liệu để thực hiện luận văn này.

Trong quá trình hoàn thành luận văn do thời gian và khả năng kiến thức còn hạn chế nên khó tránh khỏi những sai sót. Kính mong nhận được sự cảm thông, góp ý của các thầy các cô.

Hà nội, ngày tháng năm 2021.

Người viết

Nguyễn Đình Quý

DANH MỤC HÌNH VẼ

DANH MỤC BẢNG BIỂU

DANH MỤC TỪ VIẾT TẮT VÀ THUẬT NGỮ

STT	Ký hiệu, viết tắt	Từ Tiếng Anh	Chú giải
1	AI	Artificial Intelligent	Trí tuệ nhân tạo
2	QA	Question Answering systems	Hệ thống hỏi đáp
3	IR	Information retrieval	Truy xuất thông tin
4	RNN	Recurrent Neural Network	Mạng nơ-ron hồi quy
5	LSTM	Long short-term memory	Mạng bộ nhớ dài ngắn
6	NLP	Natural language processing	Xử lý ngôn ngữ tự nhiên
7	POS	Part – Of - Speech	
8	NER	Named-entity recognition	Nhận diện thực thể có tên
9	SVM	Support Vector Machine	Máy vector hỗ trợ
10	TF	Term frequency	Tần suất thuật ngữ
11	IDF	Inverse Document Frequency	Nghịch đảo tần suất của văn bản
12	**Precision**		Độ chính xác (độ phủ)
13	Recall		Độ chính xác (độ hồi tưởng)

MỞ ĐẦU

Hiện nay trường đại học Xây dựng có khoảng 15.000 sinh viên và học viên đang theo học. Hàng ngày các phòng ban của trường nhận được rất nhiều các vấn đề thắc mắc của sinh viên và học viên về chương trình đào tạo, các thông tin về lịch học, lịch thi hay các quy định của nhà trường. Kênh thông tin chủ yếu của nhà trường là thông qua website chính thức hoặc trang quản lý đào tạo của sinh viên. Các quy định hay các thông báo tới sinh viên chủ yếu dưới dạng các văn bản nên gây khó khăn cho sinh viên trong việc tiếp cận và tra cứu thông tin. Chính vì thế khi có thắc mắc, sinh viên thường bỏ qua không đọc các văn bản hay thông báo mà sử dụng kênh hỗ trợ trực tiếp từ nhà trường, hiện tại là thông qua kênh email.

Theo khảo sát của trường Đại học Xây dựng, khi một sinh viên cần hỏi vấn đề liên quan đến học tập và quy định tại trường:

- 45% số sinh viên khi cần thông tin sẽ được đáp ứng thông qua việc hỏi bạn bè trong lớp và trong trường. Trong số đó 65% hỏi trực tiếp bạn bè, 35% còn lại sẽ hỏi thông qua trang fanpage, hội nhóm trên mạng xã hội facebook.
- 15% số sinh viên sẽ tự tìm hiểu các thông báo và quy định được đăng tải trên website chính thức, website đào tạo và các kênh truyền thông của Nhà trường.
- 10% số sinh viên sẽ hỏi trực tiếp tại các phòng ban bằng cách lên trực tiếp nơi làm việc.
- 30% số sinh viên còn lại sẽ hỏi các phòng ban bằng hình thức email.

Một vấn đề đặt ra là số lượng email các câu hỏi của sinh viên gửi tới các phòng ban rất nhiều, một ngày có thể lên tới vài chục đến vài trăm câu hỏi. Vì vậy việc hỗ trợ sinh viên mà đặc biệt vào những dịp cao điểm như đăng ký môn học, thi hết học phần thường bị quá tải ở các phòng ban. Đồng thời sinh viên phải chờ đợi việc xử lý các câu hỏi và câu trả lời nên nhiều khi thông tin phản hồi không được kịp thời, gây ảnh hưởng đến quá trình học tập của sinh viên. Trong quá trình học tập của sinh viên, các nội dung liên quan đến quy định sẽ được thông báo dưới dạng văn bản

hoặc tài liệu được đăng tải trên website đào tạo của nhà trường. Sinh viên quan tâm đến thông báo thường dựa trên tiêu đề thông báo, rồi sau đó mới đến nội dung thông báo, vì vậy nhiều thông báo bị sinh viên bỏ sót. Ngoài ra một số tài liệu quy định có nội dung dài nên sinh viên thường bỏ qua không đọc. Vì vậy nếu chỉ xây dựng hệ thống để quản lý văn bản, tài liệu để sinh viên tra cứu cũng không thật sự hữu ích với sinh viên. Cần phải xây dựng công cụ để tương tác với sinh viên dưới dạng đặt câu hỏi – trả lời mới giải quyết được vấn đề này.

Chính vì vậy, việc đưa ra một hệ thống trả lời câu hỏi tự động nhằm cung cấp cho sinh viên kênh hỗ trợ nhanh chóng, đồng thời làm giảm khối lượng công việc cho các phòng ban là vô cùng cần thiết. Một trong những kỹ thuật được sử dụng phổ biến hiện nay và mang lại hiệu quả cao là kỹ thuật truy xuất thông tin. Đề tài luận văn của tôi sẽ tập trung vào tìm hiểu các kỹ thuật này, dựa trên dữ liệu được cung cấp từ nhà trường để xây dựng hệ thống trả lời tự động có kết quả trả lời tốt nhất.

Nội dung của luận văn được bố cục thành 3 chương như sau:

- ***Chương 1*** tập trung vào giới thiệu về bài toán, dữ liệu đã có và kết quả dự kiến của đề tài. Trình bày khái về hệ thống hỏi đáp tự động, các loại hệ thống hỏi đáp, lịch sử phát triển, đưa ra kiến trúc chung của hệ thống hỏi đáp đồng thời là các vấn đề cần quan tâm khi thiết kế.
- ***Chương 2*** tập trung vào lựa chọn mô hình và thuật toán để xây dựng mô hình hệ thống hỏi đáp. Trình bày về việc tìm hiểu các phương pháp tiền xử lý dữ liệu bao gồm: tách từ tiếng Việt, các hướng tiếp cận dựa trên từ và dựa trên ký tự; biểu diễn văn bản; rút trích đặc trưng văn bản như loại bỏ các stop word, trích chọn đặc trưng văn bản thành các biểu diễn của các vector; tiếp theo là đưa ra mô hình kiến trúc của hệ thống và kỹ thuật được sử dụng trong luận văn;
- ***Chương 3*** tập trung vào cài đặt, xây dựng bộ dữ liệu huấn luyện cho mô hình hỏi đáp từ dữ liệu thực tế hiện có của trường Đại học Xây dựng, sử dụng các kỹ

thuật đánh giá mô hình hỏi đáp để đánh giá hệ thống, tiếp theo là tiến hành thử nghiệm tại trường để tiếp nhận những đánh giá từ người dùng cuối.

CHƯƠNG 1. TỔNG QUAN VỀ BÀI TOÁN HỎI ĐÁP TỰ ĐỘNG

1.1. Bài toán trả lời tự động cho sinh viên trường Đại học Xây dựng

Với thực trạng tại trường Đại học Xây dựng, hàng ngày sinh viên hỏi và thắc mắc rất nhiều vấn đề liên quan đến các chính sách, quy định và quy chế. Nhà trường phải bố trí bộ phận hỗ trợ sinh viên để giải đáp các thắc mắc và giúp đỡ sinh viên khi cần thiết, hiện tại bộ phận này sẽ tiếp nhận các câu hỏi của sinh viên qua kênh email sau đó trả lời các email đó. Tuy nhiên vấn đề vào các đợt cao điểm như đăng ký môn học hay thi kết thúc học phần thì số lượng các câu hỏi tăng đột biến làm quá tải cho bộ phận hỗ trợ. Hơn nữa rất nhiều các câu hỏi thường lặp lại và được trả lời giống nhau, bộ phận hỗ trợ thường dựa vào các câu trả lời trước đó đã phản hồi để trả lời các câu hỏi tương tự.

Giả sử như nếu sinh viên hỏi một trong các câu hỏi sau đây:

1. *E thưa cô, chả hạn e trả hết môn mà tích luỹ chưa đủ 2.0 thì e có dc nhận để làm đồ án tốt nghiệp không ạ*
2. *Điều kiện để nhận ĐATN là gì ạ?*
3. *Em đã hoàn thiện hết các môn nhưng chưa đủ tiêu chuẩn ngoại ngữ thì có được nhận ĐATN không ạ?*
4. *Điểm trung bình tích lũy bao nhiêu thì được nhận đồ án tốt nghiệp ạ*

Thì đều được trả lời là: "*Em trả nợ xong tất cả các môn và đạt CĐR ngoại ngữ là đủ điều kiện nhận ĐATN. Điểm TBC tích lũy từ 2.0 trở lên là điều kiện xét TN, không áp dụng khi xét giao ĐATN"*. Như vậy là khi sinh viên hỏi một câu hỏi nào đó mà tương tự với các câu hỏi đã có thì có thể trả lời bằng câu trả lời có sẵn.

Sau một thời gian trả lời qua email, bộ phận công tác sinh viên đã thu thập được một bộ các câu hỏi của sinh viên và câu trả lời do cơ quan chức năng của trường gửi lại gồm khoảng 3.500 câu hỏi, câu trả lời. Dựa trên tập câu hỏi, câu trả lời này, bài toán mà luận văn hướng tới giải quyết là xây dựng hệ thống cho phép tự động trả lời câu hỏi của sinh viên trong tương lai.

Kết quả dự kiến của luận văn: Luận văn này sẽ dựa vào một tập dữ liệu có sẵn gồm các câu hỏi và câu trả lời để xây dựng công cụ trả lời tự động các câu hỏi giống với các câu hỏi đã có trong tập dữ liệu.

1.2. Khái quát hệ thống hỏi đáp tự động

Nếu như trong hệ thống trích chọn thông tin khi người dùng muốn tìm kiếm thông tin họ cần, hệ thống trích chọn thông tin sẽ nhận truy vấn đầu vào của người dùng dưới dạng các từ khóa và trả về các tài liệu liên quan có chứa từ khóa thì hệ thống hỏi đáp sẽ nhận đầu vào dưới dạng ngôn ngữ tự nhiên (thường là các câu hỏi), sau đó trả lại câu trả lời tương ứng với câu hỏi đưa vào.

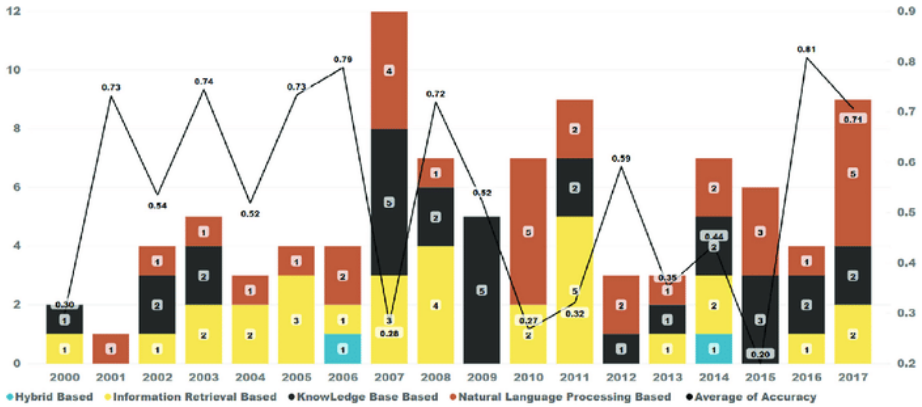
Có nhiều cách để phân loại một hệ thống hỏi đáp, dựa vào mô hình của hệ hỏi đáp có thể phân loại thành các loại như sau [2]:

- *Hệ hỏi đáp truy xuất thông tin (IR)* sử dụng máy tìm kiếm để tìm ra các câu trả lời, áp dụng các bộ lọc và xếp hạng để tìm ra trả lời gần nhất.
- *Hệ hỏi đáp dựa trên xử lý ngôn ngữ tự nhiên (NLP QA)* áp dụng kỹ thuật để hiểu ngôn ngữ tự nhiên và các phương pháp tiếp cận máy học để trích rút câu trả lời.
- *Hệ hỏi đáp dựa trên cơ sở tri thức (Knowledge Base QA)* tìm kiếm câu trả lời từ các nguồn dữ liệu có cấu trúc (hay tri thức) thay vì văn bản phi cấu trúc.
- *Hệ hỏi đáp lai là hệ hỏi đáp cho kết quả tốt* bằng cách sử dụng nhiều loại nguồn dữ liệu nhất có thể, đây là sự kết hợp giữa IR, QA, NLP QA, Knowledge Base QA. Ví dụ điển hình cho loại này là hệ hỏi đáp IBM Watson [3].

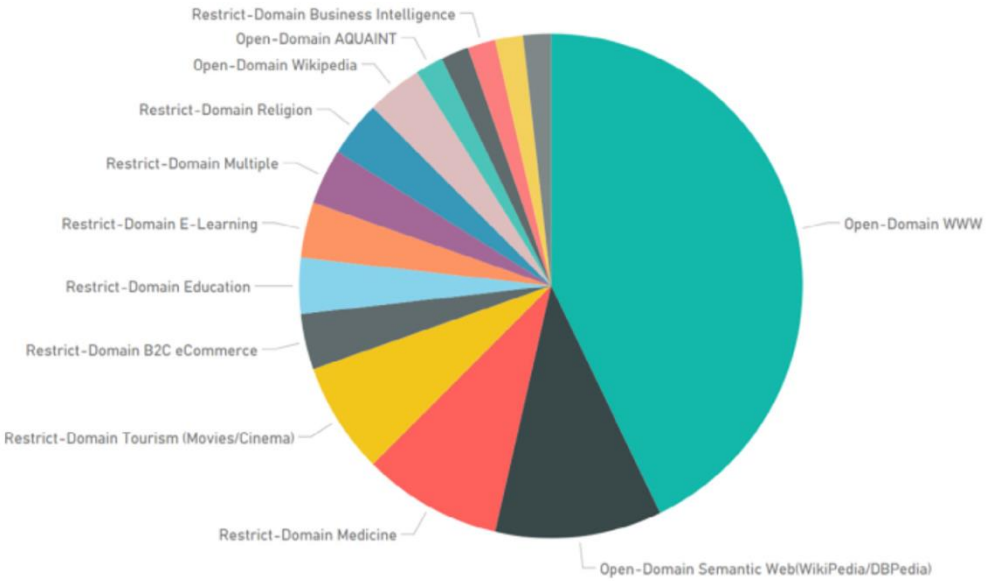
Vào những năm 1960 đã ra đời các hệ hỏi đáp sớm nhất, phải kể đến là BASEBALL [4] và LUNAR [5]. Các hệ hỏi đáp này bị giới hạn trong một lĩnh vực cụ thể nhưng nó cho chúng ta thấy tính khả thi để đưa tạo ra các tác tử tự động có khả năng hiểu và giao tiếp bằng ngôn ngữ tự nhiên để trả lời các câu hỏi. Từ bước ngoặt năm 1999 với sự đánh dấu của việc giới thiệu về QA tại hội nghị Text REtrieval Conference (TREC), các nghiên cứu về hệ hỏi đáp bắt đầu nở rộ về số

lượng và ngày càng có nhiều hơn các công bố khoa học liên quan. Trong các giai đoạn tiếp theo, *xu hướng phát triển* và *thống kê* liên quan đến hệ thống hỏi đáp được tổng hợp thông qua một cuộc khảo sát [1].

Từ 130 nghiên cứu phổ biến lấy từ tất cả 1842 nghiên cứu. 34,59% số các bài báo thực hiện hệ hỏi đáp dựa trên tri thức, 33,08% dựa trên NLP và số lượng 2 loại này cao hơn so với hệ hỏi đáp dựa trên IR, cuối cùng là chỉ có 3,76% xây dựng hệ hỏi đáp dựa trên hệ lai.



Hình 1.1: Số lượng các công bố về hệ hỏi đáp (QA) tính từ năm 2000



Hình 1.2: Cách tiếp cận hệ hỏi đáp

Trong hình 5, cách tiếp cận hệ hỏi đáp trong miền mở dựa trên World Wide Web chiếm tỉ lệ cao nhất trong số các nghiên cứu, theo sau đó là các lĩnh vực *y tế*. Ngoài ra chúng ta có thể thấy hệ hỏi đáp có sự liên hệ qua lại giữa các lĩnh vực khác nhau. Điều này cho ta thấy hệ hỏi đáp đã và đang được ứng dụng vào hầu hết các lĩnh vực trong cuộc sống và đáp ứng nhiều nhu cầu khác nhau.

Đối với bài toán cần giải quyết, tận dụng dữ liệu của bài toán gồm các câu hỏi và câu trả lời có sẵn nên luận văn này sẽ sử dụng phương pháp trả lời tự động dựa *trên truy xuất thông tin (IR).*

1.3. Truy xuất và tìm kiếm thông tin (IR)

Hệ truy xuất thông tin (IR) xuất hiện trong các hệ thống thông minh từ những năm 1960, hệ thống tìm kiếm trên máy tính sớm nhất được ra đời vào cuối những năm 1940.

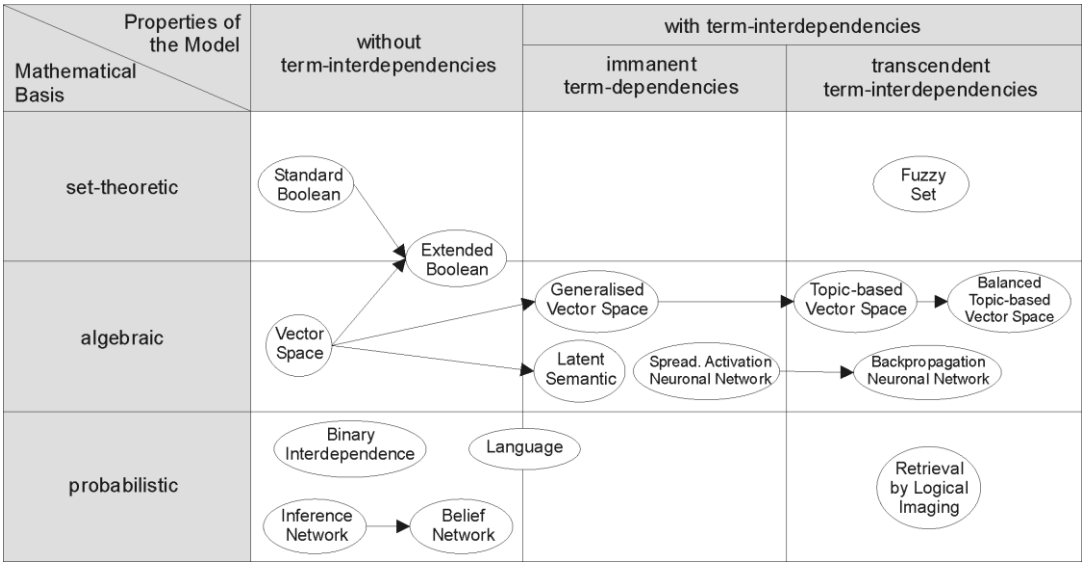
Với sự phát triển của phần cứng máy tính, cùng với sự gia tăng về tốc độ bộ xử lý và dung lượng lưu trữ đã giúp cho hệ thống tìm kiếm phát triển. Sự phát triển của hệ thống này phản ánh sự tiến bộ nhanh chóng từ các phương pháp tiếp cận dựa trên việc thu thập và lập chỉ mục và tìm kiếm thủ công sang phương pháp tự động.

Nhiệm vụ của hệ truy xuất thông tin đó là tìm ra các tài liệu hay thông tin liên quan đến truy vấn của người dùng. Hệ thống này sẽ thu thập các dữ liệu có cấu trúc hoặc các dữ liệu bán cấu trúc (ví dụ như các trang web, ảnh, video, các tài liệu…). Các tài liệu này được chuyển đổi sang dạng biểu diễn phù hợp để có thể dễ dàng thực hiện việc tìm kiếm. Mỗi mô hình IR sẽ phải thực hiện: (1) làm thế nào để biểu diễn được các tài liệu và truy vấn, và làm thế nào để lấy ra các tài liệu liên quan đến truy vấn của người dùng. Thuật ngữ (term): Dùng để chỉ thành phần của một truy vấn, ví dụ ta có truy vấn: "Thủ đô của Hà Nội là gì", thuật ngữ của truy vấn sẽ là: 'Thủ đô', 'của', 'Hà Nội'. Hiểu đơn giản, thuật ngữ là các từ trong truy vấn/văn bản mang ý nghĩa. Tài liệu: Các văn bản thông thường cần tìm kiếm, truy vấn cũng có thể coi là tài liệu.

Thông thường, mô hình xếp hạng được viết gọn trong 4 chữ D, Q, F, R. Trong đó các chữ được định nghĩa như sau:

- *D (Document collection)* là bộ sưu tập tài liệu. Mỗi tài liệu được mô hình hóa như một nhóm các thuật ngữ chỉ mục trong đó các thuật ngữ chỉ mục được giả định là độc lập với nhau.
- *Q (Query collection)* là bộ sưu tập truy vấn. Các truy vấn được kích hoạt bởi người dùng thuộc về tập hợp này. Nó cũng được mô hình hóa như một tập các thuật ngữ chỉ mục.
- *F (Framework)* Framework cho mô hình mô tả tài liệu, các câu truy vấn và mối quan hệ giữa chúng.
- *R (Ranking function)* là một hàm xếp hạng liên kết một điểm (số thực) với cặp (q_i, d_j) trong đó $q_i \in Q$ và $d_j \in D$. Cho truy vấn (q_i) các tài liệu được xếp hạng theo điểm số.

