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

BÁO CÁO ĐỒ ÁN
MÔN TRUY XUẤT THÔNG TIN
Đề tài: XÂY DỰNG HỆ THỐNG TRUY XUẤT THÔNG TIN SỬ DỤNG VECTOR SPACE MODEL VÀ BINARY INDEPENDENCE MODEL

GVHD: ThS. Nguyễn Trọng Chỉnh
Nhóm sinh viên thực hiện:

1. Bùi Quốc Thịnh	MSSV: 20520934
2. Hoàng Đình Hữu	MSSV: 20521384
3. Nguyễn Nguyên Khôi	MSSV: 21521009

ꕥ Tp. Hồ Chí Minh, 07/2023 ꕥ

NHẬN XÉT CỦA GIÁO VIÊN HƯỚNG DẪN

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

.....

……, ngày…….tháng……năm 20…

Người nhận xét

(Ký tên và ghi rõ họ tên)

BẢNG PHÂN CÔNG, ĐÁNH GIÁ THÀNH VIÊN:

Bảng 1: Bảng phân công, đánh giá thành viên

MSSV	Họ và tên	Mô hình VSM	Mô hình BIM	Thư viện Whoosh	Viết báo cáo	Thuyết trình	Làm slide
20520934	Bùi Quốc Thịnh			x	x	x	x
20521384	Hoàng Đình Hữu		x			x	x
21521009	Nguyễn Nguyên Khôi	x				x	x

MỤC LỤC

LỜI MỞ ĐẦU

Vào cuối thế kỉ 20, nghiên cứu đã chỉ ra rằng hầu hết con người thường ưu tiên nhận thông tin từ người khác hơn là từ hệ thống truy xuất thông tin. Trong thời gian đó, mọi thông tin đều được tiếp nhận qua báo chí, truyền thông, và việc du lịch cũng được lên kế hoạch tại đại lý. Tuy nhiên, trong thập kỷ qua, công cụ tìm kiếm trên web đã phát triển và đạt đến mức chất lượng mà khiến mọi người hài lòng. Việc tìm kiếm trên web đã trở thành nguồn tìm kiếm thông tin tiêu chuẩn của con người. Ví dụ, một khảo sát vào năm 2004 cho thấy "92% người dùng internet nói rằng internet là một nơi tốt để truy cập thông tin hàng ngày". Lĩnh vực truy xuất thông tin đã phát triển từ một quy tắc học thuật thành một phương tiện cơ sở cho việc tiếp cận thông tin của con người.

Bài báo cáo sẽ trình bày các khái niệm cơ bản về truy xuất thông tin, hoạt động của lĩnh vực này và đề tài chính của nhóm là xây dựng mô hình truy xuất văn bản với mô hình Vector Space và mô hình Binary Independence. Nhóm hy vọng rằng mô hình này sẽ đóng góp cho cộng đồng một công cụ hữu ích giúp người dùng tìm kiếm thông tin chính xác liên quan đến nội dung văn bản.

CHƯƠNG I: GIỚI THIỆU

1. Truy xuất thông tin là gì?

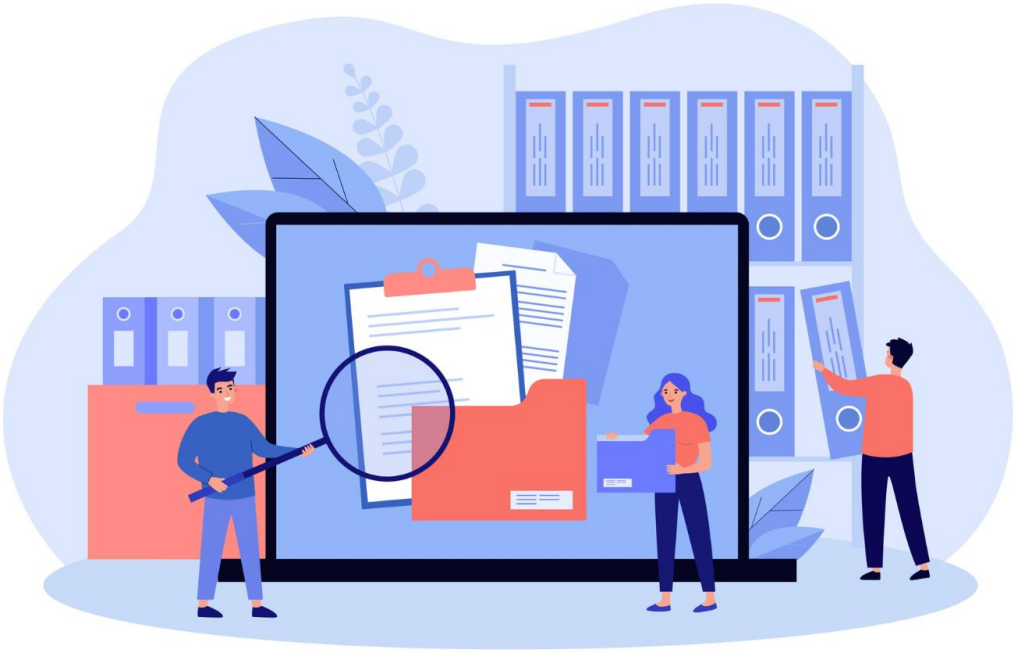
Truy xuất thông tin, là quá trình tìm kiếm và thu thập các nguồn thông tin liên quan đến thông tin mà người dùng đang tìm kiếm. Truy vấn thông tin có thể dựa trên siêu dữ liệu (metadata) hoặc chỉ mục toàn văn bản (full-text) hoặc các nội dung khác. Đây là một lĩnh vực khoa học nghiên cứu cách tìm kiếm thông tin trong các tài liệu, metadata và cơ sở dữ liệu của văn bản, hình ảnh và âm thanh.

Truy vấn thông tin xuất phát từ lý thuyết về thông tin, trong đó văn bản được coi là một hình thức mã hóa của thông tin. Nó liên quan đến việc xác định cách tìm kiếm và truy cập thông tin một cách hiệu quả, đáp ứng nhu cầu người dùng. Qua quá trình truy vấn thông tin, người dùng có thể thu thập, xử lý và sử dụng thông tin để nắm bắt kiến thức và giải quyết các vấn đề tương ứng. Đối với truy vấn dựa trên metadata, thông tin về tác giả, thời gian, định dạng và các thuộc tính khác của tài liệu có thể được sử dụng để xác định sự liên quan. Trong khi đó, truy vấn dựa trên lập chỉ mục toàn văn bản sẽ phân tích và so sánh nội dung của tài liệu để tìm kiếm các từ khóa, cụm từ hoặc mô hình ngữ nghĩa.

Ngoài việc tìm kiếm trong văn bản, truy vấn thông tin cũng có thể được áp dụng cho các dạng thông tin khác như hình ảnh và âm thanh. Ví dụ, trong trường hợp tìm kiếm hình ảnh, các thuật toán phân tích nội dung hình ảnh có thể được sử dụng để tìm kiếm các đối tượng, màu sắc, hình dáng và các thuộc tính khác của hình ảnh. Đối với âm thanh, truy vấn thông tin có thể dựa trên các thuật toán nhận dạng giọng nói hoặc trích xuất đặc trưng âm thanh để tìm kiếm các mẫu, từ khóa hoặc thông tin khác trong âm thanh.

Các hệ thống truy vấn thông tin, chẳng hạn như các công cụ tìm kiếm web, là những phần mềm giúp tổ chức và truy xuất thông tin từ các kho lưu trữ tài liệu, đặc biệt là thông tin văn bản. Nhờ vào các hệ thống này, người dùng có thể dễ dàng tìm kiếm và truy cập thông tin phù hợp theo yêu cầu của mình.

Bên cạnh việc tìm kiếm thông tin từ các nguồn đã được chỉ định, truy vấn thông tin cũng có thể mở rộng đến việc khám phá nguồn thông tin mới và tự động thu thập các tài liệu liên quan. Điều này đảm bảo người dùng nhận được thông tin cần thiết và đáng tin cậy để nâng cao kiến thức và đáp ứng nhu cầu tìm kiếm của họ.

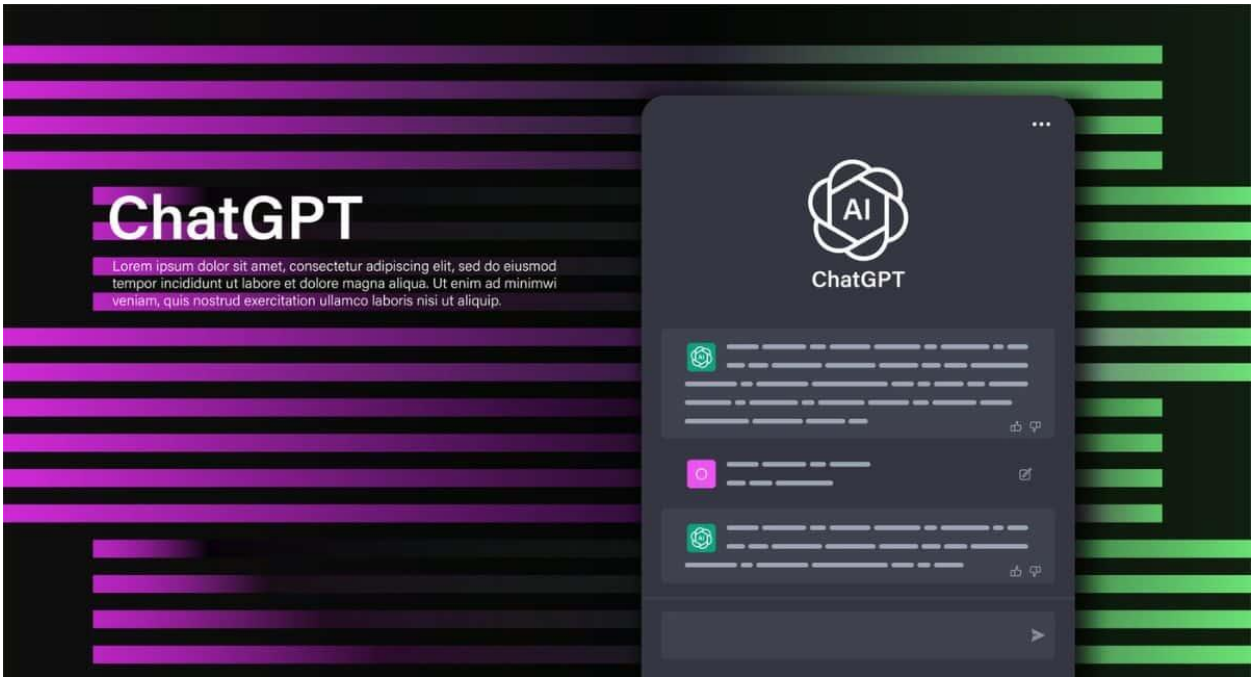


Ngoài việc liên quan đến việc tìm kiếm trên web, Truy xuất Thông tin (IR) không chỉ giới hạn ở lĩnh vực này mà còn bao gồm nhiều khía cạnh khác nhau. Một trong số đó là Trả lời câu hỏi (question answering), một lĩnh vực quan trọng trong nghiên cứu và ứng dụng của IR.

Trả lời câu hỏi (question answering) là quá trình tự động tìm kiếm và cung cấp câu trả lời chính xác cho các câu hỏi được đặt ra bằng ngôn ngữ tự nhiên. Thay vì chỉ trả về danh sách các tài liệu liên quan như trong truy vấn thông thường, trả lời câu hỏi cố gắng hiểu ý nghĩa và mục đích của câu hỏi và trực tiếp đưa ra câu trả lời.

Trả lời câu hỏi đòi hỏi sự kết hợp giữa các kỹ thuật truy vấn thông tin và xử lý ngôn ngữ tự nhiên. Quá trình này bao gồm phân tích câu hỏi, tìm kiếm thông tin phù hợp, rút trích kiến thức từ nguồn thông tin và cuối cùng đưa ra câu trả lời một cách logic và dễ hiểu.

Lĩnh vực trả lời câu hỏi đã có những tiến bộ đáng kể trong thời gian gần đây, đặc biệt là do sự phát triển của các kỹ thuật học máy và trí tuệ nhân tạo. Các hệ thống trả lời câu hỏi ngày nay đã đạt được hiệu suất tốt trong việc xử lý các câu hỏi phức tạp và đưa ra câu trả lời chính xác.



Một khía cạnh quan trọng khác của Truy xuất Thông tin (IR) là Khai thác văn bản (text mining). Khai thác văn bản là quá trình tự động phân tích và khám phá thông tin từ các nguồn văn bản không cấu trúc hoặc có cấu trúc.

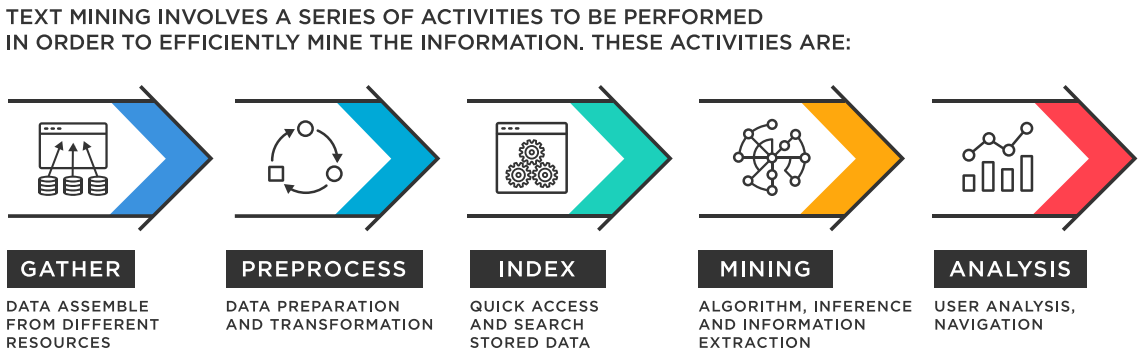
Trong khai thác văn bản, các thuật toán và phương pháp được sử dụng để tìm ra các mẫu, quy luật, thông tin ẩn và liên kết trong văn bản. Mục tiêu của khai thác văn bản là trích xuất và tạo ra kiến thức có giá trị từ dữ liệu văn bản, giúp người dùng hiểu rõ hơn về nội dung, xu hướng và mối quan hệ trong tập dữ liệu đó.

Các công nghệ trong khai thác văn bản bao gồm:

- Phân loại văn bản: Sử dụng các thuật toán máy học để xác định và phân loại văn bản vào các nhóm hoặc danh mục khác nhau dựa trên nội dung và thuộc tính của chúng.

- Rút trích thông tin: Tìm kiếm và trích xuất thông tin cụ thể từ văn bản như tên, địa chỉ, số điện thoại, ngày tháng, sự kiện quan trọng và các đối tượng khác.
- Phân tích ý kiến: Đánh giá và phân loại ý kiến và cảm xúc trong văn bản, giúp đánh giá sự hài lòng của khách hàng, phản hồi của người dùng và xu hướng chung của công chúng.
- Tóm tắt văn bản: Tạo ra các bản tóm tắt ngắn gọn từ nội dung dài và phức tạp, giúp người dùng tiết kiệm thời gian và tìm hiểu nhanh về thông tin quan trọng.
- Phát hiện và mô hình hóa liên kết: Phân tích mối quan hệ và liên kết giữa các thành phần trong văn bản như người, địa điểm, sự kiện, để xây dựng mô hình tri thức và biểu đồ liên kết.

Khai thác văn bản có ứng dụng rộng rãi trong nhiều lĩnh vực như khai thác dữ liệu web, phân tích tài chính, chăm sóc sức khỏe, phân tích phương tiện truyền thông xã hội và nhiều lĩnh vực nghiên cứu khác. Nó giúp tận dụng tri thức từ lượng lớn dữ liệu văn bản và mang lại những thông tin quan trọng và giá trị cho việc ra quyết định, dự đoán xu hướng và tìm hiểu sâu về môi trường thông tin.

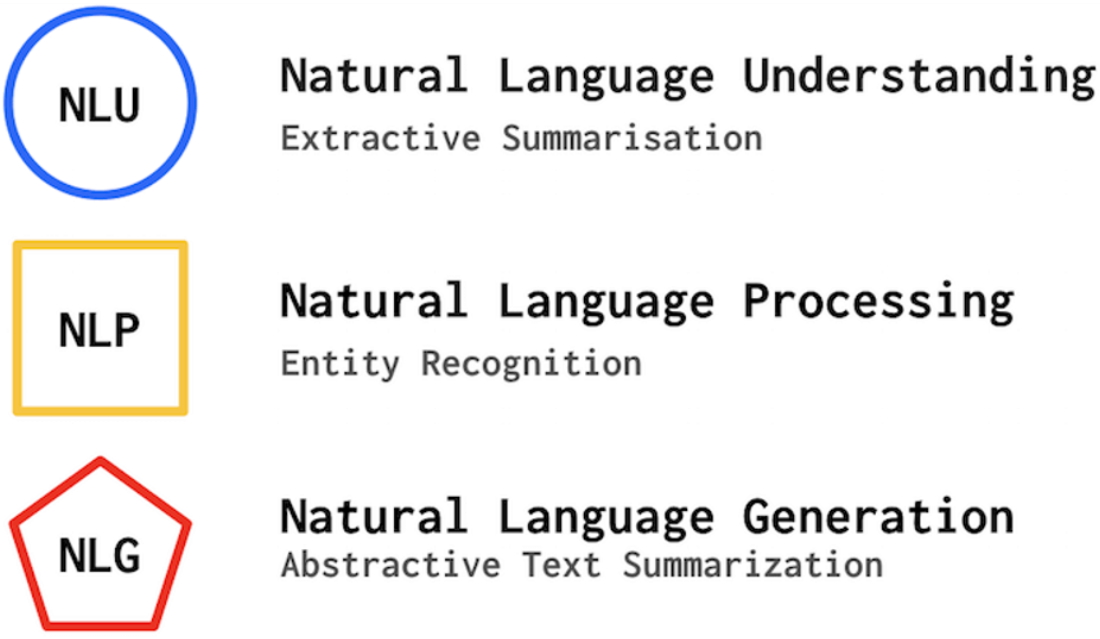


Quảng cáo trực tuyến (online advertising) là một hình thức tiếp thị và quảng bá sản phẩm, dịch vụ hoặc thông điệp thông qua Internet. Nó là quá trình hiển thị thông tin quảng cáo trên các nền tảng trực tuyến như trang web, ứng dụng di động, mạng xã hội và các kênh truyền thông khác.

Quảng cáo trực tuyến có nhiều hình thức và phương pháp khác nhau như:

- Display Advertising: Quảng cáo hiển thị, bao gồm các banner, hình ảnh, video và quảng cáo đa phương tiện khác. Được đặt trên các trang web, ứng dụng di động hoặc trong nội dung liên quan.
- Search Engine Advertising: Quảng cáo hiển thị trong kết quả tìm kiếm trên các công cụ tìm kiếm như Google, Bing, Yahoo. Thông qua việc đặt quảng cáo trong các kết quả tìm kiếm liên quan đến từ khóa cụ thể.
- Social Media Advertising: Quảng cáo trên các mạng xã hội như Facebook, Instagram, Twitter, LinkedIn. Sử dụng dữ liệu người dùng và định tuyến đến đúng đối tượng khách hàng mục tiêu.
- Video Advertising: Quảng cáo trên nền tảng video như YouTube, Vimeo, TikTok. Hiển thị trước, giữa hoặc sau video và có thể bao gồm các quảng cáo trước nội dung video.
- Native Advertising: Quảng cáo tích hợp vào nội dung tự nhiên của trang web hoặc ứng dụng di động, tạo cảm giác gần gũi và không gây phiền hà cho người dùng.

Quảng cáo trực tuyến có nhiều lợi ích như tiếp cận đến một số lượng lớn người dùng, khả năng đo lường và theo dõi hiệu quả, tăng tính tương tác và tương tác khách hàng. Nó cung cấp cơ hội tiếp cận đến đối tượng khách hàng mục tiêu cụ thể và tăng khả năng tiếp cận thị trường đa quốc gia.



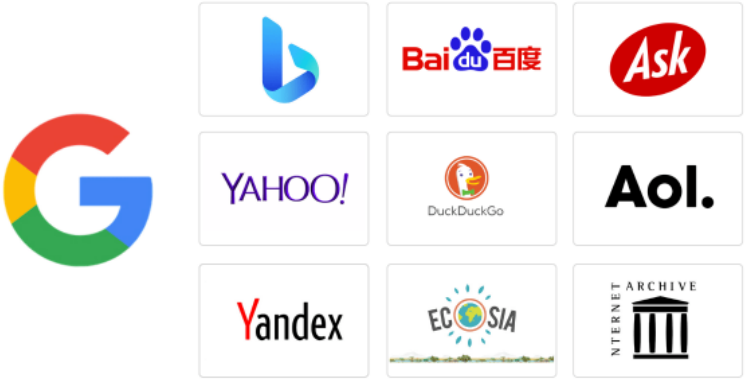
Tìm kiếm doanh nghiệp (enterprise search) là một khái niệm tổng quát để chỉ quá trình tìm kiếm thông tin trong môi trường làm việc doanh nghiệp. Nó kết hợp hai khái niệm quan trọng khác là tìm kiếm trên web (web search) và tìm kiếm trên máy tính cá nhân (desktop search).

Tìm kiếm trên web (web search): Đây là quá trình tìm kiếm thông tin trên Internet, sử dụng các công cụ tìm kiếm web như Google, Bing, Yahoo và nhiều công cụ tìm kiếm khác. Tìm kiếm trên web giúp người dùng tìm kiếm và truy cập thông tin từ các trang web trên Internet.

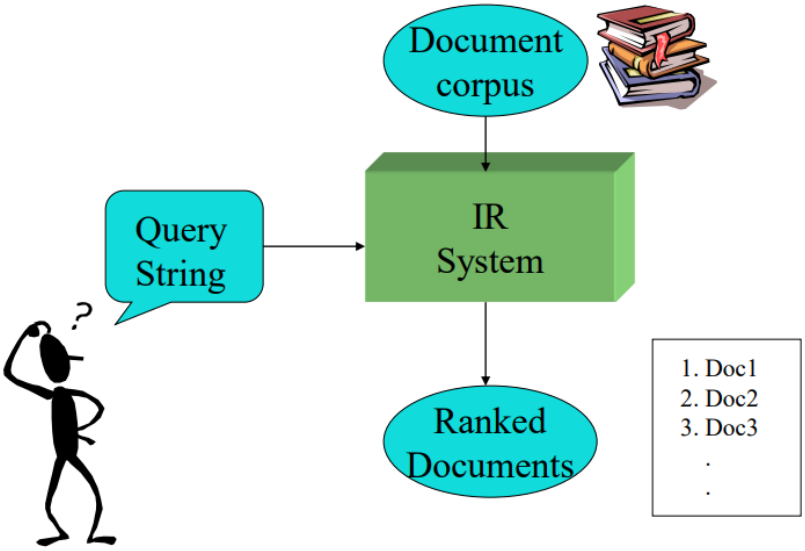
Tìm kiếm trên máy tính cá nhân (desktop search): Đây là quá trình tìm kiếm thông tin trên máy tính cá nhân hoặc trong mạng nội bộ của một tổ chức. Các công cụ tìm kiếm trên máy tính cá nhân giúp người dùng tìm kiếm và truy cập thông tin trong các tệp tin, email, tài liệu và các nội dung khác có trên máy tính của họ.

Khi kết hợp cả tìm kiếm trên web và tìm kiếm trên máy tính cá nhân, tìm kiếm doanh nghiệp (enterprise search) tạo ra một hệ thống tìm kiếm toàn diện cho tổ chức. Nó cho phép người dùng tìm kiếm thông tin từ các nguồn dữ liệu khác nhau, bao gồm cả nội dung trên web và nội dung trên máy tính nội bộ. Điều này giúp cải thiện khả năng tìm kiếm, tăng năng suất và tiết kiệm thời gian cho các thành viên trong tổ chức.

Công nghệ tìm kiếm doanh nghiệp thường bao gồm việc xây dựng các hệ thống lập chỉ mục (indexing) để tổ chức và tìm kiếm thông tin hiệu quả, sử dụng các thuật toán phân tích ngôn ngữ tự nhiên và công nghệ trí tuệ nhân tạo để cải thiện độ chính xác và phản hồi của kết quả tìm kiếm.