Các mô hình IR dựa trên toán học có thể được phân loại thêm thành 4 loại: mô hình dựa trên lý thuyết tập hợp, mô hình đại số, mô hình xác suất và mô hình truy xuất dựa trên đặc trưng. Mặc dù mỗi mô hình có các phương pháp khác nhau để biểu diễn cho các tài liệu và truy vấn, nhưng tất cả chúng đều coi mỗi tài liệu hoặc truy vấn như một *bag of terms* (túi thuật ngữ), có nghĩa là một tài liệu được mô tả bằng một tập hợp các thuật ngữ riêng biệt, trong các tài liệu thường không quan tâm đến thứ tự các thuật ngữ và vị trí [7, 8].



Hình 1.3: Phân loại các mô hình IR

1.3.1. Mô hình dựa trên lý thuyết tập hợp:

Mô hình lý thuyết tập hợp biểu diễn tài liệu dưới dạng tập hợp các từ hoặc cụm từ. Từ đó, các phép toán dùng để tính độ tương tự thường sử dụng các phép toán dựa trên lý thuyết tập hợp. Các mô hình phổ biến thuộc loại này là: Mô hình Boolean chuẩn, mô hình Boolean mở rộng và mô hình Truy xuất mờ [8].

Mô hình Boolean tiêu chuẩn [9] sử dụng khái niệm đối sánh chính xác để xác định mức độ phù hợp của tài liệu với truy vấn của người dùng, dựa trên logic boolean và lý thuyết tập hợp cổ điển. Truy vấn được đưa vào dưới dạng tập hợp các thuật ngữ, để tìm ra một tập hợp các tài liệu (các tài liệu cũng được biểu thị dưới dạng tập hợp các thuật ngữ) phù hợp nhất với truy vấn. Một số điểm hạn chế của mô hình này là: việc xác định mức độ giống nhau giữa truy vấn và các tài liệu dựa trên việc các tài liệu có chứa các thuật ngữ của truy vấn hay không mà không xem xét trọng số của các thuật ngữ; việc so sánh chính xác có thể khiến kết quả của truy vấn trả về quá ít hoặc quá nhiều tài liệu; và thêm nữa là việc dịch một truy vấn thành một biểu thức Boolean là khá khó khăn.

Mô hình Boolean mở rộng [10] được phát triển để khắc phục những thiếu sót của mô hình Boolean chuẩn, kết hợp các đặc điểm của Đại số Boolean và Mô hình không gian vectơ để sử dụng đối sánh từng phần và sử dụng trọng số của thuật ngữ. Bằng việc thực hiện như vậy, một tài liệu có thể có liên quan nếu nó phù hợp với một số thuật ngữ trong truy vấn.

Mô hình truy xuất mờ dựa trên mô hình Boolean mở rộng và lý thuyết mờ, cho phép thao tác và tích lũy trọng số cho các thuật ngữ. Logic này cho phép định nghĩa các giá trị chân lý trung gian giữa các đánh giá thông thường về đúng và sai. Có 2 mô hình truy xuất mờ cổ điển, đó là mô hình Min Max hỗn hợp [11] và mô hình Paice [12].

1.3.2. Mô hình đại số

Mô hình đại số biểu diễn các tài liệu và truy vấn dưới dạng vectơ, ma trận hoặc bộ giá trị. Một số mô hình thuộc loại này là: Mô hình không gian vectơ, mô hình

Không gian vectơ tổng quát hóa, mô hình Không gian vectơ dựa trên chủ đề, mô hình Boolean mở rộng.

Mô hình không gian vectơ biểu diễn thông tin dạng văn bản dưới dạng vectơ trong không gian gồm N chiều, trong đó N là số thuật ngữ trong tập các thuật ngữ của truy vấn và mỗi chiều tương ứng với một thuật ngữ riêng biệt). Việc xác định mức độ giống nhau giữa các văn bản có thể được tính toán dễ dàng dựa trên vector. Mô hình không gian vector cho kết quả tốt khi được sử dụng cùng với các phương pháp tiếp cận như xếp hạng tài liệu, lập chỉ mục tài liệu theo ngữ nghĩa. Mô hình này được biết đến và sử dụng rộng rãi nhất, với một số tính chất như sau: mô hình đơn giản dựa trên đại số tuyến tính; sử dụng trọng số cho thuật ngữ để chỉ ra mức độ quan trọng của thuật ngữ thay vì chỉ xét có hoặc không; cho phép tính toán mức độ tương đồng giữa các truy vấn và tài liệu; cho phép xếp hạng các tài liệu theo mức độ phù hợp của chúng; cho phép khớp từng phần. Nhưng mô hình không gian vector cũng đi kèm với một số hạn chế: các tài liệu dài được biểu diễn kém vì chúng có giá trị độ tương đồng thấp (do tích vô hướng nhỏ và vector kích thước lớn); từ khóa tìm kiếm phải khớp chính xác với các thuật ngữ trong tài liệu; các từ ngữ khác nhau nhưng có ý nghĩa giống nhau trong ngữ cảnh của tài liệu sẽ không được liên kết, dẫn đến kết quả trùng khớp "có thể bị sai"; về mặt lý thuyết, giả định rằng các thuật ngữ là độc lập về mặt thống kê; trọng số là trực quan nhưng không chính thức. Tuy nhiên, nhiều khó khăn này có thể khắc phục bằng cách tích hợp các công cụ khác nhau, bao gồm các kỹ thuật toán học như phân tách giá trị đơn lẻ (singular value decomposition) và cơ sở dữ liệu từ vựng như Wordnet.

Mô hình không gian vector khắc phục những nhược điểm của mô hình boolean là việc sử dụng trọng số cho từ chỉ mục khác trọng số nhị phân (*non-binary*). Trọng số từ chỉ mục không giới hạn bởi hai trị 0 hoặc 1, các trọng số này được sử dụng để tính toán độ đo tương tự của mỗi văn bản với câu truy vấn. Với mô hình không gian vector, các văn bản, câu truy vấn và từ chỉ mục được biểu diễn thành các vector trong không gian vector. Sử dụng các phép toán trên không gian

vector để tính toán độ đo tương tự giữa câu truy vấn và các văn bản hoặc các từ chỉ mục, kết quả sau khi tính toán có thể được xếp hạng theo độ đo tương tự với vector truy vấn. Ngoài ra, mô hình không gian vector còn hướng dẫn người dùng biết được những văn bản độ tương tự cao hơn có nội dung gần với nội dung họ cần hơn so với các văn bản khác.

Mô hình không gian vector dựa trên giả thiết là nội dung của văn bản có thể được hiểu như sự kết hợp của các từ chỉ mục. Một văn bản *d* được biểu diễn như một vector của các từ chỉ mục $d = (t_1, t_2 K , t_n)$ với t_i là từ chỉ mục thứ *i* ($1 \le i \le n$) (*các giá trị có thể là số lần xuất hiện của term* t_i *trong văn bản d*). Mỗi từ chỉ mục trong văn bản biểu diễn một chiều (*dimension*) trong không gian. Tương tự, câu truy vấn cũng được biểu diễn như một vector $q = \left(\hat{t}_1, \hat{t}_2, K , \hat{t}_n \right)$.

Sau khi đã biểu diễn tập văn bản và câu truy vấn thành các vector trong không gian vector, ta có thể sử dụng độ đo *cosines* để tính độ đo tương tự giữa các vector văn bản và vector truy vấn.

Ưu điểm của mô hình không gian vector:

- Đơn giản, dễ hiểu
- Cài đặt đơn giản
- Khắc phục các hạn chế trên mô hình Boolean

Nhược điểm mô hình không gian vector:

- Số chiều biểu diễn cho tập văn bản có thể rất lớn nên tốn nhiều không gian lưu trữ.

1.3.3. Mô hình xác suất

Cho câu truy vấn của người dùng *q* và văn bản *d* trong tập văn bản. Mô hình xác suất tính xác suất mà văn bản *d* liên quan đến cấu truy vấn của người dùng. Mô hình giả thiết xác suất liên quan của một văn bản với câu truy vấn phụ thuộc cách biểu diễn chúng. Tập văn bản kết quả được xem là liên quan và có tổng xác suất liên quan với câu truy vấn lớn nhất.

Ưu điểm của mô hình xác suất:

- Văn bản được sắp xếp dựa vào xác suất liên quan đến câu truy vấn

Nhược điểm mô hình xác suất:

- Mô hình không quan tâm đến số lần xuất hiện của từ chỉ mục trong văn bản
- Việc tính toán xác suất khá phức tạp và tốn nhiều chi phí.

Mô hình xác suất coi quá trình truy xuất tài liệu như một suy luận xác suất, trong đó các điểm tương đồng được tính như xác suất mà một tài liệu có liên quan khi đưa ra một truy vấn. trong các mô hình này thường sử dụng các định lý xác suất. Các mô hình phổ biến là: Mô hình phân biệt nhị phân, mô hình dựa vào xác suất, mô hình suy luận không chắc chắn, mô hình ngôn ngữ, mô hình phân kỳ-từ-ngẫu nhiên và phân bố ẩn Dirichlet. [8]

Mô hình Độc lập nhị phân [18] coi các tài liệu như là các vectơ nhị phân, sao cho chỉ ghi lại sự hiện diện hoặc không có các thuật ngữ trong tài liệu. Mô hình này dựa trên giả định về Tính độc lập rằng các thuật ngữ được phân phối độc lập trong tập hợp các tài liệu liên quan và các tài liệu không liên quan. Phương pháp này rất nhiều hạn chế nhưng nó mang lại kết quả tốt hơn trong nhiều bài toán. Mô hình này được coi như một thể hiện của mô hình Không gian véc tơ.

Mô hình xác định mức độ liên quan theo xác suất [19] đưa ra ước lượng xác suất tài liệu có liên quan đến truy vấn hay không bằng cách giả định rằng xác suất liên quan phụ thuộc vào truy vấn và biểu diễn tài liệu (mức độ liên quan của tài liệu tăng lên theo tần suất truy vấn). Tuy nhiên, có các hạn chế trong mô hình xác suất: các thuật ngữ trong tài liệu và truy vấn không có trọng số; và các thuật ngữ được giả định là độc lập với nhau. Một số phương pháp đã được đề xuất để giải quyết những vấn đề này, đó là một là mô hình Độc lập nhị phân, và một phương pháp dẫn xuất phổ biến dựa trên trọng số Okapi (BM25) và BM25F.

1.3.4. Mô hình ngôn ngữ

Mô hình ngôn ngữ là tập hợp các kiến thức trước đó về một ngôn ngữ nhất định, các kiến thức này có thể là các kiến thức về từ vựng, về ngữ pháp, về tần suất xuất hiện của các cụm từ, ... Một mô hình ngôn ngữ có thể được xây dựng theo hướng chuyên gia hoặc hướng dữ liệu.

Mô hình ngôn ngữ là một phân bố xác suất trên các tập văn bản, cung cấp các thông tin về phân bố xác suất tiền nghiệm (prior distribution) là các từ vựng trong bộ từ điển của một ngôn ngữ nhất định. Nói đơn giản, mô hình ngôn ngữ có thể cho biết xác suất một câu (hoặc cụm từ) thuộc một ngôn ngữ là bao nhiêu. Việc tính giá trị $p(w_1...w_n)$ trong trường hợp n vô hạn, thực tế là vô cùng khó khăn. Để giảm độ phức tạp cho việc tính toán cũng như tạo ra một hướng đi khả thi để có thể mô hình hóa ngôn ngữ, mô hình n-gram ra đời. Mô hình n-gram giả định việc mô hình ngôn ngữ là một chuỗi Markov, thỏa mãn tính chất Markov.

Các mô hình Truy xuất dựa trên đặc trưng biểu diễn tài liệu dưới dạng vectơ giá trị của các đặc trưng (hoặc chỉ các đặc trưng) và tìm cách tốt nhất để kết hợp các đặc trưng này thành một đặc trưng phù hợp duy nhất, thường bằng cách học các phương pháp xếp hạng. Hàm đặc trưng là các đặc trưng của tài liệu và truy vấn, và như vậy có thể dễ dàng kết hợp hầu hết mọi mô hình truy xuất khác nhau.

Ngoài những mô hình này, có những kỹ thuật phổ biến được sử dụng rộng rãi trong IR, chẳng hạn như:

- Sử dụng trọng số TF-IDF, TF-IDF có nghĩa là Tần suất thuật ngữ và Tần suất tài liệu nghịch đảo, là một thước đo tính điểm được sử dụng rộng rãi trong truy xuất thông tin (IR) hoặc tóm tắt. TF - IDF nhằm phản ánh mức độ liên quan của một thuật ngữ trong một tài liệu nhất định.
- Một kỹ thuật dựa trên toán học có tên là phân tách giá trị đơn lẻ được áp dụng cho các mô hình đại số dùng để giảm số chiều của không gian vectơ của một tập hợp tài liệu, do đó khiến các từ có nghĩa ngữ nghĩa chung được

hợp nhất, làm cho các truy vấn phù hợp với tài liệu liên quan ở phạm vi rộng hơn.

1.4. Kết luận chương

Một câu hỏi có thể được trả lời bằng cách tìm xem nó giống với câu hỏi nào trong bộ dữ liệu câu hỏi – câu trả lời có sẵn. Bằng cách này sinh viên có thể nhận được câu trả lời ngay sau khi hỏi mà không phải chờ đợi người hỗ trợ trả lời từng câu hỏi.

Có rất nhiều phương pháp trả lời tự động nhưng trong bài toán này, dựa vào đặc trưng của bài toán cần giải quyết và dữ liệu của bài toán nên luận văn này sẽ sử dụng phương pháp trả lời tự động dựa trên *truy xuất thông tin (IR)*. Phương pháp này sẽ tận dụng được các câu hỏi và câu trả lời có sẵn trong tập dữ liệu đã xây dựng.

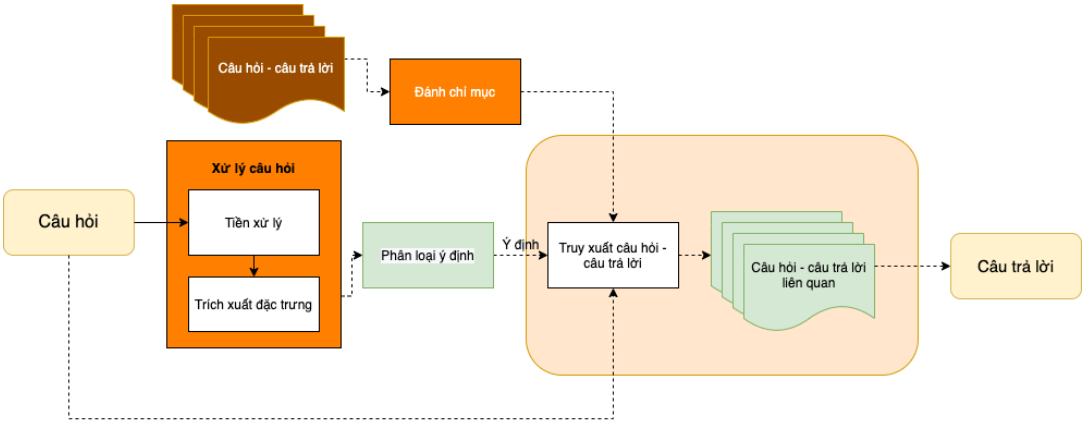
CHƯƠNG 2. PHƯƠNG PHÁP TRẢ LỜI TỰ ĐỘNG

Chương này trình bày về phương pháp trả lời tự động do học viên lựa chọn và phát triển dựa trên một số giải pháp đã có. Trước hết là kiến trúc chung của mô hình trả lời tự động, sau đó là mô tả chi tiết của từng thành phần trong mô hình. Mô hình được xây dựng để tận dụng bộ câu hỏi câu trả lời tích luỹ được tại các phòng chức năng của trường Đại học Xây dựng.

2.1. Kiến trúc mô hình

Bài toán lúc này được đặt ra như sau: có một người hỏi một câu hỏi *a*, sau đó hệ thống sẽ tìm kiếm câu hỏi *a* trong tập dữ liệu D gồm các câu hỏi - câu trả lời có sẵn đã được xây dựng từ trước. Hệ thống cần đưa ra cặp câu hỏi – câu trả lời trong D được xếp hạng cao nhất theo mức độ liên quan đến câu hỏi *a* và lấy câu trả lời ra làm câu trả lời cho câu hỏi a. Vì các câu hỏi của sinh viên hầu hết thường lặp đi lặp lại trong tập dữ liệu D nên chúng ta có thể sử dụng phương pháp truy xuất thông tin IR để xác định mức độ liên quan giữa các câu hỏi.

Tuy nhiên có một vấn đề đặt ra đó là IR sẽ xác định mức độ liên quan giữa các câu hỏi dựa trên từ khóa, nghĩa là hai câu hỏi nào có số lượng từ khóa giống nhau cao hơn sẽ được coi là liên quan đến nhau hơn. Trong một số trường hợp của tập dữ liệu hỏi đáp của trường Đại học Xây dựng nếu chỉ xét mức độ liên quan dựa trên từ khóa thì sẽ khá giống nhau, giả sử như hai câu hỏi: "*E thưa cô, chả hạn e trả hết môn mà tích luỹ chưa đủ 2.0 thì e có dc nhận để làm đồ án tốt nghiệp không ạ*" thì ý định người hỏi là hỏi về *điều kiện làm đồ án tốt nghiệp*. Trong khi câu "*vì điều kiện dịch bệnh nên e chưa thể lên trường đóng học phí được e là sinh viên năm cuối còn đồ án tốt nghiệp nữa kính mong thầy cô mở tài khoản cho e đăng kí nốt đồ án tốt nghiệp*" thì ý định câu hỏi là *đăng ký đồ án*. Như vậy để hệ thống hỏi đáp có thể trả lời chính xác câu hỏi của người dùng thì cần phải xác định được ý định của câu hỏi để thực hiện tìm kiếm hiệu quả. [Hình 2.1] thể hiện kiến trúc của mô hình hỏi đáp áp dụng để giải quyết bài toán đặt ra.



Hình 2.1: Từ câu hỏi đến câu trả lời: Mô hình xây dựng hệ thống hỏi đáp

Trong hình 2.1, hệ thống trả lời tự động có 2 thành phần chính:

- Module *xác định ý định câu hỏi* sử dụng mô hình học sâu để xác định ý định của câu hỏi, module này giống như một bộ phân loại văn bản với đầu vào là câu hỏi và đầu ra là lớp được phân loại, mỗi lớp đầu ra tương ứng với một ý định của câu hỏi.
- Module *truy xuất thông tin* để tìm kiếm câu trả lời phù hợp với câu hỏi.

Luồng xử lý của hệ thống trong hình trên được mô tả như sau:

- Đầu tiên, bộ dữ liệu gồm các câu hỏi của sinh viên trường ĐHXD và câu trả lời tương ứng vơi các câu hỏi được khởi tạp. Bộ câu hỏi và câu trả lời sẽ được tiền xử lý bằng cách loại bỏ ký tự đặc biệt, tách từ, loại bỏ từ dừng, chuẩn hóa văn bản; sau đó sẽ được chuyển thành các vector biểu diễn.
- Với câu hỏi đầu vào của sinh viên dưới dạng ngôn ngữ tự nhiên, hệ thống sẽ đưa vào một bộ phân lớp ý định để xác định ý định của câu hỏi. Bước này nhằm xác định chính xác ý định để tăng độ chính xác cho module truy xuất câu trả lời. Sau khi xác định được ý định của câu hỏi, hệ thống sẽ lọc ra trong tập dữ liệu các câu hỏi - câu trả lời có sẵn một tập con dữ liệu câu hỏi – câu trả lời mang ý định đã xác định được.
- Module truy xuất thông tin sẽ tìm kiếm trong tập con dữ liệu câu hỏi – câu trả lời xác định được ở bước trên. Việc tìm kiếm dựa trên độ giống bằng cách so sánh câu hỏi đầu vào với câu hỏi và câu trả lời trong tập dữ liệu con đó.

Với tất cả các câu trả lời được đưa ra, câu trả lời nào có độ tin tưởng cao nhất sẽ được đưa ra thành câu trả lời cho người dùng.