Quá trình truy vấn thông tin bắt đầu khi người dùng hoặc người tìm kiếm nhập câu truy vấn vào hệ thống. Câu truy vấn có thể được hiểu là một biểu diễn chính thức và rõ ràng của thông tin, tương tự như một chuỗi từ khóa trong các công cụ tìm kiếm trên web. Trong truy xuất thông tin, một câu truy vấn không thể chính xác xác định một đối tượng trong bộ tài liệu, thay vào đó có nhiều đối tượng phù hợp với câu truy vấn với mức độ tương đương khác nhau. Đối tượng có thể hiểu là một thực thể được đại diện bởi thông tin trong bộ nội dung hoặc cơ sở dữ liệu. Các câu truy vấn của người dùng sẽ khớp với thông tin trong cơ sở dữ liệu. Tuy nhiên, khác với truy vấn SQL truyền thống trong cơ sở dữ liệu, trong truy vấn thông tin, kết quả trả về có thể khớp hoặc không khớp với câu truy vấn và do đó sẽ được xếp hạng. Xếp hạng các kết quả trả về là điểm khác biệt chính giữa tìm kiếm truy vấn thông tin và tìm kiếm trong cơ sở dữ liệu.



Các đối tượng dữ liệu (data object) trong các ứng dụng khác nhau có thể là các loại tài liệu khác nhau như văn bản, hình ảnh, âm thanh, mindmap hay video. Thay vì lưu trữ trực tiếp các tài liệu này trong hệ thống IR, thường thì chúng sẽ được biểu diễn bằng các tài liệu thay thế hoặc metadata. Để đánh giá mức độ phù hợp (độ tương đồng hoặc mức độ liên quan) giữa câu truy vấn của người dùng và các đối tượng dữ liệu trong cơ sở dữ liệu, hầu hết các hệ thống truy vấn thông tin đều sử dụng một số phương pháp tính điểm số khác nhau, ví dụ như dựa trên các độ đo đánh giá khác nhau. Sau đó, các đối tượng dữ liệu sẽ

được xếp hạng theo thứ tự giảm dần của điểm số và hiển thị cho người dùng những đối tượng có điểm số cao nhất, tức là những tài liệu phù hợp và liên quan nhất với câu truy vấn. Người dùng có thể lặp lại quá trình này nếu muốn chỉnh sửa query.

Nói ngắn gọn, truy vấn thông tin là quá trình tìm kiếm:

• Các tài liệu chứa thông tin (tài liệu – Document).

• Có dạng phi cấu trúc hoặc bán cấu trúc: Free Text, XML.

• Trong các bộ lưu trữ lớn (Collections) • Đáp ứng yêu cầu.

Một hệ thống truy vấn thông tin hay còn gọi là hệ thống IR là một phần mềm máy tính có chức năng:

• Lưu trữ và quản lí thông tin trong các tài liệu (văn bản…).

• Hỗ trợ việc tìm kiếm thông tin.

• Không cung cấp thông tin hay trả lời câu hỏi một cách rõ ràng mà chỉ báo cáo về sự hiện diện và vị trí của các tài liệu có thể mang thông tin cần thiết.

2. Lập chỉ mục (Indexing)

Các hệ thống tìm kiếm hiệu quả thường sẽ không giao tiếp trực tiếp với tài liệu (hoặc truy vấn). Chúng sẽ áp dụng các phương pháp và chiến thuật khác nhau để biểu thị các khía cạnh ngữ nghĩa quan trọng của tài liệu và truy vấn. Quá trình này được gọi là xây dựng chỉ mục. Nó sẽ được trình bày trong 2 phần sau đây:

2.1. Kích thước chỉ mục (Indexing Dimensions)

Quá trình xây dựng chỉ mục nhằm mục đích biểu thị nội dung ngữ nghĩa của tài liệu (hoặc truy vấn). Trong các trường hợp thường gặp nhất, các đơn vị xây dựng chỉ mục (indexing units) này sẽ là từ cho tài liệu (words for documents), các nốt nhạc cho lĩnh vực âm nhạc (musical notes for music), giá trị màu sắc cho các bức hình (color values for pictures) Sau khi chọn được các đơn vị xây dựng chỉ mục, ta sẽ phải giải quyết các câu hỏi khác nhau để xác định một chiến lược xây dựng chỉ mục hoàn thiện. Khi miêu tả một tài liệu, ta sẽ cần phải quan tâm đến tất cả các chi tiết của nó hay chỉ các khía cạnh chính? Khi xác định mức độ đầy đủ này (tính đầy đủ của biểu diễn), người ta cũng có thể xem xét tầm quan trọng của tài liệu đối với một số mục tiêu (ví dụ: tài liệu nội bộ có thể có phạm vi bao phủ sâu

hơn). Ngoài ra, người ta phải quyết định mức chung hoặc mức độ chính xác được áp dụng cho các đơn vị chỉ mục đã lựa chọn: chọn các thuật ngữ chỉ mục phổ biến (ví dụ: "drink"), hoặc yêu cầu việc sử dụng các thuật ngữ chuyên biệt hơn ("softdrink), hoặc thậm chí là các từ ngữ rất chuyên sâu ("iced tea", "herbal iced tea", "Nestea"). Truyền thống, các librarians đã áp dụng chiến lược xây dựng chỉ mục thủ công với mong muốn tạo ra một biểu diễn tốt hơn cho các đối tượng có thể tìm kiếm của họ. Việc xây dựng chỉ mục thủ công như vậy thường dựa vào việc sử dụng các từ vựng được kiểm soát để đạt được tính nhất quán cao hơn và nâng cao chất lượng xây dựng chỉ mục thủ công. Tuy nhiên, trong khi các từ vựng được kiểm soát sẽ tăng tính nhất quán giữa các bộ chỉ mục, các nghiên cứu khác nhau cho thấy rằng các bộ chỉ mục khác nhau có xu hướng sử dụng các từ khóa khác nhau khi phân loại cùng một tài liệu. Điều này minh họa, thông qua một ví dụ khác, sự đa dạng của các từ mà con người có thể tùy ý sử dụng để mô tả cùng một nội dung. Do đó, điều quan trọng là phải có một mức độ nhất quán nhất định giữa biểu diễn các tài liệu và truy vấn để tăng cơ hội mà người tìm kiếm có thể xác định thông tin họ yêu cầu.

2.2. Quá trình lập chỉ mục (Indexing Process):

Hầu hết các hệ thống IR hiện nay đều xây dựng chỉ mục các tài liệu và truy vấn một cách tự động. Cách tiếp cận như vậy được phát triển từ những năm 60, cho phép hệ thống có thể xử lí được lượng thông tin khổng lồ có sẵn trên mạng. Một thuật toán xây dựng chỉ mục tự động sẽ bao gồm 4 bước:

- Phân tích cấu trúc và tokenization: trong bước này, các tài liệu sẽ được phân tích ngữ pháp để có thể nhận ra cấu trúc của chúng (title, abstract, section, paragraphs). Đối với mỗi cấu trúc logic có liên quan, hệ thống sau đó sẽ phân tách các câu thành các word tokens (do đó có thuật ngữ tokenization và thuật ngữ này sẽ được nói chi tiết hơn ở bước tiền xử lí trong phần sau). Quá trình này có thể tương đối dễ dàng việc sử dụng chữ viết tắt có thể khiến hệ thống nhầm lẫn boundary của câu trong khi nó lại không có (vd: "don't", "I'll", "John's"…) và nó sẽ phải đưa ra quyết định liên quan đến số, kí tự đặc biệt, dấu gạch nối và cách viết hoa. Đối với các con số, không có định nghĩa nào có thể được tìm thấy. Chúng ta có thể đơn giản bỏ qua

chúng hoặc bao gồm chúng dưới dạng các đơn vị xây dựng chỉ mục. Cuối cùng chính là chữ hoa sẽ được viết thường.

- Loại bỏ stopword: xóa các từ hạn định (vd: the), giới từ (vd: from), liên từ (vd: and), đại từ (vd: you) và một vài từ xuất hiện khá thường xuyên (vd: am, is, are) (chi tiết sẽ được trình bày ở phần sau)
- Morphological normalization: ở bước thứ 3, quá trình xây dựng chỉ mục sử dụng một số loại morphological normalization nhằm cố gắng kết hợp các biến thể của từ vào cùng một gốc hoặc đưa về từ gốc.
- Weighting: đầu tiên, ta có thể giả định rằng một đơn vị chỉ mục xuất hiện thường xuyên hơn trong một tài liệu phải có tầm quan trọng cao hơn trong việc biểu diễn nội dung ngữ nghĩa của nó, giá trị này được kí hiệu là tf. Giá trị weighting thứ hai chính là df hay còn gọi là "document frequency" nghĩa là tần suất tài liệu (hai giá trị này sẽ được nói chi tiết hơn ở phần tf-idf) Nói một cách ngắn gọn và dễ hiểu thì trong việc xây dựng mô hình của nhóm, xây dựng chỉ mục tài liệu là quá trình liên kết thông tin với một file hoặc tag cụ thể cho phép dễ dàng tìm kiếm và truy xuất sau này. Chẳng hạn, các tài liệu sẽ được biểu diễn thành các vector trong mô hình Vector Space. Thông tin được xây dựng chỉ mục sau đó sẽ được lập trình thành một hệ thống quản lý tài liệu, giúp người dùng dễ dàng truy cập vào dữ liệu mà họ yêu cầu. Nếu không xây dựng chỉ mục tài liệu hiệu quả, việc truy xuất thông tin có thể rất tốn thời gian và chi phí.

3. Tiền xử lý dữ liệu (pre-processing)

Văn bản không có cấu trúc làm cho việc xử lý trở nên khó khăn hơn. Điều này càng rõ ràng hơn với văn bản trên web chứa các HTML tag, code JS, những thứ gây nhiễu. Do đó, tiền xử lý là bước quan trọng và có ảnh hưởng lớn tới kết quả truy vấn, chẳng hạn như khi V chỉ có "USA" mà query lại là "U.S.A", nếu không tiền xử lý để đồng nhất "U.S.A" thành "USA" thì sẽ không tìm được document cần thiết.

Tiền xử lý bao gồm các bước tiêu biểu sau đây:

3.1. Loại bỏ punctuation

Punctuations bao gồm các ký tự sau: !"#$%&'()*+,-./:;<=>?@[]^_`{|}~

Punctuations thường không có tác dụng gì trong việc truy vấn, thậm chí là đôi khi chúng còn gây khó khăn cho việc truy vấn, chẳng hạn như nếu không bỏ đi punctuations thì "student?" và "student" là 2 term hoàn toàn khác nhau.

3.2. Lowercase

Nghĩa là đổi chữ hoa thành chữ thường. Mục đích là để không bị case-sensitive (phân biệt chữ hoa và chữ thường). Ví dụ: "Vietnam" và "vietnam" là hai term hoàn toàn khác nhau.

3.3. Tokenization

Tokenization là quá trình phân đoạn các văn bản thành các đơn vị nhỏ nhất, được gọi là token. Mức độ tokenization tùy thuộc vào cách thức phân loại các đơn vị này và mục đích sử dụng của việc tokenization. Ở mức độ đơn giản nhất, tokenization có thể dựa trên các kí tự trắng hoặc dấu câu để cắt các từ và câu. Đây được coi là một tokenization đơn giản, tạo ra các token là các từ và dấu câu. Tuy nhiên, đây không phải là phương pháp tokenization thông dụng nhất và không phải là phương pháp tokenization được dùng trong các ứng dụng phức tạp hơn. Ở mức độ cao hơn, các phương pháp tokenization sử dụng các quy tắc ngữ pháp và ngữ nghĩa để phân tích cấu trúc và phân loại các đơn vị trong văn bản. Các phương pháp tokenization này có thể sử dụng các công cụ như từ điển, máy học, hoặc phân tích cấu trúc để tạo ra các token biểu thị cho các thành phần ngữ pháp như danh từ, động từ và tính từ.

3.4. Stopword

Stopword là những từ phổ biến trong ngôn ngữ tự nhiên, không mang nhiều ý nghĩa cho văn bản. Ví dụ như các từ "he", "she" "I",… xuất hiện trong rất nhiều văn bản, nên truy vấn bằng những từ như vậy sẽ không có tác dụng gì nhiều cho xếp hạng kết quả trả về, ngoài ra còn để tiết kiệm bộ nhớ. Tuy nhiên Có một số trường hợp trong xử lý ngôn ngữ tự nhiên mà loại bỏ stopwords không cần thiết hoặc thậm chí là không mong muốn. Ví dụ:

• Khi phân tích ngôn ngữ tự nhiên trong một chủ đề đặc biệt, các stopwords có thể có ý nghĩa quan trọng. Ví dụ, trong một tài liệu về lịch sử của các từ ngữ, các stopwords như "of", "in", "and" có thể là rất quan trọng để hiểu được bối cảnh và mối quan hệ giữa các từ.

• Khi sử dụng các thuật toán phân tích ngôn ngữ tự nhiên phức tạp, loại bỏ stopwords có thể làm mất đi một số thông tin quan trọng. Ví dụ, trong các mô hình deep learning như Transformers, các stopwords có thể giúp mô hình hiểu được cấu trúc ngữ pháp của câu.