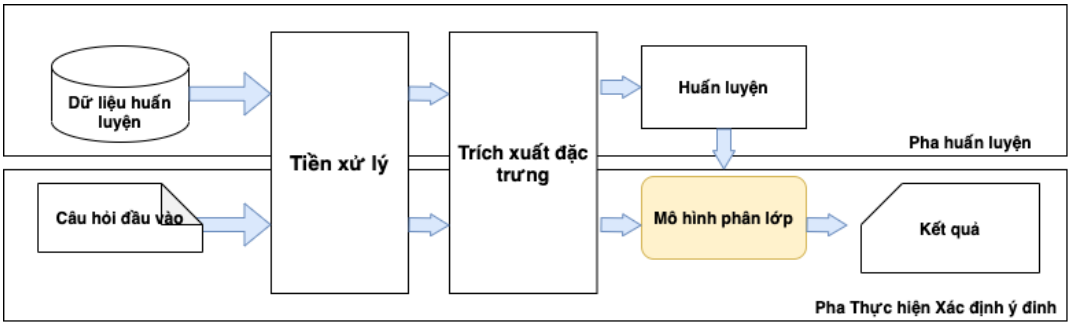
2.2. Phân loại ý định

Phân loại ý định (intent classification) là việc xác định ý định của người hỏi khi tương tác với hệ thống hỏi đáp thông qua câu hỏi hay câu truy vấn của người dùng. Ví dụ đối với câu hỏi: "Em thưa thầy cho em hỏi để đóng học phí thì đóng bằng cách nào ạ?", thì ý định của người dùng là họ đang muốn hỏi về '*học phí*', ví dụ khác như câu hỏi : "Điều kiện được học bổng trường mình là gì ạ?" thì ý định của người dùng là hỏi về '*Học bổng*. Phân tích câu hỏi là pha đầu tiên trong kiến trúc chung của hệ thống hỏi đáp, có nhiệm vụ tìm ra các thông tin cần thiết làm đầu vào cho quá trình xử lý của các pha sau (trích chọn câu trả lời). Phân loại ý định thuộc vào pha phân tích câu hỏi, vì vậy phân loại ý định câu hỏi có vai trò hết sức quan trọng, ảnh hưởng trực tiếp đến hoạt động của toàn bộ hệ thống. Nếu xác định sai ý định câu hỏi thì sẽ không thể tìm ra được câu trả lời. Dựa vào các câu hỏi thu thập được, có thể chia các câu hỏi thành 10 ý định khác nhau của câu hỏi, bao gồm:

- Các câu hỏi thắc mắc về Điểm
- Các câu hỏi thắc mắc về Học bổng
- Các câu hỏi thắc mắc về việc đăng ký môn học
- Các câu hỏi thắc mắc về học phí
- Các câu hỏi thắc mắc về lịch học
- Câu hỏi liên quan tài khoản sử dụng các hệ thống phục vụ sinh viên trong trường
- Câu hỏi liên quan đến các thủ tục hành chính của sinh viên
- Câu hỏi liên quan đến tốt nghiệp, đồ án tốt nghiệp và thực tập tốt nghiệp
- Các câu hỏi về chứng chỉ tiếng anh TOEIC
- Các câu hỏi không thuộc vào 1 trong 9 nhóm trên.

2.2.1. Luồng xử lý phương pháp xác định ý định của câu hỏi

[Hình 2.2] mô tả luồng xử lý của bài toán phân loại ý định của câu hỏi. Trong hình vẽ chia ra thành 2 pha: pha huấn luyện mô hình và pha áp dụng mô hình vào để dự đoán.



Hình 2.2: Thuật toán phân lớp ý định của câu hỏi

Pha huấn luyện được thực hiện như sau:

a. Tiền xử lý

Dữ liệu trong tập dữ liệu huấn luyện sẽ được đưa qua bước *tiền xử lý*. Bước này sẽ tiến hành một số kỹ thuật sau đây:

- *Chuẩn hóa câu hỏi:* Bước này sẽ chuẩn hóa câu hỏi đầu vào bằng cách loại bỏ các ký tự đặc biệt, đưa các câu hỏi về chữ thường, sửa lỗi chính tả và thay thế từ viết tắt dựa trên từ điển các từ viết tắt.
- *Tách từ Tiếng Việt:* Bước này sẽ tiến hành tách câu hỏi thành các từ Tiếng Việt.
- *Loại bỏ từ dừng:* Bước này sẽ tiến hành loại bỏ các từ không mang ý nghĩa trong câu.

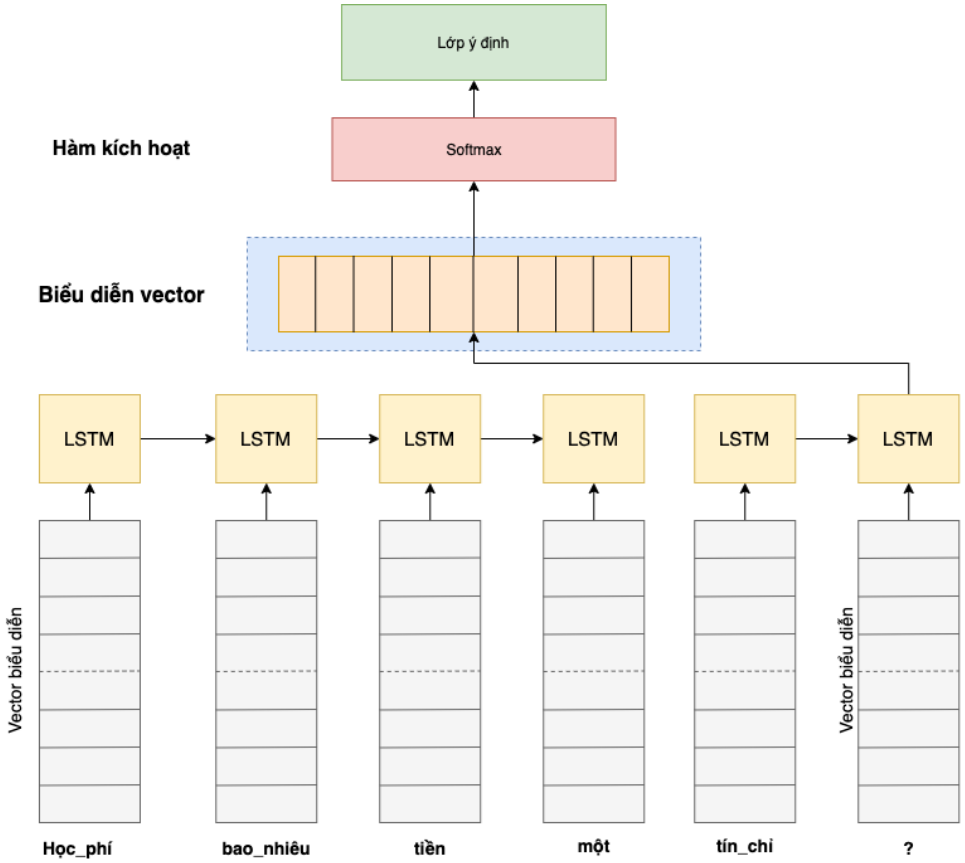
Sau khi thực hiện xong quá trình tiền xử lý, văn bản sẽ được đưa vào bước 2.

b. Trích xuất đặc trưng

Bước này sẽ thực hiện nhiệm vụ đưa văn bản thành các vector biểu diễn. Dựa trên vector biểu diễn từ có thể gộp lại để tạo thành biểu diễn văn bản. Vector biểu diễn văn bản có chiều dài cố định, văn bản nào có chiều dài nhỏ hơn chiều dài này sẽ được thêm padding để đưa về ma trận biểu diễn có kích thước giống nhau giữa các văn bản.

c. Mô hình phân lớp ý định

Mô hình phân lớp ý định sử dụng mạng nơ-ron học sâu LSTM được mô tả chi tiết như [hình 2.3].



Hình 2.3: Mô hình phân lớp ý định câu hỏi

Kiến trúc mô hình phân loại ý định với số lượng các tham số như sau:

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
lstm (LSTM)                  (None, 128)               127488
_________________________________________________________________
dropout (Dropout)            (None, 128)               0
_________________________________________________________________
dense (Dense)                (None, 64)                8256
_________________________________________________________________
dense_1 (Dense)              (None, 10)                650
=================================================================
Total params: 136,394
Trainable params: 136,394
Non-trainable params: 0
```

2.2.2. Tiền xử lý dữ liệu

a. Phương pháp tách từ tiếng việt

Tách từ, về mặt biểu hiện, là gom nhóm các từ đơn liền kề thành một cụm từ có ý nghĩa. Ví dụ: "Cách tách từ cho Tiếng Việt." sau khi tách từ thì thành "Cách tách từ cho Tiếng_Việt ." Về hình thức, các từ đơn được gom nhóm với nhau bằng cách nối với nhau bằng ký tự gạch dưới "_", trong trường hợp này là từ Tiếng_Việt. Sau khi thực hiện tách từ thì mỗi từ (token) trong câu được cách nhau bởi một khoảng trắng, trong trường hợp này như "Tiếng_Việt ." thì từ "Tiếng_Việt" cách đấu "." bởi 1 khoảng trắng. Đây là quy ước chung cho tất cả các ngôn ngữ của bài toán tách từ trong xử lý ngôn ngữ tự nhiên. Việc quy ước như vậy là để tạo thành chuẩn chung và để dễ xử lý hơn trong lập trình.

Đối với các bài toán dựa trên xử lý ngôn ngữ tự nhiên thì tách từ là một bước quan trọng, ảnh hưởng trực tiếp đến chất lượng mô hình. Khác với tiếng Anh, một từ tiếng Việt có thể được tạo bởi nhiều hơn một âm. Ví dụ từ (word) “cá_nhân” được tạo lên bởi 2 âm (syllable) là “cá” và “nhân”. Trong khi hai từ đơn “cá” và từ đơn “nhân” lại có thể mang ý nghĩa khác. Do vậy, tách từ tiếng Việt là bước quan trọng chúng ta cần thực hiện trước khi đưa dữ liệu vào các bước tiếp theo, ví dụ như word embedding.

Các phương pháp phổ biến:

- **Sử dụng bộ từ điển:** Ví dụ: Thuật toán so khớp từ dài nhất (longest matching) Thuật toán này khá đơn giản, đối với 1 đoạn text đầu vào “Thuế thu nhập cá nhân” thì chúng ta thực hiện như sau (giải thuật đơn giản hoá):