• Trong một số trường hợp, loại bỏ stopwords có thể làm mất đi một số thông tin quan trọng trong văn bản và làm suy giảm chất lượng kết quả xử lý ngôn ngữ tự nhiên. Ví dụ, trong một bài báo khoa học, các stopwords có thể là rất quan trọng để hiểu được ý nghĩa và kết quả của nghiên cứu. Nên tùy trường hợp mà chúng ta có thể lựa chọn sẽ loại bỏ stopwords hay không.

3.5. Stemming

Là quá trình chuyển đổi các từ về dạng gốc của chúng bằng cách bỏ đi các tiền tố (prefix) hoặc các hậu tố (suffix). Mục đích là giảm bớt sự phong phú của các từ trong văn bản và làm cho các từ giống nhau càng giống hơn. Việc sử dụng quá trình stemming có thể nâng cao hiệu quả cho các ứng dụng NLP như tìm kiếm thông tin, phân loại văn bản hoặc dịch máy. Khi các từ giống nhau hơn, các thuật toán có thể xử lí chúng dễ dàng hơn, giảm thiểu các trường hợp 1 từ xuất hiện ở nhiều hình thức khác nhau trong văn bản.

4. Các thành phần của hệ thống truy xuất thông tin dưới dạng văn bản

Bốn thành phần của mỗi hệ thống truy vấn thông tin văn bản là: Tập tài liệu, câu truy vấn, mô hình truy vấn thông tin văn bản và công thức tính độ liên quan giữa các tài liệu trong tập tài liệu và câu truy vấn. Mỗi thành phần đều có vai trò quan trọng trong tổng thể hệ thống truy vấn thông tin văn bản, điều này sẽ được giải thích thông qua chức năng của các thành phần này trong các bước hoạt động của mô hình truy vấn thông tin văn bản trong các phần sau của báo cáo này.

4.1. Tập tài liệu (Document)

Các tài liệu trong tập dữ liệu là các file lưu trữ văn bản dưới một số định dạng nhất định (thường là file txt), thường là vật liệu chứa thông tin phi cấu trúc như đã nêu trên. Khi xây dựng hệ thống truy vấn thông tin văn bản, các tài liệu trong tập dữ liệu sẽ được biểu diễn dưới dạng các mô hình lưu trữ như mô hình Binary Independence, mô hình Vector Space,… và được đánh chỉ mục để tiện lợi cho việc lưu trữ và truy xuất thông tin.

4.2. Câu truy vấn

Nhu cầu tìm kiếm thông tin được người dùng biểu thị ở dạng ngôn ngữ tự nhiên, điều này gây ra một số khó khăn cho máy tính trong việc xử lý những yêu cầu này. Từ câu truy vấn của người dùng người dạng ngôn ngữ tự nhiên, cần phải xử lý thông qua một số khâu tiền xử lý và chuyển về dưới dạng ngôn ngữ hình thức trước khi thực hiện truy vấn thông tin.

4.3. Mô hình truy vấn

Cần phải chọn và xây dựng một mô hình truy vấn thông tin phù hợp với các yêu cầu cần thiết đối với bài toán như khả năng tổ chức và lưu trữ các tài liệu tiện lợi cho việc truy vấn,

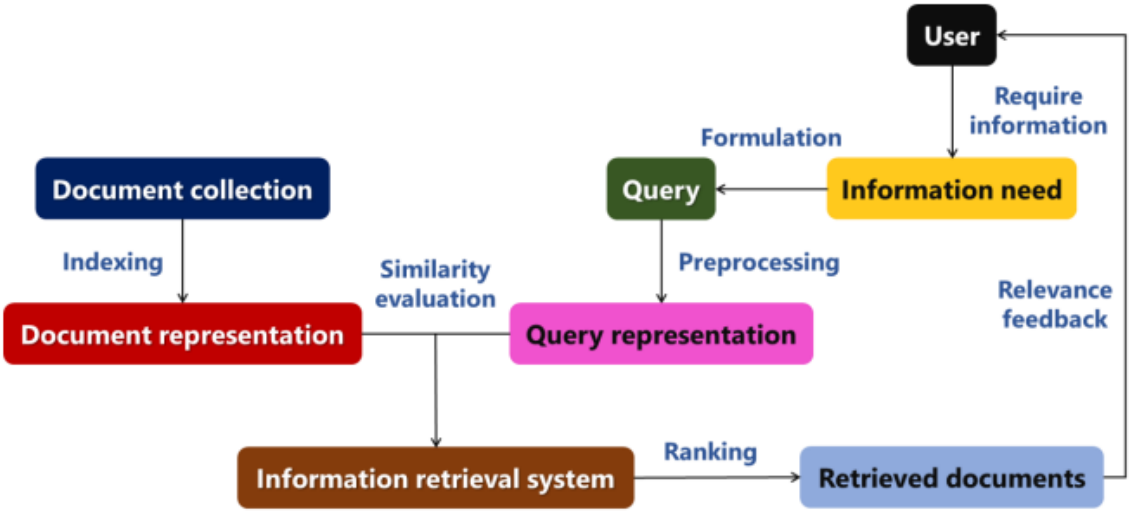
khả năng kiểm tra mức độ liên quan giữa câu truy vấn và tài liệu và khả năng xếp hạng kết quả truy vấn.

4.4. Công thức tính độ liên quan (Công thức xếp hạng)

Để có thể tìm được những tài liệu thỏa mãn/phù hợp với nhu cầu tìm kiếm thông tin của người dùng, hệ thống cần phải tiến hành đánh giá độ liên quan giữa các tài liệu và câu truy vấn. Mức độ liên quan giữa câu truy vấn và các tài liệu trong tập dữ liệu có thể được đánh giá thông qua một số độ đo tính độ tương đồng. Việc chọn công thức tính độ liên quan giữa các tài liệu và câu truy vấn cần phải phù hợp với cách biểu diễn của các đối tượng này. Tùy vào tính chất của công thức mà chúng ta sẽ xem xét xem giá trị của độ đo lớn hay nhỏ thì độ liên quan càng lớn. Dựa trên mức độ liên quan giữa các tài liệu và câu truy vấn, chúng ta thu được một danh sách xếp hạng với các tài liệu được sắp xếp theo thứ tự độ liên quan so với câu truy vấn giảm dần.

5. Quá trình truy vấn thông tin văn bản

Hoạt động của quy trình truy vấn thông tin văn bản gồm các bước sau: biểu diễn các tài liệu trong tập tài liệu; biểu diễn câu truy vấn; đánh giá mức độ liên quan giữa các tài liệu và câu truy vấn; xếp hạng và trả kết quả truy vấn cho người dùng.



5.1. Biểu diễn các tài liệu trong tập dữ liệu

Các tài liệu trong tập dữ liệu là các file lưu trữ văn bản dưới một số định dạng nhất định, thường là thông tin phi cấu trúc như đã nêu trên. Khi xây dựng hệ thống truy vấn thông tin văn bản, các tài liệu trong tập dữ liệu sẽ được biểu diễn dưới dạng các mô hình lưu trữ như mô hình Binary Independence, mô hình Vector Space,… và được đánh chỉ mục để tiện lợi cho việc lưu trữ và truy xuất thông tin. Lập chỉ mục tài liệu là quá trình kết nối thông tin với một tệp hoặc thẻ cụ thể cho phép dễ dàng tìm thấy và truy xuất sau này. Chẳng hạn, các tài liệu sẽ được biểu diễn thành các vector trong mô hình Vector Space. Thông tin được lập chỉ mục sau đó sẽ được lập trình thành một hệ thống quản lý tài liệu, giúp người dùng dễ dàng truy cập vào dữ liệu mà họ yêu cầu. Nếu không lập chỉ mục tài liệu hiệu quả, việc truy xuất thông tin có thể rất tốn thời gian và chi phí.

5.2. Biểu diễn câu truy vấn

Quy trình truy vấn thông tin văn bản được khởi động khi người dùng tạo bất kỳ câu truy vấn nào vào hệ thống thông qua một số giao diện đồ họa được cung cấp. Các câu truy vấn này biểu diễn nhu cầu tìm kiếm thông tin của người dùng dưới dạng ngôn ngữ hình thức, chẳng hạn như một số ngôn ngữ Toán học như ngôn ngữ của Đại số Boolean (các phép toán AND, OR, NOT,…). Từ câu truy vấn của người dùng người dạng ngôn ngữ tự nhiên, cần phải xử lý thông qua một số khâu tiền xử lý như loại bỏ các stopwords, normalization, stemming, lemmatization,… và chuyển về dưới dạng ngôn ngữ hình thức trước khi thực hiện truy vấn thông tin. Cần phải chú ý là cách biểu diễn câu truy vấn cần phải phù hợp với cách biểu diễn các tài liệu trong tập tài liệu để tiện lợi cho việc truy vấn thông tin.

5.3. Đánh giá mức độ liên quan giữa các tài liệu và câu truy vấn

Để có thể tìm được những tài liệu thỏa mãn/phù hợp với câu truy vấn và nhu cầu tìm kiếm thông tin của người dùng, hệ thống cần phải tiến hành đánh giá độ tương đồng (mức độ liên quan) giữa các tài liệu trong tập dữ liệu và câu truy vấn. Độ tương đồng (mức độ liên quan) giữa câu truy vấn và các tài liệu trong tập dữ liệu có thể được đánh giá thông qua một số độ đo tính độ tương đồng như Cosine Similarity, Euclidean Distance,… Việc chọn công thức tính độ liên quan giữa các tài liệu và câu truy vấn cần phải phù hợp với cách biểu diễn của các đối tượng này ở bước trước. Chẳng hạn, ở các khâu trước, hệ thống sẽ

biểu diễn các tài liệu trong tập dữ liệu và câu truy vấn dưới dạng các Vector trong mô hình Vector Space. Khi đó, chúng ta có thể đánh giá độ liên quan giữa các tài liệu và câu truy vấn bằng cách tính giá trị cosine của góc tạo bởi các vector biểu diễn các tài liệu và vector biểu diễn câu truy vấn bằng công thức Cosine Similarity, hay một cách khác là tính khoảng cách giữa các vector biểu diễn các tài liệu và vector biểu diễn câu truy vấn bằng công thức Euclidean Distance. Giá trị cosine hay khoảng cách này được xem như độ tương đồng, mức độ liên quan giữa các tài liệu và câu truy vấn. Chúng ta có thể đánh giá độ liên quan giữa các tài liệu và câu truy vấn thông qua giá trị của các độ đo đánh gá này. Tùy vào tính chất của công thức mà chúng ta sẽ xem xét xem giá trị của độ đo lớn hay nhỏ thì độ liên quan càng lớn.

5.4. Xếp hạng và trả kết quả truy vấn cho người dùng

Khâu này chỉ có ở một số mô hình truy vấn thông tin có khả năng xếp hạng kết quả truy vấn như mô hình Binary Independence, Vector Space. Trong truy vấn thông tin, truy vấn đơn không khớp với đúng đối tượng dữ liệu mà nó khớp với một số bộ sưu tập các đối tượng dữ liệu. Do đó, cần phải xếp hạng các kết quả truy vấn để có thể tìm ra được những tài liệu liên quan nhất với thông tin cần tìm. Việc xếp hạng các tài liệu có liên quan được thực hiện với mục đích để tìm ra tài liệu liên quan nhất với câu truy vấn đã cho. Chẳng hạn, đối với độ đo log-odds, số điểm (giá trị độ đo) giữa tài liệu và truy vấn càng lớn thì chúng càng liên quan đến nhau. Tương tự, đối với Cosine Similarity, số điểm (giá trị độ đo) của độ đo cosine giữa tài liệu và câu truy vấn càng lớn thì chúng càng liên quan đến nhau. Đối với Euclidean Distance thì ngược lại, số điểm (giá trị độ đo) của độ đo cosine giữa tài liệu và câu truy vấn càng nhỏ thì chúng càng liên quan đến nhau. Sau khi đã tính được độ liên quan giữa câu truy vấn và các tài liệu trong tập dữ liệu, chúng ta cần phải xếp hạng (sắp xếp thứ tự) các tài liệu theo mức độ liên quan theo thứ tự giảm dần một cách phù hợp. Điều này có nghĩa là những tài liệu liên quan nhất với câu truy vấn sẽ được đưa lên phía trên cùng của danh sách kết quả xếp hạng trả về, càng xuống phía dưới danh sách này thì mức độ liên quan sẽ giảm dần. Chẳng hạn, đối với độ đo log-odds và độ đo Cosine Similarity, xếp hạng các tài liệu theo thứ tự độ liên quan giảm dần đồng nghĩa với việc chúng ta cần với phải sắp xếp các tài liệu theo thứ tự giá trị cosine giảm dần. Còn đối với độ đo Euclidean

Distance, việc xếp hạng các tài liệu theo thứ tự độ liên quan giảm dần đồng nghĩa với việc chúng ta cần với phải sắp xếp các tài liệu theo thứ tự giá trị độ đo Euclidean Distance cosine tăng dần. Đây chính là điểm khác biệt chính giữa tìm kiếm trong cơ sở dữ liệu truyền thống và truy vấn thông tin. Sau khi đã tính được độ liên quan giữa câu truy vấn và các tài liệu trong tập dữ liệu, chúng ta cần phải xếp hạng kết quả truy vấn dựa trên độ tương đồng (mức độ liên quan), trong đó danh sách xếp hạng sẽ chứa các tài liệu trả về theo thứ tự độ tương đồng giảm dần (tức những tài liệu phù hợp nhất sẽ nằm phía trên cùng của danh sách kết quả trả về, càng xuống phía dưới danh sách này thì mức độ liên quan sẽ giảm dần.

CHƯƠNG II: CÁC MÔ HÌNH

1. Khái niệm Tf-idf

Tf-idf là viết tắt của cụm từ: Term frequence -Inverse document frequency . Đây là một kĩ thuật rất phổ biến, được dùng trong nhiều bài toàn NLP và khai phá dữ liệu dạng văn bản với mục đích: tính weight (độ quan trọng) của word trong một văn bản cụ thể, văn bản đó nằm trong một tập nhiều văn bản khác nhau. Bản thân tên gọi này đã thể hiện được nội dung thuật toán.

• Term frequence: tần xuất xuất hiện của từ w trong văn bản d

• Inverse document frequency: Nghịch đảo của '(số văn bản chứa từ w)/(tổng số văn bản)''.

1.1. TF

Giả sử:

- Có 1 tập D gồm M văn bản.
- Văn bản d ⊆⊆ D có m từ, từ vựng w xuất hiện c(w, d) lần.
- từ vựng w xuất hiện trong f(w, D) văn bản.

TF: Term frequence - tần xuất xuất hiện của w trong d. Một từ w xuất hiện trong văn bản d càng nhiều thì độ quan trọng của nó càng cao. Ta có:

$$Tf(w,d) = \frac{c(w,d)}{len(d)}$$

1.2. IDF

Lí tưởng của IDF: một từ xuất hiện trên càng nhiều văn bản, độ quan trọng từ đó càng giảm. VD các stop word như a, an, the, i .... xuất hiện trên hầu hết các văn bản, ngữ cảnh, độ quan trọng của những từ này thường rất thấp.

Định nghĩa công thức:

$$\text{IDF(w, D)} = \log \frac{M}{f(w,D)}$$

Vậy tf-idf của từ w trong văn bản d trên tập D là:

$$\text{TF-IDF(w, d, D)} = \frac{c(w,d)}{len(d)} * \frac{M}{f(w,D)}$$

2. BIM (Binary Independence Model)

2.1. Ý tưởng

Các vector biểu diễn tài liệu và truy vấn là các phân phối xác suất của các từ trong từ điển. Mỗi từ có một giá trị xác suất trong vector. Điểm xếp hạng của tài liệu sẽ bằng tổng các tích của xác suất của các từ trong truy vấn và xác suất của từng từ trong tài liệu.

Tần số của mỗi term chỉ có thể là 0 (không có) hoặc 1 (có).

2.2. Công thức

Ta có hai giả thuyết sau:

- Giả thuyết độc lập: Giả thiết này đơn giản hóa việc tính toán và khá hiệu quả.

$$p(d|q,r) \prod_{i \in [1,m]} p(td_i|q,r)$$

$$p(d|q,-r) \prod_{i \in [1,m]} p(td_i|q,-r)$$

- Các term của câu truy vấn là yếu tố duy nhất xác định sự liên quan giữa tài liệu và truy vấn:

Với term tdi$\notin$q thì p(tdi |q,r) không phụ thuộc vào r:

$$p(td_i|q,r) = p(td_i|q,-r)$$

➔ Chỉ cần tính xác suất các term trong truy vấn q

$$p(d|q,r)\prod_{td\in q} p(td|q,r)$$

$$p(d|q,-r)\prod_{td\in q} p(td|q,-r)$$

Dựa trên hai giả thiết trên, sự liên quan được thể hiện qua giá trị:

$$\prod_{td\in q}\frac{p(td|r,q)}{p(td|-r,q)}$$

Việc ước lượng giá trị p(td|r,q) và p(td|¬r,q) được thực hiện theo hai trường hợp:

- Trường hợp không có ngữ liệu mẫu:

  Độ liên quan giữa tài liệu và truy vấn:

  $$rel(d,q) = \prod_{td\in q}\frac{p(td|r,q)}{p(td|-r,q)} = \prod_{td\in q}\frac{0.5*N}{N_{td}}$$

  Sử dụng độ đo log-odds:

  $$rel(d,q) = \prod_{td\in q}\log\left(\frac{0.5*N}{N_{td}}\right)$$

  ➔ Trọng số của mỗi term là wtd=log(0.5*N/N_{td})

- Trường hợp có ngữ liệu mẫu

  r_{td} là số tài liệu liên quan chứa term td

  N_R là tổng số tài liệu liên quan

  Ước lượng các xác suất

$$p(td|r,q) = \frac{r_{td}}{N_R}$$

$$p(td|\text{-}r,q) = \frac{N_{td} - r_{td}}{N - N_R}$$

Để tránh trường hợp $r_{td}=0$ và $r_{td}=N_{td}$, thực hiện smoothing:

$$p(td|r,q) = \frac{r+0.5}{N_R+1}$$

$$p(td|\text{-}r,q) = \frac{N_{td} - r_{td}+0.5}{N - N_R+1}$$

Độ liên quan: Rel (d,q)

3. VSM (Vector Space Model)

3.1. Định nghĩa:

Là một mô hình Đại số biểu diễn các document (tài liệu) và các query (câu truy vấn) dưới dạng các vector, các document và query đều được biểu diễn bởi mô hình Bag-of-words, trong đó mỗi chiều của vector thể hiện trọng số (mức độ quan trọng) của một term tương ứng trong một document hay một query. Từ đó, xếp hạng các kết quả trả về dựa vào relevance (độ liên quan) giữa các vector document V(d) và vector query V(q). Mô hình Bag-of-words (túi từ) hay BoW, là một cách để rút trích đặc trưng từ văn bản. BoW biểu diễn một văn bản bằng các từ và tần suất xuất hiện của các từ đó. Nó được gọi là "túi" các từ, vì bất kỳ thông tin nào về thứ tự hoặc cấu trúc của các từ trong văn bản đều bị loại bỏ. Mô hình chỉ quan tâm đến việc các từ đã biết có xuất hiện trong tài liệu hay không, chứ không phải nằm ở đâu trong tài liệu.

3.2. Gán trọng số cho term (term weighting)

Trong một document hay một query thì luôn có những term chứa nhiều thông tin hơn những term khác, cũng có thể gọi là từ khóa, ngược lại thì cũng có những term sẽ không chứa nhiều thông tin (ví dụ như là các stopwords). Nên trong giai đoạn lập dictionary và xử lý query thì ta muốn những term quan trọng có trọng số cao, những term ít quan trọng sẽ có trọng số thấp.

- Term frequency (tf):

Một document d có term t xuất hiện nhiều lần, mà term t lại xuất hiện trong query q, tức là document d này ít nhiều có liên quan với query q hơn là những document không có hoặc có term t xuất hiện với số lần xuất hiện ít hơn document d. Do đó, những document như vậy nên có độ liên quan cao hơn những document khác. Ví dụ, một document có từ "car" xuất hiện 3 lần nên được trả về trước một document chỉ có từ "car" xuất hiện 1 lần đối với query "buying car". Để làm được việc này, chúng ta cần gán trọng số lớn cho những term xuất hiện nhiều lần trong một document bằng số lần xuất hiện (hay tần suất) của những term đó trong document đó. Gọi f(t, d) là số lần xuất hiện của term t trong document d, tf(t, d) sẽ được tính như sau:

$$tf(t, d) = f(t, d)$$

Nhưng đây lại là một ý tưởng rất xấu, giả sử nếu document d1 có 30 từ "car" và document d2 có 3 từ "car" thì không có nghĩa là d1 sẽ liên quan gấp 10 lần d2 với ví dụ trên. Do đó, cần phải điều chỉnh lại sao cho độ liên quan d1 vẫn cao hơn d2 nhưng không phải tuyến tính, điều chỉnh lại công thức tính tf(t, d) như sau:

$$tf(t,d) = \begin{cases} 1 + logf(t,d), \text{nếu } f(t,d) > 0 \\ 0, \text{ nếu } f(t,d) \leq 0 \end{cases}$$

Sử dụng 1 + logf(t, d) để tránh trường hợp tf(t, d) sẽ cho cùng một kết quả với f(t, d) = 1 và f(t, d) = 0 nếu chỉ dùng mỗi logf(t, d).

Với công thức trên, giá trị tf(t, d) ∈ [0; +∞).

- Inverse document frequency (idf):

Giá trị tf chỉ quan tâm đến tần suất của một term trong một document chứ không quan tâm đó là term gì, ví dụ có 1000 document, term "car" xuất hiện trong cả 1000 document thì đương nhiên term "car" trong query "buying car" sẽ không cung cấp được thông tin gì vào việc xếp hạng các document trả về, dù cho có vài document có rất nhiều từ "car" thì cũng không thể chắc rằng độ liên quan của những document này cao hơn những document khác,

bởi vì term “car” phổ biến đến nỗi nằm trong tất cả document, thay vào đó là chỉ có term “buying” góp phần vào việc xếp hạng. Nên cần phải điều chỉnh lại trọng số của các term. Ý tưởng của idf chính là những term xuất hiện trong nhiều document thì sẽ ít quan trọng hơn những term xuất hiện trong ít document. Nói cách khác, những term hiếm nên có giá trị cao hơn những term phổ biến. Với tổng số lượng document là N:

- df(t) là tần suất document của term t (số lượng document chứa term t)
- df(t) là độ đo nghịch đảo của mức độ thông tin của term t
- df(t) ∈ [1, N], do vocabulary V được tạo thành từ các term trong collection C, nên tất nhiên các term trong V sẽ nằm trong một document d ∈ C, từ đây suy ra được miền giá trị của df(t).

Công thức idf của term t được định nghĩa bởi:

$$idf(t) = log\left(\frac{N}{df(t)}\right)$$

idf(t) = 0 khi term t nằm trong tất cả document, do đó sẽ không có giá trị trong việc xếp hạng các kết quả trả về, đồng thời cũng giải quyết được việc có nên loại bỏ các stopwords hay không và nếu có thì nên loại những stopwords nào. Chúng ta muốn loại bỏ các stopwords bởi vì các chúng là những từ xuất hiện quá nhiều mà lại không cung cấp thông tin gì cho việc truy vấn. Khi một term t nằm trong tất cả document tức là term t đã không còn cung cấp thông tin gì cho việc truy vấn nên trọng số idf(t) = 0 đã trả lời cho những câu hỏi trên. Nhưng trong thực tế, không phải toàn bộ các document đều chứa stopword. Và dù sao thì nếu query có chứa stopword thì ít ra các văn bản có chứa stopword cũng có một độ liên quan nhất định tới query hơn là những document không chứa stopword. Vậy có nên loại bỏ stopword hay không, điều này còn phụ thuộc vào nhiều yếu tố.

Giá trị idf(t) của term t trong query được lấy từ idf(t) trong dictionary. Do idf là trọng số dùng để phân biệt độ quan trọng giữa các term trong vocabulary V. Như phần ý tưởng phía trên, tập hợp term T chỉ xuất hiện trong một số lượng nhỏ document D, tức là các term t ∈

T sẽ có trọng số lớn trong các document D (cũng có thể coi các term T như là từ khóa cho các document D). Từ đó nâng cao độ liên quan của các document D chứa những term t ∈ T với query nếu query cũng chứa những term t ∈ T; nên các term quan trọng trong các document D tức là cũng quan trọng trong query, nên việc lấy idf từ dictionary gán cho query cũng là hợp lý.