```
current_syllable_id = 0
while current_syllable_id and not done {
check if “word word+1 word+2” is in tri-gram-dictionary
=> {take word_word1_word2; current_syllable_id += 3}
else if “word word+1” is in bi-gram-dictionary
=> {take word_word1; current_syllable_id += 2}
else
=> {take word only; current_syllable_id += 1} }
```

Bắt đầu từ trái sang phải, với vị trí từ hiện tại chúng ta kiểm tra xem từ đó và 2 từ tiếp theo có thể ghép thành 1 từ có nghĩa hay không bằng cách kiểm tra trong từ điển tri-gram. Nếu không thể tạo ra được từ có nghĩa từ 3 từ thì ta tiếp tục kiểm tra xem từ hiện tại và từ tiếp theo có thể ghép được thành một từ có nghĩa hay không bằng cách kiểm tra trong từ điển bi-gram. Cuối cùng nếu không thể ghép được thì ta coi đó là từ đơn. Chỉ số current_syllable_id sẽ dịch 1, 2, hay 3 đơn vị tuỳ theo số từ ta ghép được.

- **Thuật toán so khớp từ cực đại:** Ngoài thuật toán so khớp từ dài nhất còn có thuật toán so khớp từ cực đại. Idea và thuật toán so khớp cực đại khá tương tự nhưng thay vì xét từng từ thì ta xét cả câu để tránh bị sai như trong câu: "Học sinh học sinh vật học" sẽ thành "Học_sinh học_sinh vật học" (theo Longest Matching).
- **Sử dụng học máy**: Ví dụ: Thuật toán Conditional Random Field (CRF) Phương pháp chung là chúng ta đưa bài toán về dạng classification và sử dụng các thuật toán classification khác nhau để học. Thuật toán CRF sử dụng xác suất có điều kiện và thống kê để mô hình hoá đầu vào là các features và đầu ra là class ID. Thuật toán này phù hợp với các bài toán classification trong NLP như tokenization (word segmentation), POS tagging, NER.

b. Phương pháp loại bỏ từ dừng

Từ dừng là những từ trong bất kỳ ngôn ngữ nào không bổ sung nhiều ý nghĩa cho một câu. Chúng có thể được bỏ qua một cách an toàn mà không làm mất đi ý nghĩa của câu. Đối với một số công cụ tìm kiếm, đây là một số từ chức năng ngắn, phổ biến nhất, chẳng hạn như, is, at, which, and on. Trong trường hợp này, các từ dừng có thể gây ra vấn đề khi tìm kiếm các cụm từ bao gồm chúng, đặc biệt là trong các tên như "The Who" hoặc "Take That". Một số ưu và nhược điểm của việc loại bỏ từ dừng trong NLP.

Ưu điểm:

- Các từ dừng thường bị xóa khỏi văn bản trước khi huấn luyện mô hình học sâu và học máy vì các từ dừng xuất hiện rất nhiều và không mang ý nghĩa, do đó

cung cấp rất ít hoặc không có thông tin có thể được sử dụng để phân loại hoặc phân cụm.

- Khi loại bỏ các từ dừng, kích thước tập dữ liệu giảm và thời gian huấn luyện mô hình cũng giảm mà không ảnh hưởng lớn đến độ chính xác của mô hình.
- Loại bỏ từ dừng có khả năng giúp cải thiện hiệu suất, vì có ít từ hơn và chỉ còn lại các từ quan trọng. Do đó, độ chính xác phân loại có thể được cải thiện

Nhược điểm:

- Việc lựa chọn và loại bỏ các từ dừng không đúng cách có thể thay đổi ý nghĩa của văn bản của chúng ta. Vì vậy, chúng ta phải cẩn thận trong việc lựa chọn từ dừng của mình.

  Ví dụ: "Bộ phim này không hay.". Nếu chúng ta loại bỏ (không) trong bước tiền xử lý, câu (phim này hay) cho biết nó là khẳng định nhưng bị diễn giải sai.

2.2.3. Trích xuất đặc trưng

Trích xuất đặc trưng là tìm cách đưa văn bản về biểu diễn dưới dạng vector hay ma trận mà các biểu diễn này vẫn thể hiện được các đặc trưng của văn bản. Các kỹ thuật trích xuất đặc trưng văn bản sẽ được giới thiệu dưới đây.

Ngôn ngữ hay cụ thể hơn là ngôn ngữ giao tiếp dưới dạng văn bản hay lời nói là một cách thức giao tiếp chỉ có ở con người mà không có ở các loài động vật khác. Một đứa trẻ phải mất đến 2 năm mới bắt đầu có nhận thức và suy nghĩ về ngôn ngữ. Một phần là do ngôn ngữ rất phức tạp, phần còn lại là muốn hiểu được ngôn ngữ cần phải có thời gian để học. Đối với con người việc tiếp cận với ngôn ngữ không hề đơn giản, vậy làm cách nào để máy tính có thể biểu diễn và hiểu được ngôn ngữ giống như con người. Đây thực sự là một bài toán lớn, trong lịch sử phát triển của khoa học máy tính, con người đã thực hiện rất nhiều các nghiên cứu chuyên sâu về việc biểu diễn ngôn ngữ với mục tiêu máy tính có thể xử lý các tác vụ về ngôn ngữ giống như con người.

Các nghiên cứu trong lĩnh vực xử lý ngôn ngữ tự nhiên bằng máy tính đưa ra nhiều phương pháp để biểu diễn ngôn ngữ khác nhau. Đơn vị nhỏ nhất của ngôn ngữ là từ (hoặc ký tự), nên các nghiên cứu tìm cách đưa biểu diễn ngôn ngữ về phương pháp biểu diễn dựa trên từ (hoặc ký tự). Vì máy tính chỉ có khả năng tính toán dựa trên số học nên người ta tìm cách đưa các biểu diễn này về dưới dạng các vector hoặc ma trận.

Các phương pháp biểu diễn từ là phương pháp ánh xạ mỗi từ vào một không gian số thực nhiều chiều nhưng có kích thước nhỏ hơn nhiều so với kích thước từ điển. Sau đây là một số phương pháp phổ biến:

c. One-hot encoding (tạm gọi mã hoá số 1):

Đây là cách đơn giản để biểu diễn ngôn ngữ sang dạng vector với số chiều là kích thước từ điển. Giống như tên của nó, chỉ ở chiều mà vị trí một từ xuất hiện trong từ điển có giá trị là 1. Các chiều khác đều có giá trị là 0.

Ví dụ đơn giản khi tập dữ liệu của ta có 3 câu:

- Câu 1: tôi đang đi tìm một_nửa của mình.
- Câu 2: tôi đã ăn một_nửa quả táo
- Câu 3: tôi đã đi tìm một_nửa quả táo

Như vậy từ điển V = { t ô i $_1$, đ a n g $_2$, đ i $_3$, t ì m $_4$, m ộ t − n ử a $_5$, c ủ a $_6$, m ì n h $_7$, đ ã $_8$, ă n $_9$, q u ả $_{10}$, t á o $_{11}$ } có kích thước S = 11. Để biểu diễn các từ bằng one-hot vector thì các từ sẽ có giá trị như sau:

- Tôi = [1 0 0 0 0 0 0 0 0 0 0]
- Đang = [0 1 0 0 0 0 0 0 0 0 0]

  …
- Mình = [0 0 0 0 0 0 1 0 0 0 0]
- Táo = [0 0 0 0 0 0 0 0 0 0 1]

Nhược điểm khi sử dụng phương pháp biểu diễn one-hot encoding đó là: không biểu diễn được các từ đồng nghĩa, giả sử như có 2 từ đồng nghĩa là “tôi” và “mình” tuy nhiên độ giống giữa 2 vector biểu diễn 2 từ này = 0 (sử dụng khoảng cách cosine); kích thước vector sẽ phụ thuộc vào số lượng từ trong từ điển, giả sử

để biểu diễn một bộ ngữ liệu mà từ điển gồm n từ (n thường rất lớn) thì sẽ cần một one-hot vector có kích thước đúng bằng n để biểu diễn; ngoài ra nhược điểm nữa là cần không gian nhớ lớn để lưu thông tin.Các mô hình phân loại sau đó dựa trên cách biểu diễn này sẽ gặp phải vấn đề biểu diễn thưa (sparsity issues).

Khác với Count Vector chỉ xét đến tần số xuất hiện của từ trong một document, tf-idf Vector quan tâm cả tần số xuất hiện của từ trong toàn bộ tập dữ liệu, chính do đặc điểm này mà tf-idf Vector có tính phân loại cao hơn so với Count Vector. tf-idf (Term Frequency-Inverse Document Frequency) Vector là một vector số thực cũng có độ dài D với D là số văn bản, nó được tính bằng tích của 2 phần bao gồm tf và idf, công thức của mỗi phần tử của vector được tính như sau:

TF- term frequency – tần số xuất hiện của 1 từ trong 1 văn bản. Cách tính:

$$\mathrm{tf}(t,d) = \frac{\mathrm{f}(t,d)}{\max\{\mathrm{f}(w,d) : w \in d\}} \quad (2.1)$$

Thương của số lần xuất hiện 1 từ trong văn bản và số lần xuất hiện nhiều nhất của một từ bất kỳ trong văn bản đó. (giá trị sẽ thuộc khoảng [0, 1])

- *f(t,d)* - số lần xuất hiện từ t trong văn bản d.
- *max{f(w,d):w∈d}* - số lần xuất hiện nhiều nhất của một từ bất kỳ trong văn bản.

DF – inverse document frequency. Tần số nghịch của 1 từ trong tập văn bản (corpus).

Tính IDF để giảm giá trị của những từ phổ biến. Mỗi từ chỉ có 1 giá trị IDF duy nhất trong tập văn bản.

$$\mathrm{idf}(t,D) = \log \frac{|D|}{|\{d \in D : t \in d\}|} \quad (2.2)$$

- $|D|$ - tổng số văn bản trong tập D
- $\{d \in D : t \in d\}$: - số văn bản chứa từ nhất định, với điều kiện {\displaystyle t}t xuất hiện trong văn bản d. Nếu từ đó không xuất hiện ở bất cứ 1 văn bản nào trong tập thì mẫu số sẽ bằng 0 => phép chia cho không không hợp lệ, vì thế người ta thường thay bằng mẫu thức $1 + |\{d \in D : t \in d\}|$.

Cơ số logarit trong công thức này không thay đổi giá trị của 1 từ mà chỉ thu hẹp khoảng giá trị của từ đó. Vì thay đổi cơ số sẽ dẫn đến việc giá trị của các từ thay đổi bởi một số nhất định và tỷ lệ giữa các trọng lượng với nhau sẽ không thay đổi. (nói cách khác, thay đổi cơ số sẽ không ảnh hưởng đến tỷ lệ giữa các giá trị IDF).

d. Ma trận đồng xuất hiện

Nhược điểm của cả hai phương pháp trên chính là việc nó chỉ chú trọng đến tần số xuất hiện của một từ, dẫn tới nó hầu như không mang ý nghĩa gì về mặt ngữ cảnh, ***Co-occurrence Matrix*** phần nào giải quyết vấn đề đó. ***Co-occurrence Matrix*** có ưu điểm là bảo tồn mối quan hệ ngữ nghĩa giữa các từ, được xây dựng dựa trên số lần xuất hiện của các cặp từ trong ***Context Window***. Một ***Context Window*** được xác định bởi kích thước và hướng của nó.

Có hai cách chính để tiếp cận việc biểu diễn từ vựng dựa trên ngữ cảnh:

- ***Dựa trên thống kê***: Đây là thuật toán không giám sát dựa trên ma trận xuất hiện của các từ. Sử dụng ma trận đồng xuất hiện, 1 từ sẽ được biểu diễn bởi các từ đi cùng với nó.
- ***Biểu diễn từ dựa trên ngữ cảnh***: Đây là phương pháp học có giám sát. Thay vì lưu thông tin xuất hiện của các từ bằng cách đếm trực tiếp như ma trận đồng xuất hiện, phương pháp này học để đoán từ lân cận của tất cả các từ.

Ma trận đồng xuất hiện: được đề xuất ở hai mức là mức document (văn bản) và mức windows (cửa sổ từ). Mức văn bản cho thông tin chung về các chủ đề hướng tới các phương pháp LSA (latent semantic analysis). Mức *cửa sổ từ* cho thông tin về cả chức năng *cú pháp* của từ và *ngữ nghĩa.*

Ma trận đồng xuất hiện của tập dữ liệu 3 câu ví dụ ở trên với window_size=1. Tức là ma trận đồng xuất hiện chỉ đếm 2 từ liền kề nhau. Nếu tăng window_size=2 thì ma trận này sẽ ghi nhận các từ là đồng xuất hiện với các từ đứng kề bên và cách một từ. Như ví dụ trên, khi tăng window_size=2 thì giá trị giữa từ tôi và đi ở ma trận đồng xuất hiện sẽ có giá trị là 2 do tôi và đi xuất hiện ở câu 1 và câu 3 và nằm trong khoảng cách window_size=2.

Counts	tôi	đang	đi	tìm	một_nửa	của	mình	đã	ăn	quả	táo
tôi	0	1	0	0	0	0	0	2	0	0	0
đang	1	0	1	0	0	0	0	0	0	0	0
đi	0	1	0	2	0	0	0	1	0	0	0
tìm	0	0	2	0	2	0	0	0	0	0	0
một_nửa	0	0	0	2	0	1	0	0	1	2	0
của	0	0	0	0	1	0	1	0	0	0	0
mình	0	0	0	0	0	1	0	0	0	0	0
đã	2	0	1	0	0	0	0	0	1	0	0
ăn	0	0	0	0	1	0	0	1	0	0	0
quả	0	0	0	0	2	0	0	0	0	0	2
táo	0	0	0	0	0	0	0	0	0	2	0

Hình 2.4: Ma trận đồng xuất hiện

e. Word embeddings (Tập nhúng từ)

Do các vấn đề của ma trận đồng xuất hiện nên đã có nhiều nghiên cứu hướng theo giải pháp học biểu diễn ở số chiều thấp hơn. Trước khi word2vec ra đời đã có nhiều nghiên cứu như bài của nhóm Bengio năm 2003. Nhưng những giải pháp này vẫn gặp vấn đề về chi phí tính toán. Cho đến năm 2013, nhóm Mikolov đã đóng góp vào NLP với giải pháp mới là word2Vec. Từ thời điểm này hàng loạt bài toán NLP được giải quyết với độ chính xác cao hơn nhiều so với trước.

Tổng quát phương pháp và ý tưởng chính của **word2vec** như sau:

- Thay vì lưu thông tin xuất hiện của các từ bằng cách đếm trực tiếp như ***ma trận đồng xuất hiện***, ***word2vec*** học để đoán từ lân cận của tất cả các từ.
- Các giải pháp sau đó như ***Glove*** cũng tương tự word2vec được đề xuất bởi nhóm Pennington năm 2014.
- Tính toán nhanh hơn và dễ dàng thêm dữ liệu mới vào trong mô hình học.

Dựa vào ưu và nhược điểm của từng phương pháp cùng với yêu cầu của mô hình phân loại ý định, luận văn này sẽ sử dụng mô hình Skip-gram để thực hiện bước trích xuất đặc trưng :

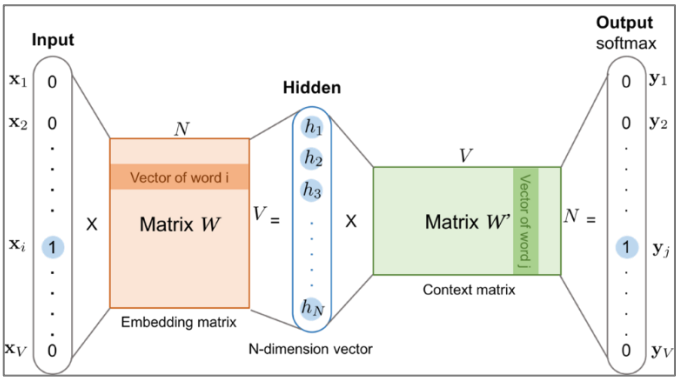
Skip-Gram Model Sử dụng cửa sổ trượt với kích thước cố định để di chuyển từ trái qua phải của câu. Từ ở giữa là "target" và các từ bên trái và phải trong cửa sổ đó là các từ thể hiện ngữ cảnh. Mô hình skip-gram được huấn luyện để dự đoán xác suất của từ theo ngữ cảnh đưa ra.

Ví dụ: "***The man who passes the sentence should swing the sword.***" – Ned Stark

Các cặp từ đích - các từ ngữ cảnh như là mẫu để huấn luyện được tạo bởi cửa sổ kích thước là 5 dọc theo câu.

Cửa sổ kích thước (kích cỡ = 5)	**Từ mục tiêu**	**Ngữ cảnh**
[The man who]	the	man, who
[The man who passes]	man	the, who, passes
[The man who passes the]	who	the, man, passes, the
[man who passes the sentence]	passes	man, who, the, sentence
…	…	…
[sentence should swing the sword]	swing	sentence, should, the, sword
[should swing the sword]	the	should, swing, sword
[swing the sword]	sword	swing, the

Mỗi cặp từ tập trung ngữ cảnh được coi là 1 quan sát trong tập dữ liệu. Ví dụ từ mục đích "swing" ở trường hợp trên sẽ tạo ra các mẫu để đưa vào huấn luyện gồm ("swing", "sentence"), ("swing", "should"), ("swing", "the"), and ("swing", "sword").

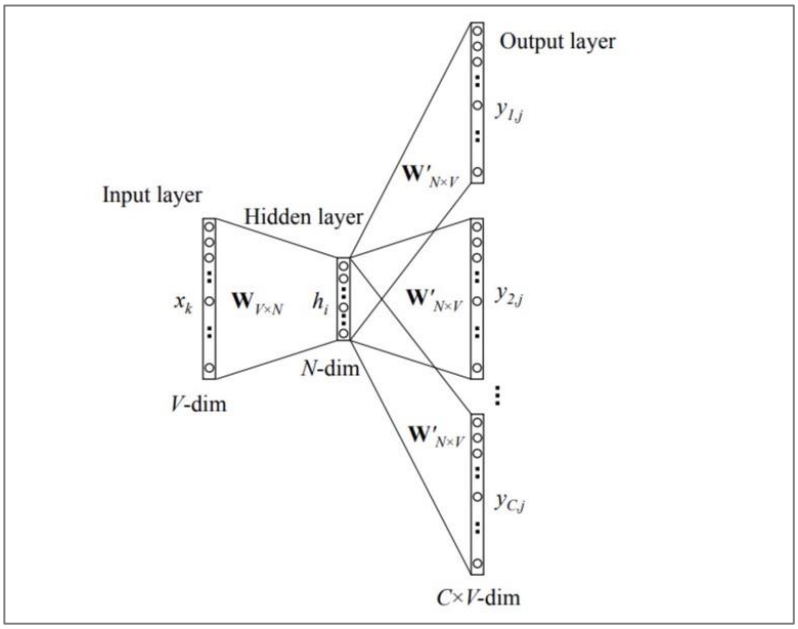


Hình 2.5: Mô hình skip-gram.

Trong [hình 2.5] cả đầu vào x và đầu ra y đều là biểu diễn dưới dạng dạng one-hot-encoding. Các hidden layer là word embedding có kích thước N

Đưa ra từ vựng có kích thước V, chúng sẽ học từ các biểu diễn từ bằng vector có kích thước N. Mô hình học cách dự đoán 1 từ ngữ cảnh từ 1 từ target (đích) tại 1 thời điểm.

Cho từ mục tiêu wc tại vị trí c trong câu văn bản, khi đó đầu vào của mô hình Skip-gram cũng chính là từ mục tiêu w_c và đầu ra của mô hình là các từ ngữ cảnh (w_{c-m},..., w_{c-1},w_{c+1},...w_{c+m}) xung quanh từ w_c trong phạm vi m.



Hình 2.6: Ảnh minh họa cho mô hình Skip-gram ở dạng tổng quát

Mô hình Skip-gram tổng quát được thể hiện trong hình bên trên với đầu vào gồm một từ mục tiêu duy nhất, đầu ra gồm C từ ngữ cảnh xung quanh từ mục tiêu đầu vào, V là kích thước của tập từ vựng trong tập ngữ liệu dùng để huấn luyện và hyperparameter N là kích thước của hidden layer. Các unit thuộc các layer kế cận nhau được kết nối theo kiểu fully connected. Gọi vwI là vector đầu vào đại diện cho từ đầu vào duy nhất w^I. Các từ trong câu đầu vào của mô hình được chuyển về dưới dạng vector one-hot $x^{(k)}$:

$$x^{(k)} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_k \\ \vdots \\ x_V \end{bmatrix}$$

Ma trận W với kích thước V×N là ma trận trọng số từ lớp đầu vào đến lớp ẩn có dạng như sau:

$$W_{V\times N} = \begin{bmatrix} w_{11} & w_{12} & \cdots & w_{1N} \\ w_{21} & w_{22} & \cdots & w_{2N} \\ \vdots & \vdots & \ddots & \vdots \\ w_{\mathrm{V}1} & w_{\mathrm{V}2} & \cdots & w_{VN} \end{bmatrix}$$

Trong ma trận W, mỗi hàng thứ ii của ma trận chính là vector đại diện tương ứng cho từ thứ ii trong tập từ vựng và N do chúng ta định nghĩa. Ma trận này thu được sau khi huấn luyện là kết quả cần quan tâm do chứa các vector đại diện cho các từ trong tập từ vựng. Ma trận h của lớp ẩn kích thước là N×1 có dạng như sau:

$$h = \begin{bmatrix} h_1 \\ h_2 \\ \vdots \\ h_N \end{bmatrix}$$

Mỗi phần tử của ma trận h tương ứng với output của mỗi hidden layer unit. Activation function của các hidden layer unit đều là hàm tuyến tính φ(x)=x. Ma trận W′ có chiều N×V là ma trận trọng số từ lớp ẩn đến lớp đầu ra có dạng như sau:

$$W'_{N\times V} = \begin{bmatrix} w'_{11} & w'_{12} & \cdots & w'_{1V} \\ w'_{21} & w'_{22} & \cdots & w'_{2V} \\ \vdots & \vdots & \ddots & \vdots \\ w'_{N1} & w'_{N2} & \cdots & w'_{NV} \end{bmatrix}$$

Trong đầu ra thay vì chỉ có một phân phối, chúng ta tạo ra C phân phối. Gọi yc,j là phần tử thứ j trong vector đầu ra thứ c với c=1,2,…C. Do $x^{(k)}$ là vector one-hot đầu vào duy nhất nênh được tính như sau:

$$h = W_{(k,\cdot)}^{T} = v_{w_I}^{T} \qquad (2.3)$$

Giá trị của $y_{c,j}$ biểu diễn cho xác suất xuất hiện của từ thứ j trong tập từ vựng gồm V từ ở đầu ra thứ c được tính như sau:

$$p\left(w_{c,j} = w_{O,c} | w_I\right) = y_{c,j} = \frac{\exp\left(u_{c,j}\right)}{\sum_{j'=1}^{V} \exp\left(u_{j'}\right)} \qquad (2.4)$$

Trong đó $w_{c,j}$ là từ thứ j trong tập từ vựng gồm V từ tương ứng ở đầu ra thứ c và $w_{O,c}$ là từ ngữ cảnh đầu ra thứ c hiện tại. Do các đầu ra đều sử dụng chung các trọng số nên $u_{c,j}$ được tính bằng công thức sau:

$$u_{c,j} = u_j = v_{w_j}^{'\,T} \cdot h \text{ với } c = 1, 2, \ldots, C \qquad (2.5)$$

Trong đó v'_{wj} là vector đầu ra của từ thứ j trong tập từ vựng wj và lấy từ cột tương ứng của ma trận trọng số W′. Hàm mất mát E được cho bởi công thức sau:

$$\begin{aligned} E &= -\log p\left(w_{O,1}, w_{O,2}, \ldots, w_{O,C} | w_I\right) \\ &= -\log \prod_{c=1}^{C} \frac{\exp\left(u_{c,j_c^*}\right)}{\sum_{j'=1}^{V} \exp\left(u_{j'}\right)} \\ &= -\sum_{c=1}^{C} u_{j_c^*} + C \cdot \log \sum_{j'=1}^{V} \exp\left(u_{j'}\right) \end{aligned} \qquad (2.6)$$

Trong đó j_c^* là vị trí hiện tại của từ ngữ cảnh thứ c trong tập từ vựng. Các trọng số từ lớp ẩn đến lớp đầu ra trong ma trận W′ được cập nhật theo phương trình sau:

$$v'^{(new)}_{w_j} = v'^{(old)}_{w_j} - \eta \cdot EI_j \cdot h \text{ với } j = 1, 2, \ldots, V$$

Trong đó EI_j được tính như sau:

$$EI_j = \sum_{c=1}^{C} \frac{\partial E}{\partial u_{c,j}} \qquad (2.7)$$

Tiếp đến các trọng số trong ma trận W từ lớp đầu vào đến lớp ẩn được cập nhật như sau:

$$v_{w_I}{}^{(new)} = v_{w_I}{}^{(old)} - \eta \cdot EH^T \qquad (2.8)$$

Trong đó EH là một vector có N chiều và mỗi phần tử của nó được tính như sau:

$$EH_i = \sum_{j=1}^{V} EI_j \cdot w'_{ij} \qquad (2.9)$$

2.2.4. Mô hình phân lớp

Để phân loại văn bản với đầu vào là các vector biểu diễn, luận văn sẽ sử dụng mạng nơ-ron học sâu để tiến hành phân lớp. Có nhiều kiến trúc mạng nơ-ron khác nhau, để giải quyết bài toán này tôi sử dụng Mạng bộ nhớ dài-ngắn (Long Short Term Memory networks) là một dạng của mạng hồi quy RNN. Trước tiên tôi sẽ trình bày sơ lược về cấu trúc mạng RNN sau đó sẽ đi sâu vào trình bày về mạng LSTM.

a. RNN

Sự hiểu biết về ngôn ngữ tự nhiên được gắn liền với suy nghĩ của con người. Trong quá trình đọc, việc hiểu một văn bản không chỉ dừng lại ở việc hiểu từng từ đơn lẻ, mà phải thông qua thứ tự sắp xếp chúng. Nói cách khác, cần phải có mô hình có thể mô hình hóa các văn bản dựa trên các từ riêng lẻ và thứ tự sắp xếp của chúng.

Với mạng nơ-ron truyền thẳng truyền thống không phù hợp để ngoại suy thông tin từ thứ tự các đầu dưới dạng tuần tự theo thời gian (dạng time series), chúng chỉ quan tâm đến đầu vào của mạng và thực hiện biến đổi qua các layer của mạng.

Việc coi các từ đầu vào độc lập với các từ khác hay độc lập với ngữ cảnh sẽ khiến mô hình NLP gặp nhiều hạn chế. Thật vậy, giống như quá trình đọc hiểu văn bản của con người, khi đọc một bài báo chúng ta không cần phải nhớ chính xác hết toàn bộ các từ và thứ tự các từ trong văn bản, nhưng để rút được ra ý nghĩa mà văn bản muốn biểu đạt thì chúng ta cần phải có "trí nhớ" để ghi nhớ lại các thông tin có liên quan trước đó.

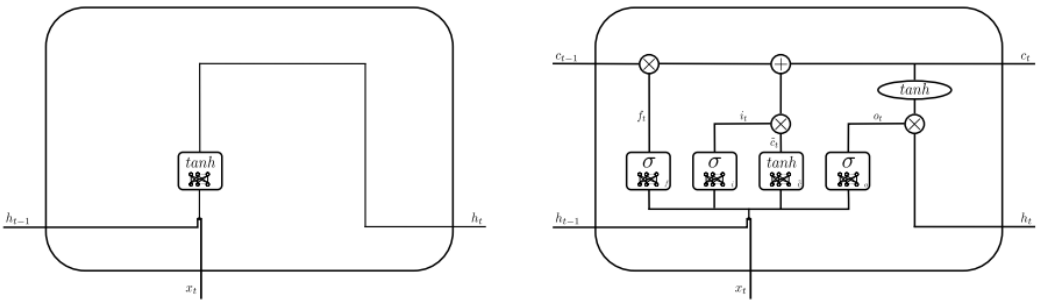
Điều này dẫn đến sự ra đời của Mạng RNN (Recurrent Neural Network) sinh ra để giải quyết vấn đề đó. Mạng này chứa các vòng lặp bên trong cho phép thông tin có thể lưu lại được. Đôi lúc ta chỉ cần xem lại thông tin đằng trước là đủ để biết được tình huống hiện tại. Ví dụ, ta có câu: "các đám may trên bầu trời" thì ta chỉ cần đọc tới "các đám may trên bầu" là đủ biết được chữ tiếp theo là "trời" rồi. Trong tình huống này, khoảng cách tới thông tin có được cần để dự đoán là nhỏ, nên RNN hoàn toàn có thể học được. Một vấn đề gặp phải đối với mạng RNN đó là việc ghi nhớ xa. Trong nhiều tình huống ta buộc phải sử dụng nhiều ngữ cảnh hơn để suy luận. Ví dụ, dự đoán chữ cuối cùng trong đoạn: "*Vừa qua em có nộp học phí nhưng nhà trường thông báo đình chỉ học do không nộp học phí*.". Rõ ràng là các thông tin gần ("*do không nộp*") chỉ có phép ta biết được đằng sau nó sẽ là tên của một danh từ nào đó, có thể là "học phí", "giấy tờ", "bài thi",… còn không thể nào biết được đó là danh từ gì. Muốn biết là từ gì, thì ta cần phải có thêm ngữ cảnh

"*Vừa qua em có nộp học phí*" nữa mới có thể suy luận được. Rõ ràng là khoảng cách thông tin lúc này khá xa.

Về lý thuyết, RNN có thể giữ lại ở trạng thái của chúng thông tin mà thuật toán học tìm thấy trong một chuỗi các đầu vào trong quá trình huấn luyện. Tuy nhiên, trong sự lan truyền ngược của gradient theo thời gian, các giá trị đầu ra ảnh hưởng đến hàm chi phí làm cho hàm chi phí quá nhỏ và tiến dần về 0, chỉ sau một vài bước, do hàm chi phí quá nhỏ không còn giúp ích được gì cho việc học tham số (vấn đề biến mất gradient) nên việc học không hiệu quả.

b. Mạng LSTM

Mạng bộ nhớ dài-ngắn (Long Short Term Memory networks), thường được gọi là LSTM - là một dạng đặc biệt của RNN, nó có khả năng học được các phụ thuộc xa của ngữ cảnh. LSTM được giới thiệu bởi Hochreiter & Schmidhuber (1997), và sau đó đã được cải tiến và phổ biến bởi rất nhiều người trong ngành. Chúng hoạt động cực kì hiệu quả trên nhiều bài toán khác nhau nên dần đã trở nên phổ biến như hiện nay.



Hình 2.7: Biểu diễn của mô hình LSTM và RNN

LSTM được thiết kế để tránh được vấn đề phụ thuộc xa (long-term dependency). Việc nhớ thông tin trong thời gian dài là đặc tính mặc định của chúng, chứ ta không cần phải huấn luyện nó để có thể nhớ được. Tức là ngay nội tại của nó đã có thể ghi nhớ được mà không cần bất kì can thiệp nào.

Mọi mạng hồi quy đều có dạng là một chuỗi các mô-đun lặp đi lặp lại của mạng nơ-ron. Với mạng RNN chuẩn, các mô-dun này có cấu trúc rất đơn giản, thường là một tầng tanh.

Để khắc phục (ít nhất một phần) vấn đề của trí nhớ ngắn hạn, kiến trúc Long short-term memory (LSTM) đã được giới thiệu. Không giống như mạng RNN cổ điển, LSTM có cấu trúc phức tạp hơn nhiều, trong đó một mạng nơ-ron đơn được thay thế bằng bốn mạng tương tác với nhau. Yếu tố đặc biệt của LSTM là trạng thái tế bào c, cho phép thông tin truyền dọc theo chuỗi thông qua việc truyền thẳng một cách đơn giản. Việc bổ sung hoặc loại bỏ thông tin được quy định bởi ba cấu trúc, được gọi là "cổng", mỗi cấu trúc có các mục tiêu cụ thể. Các cổng đều đóng vai trò có nhiệm vụ sàng lọc thông tin với mỗi mục đích khác nhau:

• *Forget gate*: Có nhiệm vụ loại bỏ những thông tin không cần thiết nhận được khỏi cell internal state

• *Input gate*: Có nhiệm vụ chọn lọc những thông tin cần thiết nào được thêm vào cell internal state

• *Output gate*: Có nhiệm vụ xác định những thông tin nào từ cell internal state được sử dụng như đầu ra

2.2.5. Tăng cường dữ liệu để huấn luyện mô hình phân lớp ý định

Phân lớp ý định thực tế là bài toán phân lớp văn bản, một trong những vấn đề gặp phải với hầu hết các bài toán về học máy nói chung và bài toán về phân lớp nói riêng là vấn đề về thiếu dữ liệu. "Deep learning is a data-hungry framework". Tạm dịch câu này là Học sâu là 1 framework luôn "đói dữ liệu". Câu này có ý nghĩa là dữ liệu là một phần quan trọng trong học sâu nói riêng và trong học máy nói chung. Và bởi vì deep learning là thuật toán dựa trên data (data-driven approach), và càng nhiều data thì càng dễ dẫn đến chất lượng các ứng dụng học máy được cải thiện. Sau khi trình bày các kỹ thuật để xác định ý định của câu hỏi tôi sẽ trình bày một kỹ thuật được áp dụng để bổ sung dữ liệu huấn luyện cho mô hình phân lớp ý định để tăng độ chính xác của mô hình.

Tăng cường dữ liệu là một khái niệm khá phổ biến trong deep learning mà chắc hẳn ai đang nghiên cứu cũng đã từng nghe hoặc sử dụng đến. Cụ thể hơn, Data Augmentation là kỹ thuật tạo ra thêm dữ liệu để bổ sung cho tập dữ liệu để giúp mô hình khái quát tốt hơn. Các kỹ thuật data augmentation được sử dụng nhiều trong thị giác máy tính, thuật toán supervised learning…

Vậy nếu giờ chúng ta phải xử lý bài toán có giữ liệu giới hạn thì phải làm sao? Không đủ dữ liệu sẽ dẫn tới vấn đề như

- Thiếu tính khái quát: Over-fitting hay như một kiểu học vẹt, kết quả trên tập train thì kết quả trên test thì thấp.
- Chất lượng dự đoán sẽ không ổn định:
- Nhiễu với đầu vào ảnh hưởng lớn tới chất lượng dự đoán
- Mất cân bằng giữa các lớp cần phân loại dẫn đến kết quả đánh giá thiếu chính xác…

Và phần này sẽ tập trung chính vào kỹ thuật sinh câu hỏi tương ứng với ý định của người dùng để bổ sung thêm dữ liệu vào tập dữ liệu huấn luyện.

Tất cả các câu hỏi của sinh viên trong trường được chia ra thành các class như sau, mỗi class tương ứng với ý định hỏi của người dùng. Như vậy việc xác định ý định chính là việc phân lớp 1 câu hỏi thuộc vào class nào:

Như [hình 3.1] chúng ta có thể thấy số lượng các câu hỏi trong các class là không đều nhau, các câu hỏi thuộc class TOEIC chỉ khoảng 50 câu hỏi, trong khi các câu hỏi trong class DKMH lại là gần 480 câu hỏi. Việc mất cân bằng dữ liệu này sẽ ảnh hưởng nhiều đến chất lượng của mô hình. Có nhiều phương pháp để xử lý mất cân bằng dữ liệu, tuy nhiên các phương pháp chủ yếu tập trung vào việc phân chia tập dữ liệu huấn luyện và kiểm tra chứ không tập trung vào việc bổ sung thêm dữ liệu. Việc bổ sung dữ liệu sẽ giúp cải thiện mô hình một cách tốt hơn.

Kỹ thuật này được gọi là kỹ thuật tăng cường dữ liệu, để áp dụng kỹ thuật tăng cường dữ liệu cho bài toán này tôi sử dụng BERT được trình bày chi tiết dưới đây.

a. BERT

BERT là viết tắt của Bidirectional Encoder Representations from Transformers [22] là một mô hình ngôn ngữ được huấn luyện dựa trên tập dữ liệu văn bản rất lớn, mô hình học được cách biểu diễn vector của các từ theo ngữ cảnh 2 chiều của từ, thường được sử dụng để transfer sang các bài toán khác trong lĩnh vực xử lý ngôn ngữ tự nhiên. BERT đã thành công trong việc cải thiện những tác vụ gần đây trong việc tìm ra biểu diễn của từ trong không gian thông qua ngữ cảnh của nó.

BERT là một kỹ thuật đơn giản nhưng lại mang lại hiệu quả cực lớn trong thực tế. Nó đã thu được kết quả tối ưu mới nhất cho 11 nhiệm vụ xử lý ngôn ngữ tự nhiên, bao gồm việc đẩy kết quả của nhiệm vụ GLUE benchmark lên 80.4%(cải tiến thêm 7.6%) và SQuAD v.1.1 với F1 score trên tập test đạt 93.2%(cải tiến thêm 1.5%), tốt hơn con người 2%.

Có 2 chiến lược để sử dụng các biểu diễn ngôn ngữ được huấn luyện cho các nhiệm vụ về sau, gồm feature-based và fine-tuning.

Chúng ta đào tạo BERT bằng cách sử dụng 2 nhiệm vụ dự đoán không giám sát được gọi là Masked LM và Next Sentence Prediction. Cả 2 sẽ được trình bày ngay trong phần nội dung dưới đây.

b. Các nhiệm vụ của BERT

Masked LM

Một cách trực quan, một mô hình học sâu được huấn luyện dựa trên ngữ cảnh 2 chiều giống với tự nhiên và mạnh mẽ hơn nhiều so với một mô hình chỉ dùng ngữ cảnh từ trái qua phải (hoặc ngược lại).Tuy nhiên, hầu hết các mô hình ngôn ngữ trước đây chỉ có thể huấn luyện từ trái qua phải hoặc từ phải qua trái. Lý do được lý giải là vì khi sử dụng ngữ cảnh 2 chiều sẽ gây ra một nghịch lý là một từ có thể gián tiếp tự nhìn thấy nó trong một ngữ cảnh nhiều lớp.

Để đào tạo một mô hình tìm ra biểu diễn từ dựa vào ngữ cảnh 2 chiều, chúng ta sử dụng một cách tiếp cận đơn giản để che giấu đi một số token đầu vào một cách ngẫu nhiên và sau đó chúng ta chỉ dự đoán các token được giấu đi đó và gọi nhiệm vụ này như là một "masked LM"(MLM). Trong trường hợp này, các hidden vectors

ở lớp cuối cùng tương ứng với các tokens được ẩn đi được đưa vào 1 lớp softmax trên toàn bộ tự vựng để dự đoán. Các nhà nghiên cứu của Google đã thử nghiệm mask 15% tất cả các token lấy từ từ điển của WordPiece trong câu một cách ngẫu nhiên là chỉ dự đoán các từ được mask.

Mặc dù điều này cho phép chúng ta có được một mô hình đào tạo 2 chiều, nhưng có 2 nhược điểm tồn tại:

- Đầu tiên là chúng ta đang tạo ra một sự không phù hợp giữa pre-train và fine-tuning vì các token được [MASK] không bao giờ được nhìn thấy trong quá trình tinh chỉnh mô hình. Để giảm thiểu điều này, chúng ta sẽ không phải lúc nào cũng thay thế các từ được giấu đi bằng token [MASK]. Thay vào đó, trình tạo dữ liệu đào tạo chọn 15% tokens một cách ngẫu nhiên và thực hiện các bước như sau:

  Ví dụ với câu: "con_chó của tôi đẹp quá" Từ được chọn để mask là từ "đẹp".

  - Thay thế 80% từ được chọn trong dữ liệu huấn luyện thành token [MASK] --> "con_chó của tôi [MASK] quá"
  - 10% các từ được chọn sẽ được thay thế bởi 1 từ ngẫu nhiên. --> "con_chó của tôi máy_tính quá"
  - 10% còn lại được giữ không thay đổi --> "con_chó của tôi đẹp quá"

  Transformer encoder không hề biết được từ nào sẽ được yêu cầu dự đoán hoặc từ nào đã được thay thế bằng một từ ngẫu nhiên, do đó, nó buộc phải giữ một biểu diễn theo ngữ cảnh của mối token đầu vào. Ngoài ra, do thay thế 1.5% tất cả các tokens bằng một từ ngẫu nhiên nên điều này dường như sẽ không làm ảnh hưởng tới khả năng hiểu ngôn ngữ của mô hình.
- Nhược điểm thứ 2 của việc sử dụng MLM là chỉ có 15% tokens được dự đoán trong mỗi lô, điều này gợi ý cho ta 1 điều là có thể cần thêm các các bước sử dụng các pre-train model khác để mô hình hội tụ.

Dự đoán câu tiếp theo

Một số nhiệm vụ quan trọng trong xử lý ngôn ngữ tự nhiên như Question Answering yêu cầu sự hiểu biết dựa trên mối quan hệ giữa 2 câu văn bản, không trực tiếp sử dụng được các mô hình ngôn ngữ. Để đào tạo được mô hình hiểu được mối quan hệ giữa các câu, chúng ta cần xây dựng một mô hình dự đoán câu tiếp theo dựa vào câu hiện tại, dữ liệu huẩn luyện có thể là một corpus bất kỳ nào. Cụ thể, khi chọn câu A và câu B cho mỗi training sample, 50% khả năng câu B là câu tiếp theo sau câu A và 50% còn lại là một câu ngẫu nhiên nào đó trong corpus.