- Tf-idf:

Công thức tính tf- idf của một term t trong document d chỉ đơn giản là:

$$\text{tf-idf}(t,d) = \text{tf}(t,d) \times \text{idf}(t)$$

Trọng số tf- idf(t, d) sẽ:

- Rất lớn khi term t xuất hiện rất nhiều lần trong một số lượng nhỏ document (do đó làm tăng độ liên quan của những document này với query khi query có chứa term t).
- Thấp hơn khi term t xuất hiện ít lần trong một document, hoặc xuất hiện trong nhiều document (do đó làm giảm độ liên quan của document với query khi query có chứa term t).
- Thấp nhất khi term t xuất hiện trong tất cả document.

Cần lưu ý: tf(t, d) ∈ [0; +∞) và idf(t) ∈ [0; log(N)], nên giá trị của tf-idf(t, d) sẽ luôn nằm trong khoảng [0; +∞).

3.3. Cosine similarity

Cách tiêu chuẩn để định lượng similarity giữa 2 vector là dùng Cosine similarity, chỉ đơn giản là tính cos($\vec{q}$, $\vec{d}$), công thức tính cos($\vec{q}$, $\vec{d}$) được biểu diễn bởi:

$$cos(\vec{q},\vec{d}) = \frac{\vec{q}\cdot\vec{d}}{|\vec{q}||\vec{d}|} = \frac{\vec{q}}{|\vec{q}|}\cdot\frac{\vec{d}}{|\vec{d}|} = \frac{\sum_{i=1}^{|V|} q_i d_i}{\sqrt{\sum_{i=1}^{|V|} q_i^2}\sqrt{\sum_{i=1}^{|V|} d_i^2}}$$

2 vector nằm gần nhau hay nói cách khác là góc giữa 2 vector này nhỏ, góc càng nhỏ thì cos($\vec{q}$, $\vec{d}$) càng lớn, nên độ liên quan giữa 2 vector sẽ tỉ lệ thuận với cos($\vec{q}$, $\vec{d}$) và nằm trong khoảng [−1; 1].

Sau khi đã tính toán xong cos cho từng cặp document-query, chúng ta chỉ đơn giản là trả về top K kết quả cho người dùng.

3.4. Chuẩn hóa

Khi tách tích vô hướng và tích độ dài trong công thức cos thì phần thương $\frac{\vec{x}}{|\vec{x}|}$ chính là chúng ta đang normalize vector (chuẩn hóa vector hay còn gọi là chuyển về vector đơn vị). Số chiều của mỗi vector là |V|, công thức tính độ dài của các vector biểu diễn các document trong collection C:

$$|\vec{x}| = \sqrt{\sum_{i=1}^{|V|} x_i^2}$$

Sau khi đã có độ dài thì chúng ta chỉ cần chia vector cho độ dài của nó là đã có thể chuẩn hóa vector (chuyển về thành vector đơn vị).

Đối với vector query, chúng ta phải đợi người dùng nhập query vào thì mới có thể chuẩn hóa vector query được, nhưng chúng ta có thể chuẩn hóa các vector document khi đã tính xong tf- idf. Và khi người dùng nhập query vào, tạo vector query, vector query sẽ được chuẩn hóa thì cos($\vec{q}$, $\vec{d}$) chỉ còn là:

$$\cos(\vec{q}, \vec{d}) = \vec{q} \cdot \vec{d} = \sum_{i=1}^{|V|} q_i d_i$$

với term $t \in q \cap d$

Bằng cách chuẩn hóa các vector document từ trước, chúng ta đã giảm một lượng đáng kể thời gian và chi phí tính toán khi người dùng truy vấn.

3.5. Ưu điểm và nhược điểm

Ưu điểm:

• Dễ hiểu và dễ cài đặt.

• Đã được nghiên cứu từ lâu và thực nghiệm cho kết quả tốt và khả thi.

• Hiện tại là phương pháp được sử dụng rộng rãi.

• Có nhiều công thức tính TF.IDF khác nhau.

Nhược điểm:

• Để cho kết quả đúng đắn, ta giả định:

  o Các câu là độc lập với nhau.

  o Các truy vấn và tài liệu cùng loại với nhau.

• Có nhiều tham số để hiệu chỉnh:

  o Bộ từ điển.

  o Tham số trong các hàm chuẩn hoá.

  o Threshold để chọn ra top kết quả.

CHƯƠNG III: TẬP NGỮ LIỆU

Cranfield là một tập tài liệu rất nổi tiếng và có tầm ảnh hưởng trong lĩnh vực truy xuất thông tin, bản thân tập tài liệu này được tạo thành từ một chuỗi các nghiên cứu thực nghiệm trong truy xuất thông tin được thực hiện bởi Cyril W.Cleverdon ở đại học hàng không, ngày nay được biết đến với cái tên đại học Cranfield và bản thân chuỗi nghiên cứu này cũng được gọi là nghiên cứu Cranfield. Nghiên cứu này được thực hiện vào năm 1960 với mục đích là để đánh giá tính hiệu quả của hệ thống chỉ mục (indexing systems), nghiên cứu được chia làm thành 2 phần chính. Ở phần đầu tiên, một số phương pháp index hiện có được đem ra so sánh để kiểm tra hiệu quả của chúng. Các truy vấn được tạo ra bởi các tác giả của các bài báo trong collection và sau đó được các chuyên gia trong hệ thống đó chuyển thành chỉ mục tra cứu (index lookups). Trong loạt chuỗi nghiên cứu đầu tiên này, một phương pháp đã phát triển từ kém hiệu quả nhất thành hiệu quả nhất sau khi thực hiện những thay đổi nhỏ đối với cách sắp xếp dữ liệu được ghi trên thẻ chỉ mục (index cards). Kết luận sau phần đầu tiên này dường như là phương pháp luận sẽ ít quan trọng hơn là chi tiết cụ thể của việc triển khai nó. Điều này cũng đã dẫn đến cuộc tranh luận lớn về phương pháp luận của nghiên cứu. Sự phê bình trên cũng đã dẫn đến loạt phần thứ nghiệm thứ hai. hiện được gọi là Cranfield 2. Cranfield 2 đã cố gắng thu thập thêm thông tin chi tiết bằng cách đảo ngược phương pháp. Nếu Cranfield 1 đã kiểm tra khả năng mà các chuyên gia có thể tìm thấy được nguồn tài nguyên cụ thể (specific resource) dựa vào hệ thống chỉ mục thì ở Cranfield 2 lại nghiên cứu kết quả đặt câu hỏi bằng ngôn ngữ của con người để xem liệu hệ thống chỉ mục có cung cấp câu trả lời phù hợp hay không, bất kể đó có phải là tài liệu mục tiêu ban đầu của nó hay không. Với phần kết của Cranfield 2 được hoàn thành vào năm 1967, toàn bộ kho văn bản (corpus) đã được published thành dạng machine-readable. Và ngày nay nó được biết đến là tập tài liệu Cranfield hay còn gọi là Cranfield 1400. Cái tên này đề cập đến số lượng tài liệu trong collection, bao gồm 1398 abstracts. Collection này cũng bao gồm 225 câu truy vấn và đánh giá mức độ của tất cả các cặp tài liệu-truy vấn có được từ các lần chạy thử nghiệm. Database chính của abstracts là khoảng 1.6MB. Toàn bộ collection of abstracts, chỉ mục kết quả (resulting indexes) và kết quả sau đó đều được phân phối ở định dạng điện tử và được sử dụng rộng rãi trong nhiều thập kỉ.

CHƯƠNG IV: PHƯƠNG PHÁP GIẢI QUYẾT BÀI TOÁN

1. Xây dựng từ điển

Chúng ta đã bàn về các thao tác xử lý dữ liệu ở phần trên. Bây giờ, chúng ta sẽ tiến hành xây dựng dictionary từ bước đầu cho tới bước cuối cùng, nghĩa là từ khi trích xuất các file .txt có trong tập Cranfield cho tới khi tạo được dictionary cho tất cả các term trong bộ dữ liệu. Ở đây nhóm sẽ tiến hành xây dựng từ điển bằng hai cách khác nhau: 1. Bước lập chỉ mục và xây dựng từ điển được tách ra làm hai phần (có nghĩa là lập chỉ mục trước sau đó lấy những thông tin cần thiết trong chỉ mục để xây dựng từ điển, sử dụng cho mô hình BIM với trường hợp không có ngữ liệu mẫu), 2. Lập chỉ mục và xây dựng từ điển xuyên suốt trong một quá trình (có nghĩa là xuyên suốt quá trình xây dựng từ điển, mọi thay đổi đều chỉ sử dụng trên một từ điển duy nhất và tiến hành cập nhật lại những thay đổi trực tiếp ở trên đó, sử dụng cho mô hình VSM). Mỗi cách xây dựng từ điển sẽ sử dụng cho mỗi mô hình khác nhau. Bây giờ chúng ta sẽ tiến hành xây dựng từ điển theo thứ tự sau:

• Giải nén và đọc thư mục chứa các file .txt (thư mục Crandfield) (đây là bước hệ thống nhận vào các document).

• Tiền xử lý dữ liệu có trong các file .txt (tách từ, xóa stopword, stemming, lemma).

1.1. Lập chỉ mục và xây dựng từ điển được tách ra làm hai phần

Lập chỉ mục bằng cách tính số lần xuất hiện của term trong toàn bộ tập tài liệu (frequency), tính số lượng tài liệu chứa term đó (quantity) đồng thời trích xuất chỉ số tài liệu chứa term đó (index_doc) và lưu lại trong một mảng.

Ví dụ: Chỉ mục sau khi được tạo sẽ có cấu trúc như sau:

([‘abl’, ‘accomplish’, ‘achiev’, ‘across’,….], [11, 2, 1, 2, …], [10, 2, 1, 2, …], [[8, 15, 18, 19, 20, 22, 23, 25, 28, 29], [2, 3], [21], [4, 30],….])

Trong đó chỉ mục được lưu lần lượt từ trái sang phải là words (terms), tần số (frequency), số lượng tài liệu chứa term (quantity), chỉ số tài liệu chứa term (index_doc).

Truy xuất chỉ mục lấy danh sách words, quantity và index_doc để xây dựng dictionary.

Ví dụ: Từ điển sau khi được tạo sẽ có cấu trúc như sau:

```
{'abl': {'tq|r,q': 0.5,
         'td|-r,q': 0.3333333333333333,
         'weight': {8: 0.17609125905568124,
                    15: 0.17609125905568124,
                    …}},
 'accomplish': {'tq|r,q': 0.5,
                'td|-r,q': 0.06666666666666667,
                'weight': {2: 0.8750612633916999,
                           3: 0.8750612633916999}},
 …}
```

1.2. Lập chỉ mục và xây dựng từ điển xuyên suốt trong một quá trình

Đánh số cho các file bằng docID để lập chỉ mục.

Lấy lời bài hát từ các document (như đã nói ở trên) và tiền xử lý chúng, sau đó tạo ra một danh sách gồm token-docID và xếp thứ tự danh sách này theo chữ cái, chữ số (giảm dần); chữ số ưu tiên hơn chữ cái, nếu chữ số bằng nhau thì xem xét chữ cái, nếu chữ cái bằng nhau thì so sánh docID.

Từ danh sách token-docID ở trên, tạo ra dictionary. Dùng các công thức ở chương 2 để tính các trọng số $tf(t, d)$, $idf(t)$ và $tf\text{-}idf(t, d)$ cho các term.

Biến đổi (chuyển về vector có độ dài bằng 1) các vector document.

Phần này chỉ tóm tắt lại các bước đã làm ở trên, đã được điều chỉnh cho phù hợp với dữ liệu.

Ví dụ: Từ điển sau khi được tạo sẽ có cấu trúc như sau:

```
{abl: {idf: 0.0379,
       id tf-idf: {0: 0.0061,
                   1: 0.0067,
                   2: 0.0026,
                  ...}},
 accomplish: {idf: 0.7675,
              id tf-idf: {19: 0.041,
                          20: 0.1071,
                          25: 0.0605,
                          ...}},
 ...}
```

Đúng như lý thuyết ở trên, với mỗi term sẽ lưu một giá trị idf và một loạt các cặp id tf-idf. Ví dụ ở trên là tất cả các vector document đã được normalize.

2. Xử lý câu truy vấn

Tới bước này là chúng ta đã tạo xong từ điển. Bây giờ chúng ta sẽ tiến hành xử lý truy vấn theo thứ tự sau:

2.1. Đối với BIM

Tiền xử lý truy vấn (sử dụng cùng một phương pháp tiền xử lý đã được áp dụng với các tài liệu).

Liệt kê tất cả các term riêng biệt trong câu truy vấn.

2.2. Đối với VSM

Tiền xử lý truy vấn (sử dụng cùng một phương pháp tiền xử lý đã được áp dụng với các tài liệu). Loại bỏ các tokens không có trong từ điển.

Lập chỉ mục các token.

Thống kê tf của từng term.

Tính trọng số cho các term.

- Tính $tf(t, q)$.
- Giá trị $idf(t)$ của term trong query được lấy từ term tương ứng trong dictionary.
- Tính $tf\text{-}idf(t, q)$.

Chuẩn hóa (chuyển về vector đơn vị) vector query.

Phần chi tiết về các thao tác trên đã được trình bày ở trên, tại đây chỉ liệt kê lại các thao tác này cụ thể hơn.

3. Các mô hình đề xuất

Chúng em đã nghiên cứu và chọn một số phương pháp truy xuất thông tin phổ biến để áp dụng cho bài toán này, dựa trên sơ cở mô hình truy xuất thông tin Binary Independence và Vector Space. Chúng em đã kết hợp các phương pháp này để tạo ra các mô hình truy xuất thông tin phù hợp cho bài toán như sau. Với mô hình Binary Independence, chúng em sử dụng một phương pháp rút gọn từ và phương pháp tính trọng số của từ là log-odds. Với mô hình Vector Space, chúng em sử dụng một phương pháp rút gọn từ và phương pháp tính trọng số của từ là TF-IDF. Tất cả các mô hình đều loại bỏ stopword. Với các mô hình dựa trên Vector Space, chúng em sử dụng công thức Cosine Similarity để tính độ tương đồng. Các mô hình này đều được xây dựng trên nền tảng của Binary Independence hoặc Vector Space và kết hợp các phương pháp như sau:

Binary Independence stemming với PorterStemmer.

Vector Space stemming với PorterStemmer.

Chúng em có tổng cộng 2 mô hình truy xuất thông tin để thử nghiệm cho bài toán. Bảng sau đây thống kê các mô hình truy xuất thông tin do nhóm đề xuất. Chúng em đã cài đặt thử nghiệm 2 mô hình này và so sánh kết quả thực nghiệm về thời gian cài đặt và thời gian chạy để có thể cải thiện hiệu quả của chúng. Chúng em đã chọn ra mô hình tốt nhất và hiệu quả nhất của từng loại Binary Independence và Vector Space để so sánh với kết quả query khi dùng thư viện Whoosh với mô hình BM25 (có sẵn trong thư viện Whoosh).

4. Các phương pháp áp dụng

Chúng em đã nói về các thao tác xử lý như Term selection (các thao tác xử lý từ ngữ, bao gồm loại bỏ stopwords và rút gọn từ), Term weighting (tính trọng số của từ) và Similarity measure (tính mức độ liên quan giữa các tài liệu và câu truy vấn) ở trên.

Trong phần này, chúng em chỉ nêu ra các phương pháp cụ thể của các thao tác trên mà chúng em dùng để xây dựng các mô hình đã đề xuất ở trên:

- Phương pháp tách từ dùng hàm word_tokenize của thư viện NLTK. Loại bỏ những stopwords trong danh sách stopwords mặc định của thư viện NLTK.
- Phương pháp rút gọn từ dùng PorterStemmer.
- Hai phương pháp tính độ tương đồng (mức độ liên quan) giữa các tài liệu và câu truy vấn cho các mô hình dựa trên hai mô hình cơ sở là mô hình Binary Independence và Vector Space là dùng công thức Cosine Similarity và dùng độ đo Log-odds.

4.1. Phương pháp tách từ

4.1.1. Tách từ sử dụng word_tokenize

4.1.1.1. Đặc điểm và chức năng

Dùng các ký tự khoảng trắng, dấu xuống dòng và tab để tách các từ.

4.1.1.2. Lý do chọn phương pháp

Tiếng Anh chủ yếu có các từ đơn, mỗi từ đơn là một từ khác biệt. Vì vậy, các từ trong câu/văn bản thường cách nhau bởi các ký tự khoảng trắng, dấu xuống dòng và tab. Chúng em chọn phương pháp này vì nó thích hợp với đặc điểm này của từ vựng trong tiếng Anh.

Ví dụ 1:

Khi tách từ cho câu tiếng Anh: "Firm \trelated\nfactors" bằng hàm word_tokenize, chúng ta sẽ thu được kết quả ['Firm', 'related', 'factors'].

Ví dụ 2:

Khi tách từ cho câu tiếng anh: "This is a\n\ttext" bằng hàm word_tokenize, chúng ta sẽ thu được kết quả là ['This', 'is', 'a', 'text'].

4.2. Các phương pháp chọn Term

4.2.1. Phương pháp loại bỏ các stopwords nằm trong danh sách stopwords mặc định của thư viện NLTK dành cho tiếng Anh

4.2.1.1. Lý do lựa chọn phương pháp

Vì tập dữ liệu Cranfield mà chúng em dùng cho bài toán là tập dữ liệu tiếng Anh và vẫn có nhiễu, nên chúng em chọn dùng danh sách stopwords mặc định của thư viện NLTK cho tiếng Anh và thêm một số từ vào danh sách stopwords để loại bỏ. Lý do chúng em chọn danh sách stopwords này là vì nó khá đầy đủ, đa dạng, hay được dùng trong các hệ thống truy vấn thông tin và phù hợp với mục tiêu của bài toán là xử lý các bài hát tiếng Anh.

4.2.1.2. Danh sách stopwords mặc định của thư viện NLTK dành cho tiếng Anh

Danh sách stopwords mặc định của thư viện NLTK dành cho tiếng Anh bao gồm 179 stopwords: 'i', 'me', 'my', 'myself', 'we', 'our', 'ours', 'ourselves', 'you', "you're", "you've", "you'll", "you'd", 'your', 'yours', 'yourself', 'yourselves', 'he', 'him', 'his', 'himself', 'she', "she's", 'her', 'hers', 'herself', 'it', "it's", 'its', 'itself', 'they', 'them', 'their', 'theirs', 'themselves', 'what', 'which', 'who', 'whom', 'this', 'that', "that'll", 'these', 'those', 'am', 'is', 'are', 'was',

'were', 'be', 'been', 'being', 'have', 'has', 'had', 'having', 'do', 'does', 'did', 'doing', 'a', 'an', 'the', 'and', 'but', 'if', 'or', 'because', 'as', 'until', 'while', 'of', 'at', 'by', 'for', 'with', 'about', 'against', 'between', 'into', 'through', 'during', 'before', 'after', 'above', 'below', 'to', 'from', 'up', 'down', 'in', 'out', 'on', 'off', 'over', 'under', 'again', 'further', 'then', 'once', 'here', 'there', 'when', 'where', 'why', 'how', 'all', 'any', 'both', 'each', 'few', 'more', 'most', 'other', 'some', 'such', 'no', 'nor', 'not', 'only', 'own', 'same', 'so', 'than', 'too', 'very', 's', 't', 'can', 'will', 'just', 'don', "don't", 'should', "should've", 'now', 'd', 'll', 'm', 'o', 're', 've', 'y', 'ain', 'aren', "aren't", 'couldn', "couldn't", 'didn', "didn't", 'doesn', "doesn't", 'hadn', "hadn't", 'hasn', "hasn't", 'haven', "haven't", 'isn', "isn't", 'ma', 'mightn', "mightn't", 'mustn', "mustn't", 'needn', "needn't", 'shan', "shan't", 'shouldn', "shouldn't", 'wasn', "wasn't", 'weren', "weren't", 'won', "won't", 'wouldn', "wouldn't".

4.2.1.3. Quy trình thực hiện

Chúng ta sẽ so sánh các từ trong câu/văn bản với các stopwords trong danh sách kia. Nếu trùng, chúng ta sẽ bỏ các stopwords đó. Nếu không, chúng ta sẽ để nguyên từ đó.

4.2.1.4. Ưu điểm của việc loại bỏ các stopwords

Bằng cách bỏ các stopwords, chúng ta có thể giảm được nhiều tài nguyên bộ nhớ để lưu trữ các stopwords và thời gian xử lý các văn bản có chứa các stopwords, do đó giúp tiết kiệm được nhiều chi phí, công sức và thời gian cho các việc như lưu trữ và xử lý các stopwords.

Ví dụ 1

Cho câu văn tiếng Anh: “this will serve to ensure that starbucks is able to remain relevant in the coffee industry”. Các stopwords có trong câu bao gồm “this”, “will”, “to”, “that”, “is”, “in”, “the”. Sau khi tách từ và loại bỏ các stopwords có trong câu, chúng ta thu được các tokens của câu bao gồm 'serve', 'ensure', 'starbucks', 'able', 'remain', 'relevant', 'coffee', 'industry'.

Ví dụ 2

Cho câu văn tiếng Anh: “an innovator is a person who develops new ideas and looks for efficient ways of accomplishing a certain task.”. Các stopwords có trong câu bao gồm ‘an’,

‘is’, ‘a’, ‘who’, ‘and’, ‘for’, ‘of’. Sau khi tách từ và loại bỏ các stopwords có trong câu, chúng ta thu được các tokens của câu bao gồm 'innovator', 'person', 'develops', 'new', 'ideas', 'looks', 'efficient', 'ways', 'accomplishing', 'certain', 'task', '.'.

4.2.2. Các phương pháp rút gọn từ

4.2.2.1. Rút gọn từ bằng phương pháp Stemming sử dụng PorterStemmer

4.2.2.1.1. Giới thiệu chung

PorterStemmer là công cụ Stemming cổ nhất được tạo ra từ năm 1979. Đây là một trong những công cụ Stemming hay được dùng nhất.

4.2.2.1.2. Đặc điểm và chức năng

PorterStemmer là công cụ stemming từ bằng cách cắt bỏ các ký tự cuối (hậu tố) của từ đó. Ví dụ, PorterStemmer cho ra dạng từ gốc của từ “cats” là “cat” bằng cách đơn giản bỏ ký tự “-s” ở cuối từ “cats”, là hậu tố của từ. Hậu tố này được thêm vào từ gốc để tạo ra dạng biến thể là danh từ số nhiều của từ gốc (“cat” thêm hậu tố “-s” thành “cats” để chỉ “cat” ở số nhiều). Nhưng nếu xét các từ “trouble”, “troubling” và “troubled” thì chúng đều có kết quả stemming là “troubl” vì PorterStemmer không theo các quy tắc của ngôn ngữ học, mà theo bộ quy tắc riêng cho các trường hợp khác nhau được áp dụng theo từng giai đoạn. Đây là lý do tại sao PorterStemmer không luôn cho ra kết quả stemming là từ tiếng Anh có nghĩa. Nó không có một bảng tra cứu các từ gốc có nghĩa mà áp dụng các quy tắc rút gọn dựa trên các luật rút gọn.

Từ thu được có thể không tồn tại/không có trong từ điển. Có thể làm mất ý nghĩa của từ được rút gọn.

PorterStemmer thực hiện rút gọn từ dựa trên các luật biến đổi.

Ưu điểm của công cụ này là từ được rút gọn ở mức độ vừa đủ, không làm mất đi hoàn toàn khả năng nhận diện từ. PorterStemmer được nổi tiếng với sự đơn giản và nhanh chóng.

Thường có ích trong các mô hình truy vấn thông tin dùng vector. Các tài liệu trong tập dữ liệu sẽ được biểu diễn dưới dạng các vector của các từ hoặc thuật ngữ. Các từ cùng gốc sẽ có ý nghĩa tương tự. Ví dụ, chúng ta có từ gốc là "connect", một số biến thể của nó như "connections", "connected", "connecting" và "connection".

PorterStemmer thường được dùng trong lĩnh vực Khai thác dữ liệu và Truy vấn thông tin. Tuy nhiên, các ứng dụng của nó thường chỉ hạn chế trong phạm vi các từ tiếng Anh và trong các trường hợp kết quả stemming không cần thiết phải là một từ có nghĩa, có tồn tại.