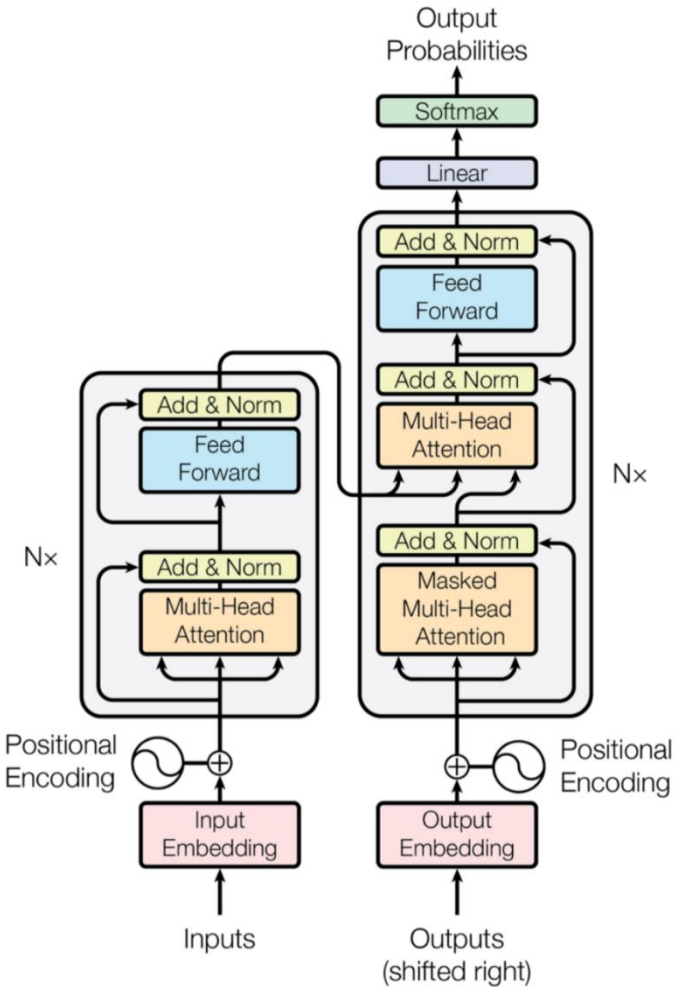
Ví dụ:

- Input: [CLS] người đàn_ông làm [MASK] tại cửa_hàng [SEP] anh_ta rất [MASK] và thân_thiện [SEP]
- Label: isNext
- Input: [CLS] người đàn_ông làm [MASK] tại cửa_hàng [SEP] cô_ta đang cầm súng [SEP]
- Label: notNext

Chúng ta chọn những câu notNext một cách ngẫu nhiên và mô hình cuối cùng đạt được độ chính xác 97%-98% trong nhiệm vụ này.

c. Kiến trúc mô hình

Trong mô hình transformer, đây là một lớp mô hình seq2seq gồm 2 pha là encoder và decoder. Mô hình hoàn toàn không sử dụng các kiến trúc Recurrent Neural Network của RNN mà chỉ sử dụng các layers attention để embedding các từ trong câu. Kiến trúc cụ thể của mô hình như sau:



Hình 2.8: Sơ đồ kiến trúc transformer kết hợp với attention.

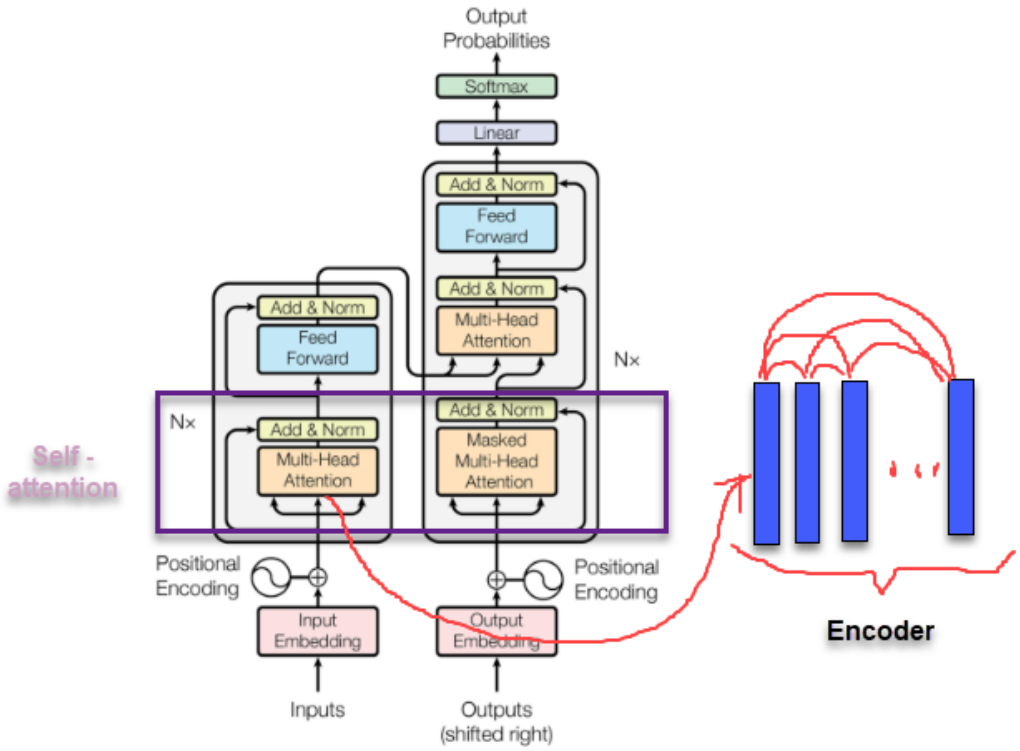
Mô hình sẽ bao gồm 2 phase.

- **Encoder**: Bao gồm 6 layers liên tiếp nhau. Mỗi một layer sẽ bao gồm một sub-layer là Multi-Head Attention kết hợp với fully-connected layer như mô tả ở nhánh encoder bên trái của hình vẽ. Kết thúc quá trình encoder ta thu được một vector embedding output cho mỗi từ.
- **Decoder**: Kiến trúc cũng bao gồm các layers liên tiếp nhau. Mỗi một layer của Decoder cũng có các sub-layers gần tương tự như layer của Encoder nhưng bổ sung thêm sub-layer đầu tiên là Masked Multi-Head Attention có tác dụng loại bỏ các từ trong tương lai khỏi quá trình attention.

  Các tiến trình self-attention và encoder-decoder attention

Trong kiến trúc transformer chúng ta áp dụng 2 dạng attention khác nhau tại từng bước huấn luyện.

self-attention: Được sử dụng trong cùng một câu input, tại encoder hoặc tại decoder. Đây chính là attention được áp dụng tại các Multi-Head Attention ở đầu vào của cả 2 phase encoder và decoder.



Hình 2.9: Sơ đồ vị trí áp dụng self-attention trong kiến trúc transformer.

Các véc tơ embedding của cùng một chuỗi encoder hoặc decoder tự liên kết với nhau để tính toán attention như hình bên phải.

d. phoBERT [22]

phoBERT là mô hình dựa trên BERT nhưng được huấn luyện bằng tập dữ liệu Tiếng Việt rất lớn. Để thực hiện tạo thêm dữ liệu huấn luyện tôi tiến hành hiệu chỉnh lại mô hình BERT, tôi có sử dụng lại các trọng số đã được huấn luyện cho tập dữ liệu Tiếng Việt phoBERT [22]

BERT cung cấp 2 mô hình với kích thước khác nhau để sử dụng: Mô hình $BERT_{base}$ và mô hình $BERT_{large}$. $BERT_{large}$ cho ra kết quả tốt hơn BERTbase (tăng 3% độ chính xác đối với tác vụ SQuAD), nhưng cũng đi kèm với yêu cầu tài nguyên phần cứng lớn. Hiện tại, không thể tái tạo hầu hết các kết quả $BERT_{large}$ theo như bài báo bằng cách sử dụng GPU có RAM 12GB - 16GB, vì kích thước batch size tối đa có thể được đưa vào bộ nhớ quá nhỏ, do đó tôi sẽ sử dụng BERTbase với các tham số sau:

- Số lớp (số khối transfomer) L = 12.
- Số lượng tham số mô hình: 110 triệu.
- Trong mỗi khối transformer:
    - Số lượng self-attention (trong mỗi khối transformer) A = 12.
    - Kích thước của mỗi lớp feed-forward/filter H = 768

PhoBERT được train trên khoảng 20GB dữ liệu bao gồm khoảng 1GB Vietnamese Wikipedia corpus và 19GB còn lại lấy từ Vietnamese news corpus. [22] Đây là một lượng dữ liệu khá tốt để huấn luyện BERT.

e. Các bước thực hiện

Để thực hiện tăng cường dữ liệu cho mô hình phân loại ý định tôi thực hiện như sau [23]:

- ***Bước 1:*** Giả sử ta có mô hình phân lớp như đã được huấn luyện trong phần 3.3. Gọi mô hình giả thiết này là $h = \mathcal{A}(D_{train})$ dựa trên tập dữ liệu có sẵn là D_{train}. Mô hình này sẽ được sử dụng để lọc kết quả trong bước 4.
- ***Bước 2:*** **Fine-tune lại mô hình ngôn ngữ**

    Bước này tiến hành độc lập với bước 1, tôi fine-tune mô hình ngôn ngữ $\mathcal{G}$ với nhiệm vụ tổng hợp các câu được gán nhãn, để thu được mô hình ngôn ngữ tinh chỉnh $\mathcal{G}_{tuned}$. Ở đây, $\mathcal{G}$ được fine-tune cụ thể theo miền ngôn ngữ của D_{train} (gồm các câu, từ vựng, văn phong, v.v.), cũng như các lớp trong D_{train}.

    Mục tiêu của việc fine-tune đó là chúng ta có thể sử dụng $\mathcal{G}_{tuned}$. để tạo một tập hợp câu có kích thước bất kỳ và mỗi câu được gắn nhãn bằng một lớp.

Ở đây tôi sử dụng $\mathcal{G}$ là mô hình BERT, tôi tiến hành fine-tune lại BERT với dữ liệu huấn luyện $D_{train} = \{(x_i, y_i)\}_{i=1}^{n}$. Sau đó tôi tiến hành nối các câu trong D_{train} lại để được U có dạng như sau:

$$U^* = y_1 SEP x_1 EOS y_2 SEP x_2 EOS y_3 SEP x_3 \dots y_n SEP x_n EOS$$

Ở đây tôi sử dụng token `SEP` để phân cách giữa nhãn của câu và câu tương ứng. Token `EOS` dúng để kết thúc một câu và tách nó khỏi nhãn của câu sau.

Để huấn luyện tôi vẫn sử dụng hàm mất mát được dùng trong mô hình BERT:

$$J_\theta = -\sum_j log P_\theta(\omega^j \mid \omega^{j-k}, \dots, \omega^{j-1}) \quad (2.10)$$

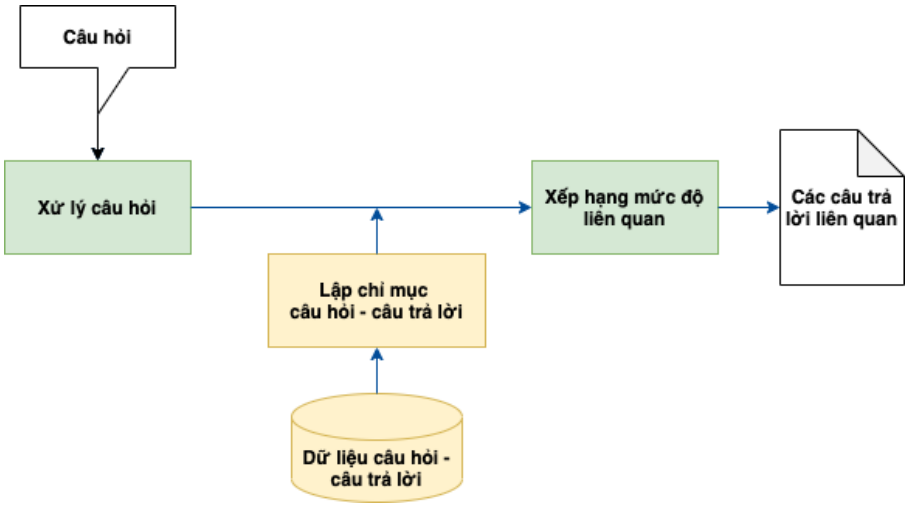
Các lớp mạng cách huấn luyện mô hình được giữ nguyên, tuy nhiên tôi sử dụng tập dữ liệu U^* thay vì U để huấn luyện cho mô hình này. Sau quá trình huấn luyện chúng ta được mô hình $\mathcal{G}_{tuned}$

- ***Bước 3:*** **Sinh ra câu mới.** Từ mô hình $\mathcal{G}_{tuned}$ tôi tiến hành sinh ra các câu mới bằng cách: Với mỗi nhãn thuộc vào trong tập nhãn y ∈ {1, ..., q} (với q là số lượng các nhãn), chúng ta có thể sử dụng mô hình ngôn ngữ $\mathcal{G}_{tuned}$ để dự đoán từ tiếp theo của chuỗi "y SEP" cho đến khi gặp token EOS thì dừng lại và kết thúc câu đã tạo. Thực hiện tương tự theo cách này cho mỗi lớp, ta có thể sinh ra được số lượng câu bất kỳ. Tôi đã tiến hành sinh ra các câu để cân bằng dữ liệu giữa các lớp và tạo ra thêm dữ liệu cho các lớp. Gọi tập dữ liệu được sinh ra là $D^* = \{(x'_i, y'_i)\}_{i=1}^{N}$.
- ***Bước 4:*** **Lọc lại các câu đã được sinh ra.** Từ các dữ liệu được sinh ra ở bước số 3 tôi sẽ tiến hành đưa vào mô hình h trong bước 1 để loại bỏ những câu kém chất lượng. Bằng cách đưa từng câu x'_i vào h, ta có $\hat{y}'_i = h(x'_i)$. Nếu $\hat{y}'_i = y'_i$ nghĩa là nhãn dự đoán của mô hình và nhãn thực tế giống nhau thì ta sẽ bổ sung câu nay vào tập $D_{synthesized}$. Như vậy $D_{synthesized} \subseteq D^*$.

Như vậy sau bước sinh dữ liệu ta có tập dữ liệu $D_{new} = D_{train} \cup D_{synthesized}$.

2.3. Tìm kiếm và truy xuất thông tin.

Khi chúng ta tìm kiếm tài liệu, chúng ta muốn hệ thống IR truy xuất các tài liệu giống với truy vấn của chúng ta. Để làm như vậy, chúng ta cần một mô hình truy xuất thông tin tốt. Một mô hình truy xuất được coi là tốt nếu tài liệu được truy xuất đúng với yêu cầu của truy vấn. Đây là nền tảng của thuật toán xếp hạng mức độ liên quan được sử dụng trong công cụ tìm kiếm. Sau đây tôi sẽ trình bày về thuật toán xếp hạng văn bản phù hợp Okapi BM25 được áp dụng trong luận văn để tìm kiếm câu hỏi trong tập dữ liệu câu hỏi – câu trả lời để xác định được câu trả lời gần đúng với câu hỏi của người dùng.



Hình 2.10: Kiến trúc mô hình truy xuất thông tin

Trong tìm kiếm thông tin, để xếp hạng các văn bản phù hợp với truy vấn của người dùng, người ta thường sử dụng thuật toán Okapi BM25. Thuật toán này dựa trên mô hình xác suất , được phát minh ra vào những năm 1970 – 1980. Phương pháp có tên BM25 (BM – best match), nhưng người ta thường gọi “Okapi BM25”, vì lần đầu tiên công thức được sử dụng trong hệ thống tìm kiếm Okapi, được sáng lập tại trường đại học London những năm 1980 và 1990.

BM25 là một phương pháp xếp hạng được sử dụng rộng rãi trong tìm kiếm. Trong Web search những hàm xếp hạng này thường được sử dụng như một phần của các phương pháp tích hợp để dùng trong machine learning, xếp hạng.

Một trong những kỹ thuật tìm kiếm nỗi tiếng hiện nay đang sử dụng thuật toán này là Elasticsearch. Khi tìm kiếm, Elascticsearch trả về cho người dùng ngoài các kết quả tìm được, còn có đánh giá độ liên quan của kết quả dựa trên giá trị thực dương score. Elastic search sẽ sắp xếp các kết quả trả về của các query theo thứ tự score giảm dần.

2.3.1. Một số khái niệm

Thuật ngữ (term): Dùng để chỉ thành phần của một truy vấn, ví dụ ta có truy vấn: "Thủ đô của Hà Nội là gì", thuật ngữ của truy vấn sẽ là: 'Thủ đô', 'của', 'Hà Nội'. Hiểu đơn giản, thuật ngữ là các từ trong truy vấn/văn bản mang ý nghĩa.

- Tài liệu: Các văn bản thông thường cần tìm kiếm, truy vấn cũng có thể coi là tài liệu.
- Tần suất thuật ngữ hay còn gọi là tf: tần suất thuật ngữ xuất hiện trong tài liệu? 3 lần? 10 lần?
- Tần suất tài liệu nghịch đảo hay còn gọi là idf: được tính bằng số lượng tài liệu mà thuật ngữ xuất hiện. Tần suất tài liệu nghịch đảo (1 / df) cho biết mức độ quan trọng của thuật ngữ. Thuật ngữ có phải là một từ hiếm (chỉ xảy ra trong một tài liệu) hay không? Hay thuật ngữ này phổ biến (xảy ra trong gần như tất cả các tài liệu)?

Sử dụng hai yếu tố này, TF-IDF cho biết độ tương đối của một thuật ngữ trong một tài liệu nào đó. Nếu một thuật ngữ phổ biến trong tài liệu này, nhưng hiếm ở tài liệu khác, thì điểm TF-IDF sẽ cao và tài liệu có điểm TF-IDF cao hơn sẽ được coi là phù hợp với cụm từ tìm kiếm. BM25 cải thiện dựa trên TF-IDF bằng cách sử dụng mức độ liên quan với một bài toán xác suất. BM25 sẽ đưa ra điểm liên quan, để xác định xem một truy vấn có mức độ liên quan thế nào đến các tài liệu. Sau đó xếp hạng các điểm liên quan đó để đưa ra kết quả các tài liệu phù hợp với truy vấn.

2.3.2. Công thức tính BM25

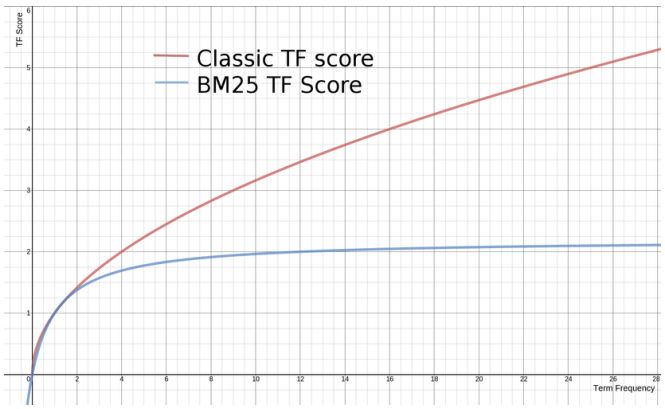
Để xác định mức độ liên quan giữa một truy vấn (tài liệu) với một tài liệu khác, chúng ta có thể sử dụng công thức tính BM25 như sau:

$$\text{BM25}(D,Q) = \sum_{i=1}^{n} IDF(q_i, D) \frac{f(q_i,D)\cdot(k_1+1)}{f(q_i)+k_1\cdot(1-b+b\cdot|D|/d_{avg}))} \qquad (2.11)$$

Trong đó:

- $f(q_i,D)$ Là số lần mà term qi xuất hiện trong tất cả các tài liệu D
- $|D|$ là số từ trong tất cả các tài liệu D
- d_{avg} là số lượng từ trung bình trong mỗi tài liệu
- b và k_1 là các tham số của BM25
- $f(q_i,D)$ cho ta thấy rằng nếu một từ xuất hiện trong tài liệu càng nhiều thì điểm của tài liệu càng cao.

Với tham số k1, xác định tính bão hòa tần suất. Giá trị càng cao, độ bão hòa càng chậm. Nghĩa là nếu một từ xuất hiện nhiều sẽ làm điểm của tài liệu cao, nhưng sẽ nhiều với một mức độ nào đó và mức độ ảnh hưởng tới điểm sẽ giảm dần.



Hình 2.11: Sự ảnh hưởng của TF tới Score

$|D|/d_{avg}$ ở mẫu số có nghĩa là tài liệu dài hơn các tài liệu trung bình sẽ dẫn đến mẫu số lớn hơn, dẫn đến giảm điểm. Thực tế cho ta thấy là nếu càng nhiều thuật ngữ trong tài liệu mà không khớp với truy vấn đầu vào thì điểm của tài liệu càng thấp. Nói cách khác, nếu một tài liệu dài 300 trang đề cập đến cụm từ truy vấn một

lần, thì nó ít có khả năng liên quan đến truy vấn hơn so với một tài liệu ngắn đề cập đến truy vấn một lần.

Đối với phần tần suất tài liệu nghịch đảo, IDF(q_i,D). Với tập ngữ liệu gồm N tài liệu, IDF cho thuật ngữ q_i được tính như sau:

$$\text{IDF}(q_i, D) = \log \frac{N - N(q_i) + 0.5}{N(q_i) + 0.5} \quad (2.12)$$

Với N(q_i) là số lượng các tài liệu trong ngữ liệu chứa q_i.Phần tần suất tài liệu nghịch đảo giống với TF-IDF, có vai trò đảm bảo các từ hiếm hơn sẽ có điểm cao hơn và đóng góp nhiều hơn vào điểm xếp hạng.

Lưu ý rằng công thức IDF ở trên có một nhược điểm khi sử dụng nó cho các cụm từ xuất hiện trong hơn một nửa kho ngữ liệu IDF sẽ là giá trị âm, dẫn đến điểm xếp hạng trở thành số âm. ví dụ. nếu chúng ta có 10 tài liệu trong kho ngữ liệu và thuật ngữ "là" xuất hiện trong 6 tài liệu đó, IDF của nó sẽ là log (10 - 6 + 0.5 / 6 + 0.5) = log (4.5 / 6.5). Mặc dù trong quá trình tiền xử lý chúng ta đã loại bỏ các stop-words(từ dừng) vì các từ này ít mang ý nghĩa trong câu, tuy nhiên ta vẫn cần phải tính đến trường hợp này.

Thêm 1 vào biểu thức:

$$\text{IDF}(q_i) = \log\left(1 + \frac{N - N(q_i) + 0.5}{N(q_i) + 0.5}\right) \quad (2.13)$$

Đối với cụm từ dẫn đến giá trị IDF âm, hãy hoán đổi nó với một giá trị dương nhỏ, thường được ký hiệu là epsilon.

2.3.3. Đánh giá mô hình IR

Trong các hệ thống tìm kiếm và truy xuất thông tin, ngoài phát triển những kỹ thuật tìm kiếm, chúng ta cũng cần các phương pháp để đánh giá hiệu quả của các hệ thống này. Với việc đánh giá các hệ thống một cách trực tiếp dựa trên các chỉ số một cách trực quan, ta có thể đánh giá được chất lượng và hiệu quả của hệ thống truy xuất thông tin, từ đó biết được kĩ thuật tìm kiếm sử dụng có phù hợp với bài toán và dữ liệu không để lựa chọn và điều chỉnh cho phù hợp. Phương pháp đánh

giá hệ thống truy xuất thông tin tập trung vào hai khái niệm chính cho một kết quả là relevant (liên quan) và non-relevant (không liên quan).

Một kết quả được cho là relevant nếu nó giải quyết được lượng thông tin người dùng cần. Ví dụ: lượng thông tin người dùng cần là: "Bệnh viện tốt nhất tại Hà Nội", những kết quả như: "Bệnh viện Bạch Mai", "Bệnh viện Nhiệt đới Trung ương" rõ ràng giải quyết nhu cầu thông tin hơn là những kết quả như "quán ăn tốt nhất tại Hà Nội" mặc dù nó chứa nhiều từ trong câu truy vấn hơn các kết quả trên. Sự khác biệt này đôi khi bị hiểu lầm vì khái niệm "nhu cầu thông tin" trong thực tế không phải luôn luôn được chỉ ra rõ ràng. Do đó, trước khi xây dựng hệ thống tìm kiếm, nhu câu thông tin của người dùng cần được xác định và chỉ ra rõ ràng. Nếu người dùng nhập python vào ô tìm kiếm, họ có thể muốn tìm mua một con trăn, họ cũng có thể muốn học lập trình. Vì lí do đó nên hệ thống đánh giá nói chung và tập test nói riêng cần được xây dựng dựa trên người dùng.

Sau đây tôi sẽ trình bày một số phương pháp để đánh giá một hệ thống truy xuất thông tin.

a. Precision và Recall.

Giả sử hệ thống tìm kiếm trả về một tập kết quả cho một câu truy vấn. Hai độ đo trên được định nghĩa như sau:

- **Precision (P):** là độ đo được tính bằng tỉ lệ số lượng kết quả *relevant* trên tổng số lượng kết quả trả về.

$$Precision = \frac{relevant\ items\ retrieved}{retrieved\ items} = P((relevant|retrieved) \quad (2.14)$$

- **Recall (R):** là độ đo được tính bằng tỉ lệ số lượng kết quả *relevant* trên tổng số lượng kết quả *relevant* trong tập test.

$$Recall = \frac{relevant\ items\ retrieved}{relevant\ items} = P(retrieved|relevant) \quad (2.15)$$

Một cách giải thích trực quan hơn, ta có bảng dưới đây

	RELEVANT	**NONRELEVANT**
RETRIEVED	true positives (tp)	false positives (fp)
NOT RETRIEVED	false negatives (fn)	true negatives (tn)

Khi đó:

$$P = \frac{tp}{tp + fp} \quad (2.16)$$

$$R = \frac{tp}{tp + fn} \quad (2.17)$$

Một độ đo khác mà chúng ta có thể gặp khi đọc các tài liệu về hệ thống tìm kiếm đó là **Accuracy (A)**. Độ đo này được tính bằng tỉ lệ giữa các kết quả được phân loại đúng (bao gồm cả *relevant* và *non-relevant*) trên tổng số kết quả. Cụ thể:

$$A = \frac{tp + tn}{tp + fp + tn + fn} \quad (2.18)$$

Truy nhiên độ đo này không quá phù hợp với hệ thống tìm kiếm thông tin. Lí do là vì trong hầu hết các trường hợp, các dữ liệu *non-relevant* thường lớn hơn rất nhiều so với dữ liệu *relevant* vì thế khi cải thiện độ đo này, sự ảnh hưởng của chúng ta lên sự chính xác thực sự của kết quả trả về là rất thấp. Trong thực tế, độ đo này phù hợp hơn và thường được sự dụng trong các hệ thống học máy.

Điểm mạnh của 2 độ đo **Precision** và **Recall** là ta có thể đánh giá trực quan hệ thống vừa dựa trên kết quả thực tế đã trả về, vừa dựa trên kết quả đáng ra phải được trả về. Một hệ thống lý tưởng sẽ trả về tập kết quả là tất cả các kết quả trong tập test, khí đó giá trị của P=R=1. Có thể nhận thấy rằng **Precision** và **Recall** là 2 giá trị liên quan mật thiết đến nhau. Cụ thể là khi **P** tăng thì **R** sẽ giảm và ngược lại:

- Khi P tăng thì R giảm: Để P đạt giá trị cao - các kết quả trả về có độ chính xác lớn, ta chỉ cần hạn chế số lượng kết quả trả về. Ví dụ đơn giản nếu kết

quả trả về là 1 và kết quả đó nằm trong 50 kết quả của tập test thì giá trị của P=1. Tuy nhiên khi đó giá trị của R sẽ rất nhỏ vì có quá ít kết quả được trả về, cụ thể trong trường hợp trên giá trị của R chỉ là 1/50

- Khi R tăng thì P giảm: Rõ ràng là theo chiều tăng của số lượng các kết quả trả về, các kết quả trong tập test sẽ dễ được trả về hơn, dẫn đến R tăng, tuy nhiên khi có quá nhiều kết quả trả về, độ chính xác P sẽ bị giảm đi đáng kể. Ví dụ trả về 1000 kết quả trong đó bao gồm 10/10 kết quả trong tập test thì R=1 trong khi P chỉ là 10/1000.

b. Precision@K

Trong các hệ thống truy xuất thông tin hiện đại, precision không có nhiều ý nghĩa trong việc đánh giá mô hình, bởi vì nhiều truy vấn đưa ra hàng nghìn tài liệu liên quan và sẽ ít người dùng quan tâm đến việc đọc tất cả các tài liệu. Độ chính xác ở k tài liệu đầu tiên P@k cho biết bao nhiêu phần trăm các tài liệu liên quan nằm trong top k tài liệu nhưng không tính đến các vị trí của các tài liệu liên quan trong số k tài liệu trả về đó. Ví dụ P@10 biểu thị tỉ lệ bao nhiêu tài liệu liên quan nằm trong top 10 tài liệu trả về.

c. Trung bình các precision (Mean average precision - MAP)

Trung bình các precision được tính dựa trên toàn bộ các truy vấn trong tập kiểm thử, MAP sẽ tính bằng cách lấy trung bình các precision của tất cả các truy vấn theo công thức sau:

$$MAP = \frac{\sum_{q=1}^{Q} AveP(q)}{Q} \quad (2.19)$$

Trong đó Q là số lượng các truy vấn trong tập kiểm thử.

d. Discounted cumulative gain - DCG

DCG sử dụng thang đo mức độ phù hợp của các tài liệu từ tập hợp kết quả để đánh giá mức độ hữu ích hoặc của một tài liệu dựa trên vị trí của nó trong danh sách kết quả. Tiền đề của DCG là các tài liệu có liên quan cao xuất hiện ở vị trí thấp hơn

trong danh sách kết quả tìm kiếm sẽ bị phạt vì giá trị mức độ liên quan được phân loại bị giảm theo tỷ lệ logarit với vị trí của kết quả.

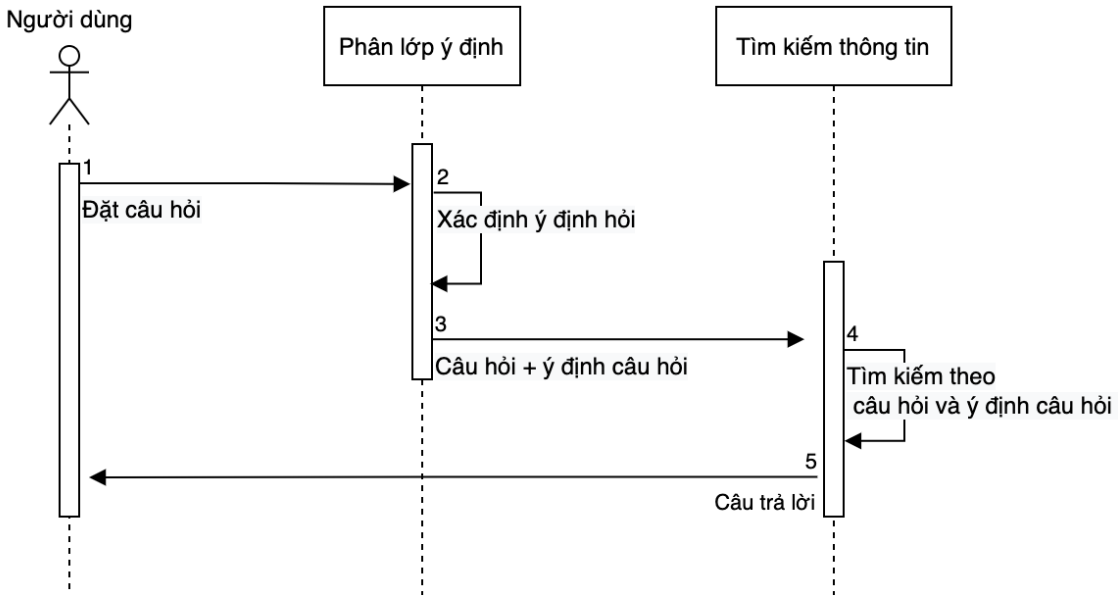
DCG ở vị trí p được định nghĩa là:

$$\mathrm{DCG_p} = rel_1 + \sum_{i=2}^{p} \frac{rel_i}{\log_2(i+1)} \qquad (2.20)$$

Hay được viết gọn lại thành:

$$\mathrm{DCG_p} = \sum_{i=1}^{p} \frac{2^{rel_i} - 1}{\log(i+1)} \qquad (2.21)$$

2.4. Kết hợp xác định ý định và truy xuất thông tin



Hình 2.12: Biểu đồ tuần tự các bước kết hợp xác định ý định và truy xuất thông tin

Hình 2.11 mô tả các bước của mô hình trả lời tự động bằng cách kết hợp xác định ý định câu hỏi và truy xuất thông tin. Các bước trong hình được mô tả như sau:

- *Bước 1:* Người dùng (là sinh viên) đặt câu hỏi cho hệ thống dưới dạng ngôn ngữ tự nhiên.

- *Bước 2 + 3:* Câu hỏi sẽ được đưa vào module phân lớp ý định để tìm ra ý định của câu hỏi. Sau khi xác định được ý định của câu hỏi, hệ thống tiếp tục đưa câu hỏi và ý định của người hỏi sang module truy xuất thông tin ở bước số 4.
- *Bước 4:* Tại bước 4 module truy xuất thông tin tiến hành tìm trong tập dữ liệu câu hỏi – câu trả lời có sẵn thỏa mãn điều kiện: 1 – câu hỏi trong tập dữ liệu phải thuộc loại ý định trong bước 2 và 2 - câu hỏi của người dùng phải gần giống nhất với câu hỏi và câu trả lời trong tập dữ liệu. Bước này sẽ xác định ra cặp câu hỏi – câu trả lời trong tập dữ liệu phù hợp với câu hỏi của người dùng.
- *Bước 5:* Từ cặp câu hỏi – câu trả lời tìm ra ở bước 4, hệ thống sẽ lấy câu trả lời để làm câu trả lời cho câu hỏi ở bước số 1.

2.4.1. Tổ chức dữ liệu để tìm kiếm thông tin theo ý định

Để tổ chức cấu trúc dữ liệu cho mô hình IR phù hợp với việc tìm kiếm theo ý định, mỗi cặp câu hỏi (q) và câu trả lời (t) sẽ dưới dạng d = {q, t, i}, trong đó *i* là ý định của câu hỏi. Mỗi cặp câu hỏi – câu trả lời trước khi đưa vào tập dữ liệu câu hỏi - câu trả lời *D* sẽ được gán nhãn ý định *i* bằng tay.

Ví dụ dữ liệu trong tập dữ liệu sẽ như sau:

Câu hỏi (q)	**Câu trả lời (t)**	**Ý định (i)**
Điểm trung bình tích lũy bao nhiêu thì được nhận đồ án tốt nghiệp ạ?	*Em trả nợ xong tất cả các môn và đạt CĐR ngoại ngữ là đủ điều kiện nhận ĐATN.* *Điểm TBC tích luỹ từ 2.0 trở lên là điều kiện xét TN, không áp dụng khi xét giao ĐATN.*	*"**Tốt nghiệp**"*

Bảng 2.1: Ví dụ dữ liệu lưu trong IR

Sau khi nhận được dữ liệu các câu hỏi của sinh viên và câu trả lời của các phòng ban gửi lại, tôi tiến hành đánh nhãn cho từng câu hỏi với ý định tương ứng của câu hỏi. Ví dự với câu hỏi "*Điểm trung bình tích lũy bao nhiêu thì được nhận đồ án tốt nghiệp ạ?"* như ở bên trên thì ý định câu hỏi sẽ được xác định bằng cách thủ công theo ý hiểu của người đánh nhãn. Câu hỏi sau khi được đánh nhãn và câu trả lời sẽ được đưa vào tập dữ liệu tìm kiếm của module truy xuất thông tin.

2.4.2. Tìm kiếm theo ý định và câu hỏi

Việc kết hợp *module xác định ý định câu hỏi* và *module truy xuất thông tin* giúp cho mô hình tìm kiếm chính xác hơn. Do mô hình truy xuất thông tin IR tìm kiếm bằng cách xếp hạng các văn bản theo mức độ liên quan dựa trên từ khóa nên với 2 câu hỏi có thể có nhiều từ khóa giống nhau nhưng có thể lại không giống nhau về mục đích muốn hỏi. Nếu đưa ra câu trả lời mà không phù hợp mục đích hay ý định của người hỏi sẽ làm hệ thống trả lời sai. Vì vậy muốn hệ thống trả lời đúng mục đích câu hỏi thì cần phải có thêm bước xác định ý định của câu hỏi.

Giả sử người dùng có một câu hỏi a, sau khi đưa a vào mô hình xác định ý định câu hỏi, ta xác định được ý định của câu hỏi là *i'*. Thay vì tìm kiếm trong tập D câu hỏi giống với câu hỏi a thì ta sẽ tìm câu hỏi a trong tập D' với D' $\subset$ D, D' là tập câu hỏi - câu trả lời chỉ chứa các ý định *i'*.

Như vậy với câu hỏi a, tập dữ liệu D gồm các câu hỏi - câu trả lời có sẵn đã được xây dựng từ trước. Để tìm ra câu hỏi $d \in D$ sao cho d giống với a nhất với i là ý định của câu hỏi:

$$P(d \mid D, a) = \sum_{i \in I} P(i \mid a)\, P(d \mid D, a, i) \qquad (2.22)$$

Với $P(i \mid a)$ là xác suất câu hỏi được phân loại là i được tính thông qua mô hình phân loại ý định. Vì mỗi câu hỏi chỉ mang 1 ý định nên $P(i \mid a) = 1$ nếu câu hỏi mang ý định i và $\mathrm{P(i \mid a)} = 0$ nếu câu hỏi không mang ý định i. $P(d \mid D,\ a,\ i)$ được tính thông qua mô hình truy xuất thông tin IR.

Ví dụ ta tập dữ liệu câu hỏi – câu trả lời có sẵn trong module truy xuất thông tin IR như sau:

STT	Câu hỏi	Câu trả lời	Ý định hỏi
1	*Điểm trung bình tích lũy bao nhiêu thì được nhận đồ án tốt nghiệp ạ?*	*Em trả nợ xong tất cả các môn và đạt CĐR ngoại ngữ là đủ điều kiện nhận ĐATN. Điểm TBC tích luỹ từ 2.0 trở lên là điều kiện xét TN, không áp dụng khi xét giao ĐATN.*	Tốt nghiệp
2	*Tôi tên Nguyễn Văn Nam nhưng trên văn bằng cấp cho tôi tôi lại tên là Nguyễn Văn Nám, mà quy định thì chỉ được lấy văn bằng 1 lần thôi vậy trường hợp của tôi giải quyết thế nào?*	*Bản chính văn bằng, chứng chỉ được cấp một lần trừ trường hợp văn bằng, chứng chỉ đã cấp cho người học nhưng phát hiện bị viết sai do lỗi của đơn vị cấp văn bằng, chứng chỉ thì đơn vị đã cấp có trách nhiệm cấp lại cho người học.*	Thủ tục sinh viên
3	*Em đã đăng ký thực tập doanh nghiệp trong học kỳ này, vậy khi nào em phải báo cáo lại kết quả thực tập?*	*Sinh viên đăng ký thực tập doanh nghiệp trong học kỳ nào sẽ phải báo cáo kết quả thực tập vào cuối học kỳ đó theo kế hoạch của khoa/bộ môn quản lý ngành đào tạo.*	Tốt nghiệp
4	*Điểm cao nhất trong các lần học sẽ được chọn để tính vào đâu?*	*tính vào ĐTBTLTK, ĐTBCTL*	Điểm
5	*Bằng tốt nghiệp của tôi ghi sai ngày tháng năm sinh của tôi, nhưng hiện tại tôi đang cần gấp, vậy tôi*	*Bằng chỉ được cấp cho sinh viên khi đã ghi đầy đủ, chính xác các nội dung trên bằng*	Tốt nghiệp

	có thể lấy bằng luôn mà không cần sửa đổi được không?		
6	*Bằng tốt nghiệp phải đáp ứng tiêu chí nào thì mới được cấp cho sinh viên?*	*Bằng chỉ được cấp cho sinh viên khi đã ghi đầy đủ, chính xác các nội dung trên bằng.*	Tốt nghiệp
7	*Điểm trung bình tích lũy bao nhiêu thì được nhận học bổng?*	*Điểm trung bình tích lũy để được xét học bổng phải tối thiểu từ 2.5 hệ 4.*	Học bổng

Khi có một sinh viên hỏi câu hỏi *q* như sau: "Điểm trung bình tích lũy của em được 2.0 thì có được nhận đồ án tốt nghiệp không ạ?". Đầu tiên câu hỏi này sẽ được đưa vào module xác định ý định, ý định câu hỏi được module phân loại xác định là hỏi về "Tốt nghiệp". Với ý định này của câu hỏi, tập dữ liệu để tìm kiếm chỉ còn tập câu hỏi – câu trả lời D' = { 1, 3, 5, 6 } do các câu hỏi – câu trả lời {2, 4, 7} bị loại ra vì không thuộc ý định hỏi về "Tốt nghiệp". Sau đó hệ thống tiếp tục tìm trong tập D' xem câu hỏi *q* giống với câu hỏi – câu trả lời nào nhất bằng cách xếp hạng mức độ liên quan của câu hỏi *q* với lần lượt câu hỏi – câu trả lời trong D' = {1, 3, 5, 7}. Giả sử câu hỏi *q* liên quan nhất đến cặp câu hỏi – câu trả lời 1, hệ thống sẽ lấy ra câu trả lời của cặp câu hỏi – câu trả lời 1 để làm câu trả lời cho câu hỏi *q*. Câu trả lời sẽ là: *"Em trả nợ xong tất cả các môn và đạt CĐR ngoại ngữ là đủ điều kiện nhận ĐATN. Điểm TBC tích luỹ từ 2.0 trở lên là điều kiện xét TN, không áp dụng khi xét giao ĐATN."*

Như vậy trong chương 2 luận văn đã trình bày về kiến trúc chung của mô hình hỏi đáp sau đó đi sâu vào trình bày các kỹ thuật đã sử dụng để xây dựng nên mô hình bao gồm kỹ thuật xác định ý định của câu hỏi, kỹ thuật tìm kiếm và truy xuất thông tin IR. Trong chương 3 luận văn sẽ tiếp tục trình bày về các thực nghiệm và kết quả dựa trên các kỹ thuật đã thực hiện trong chương 2.

CHƯƠNG 3. THỰC NGHIỆM VÀ KẾT QUẢ

Chương này tập trung vào trình bày chi tiết các thực nghiệm và kết quả thực nghiệm trong quá trình xây dựng mô hình. Nội dung cụ thể sẽ được trình bày trong các phần dưới đây.

3.1. Các bước cài đặt

Trong phần này sẽ mô tả quy trình từng bước để xây dựng hệ hỏi đáp, bao gồm các thiết lập và môi trường:

- Xây dựng module *truy xuất và tìm kiếm thông tin*
- Xây dựng module *phân loại ý định*
- *Fine-tune mô hình BERT* với pretrained của phoBERT để cải thiện độ chính xác của module phân loại ý định
- *Xây dựng dataset* để huấn luyện và kiểm thử
- *Tiến hành kiểm thử*

Tất cả mã nguồn để xây dựng hệ thống này được viết chính bằng Python, chạy trên phiên bản môi trường Python3.x

3.1.1. Dữ liệu huấn luyện

Dữ liệu huấn luyện cho bài toán được thu thập và gán nhãn thủ công dựa trên kênh hỗ trợ sinh viên của trường Đại học Xây dựng. Tập dữ liệu có tổng cộng 3.500 cặp câu hỏi – câu trả lời.

3.2. Cài đặt module truy xuất thông tin

Module truy xuất thông tin sẽ nhận truy vấn dưới dạng các câu hỏi, sau đó sẽ đo độ tương tự giữa câu hỏi đầu vào và câu hỏi – câu trả lời trong cơ sở dữ liệu và tiến hành xếp hạng để đưa ra câu trả lời gần đúng nhất với câu hỏi.

Trong nội dung luận văn này, tôi sử dụng thuật toán OKAPI BM25 như đã trình bày ở chương trước để cài đặt cho module truy xuất thông tin. Để cài đặt tôi sử dụng code python thuần, không sử dụng các thư viện nâng cao khác. Ngoài việc cài

đặt thuật toán bằng tay chúng ta có thể sử dụng phần mềm có sẵn là Elastic search. Trong quá trình thử nghiệm và đánh giá, tôi có so sánh chất lượng thuật toán khi cài đặt bằng python và chất lượng khi sử dụng Elastic Search, 2 phương pháp cho kết quả giống nhau.

Kiến trúc module truy xuất thông tin được mô tả như hình [ ]. Trong kiến trúc này sẽ gồm 3 module nhỏ: Module *tiền xử lý văn bản, xếp hạng văn bản* và *đánh chỉ mục tài liệu.*

3.2.1. Tiền xử lý văn bản

Để việc truy xuất thông tin hiệu quả dựa trên Tiếng Việt, các tài liệu dưới dạng câu hỏi - câu trả lời có sẵn khi được đưa vào trong module truy xuất thông tin cần phải được tiền xử lý. Trong bước tiền xử lý sẽ thực hiện loại bỏ các ký tự đặc biệt, tách từ tiếng việt, loại bỏ từ dừng.

- *Loại bỏ các ký tự đặc biệt*: Tiến hành xóa các ký tự không đóng góp vào ý nghĩa của câu.
- *Đưa về các ký tự thường*: Đưa các chữ viết hoa về chữ thường.
- *Tách từ Tiếng Việt*: Tiếng Việt khác với tiếng Anh và các ngôn ngữ khác đó là một từ được ghép từ một hoặc nhiều tiếng (tiếng là đơn vị cấu tạo nên từ), chúng ta cần hợp nhất các tiếng để tạo thành từ. Trong bước này tôi sử dụng thư viện tách từ *VnCoreNLP*, thư viện này được huấn luyện trên tập dữ liệu gồm 75 nghìn câu văn bản được tách từ thủ công, tập này được chia sẻ trong cuộc thi VLSP 2013.

  VnCoreNLP hiện có kết quả tốt nhất trong số các mô hình tương tự như vnTokenizer (pyvi), JvnSegmenter… với độ chính xác F1-Scorce = 97.9%

Model	F1 (%)	Speed (words/second)
VnCoreNLP (i.e.	97.90	62k / _

RDRsegmenter)		
UETsegmenter	97.87	48k / 33k*
vnTokenizer	97.33	_ / 5k*
JVnSegmenter-Maxent	97.00	_ / 1k*
JVnSegmenter-CRFs	97.06	_ / 1k*
DongDu	96.90	_ / 17k*

Bảng 3.1: So sánh VnCoreNLP và một số tokenizer

- *Loại bỏ từ dừng:* Việc loại bỏ từ dừng được dựa theo từ điển chứa các từ dừng.

3.2.2. Đánh chỉ mục tài liệu

Bước đầu tiên trong việc đánh chỉ mục tài liệu đó là định nghĩ cấu trúc của tài liệu trong hệ hỏi đáp trường Đại học Xây dựng. Một tài liệu sẽ gồm các trường:

- *Nội dung câu hỏi*
- *Câu trả lời*, trường này sẽ lưu nội dung câu trả lời tương ứng với câu hỏi
- *Ý định của câu hỏi*, trường này sẽ nhằm xác định ý định của câu hỏi để giúp cho việc truy xuất thông tin chính xác hơn. Về việc xác định ý định câu hỏi tôi sẽ trình bày ở phần sau.

  Ví dụ về các trường của một tài liệu:

```
{
"cau_hoi": [
      "Em muốn xin thi lại thì phải như thế nào ạ",
      "Nhà trường có tổ chức thi lại hay không ạ?",
      "Khi trượt môn thì có được thi lại hay không vậy các thầy cô?",
      "E muốn đăng kí thi lại một môn thì cần những điều kiện gì vậy ạ"
],
```

```
    "cau_tra_loi": "Trường ĐHXD không có hình thức thi lại, em
trượt môn thì phải học lại để trả nợ môn"
    }
```

Tài liệu sẽ được đánh chỉ mục để đưa vào thuật toán xếp hạng Okapi BM25. Cụ thể tài liệu sẽ được đánh chỉ mục theo phương pháp tf-idf vector được mô tả trong phần [2.3.2]

3.2.3. Xếp hạng văn bản

Để xếp hạng văn bản, tôi sử dụng thuật toán Okapi BM25 đã được trình bày trong phần [2.3.2].

3.2.4. Kết quả thực nghiệm

Để thực nghiệm mô hình IR tôi tiến hành chia tập dữ liệu như sau:

Về bộ câu hỏi test thì tôi chọn ngẫu nhiên 50 câu hỏi trong các cặp câu hỏi - câu trả lời được lưu trong elasticsearch.

- Đầu tiên cần lưu hết tất cả 400 cặp câu hỏi - câu trả lời trong tập dữ liệu vào trong elasticsearch.
- Sau đó trong 400 cặp câu hỏi - câu trả lời đó tôi lấy ra 50 câu hỏi để test. Trong quá trình test, khi truy vấn với nội dung của câu hỏi tôi thêm điều kiện để loại bỏ id của câu hỏi vừa lấy ra để đảm bảo câu hỏi đó không có trong kết quả trả về.
- Ví dụ với cặp câu hỏi-câu trả lời như sau:

```
        {
            "cau_hoi": [
                "Em muốn xin thi lại thì phải như thế nào ạ",
                "Nhà trường có tổ chức thi lại hay không ạ?",
                "Khi trượt môn thì có được thi lại hay không vậy các thầy cô?",
                "E muốn đăng kí thi lại một môn thì cần những điều kiện gì vậy
ạ"
            ],
```

"cau_tra_loi": "Trường ĐHXD không có hình thức thi lại, em trượt môn thì phải học lại để trả nợ môn"

}

Tôi sẽ lưu thành 4 document trong elasticsearch. Sau đó tôi sẽ lấy ra các câu hỏi để test:

Ví dụ câu. "Em muốn xin thi lại thì phải như thế nào ạ" relevant đến 3 câu hỏi còn lại:

- "Nhà trường có tổ chức thi lại hay không ạ?",
- "Khi trượt môn thì có được thi lại hay không vậy các thầy cô?",
- "E muốn đăng kí thi lại một môn thì cần những điều kiện gì vậy ạ? Nếu xét điểm quá trình..."

Để lấy ra được câu trả lời đúng nhất với câu hỏi đưa vào, module xếp hạng văn bản sẽ tìm cách so sánh câu hỏi của người dùng với câu hỏi và câu trả lời trong tập dữ liệu xây dựng từ trước. Lý do cho việc lựa chọn phương pháp này trên các thử nghiệm: so sánh *câu hỏi với câu hỏi*, *câu hỏi với câu trả lời* và so sánh *câu hỏi với cả câu hỏi và câu trả lời*. Trong thử nghiệm này tôi tiến hành đánh giá với các thuật toán tách từ (tokenizer) khác nhau với các tham số k-top và các phương pháp đo khác nhau. Kết quả được thể hiện theo bảng bên dưới.

Cách 1: ***Tìm câu hỏi theo câu hỏi:***

Tìm câu hỏi theo câu trả lời						
Ngày tháng	**Nội dung**			**Kết quả**		
	INDEXING	*TOKENIZER*	*K*	Recal@K	NDCG@K	MAP@K
12/01/2021	TF-IDF	DEFAULT	1	0,180	0,180	0,180
12/01/2021			3	0,340	0,270	0,247
04/01/2021			10	0,480	0,321	0,271
12/01/2021	TF-IDF	ViTokenizer	1	0,240	0,240	0,240
12/01/2021			3	0,380	0,320	0,300
08/01/2021			10	0,480	0,358	0,319
12/01/2021	BM25	DEFAULT	1	0,140	0,140	0,140
12/01/2021			3	0,320	0,240	0,213
04/01/2021			10	0,400	0,268	0,225
12/01/2021	BM25	ViTokenizer	1	0,200	0,200	0,200
12/01/2021			3	0,300	0,255	0,240
08/01/2021			10	0,400	0,293	0,259

Bảng 3.1: Kết quả tìm kiếm câu hỏi theo câu hỏi

Trong kết quả đo này, giá trị các phép đo khi thực hiện tìm câu hỏi theo câu trả lời cho kết quả khá thấp, với chỉ số MAP@K với k = {1, 3} chỉ đạt trong khoảng [0.180, 0.271].

Sau đó tôi tiến hành thử nghiệm với việc thực hiện tìm câu hỏi theo câu hỏi, kết quả thử nghiệm cũng được cho như bảng dưới đây:

Cách 2: ***Tìm câu hỏi theo câu trả lời***

Việc tìm câu hỏi theo câu hỏi cho kết quả tốt hơn, với phương pháp đo MAP@K, K = {1, 3, 10} kết quả của việc tìm kiếm nằm trong khoảng [0.45; 0.53].

Tìm câu hỏi theo câu hỏi						
Ngày tháng	**Nội dung**			**Kết quả**		
	INDEXING	*TOKENIZER*	*K*	Recal@K	NDCG@K	MAP@K
22/01/2021	TF-IDF	DEFAULT	1	0,206	0,457	0,457
22/01/2021			3	0,387	0,409	0,512
22/01/2021			10	0,575	0,488	0,495
22/01/2021	TF-IDF	ViTokenizer	1	0,201	0,449	0,449
22/01/2021			3	0,410	0,422	0,530
22/01/2021			10	0,598	0,497	0,500
25/01/2021	BM25	DEFAULT	1	0,206	0,457	0,457
25/01/2021			3	0,387	0,409	0,512
25/01/2021			10	0,575	0,488	0,495
25/01/2021	BM25	ViTokenizer	1	0,204	0,457	0,457
25/01/2021			3	0,410	0,424	0,533
25/01/2021			10	0,598	0,499	0,504

Bảng 3.2: Kết quả tìm kiếm câu hỏi theo câu trả lời

Sau khi thử nghiệm tìm kiếm bằng cách so sánh câu hỏi với câu hỏi, câu hỏi với câu trả lời tôi tiến hành kết hợp 2 phương pháp này lại và tiến hành đo bằng cách tìm kiếm câu hỏi với cả câu hỏi và câu trả lời. Kết quả được cho như bảng dưới đây:

Cách 3: ***Tìm câu hỏi theo câu hỏi và câu trả lời***

Tìm câu hỏi theo câu hỏi và câu trả lời						
Ngày tháng	**Nội dung**			**Kết quả**		
	INDEXING	*TOKENIZER*	*K*	Recal@K	NDCG@K	MAP@K
28/01/2021	TF-IDF	DEFAULT	1	0,207	0,457	0,457
28/01/2021			3	0,404	0,424	0,510
28/01/2021			10	0,620	0,514	0,509
28/01/2021	TF-IDF	ViTokenizer	1	0,201	0,449	0,449
28/01/2021			3	0,435	0,440	0,518
28/01/2021			10	0,613	0,512	0,524
01/02/2021	BM25	DEFAULT	1	0,207	0,457	0,457
01/02/2021			3	0,404	0,424	0,512
01/02/2021			10	0,620	0,514	0,509
01/02/2021	BM25	ViTokenizer	1	0,201	0,449	0,449
01/02/2021			3	0,435	0,440	0,519
01/02/2021			10	0,613	0,512	0,526

Bảng 3.3: Kết quả áp dụng IR tìm câu hỏi theo câu hỏi và câu trả lời

Kết quả nhìn chung không thay đổi mấy so với cách số 2. Như vậy thông qua cả 3 thử nghiệm với các tham số khác nhau tôi nhận thấy việc thực hiện truy xuất câu trả lời bằng cách so sánh câu hỏi với câu hỏi và so sánh câu hỏi với câu trả lời cho kết quả khá thấp, vì vậy nếu xây dựng hệ hỏi đáp chỉ dựa trên câu hỏi và câu trả lời là *không khả thi*.

Sau khi thực hiện tìm kiếm và nhận thấy kết quả khá thấp, tôi tiến hành phân tích lại tập dữ liệu hỏi đáp. Tôi tiến hành đưa ra một phương pháp thự hiện sử dụng việc phân tích ý định của câu hỏi. Mỗi câu hỏi sẽ thực hiện intent detection hoặc intent classification. Intent là ý định của người hỏi, Ví dụ trong câu "E thưa cô, chả hạn e trả hết môn mà tích luỹ chưa đủ 2.0 thì e có dc nhận để làm đồ án tốt nghiệp không ạ" thì ý định người dùng là hỏi về điều kiện làm đồ án tốt nghiệp. Trong khi câu "vì điều kiện dịch bệnh nên e chưa thể lên trường đóng học phí được e là sinh viên năm cuối còn đồ án tốt nghiệp nữa kính mong thầy cô mở tài khoản cho e đăng kí nốt đồ án tốt nghiệp" thì intent là đăng ký đồ án. Nếu chỉ dùng từ khoá như phương pháp được đề cập trước đó (BM25, TF-IDF) thì nội dung hai câu này khá giống nhau.

3.3. Cài đặt mô hình phân lớp ý định

Phân lớp ý định (intent detection) thực hiện bằng phương pháp text classification, tức là với mỗi câu hỏi thì người ta tiến hành phân loại vào một trong số các loại intent định nghĩa trước.

3.3.1. Xây dựng mô hình phân lớp ý định

Để thực hiện được bằng phương pháp này tôi tiến hành như sau:

a. Đánh nhãn dữ liệu

Để xây dựng mô hình xác định ý định câu hỏi, tôi sẽ sử dụng ontology là các cặp "câu hỏi - ý định" được thu thập từ sinh viên trường Đại học Xây dựng. Tôi sẽ đưa bài toán về việc xây dựng một mô hình phân lớp với các class là các ý định của người hỏi. Ví dụ sau đây là các câu hỏi trong tập dữ liệu:

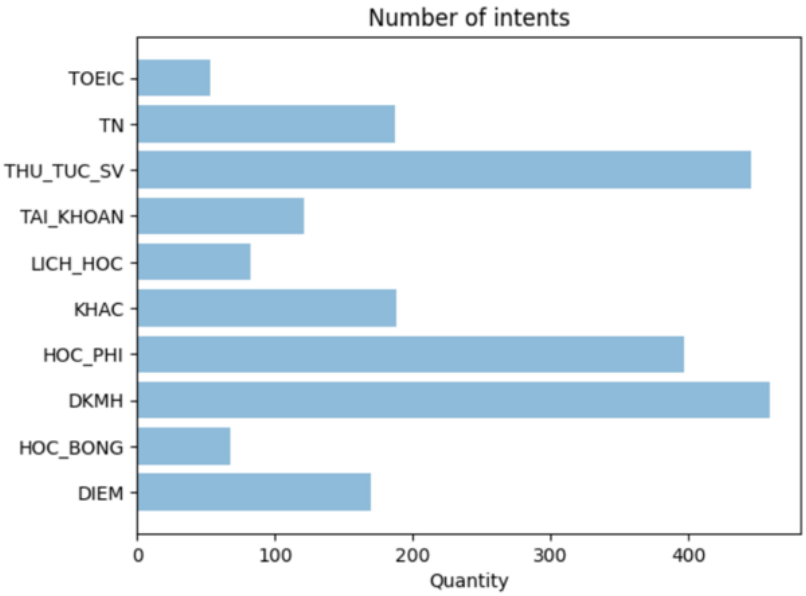

```
[
  {
    "content": " hiện tại em bị quên mất khẩu gmail trường 18145015 student hcmute edu vnmong phòng đào tạo giúp em reset mật khẩu",
    "intent": "TAI_KHOAN"
  },
  {
    "content": "Tổng số tín chỉ của các chương trình đào tạo tối thiểu là bao nhiêu và tối đa là bao nhiêu?",
    "intent": "KHAC",
    "answer": "tối thiểu là 120 và tối đa là 140"
  },
  {
    "content": "Thầy cô cho e hỏi, e đang được gọi đi nghĩa vụ quân sự ở quê, giờ e muốn xin giấy xác nhận vẫn đang là sinh viên của trường thì xin ở đâu ạ ?",
    "intent": "THU_TUC_SV"
  }
]
```


Toàn bộ các nhãn intent được đánh một cách thủ công. Các câu hỏi được chia thành 10 nhóm ý định: ['DIEM', 'HOC_BONG', 'DKMH', 'HOC_PHI', 'KHAC', 'LICH_HOC', 'TAI_KHOAN', 'THU_TUC_SV', 'TN', 'TOEIC'] Trong đó:

- 'DIEM' bao gồm các câu hỏi thắc mắc về Điểm
- 'HOC_BONG' bao gồm các câu hỏi thắc mắc về Học bổng
- 'DKMH' bao gồm các câu hỏi thắc mắc về việc đăng ký môn học
- 'HOC_PHI' bao gồm các câu hỏi thắc mắc về học phí

- …

- ‘KHAC’ bao gồm các câu hỏi không thuộc vào 1 trong 9 nhóm trên



Hình 3.1: Số lượng câu hỏi trong các intent

b. Phân lớp ý định bằng mô hình SVM:

Để thực hiện phân lớp ý định câu hỏi tôi tiến hành thử nghiệm với mô hình SVM. Câu hỏi sẽ được biểu diễn bằng cách cộng các vector biểu diễn từ (R^n) lại thành 1 vector duy nhất (R^n). Với cách biểu diễn này tôi tiến hành thử nghiệm với 2 phương pháp biểu diễn từ: *word2vec* và *one-hot encoding.*

Sau khi biểu diễn câu hỏi thành vector, tôi đưa vào *mô hình SVM* để tiến hành huấn luyện cho mô hình phân lớp. Kết quả sau đây được thực hiện với 2 phương pháp trên:

	One-hot encoding	*word2vec*
Precision	0.56	0.38
Recall	0.56	0.38
F1-score	0.56	0.38

Bảng 3.4: Kết quả bài toán phân lớp ý định bằng mô hình SVM

Nhận xét: Việc xác định ý định bằng thuật toán SVM trong bài toán này cho kết quả tương đối thấp.

c. Phân lớp ý định bằng deep learning

Vì kết quả của SVM khá thấp nên tôi tiến hành sử dụng mô hình LSTM như đã đề cập trong phần [2.4] để tiến hành phân lớp.

Trong phần này tôi thử nghiệm trên một số mô hình mạng nơ ron khác nhau để tiến hành so sánh. Các tham số áp dụng cho mô hình phân lớp như sau:

- Đối với mô hình Word2Vec Tiếng việt tôi sử dụng bộ tham số được huấn luyện dựa trên bộ ngữ liệu Tiếng Việt được tổng hợp từ Wikipedia với kích thước vector là 150, window size = 10, và kích thước tập từ vựng là 10.000. min_count = 2 để loại bỏ biểu diễn các từ có số lần xuất hiện < 2.
- Sau khi sử dụng pretrain word2vec, tôi tiến hành huấn luyện dựa trên tập dữ liệu hỏi đáp mà tôi đã xây dựng trước đó nhằm mục đích giúp mô hình học thêm cách biểu diễn những từ không có trong tập từ vựng.
- Đối với mô hình LSTM tôi tiến hành training từ đầu, còn với BERT tôi sử dụng pretrained phoBERT, sau đó thay đổi layer cuối cùng (activation layer) để đưa BERT về bài toán phân lớp với k lớp.

Kết quả sau quá trình huấn luyện được trình bày trong bảng dưới đây:

Model	Word Embedding	F1-score
LSTM	Word2Vec	0.906
LSTM	Fastext	0.912
CNN + LSTM	Word2Vec	0.866
CNN + LSTM	Fastext	0.879
BiGRU	Fastext	0.903
baseBERT	Fastext	0.933

Bảng 3.5: Kết quả huấn luyện mô hình phân loại ý định

3.3.2. Tăng cường dữ liệu cho bài toán phân lớp ý định

Sau khi tăng cường dữ liệu như đã đề cập trong phần [2.2.5] ta có tập dữ liệu

$$D_{new} = D_{train} \cup D_{synthesized}.$$

3.3.3. Kết quả huấn luyện sau khi tăng cường dữ liệu

Sử dụng tập dữ liệu D_{new} để huấn luyện mô hình trong mục tăng cường dữ liệu kết quả dựa trên tập kiểm tra như sau:

Model	Word Embedding	F1-score
LSTM	Word2Vec	0.917
LSTM	Fastext	0.923
CNN + LSTM	Word2Vec	0.871
CNN + LSTM	Fastext	0.882
BiGRU	Fastext	0.913
baseBERT	Fastext	**0.953**

Bảng 3.6: Kết quả huấn luyện mô hình phân lớp ý định sau khi fine-tune

Như vậy với việc áp dụng kỹ thuật tăng cường dữ liệu cho mô hình phân lớp ý định, độ chính xác của mô hình đã tăng đáng kể.

3.4. Kết quả thực hiện sau khi kết hợp IR và phân lớp ý định

Sau khi áp dụng các kỹ thuật để cải tiến mô hình phân lớp ý định, tôi tiến hành đưa mô hình phân lớp ý định để lọc các câu trả lời không liên quan đến ý định của câu hỏi. Kết quả được trình bày trong bảng dưới đây:

Indexing	K	NDCG@K		MAP@K	
		Kết hợp ý định	**Không** kết hợp ý định	Kết hợp ý định	**Không** kết hợp ý định
TF-IDF	1	0.821	0.449	0.821	0.449
	3	0.837	0.440	0.833	0.518
BM25	1	**0.841**	**0.449**	**0.841**	**0.449**
	3	**0.852**	**0.440**	**0.866**	**0.519**

Bảng 3.7: Kết quả bài toán sau khi kết hợp IR và phân lớp ý định

Như vậy mô hình đã cho kết quả khá tốt, với tham số indexing BM25 và sử dụng ViTokenizer cho kết quả MAP@K [3.2.4] đạt 0.866.

3.5. So sánh với các hệ thống hỏi đáp tương tự

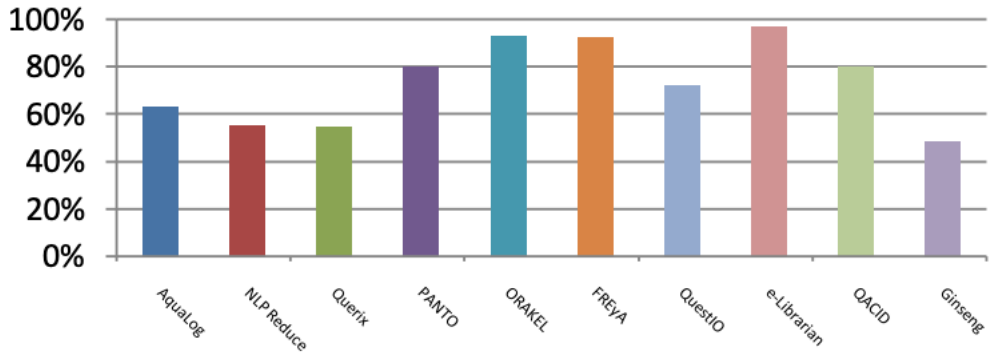
Phân loại theo lĩnh vực mà hệ thống hỏi đáp phục vụ có thể chia hệ thống hỏi đáp thành hai loại như sau:

(1) Hệ thống hỏi đáp lĩnh vực hẹp: Hệ thống liên quan đến các câu hỏi trong một lĩnh vực cụ thể như y học, giáo dục…

(2) Hệ thống hỏi đáp trong lĩnh vực rộng: Hệ thống này liên quan đến các câu hỏi gần như là về tất cả mọi thứ.

Luận văn này mục tiêu xây dựng hệ thống hỏi đáp trong lĩnh vực hẹp và cụ thể là trong việc hỗ trợ sinh viên trả lời các thắc mắc và câu hỏi trong phạm vi trường Đại học Xây dựng. Hệ thống hỏi đáp này sử dụng các câu trả lời có sẵn được chuẩn bị trước trong tập dữ liệu để trả lời các câu hỏi tương tự về ý định hỏi và nội dung câu hỏi, có thể coi đây là hệ thống hỏi đáp dựa trên tri thức.

Trong bài báo khảo sát [24], tôi tiến hành tóm tắt kết quả của các hệ thống hỏi đáp tương tự và thể hiện bằng biểu đồ sau đây, trục tung thể hiện tỉ lệ các câu trả lời là chính xác, trục hoành thể hiện các mô hình hỏi đáp:



Hình 3.2: Kết quả các hệ thống hỏi đáp dựa trên tri thức

Có thể thấy tỉ lệ các câu trả lời chính xác trên tổng số câu hỏi của các hệ hỏi đáp này nằm trong khảng 49% và 89%. Kết quả này phụ thuộc vào hai điều kiện: (1) Thuật toán và phương pháp xử lý ngôn ngữ tự nhiên được dùng trong hệ thống và (2) là miền câu hỏi của bài toán sẽ phục vụ.

KẾT LUẬN VÀ KIẾN NGHỊ

Kết quả đã đạt được

Về mặt lý thuyết: Luận văn đã tổng hợp được các kiến thức về xây dựng hệ thống hỏi đáp tự động sử dụng các kỹ thuật học máy.

Về thực nghiệm: Tác giả đã hoàn thành việc cài đặt và thử nghiệm hệ thống hỏi đáp hỗ trợ sinh viên Trường Đại học Xây dựng. Áp dụng hệ thống hỏi đáp tự động giải quyết nhu cầu hỏi đáp, hỗ trợ sinh viên trường Đại học Xây dựng mang lại hiệu quả cao, giúp sinh viên dễ dàng tiếp cận thông tin từ phía nhà trường đồng thời làm giảm khối lượng công việc tiếp nhận và giải quyết thắc mắc, nhu cầu thông tin từ phía sinh viên cho các phòng ban trong trường. Các tiếp cận xây dựng hệ thống trả lời tự động dựa trên truy xuất thông tin có thể sử dụng được dữ liệu các câu hỏi – câu trả lời được cung cấp bởi các phòng ban trong Trường phục vụ cho việc trả lời tự động. Để câu trả lời tự động dựa trên truy xuất thông tin được chính xác, việc xác định ý định câu hỏi và sử dụng ý định câu hỏi để loại bỏ các câu trả lời sai không phù hợp với câu hỏi mang lại kết quả tốt.

Ngoài những kết quả đạt được mang tính ứng dụng, luận văn đưa ra những kết quả của việc áp dụng các kỹ thuật xử lý ngôn ngữ tự nhiên cho Tiếng Việt. Các kết quả có thể làm tài liệu tham khảo cho những người quan tâm đến việc nghiên cứu về các hệ thống hỏi đáp tự động.

Những điểm còn hạn chế

Với việc tiếp cận dựa trên truy xuất thông tin vẫn còn nhiều hạn chế, hệ thống trả lời tự động hoạt động tốt với các câu hỏi giống nhau và thường lặp đi lặp lại nhưng kém hữu ích với các câu hỏi có nội dung mới của sinh viên và câu hỏi mang tính chất tư vấn và hỗ trợ cho từng cá nhân. Ngoài ra thông tin của hệ thống hỏi đáp phục vụ hỗ trợ cho sinh viên cần phải được cập nhật theo những thay đổi của nhà trường về chính sách, quy chế, quy định. Nếu không thông tin không được cập nhật thường xuyên sẽ dẫn đến lỗi thời, sai thông tin và làm ảnh hưởng đến kết

quả cũng như độ tin cậy của hệ thống hỏi đáp. Vì vậy cần phải bổ sung và cập nhật dữ liệu thường xuyên cho hệ thống.

Kiến nghị những vấn đề tiếp theo

Để kết quả luận văn có thể áp dụng vào trong thực tế, cần phải hoàn thiện dữ liệu về câu hỏi – câu trả lời một cách đầy đủ nhất. Ngoài ra thông tin liên quan đến các câu trả lời cần một chức năng có thể quản lý được các thay đổi giúp cập nhật câu trả lời, bổ sung thêm câu trả lời khi có những thay đổi về mặt thông tin do Nhà trường cung cấp. Ngoài ra luận văn cần đóng gói lại thành sản phẩm để có thể triển khai thực tế và từ đó thu thập những phản hồi của người dùng giúp cải tiến và nâng cao chất lượng và độ tin cậy của hệ thống hỏi đáp. Hệ thống cũng cần phải cập nhật liên tục để đảm bảo các thông tin luôn chính xác khi cung cấp cho người dùng.

TÀI LIỆU THAM KHẢO

[1] M. A. C. Soares and F. S. Parreiras, "A literature review on question answering tech- niques, paradigms and systems," *Journal of King Saud University-Computer and Infor- mation Sciences*, 2018.

[2] X. Yao, "Feature-driven question answering with natural language alignment," Ph.D. dissertation, Johns Hopkins University, 2014.

[3] D. A. Ferrucci, "Introduction to "this is watson"," *IBM Journal of Research and Devel- opment*, vol. 56, no. 3.4, pp. 1:1–1:15, May 2012.

[4] B. F. Green Jr, A. K. Wolf, C. Chomsky, and K. Laughery, "Baseball: an automatic question-answerer," in *Papers presented at the May 9-11, 1961, western joint IRE- AIEE-ACM computer conference*. ACM, 1961, pp. 219–224.

[5] W. A. Woods and R. Kaplan, "Lunar rocks in natural english: Explorations in natural language question answering," *Linguistic structures processing*, vol. 5, pp. 521–569, 1977.

[6] A.MishraandS.K.Jain,"Asurveyonquestionansweringsystemswithclassifi cation," *Journal of King Saud University-Computer and Information Sciences*, vol. 28, no. 3, pp. 345–361, 2016.

[7] M. Sanderson and W. B. Croft, "The history of information retrieval research," *Pro- ceedings of the IEEE*, vol. 100, no. Special Centennial Issue, pp. 1444–1451, 2012.

[8] M. Fridah Nyamisa, "A survey of information retrieval techniques," *Advances in Net- works*, vol. 5, p. 40, 01 2017.

[9] 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.

[10] G. Salton, E. A. Fox, and H. Wu, "Extended boolean information retrieval," Cornell University, Tech. Rep., 1982.

[11] E. A. Fox and S. Sharan, “A comparison of two methods for soft boolean operator interpretation in information retrieval,” 1986.

[12] C. D. Paice, “Soft evaluation of boolean search queries in information retrieval sys- tems,” *Information Technology Research Development Applications*, vol. 3, no. 1, pp. 33–41, 1984.

[13] G. Salton, A. Wong, and C.-S. Yang, “A vector space model for automatic indexing,” *Communications of the ACM*, vol. 18, no. 11, pp. 613–620, 1975.

[14] S. M. Wong, W. Ziarko, and P. C. Wong, “Generalized vector spaces model in informa- tion retrieval,” in *Proceedings of the 8th annual international ACM SIGIR conference on Research and development in information retrieval*. ACM, 1985, pp. 18–25.

[15] J.BeckerandD.Kuropka,“Topic-based vector space model,”in *Proceedings of the 6th international conference on business information systems*, 2003, pp. 7–12.

[16] A. Polyvyanyy and D. Kuropka, “A quantitative evaluation of the enhanced topic-based vector space model,” 2007.

[17] S. Deerwester, S. T. Dumais, G. W. Furnas, T. K. Landauer, and R. Harshman, “In- dexing by latent semantic analysis,” *Journal of the American society for information science*, vol. 41, no. 6, pp. 391–407, 1990.

[18] S. E. Robertson and K. S. Jones, “Relevance weighting of search terms,” *Journal of the American Society for Information science*, vol. 27, no. 3, pp. 129–146, 1976.

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

[20] C. J. Van Rijsbergen, “A non-classical logic for information retrieval,” *The computer journal*, vol. 29, no. 6, pp. 481–485, 1986.

[21] G.Amatiand C.J.Van Rijsbergen, “Probabilistic models of information retrieval based on measuring the divergence from randomness,” *ACM Transactions on Information Systems (TOIS)*, vol. 20, no. 4, pp. 357–389, 2002.

[22] Dat Quoc Nguyen and Anh Tuan Nguyen; “PhoBERT: Pre-trained language models for Vietnamese”

[23] Ateret Anaby-Tavor,1 Boaz Carmeli,1,∗ Esther Goldbraich,1 Amir Kantor,1 George Kour,1,2,∗ Segev Shlomov,1,3,∗ Naama Tepper,1 Naama Zwerdling, “Do Not Have Enough Data? Deep Learning to the Rescue!” in The Thirty-Fourth AAAI Conference on Artificial Intelligence (AAAI-20)

[24] Lopez V., Uren V., Sabou M. and Motta E., “Is Question Answering fit for the Semantic Web?: a Survey”. Universität Bielefeld, Germany, (2011)

Tôi cam đoan đã thực hiện việc kiểm tra mức độ tương đồng về nội dung luận văn thông qua phần mềm DoIT một cách trung thực và đạt mức độ kết quả tương đồng 12% toàn bộ nội dung luận văn. Bản luận văn kiểm tra qua phần mềm là bản cứng luận văn đã nộp để bảo vệ trước hội đồng. Nếu sai tôi xin chịu các hình thức kỷ luật theo quy định hiện hành của Học viện

Hà Nội, ngày 18 tháng 05 năm 2021

HỌC VIÊN CAO HỌC