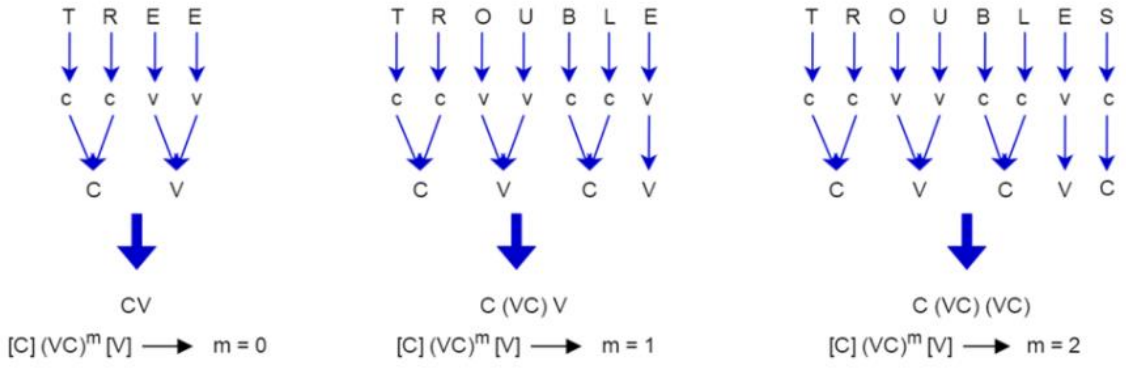
4.2.2.1.3. Các luật biến đổi của PorterStemmer

Các luật biến đổi heuristic dựa trên các quy tắc biến đổi hình thái bằng cách thêm các tiền tố và hậu tố. Các luật này được áp dụng lên từ cần phân tích theo một trình tự cố định cho đến khi không thể áp dụng nữa. Kết quả thu được là cụm 102 ký hiệu được coi là gốc từ nhưng không thể xác định được từ loại. Kết quả này có thể không phải là một từ có nghĩa, tức không có trong từ điển.

Một số kiến thức liên quan về ngôn ngữ học như sau:

Nguyên âm (ký hiệu là V) gồm các ký tự A, E, I, O, U và Y (nếu Y liền sau một phụ âm).
Phụ âm (ký hiệu là C) gồm các ký tự còn lại và Y (nếu Y không liền sau một phụ âm).

Các từ trong tiếng Anh có dạng: $[C](VC)^m[V]$.



Ví dụ:

Với m = 0: Ta có các từ như tree, tr, ee, y, by,…

Với m = 1: Ta có các từ như trouble, oats, trees, ivy,…

Với m = 2: Ta có các từ như troubles, private, oaten, robbery,…

Các quy tắc thay đổi hậu tố $S1$ thành $S2$ có dạng: $(condition)S1 \rightarrow S2$. Điều này có nghĩa là nếu một từ kết thúc bằng hậu tố $S1$ và gốc trước $S1$ thỏa mãn điều kiện đã cho, thì $S1$ được thay thế bằng $S2$.

Ví dụ:

Với m > 1, ta sẽ biến đổi "-EMENT" thành "", tức là bỏ đi hậu tố "-EMENT". Chẳng hạn như "replacement" sẽ biến thành "replac".

Các ký hiệu được sử dụng trong condition:

$*\ S$: tận cùng là một ký tự S nào đó.

$*\ v\ *$: có một nguyên âm.

$*\ d$: tận cùng là hai phụ âm (ss, ck, …).

$*\ o$: tận cùng là cvc với c thứ hai không là x, y, w.

Phần điều kiện có thể là các biểu thức có chứa các phép toán của Đại số Boolean như AND, OR, NOT. Chẳng hạn như biểu thức ($m > 1\ AND\ (*\ S\ OR\ *\ T)$) được sử dụng để thực hiện stemming với m > 1 và các từ tận cùng bằng S hoặc T. Hoặc, biểu thức ($*\ d\ AND\ NOT\ (*\ L\ OR\ *\ S\ OR\ *\ Z)$) được sử dụng để thực hiện stemming với các từ tận dùng là hai phụ âm khác L, S và Z. Các điều kiện phức tạp như thế này hiếm khi được yêu cầu sử dụng.

Trong một tập hợp các quy tắc được viết bên dưới nhau, chỉ một quy tắc được tuân theo và đây sẽ là quy tắc có S1 khớp dài nhất cho từ đã cho. Ví dụ, với

"-SSES" → "-SS"

"-IES" → "-I"

“-SS” → “-SS”

“-S” → “”

Từ “CARESSES” được chuyển thành với từ “CARESS” bởi vì “-SSES” là phần S1 có so khớp dài nhất. Đối với “CARESS” chuyển thành “CARESS” (S1 = “-SS”) và “CARES” được chuyển thành “CARE” (S1 = “-S”) thì độ dài so khớp là bằng nhau.

Sau đây là một số luật rút gọn từ được PorterStemmer sử dụng:

Luật biến đổi	Ví dụ
“-SSES” -> “-SS”	“caresses” -> “caress”
“-IES” -> “-I”	“ladies” -> “ladi”
“-SS” -> “-SS”	“pass” -> “pass”
“-S” -> “”	“dogs” -> “dog”
(m>0) “-ATIONAL” -> “-ATE”	“relational” -> “relate”
(m>0) “-TIONAL” -> “-TION”	“conditional” -> “condition”
(m>0) “-FULNESS” -> “-FUL”	“helpfulness” -> “helpful”

Một số luật rút gọn từ được PorterStemmer sử dụng

Từ ban đầu	Từ thu được sau khi sử dụng PorterStemmer
helpful	help
trouble	troubl
football	footbal
unstoppable	unstopp
dogs	dog
relationship	relationship
shipping	ship

Một số ví dụ sau khi sử dụng PorterStemmer

4.2.2.1.4. Lý do lựa chọn phương pháp

Chúng ta cùng quan sát bảng so sánh kết quả rút gọn một số từ bằng hai công cụ PorterStemmer và LancasterStemmer dưới đây:

Từ ban đầu	Từ sau khi rút gọn bằng stemming	
	PorterStemmer	LancasterStemmer
friendship	friendship	friend
destabilize	destabil	des
normalization	normal	norm
aging	age	chem
family	famili	ag
chemistry	chemistri	famy
famous	famou	fam

So sánh kết quả rút gọn một số từ bằng hai công cụ PorterStemmer và LancasterStemmer

Xét các kết quả stemming từ trong bảng trên, chúng ta thấy kết quả stemming bằng cách dùng công cụ PorterStemmer có chất lượng cao hơn, từ được stemming ở mức độ vừa đủ, không làm mất đi hoàn toàn khả năng nhận diện từ.

Ví dụ, khi stemming từ "friendship", khi dùng công cụ PorterStemmer và LancasterStemmer chúng ta có kết quả stemming từ lần lượt là "friendship" và "friend". Với kết quả của PorterStemmer là "friendship" vẫn giữ nguyên ý nghĩa so với từ gốc. Còn với kết quả của LancasterStemmer là "friend" đã mất đi ý nghĩa và chúng ta có khó thể biết được ý nghĩa của từ gốc là gì, vì từ "friend" có thể là dạng stemming của các từ như "friendship", "friendzone", "friendless", "friendlessness", "friends", "unfriend", "befriend",… Từ "friend" có rất nhiều biến thể như các từ vừa kể và từ "friendship" là một trong số đó. Từ từ "friend", chúng ta khó có thể biết được từ gốc trước khi stemming là từ "friendship". Do đó, khi xem kết quả stemming từ bằng cách dùng công cụ PorterStemmer, chúng ta có thể biết được ý nghĩa của từ gốc tương ứng trước khi stemming.

Dựa trên việc nghiên cứu và tìm hiểu các ưu điểm và chất lượng kết quả stemming từ của các công cụ Stemming, chúng em thấy công cụ PorterStemmer cho ra kết quả stemming từ khá chính xác, không bị lệch nhiều. Xem kết quả stemming từ bằng cách dùng công cụ PorterStemmer, chúng ta có thể biết được ý nghĩa của từ gốc tương ứng trước khi stemming.

4.3. Phương pháp tính trọng số Term

4.3.1. Tính trọng số của từ bằng công thức TF-IDF

Thực tế là có rất nhiều cách để tính trọng số cho term nhưng nhóm chỉ chọn ra 2 công thức phổ biến và được sử dụng trong chương trình học, đó là:

$$tf(t,\ d) = \begin{cases} 1 + logf(t,d), nếu\ f(t,d) > 0 \\ 0, nếu\ f(t,d)\ \leq 0 \end{cases}$$

Và

$$idf(t) = log\ (\frac{N}{df(t)})$$

Term frequency		Document frequency	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1
l (logarithm)	$1 + \log(\mathrm{tf}_{t,d})$	t (idf)	$\log \frac{N}{\mathrm{df}_t}$
a (augmented)	$0.5 + \frac{0.5 \times \mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0, \log \frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$
b (boolean)	$\begin{cases} 1 & \text{if } \mathrm{tf}_{t,d} > 0 \\ 0 & \text{otherwise} \end{cases}$		
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t \in d}(\mathrm{tf}_{t,d}))}$		

Một số công thức tính tf và idf

Khi chúng ta đã tính được 2 trọng số trên cho term thì kết hợp chúng lại bằng công thức sau:

$$tf\text{-}\ idf(t,\ d) = tf(t,\ d) \times idf(t)$$

4.3.2. Tính trọng số của từ sử dụng độ đo log-odds

Trọng số của term được tính bằng công thức sau:

$$W_{td} = \log(\frac{0.5*N}{N_{td}})$$

4.4. Các phương pháp tính độ liên quan (Similarity measure)

Hai phương pháp tính độ liên quan dựa trên độ tương đồng giữa 2 vector là Cosine similarity và phương pháp dùng độ đo log-odds.

4.4.1. Cosine similarity

Phương pháp Cosine Similarity được biểu diễn bởi công thức sau:

$$\cos(\vec{q}, \vec{d}) = \vec{q}.\vec{d} = \sum_{i=1}^{|V|} q_i d_i$$

4.4.2. Độ đo log-odds

$$rel(d,q) = \sum_{td \in q} \log(\frac{0.5*N}{N_{td}})$$

5. Phương pháp đánh giá mô hình

5.1. Phương pháp

Để đánh giá cho các mô hình đề xuất cho bài toán, nhóm đã dùng một số phương pháp dành cho mô hình truy xuất thông tin như Precision, Recall để tính Averave Precision (AP: độ chính xác trung bình) và từ đó tính được mean Average Precision (MAP) và từ Precision và Recall để tính Interpolation Precision (độ chính xác nội suy) theo bộ TREC 11 điểm để tính Interpolation Average Precision (độ chính xác nội suy trung bình)và từ đó tính MAP nội suy để đánh giá bài toán, ngoài ra nhóm còn đánh giá qua thời gian chạy của các mô hình đề xuất.

Precision (P)

Precision đo lường tỉ lệ giữa số lượng các tài liệu liên quan được trả về và tổng số lượng các tài liệu được trả về. Đây là một độ đo cho biết bao nhiêu phần trăm của các tài liệu được trả về là chính xác và liên quan đến truy vấn. Là độ đo được tính bằng tỉ lệ số lượng kết quả có liên quan trên tổng số lượng kết quả trả về.

Precision được tính bằng công thức sau:

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

Recall (R)

Recall đo lường tỉ lệ giữa số lượng các tài liệu liên quan được trả về và tổng số lượng các tài liệu liên quan. Đây là một độ đo cho biết bao nhiêu phần trăm các tài liệu liên quan đã được tìm ra bởi hệ thống tìm kiếm. Là độ đo được tính bằng tỉ lệ số lượng kết quả có liên quan trên tổng số lượng kết quả có liên quan trong tập test.

R được tính bằng công thức sau:

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

Interpolation Precision (Precision nội suy)

Interpolation precision là một phương pháp đánh giá precision trong truy xuất thông tin. Nó đo lường độ chính xác của các kết quả truy xuất tìm kiếm theo thứ tự, bằng cách tính precision ở mỗi điểm giao nhau trên đồ thị precision-recall và chọn giá trị cao nhất. Phương pháp này được dùng để đánh giá hiệu quả của hệ thống tìm kiếm trong các bài kiểm tra đánh giá truy xuất thông tin.

MAP

MAP là viết tắt của Mean Average Precision, là một độ đo quan trọng trong truy xuất thông tin được dùng để đánh giá hiệu quả của một hệ thống tìm kiếm. MAP tính toán trung bình của các giá trị Average Precision (AP) cho tất cả các truy vấn trong tập dữ liệu. Để tính toán AP cho một truy vấn, ta sẽ tính toán precision ở mỗi điểm giao nhau trên đồ thị precision-recall, sau đó tính toán diện tích dưới đường cong precision-recall của truy vấn đó. AP biểu diễn cho độ chính xác của các kết quả tìm kiếm của một truy vấn riêng biệt.

MAP nội suy

MAP nội suy (Interpolated Mean Averave Precision) là một biến thể của Mean Average Precision (MAP) trong truy xuất thông tin. Được đưa ra bởi người viết tài liệu thông tin Karen Sparck Jones, MAP nội suy là một cách để tính toán MAP bằng cách dùng nội suy tuyến tính trên đồ thị precision-recall. Thay vì tính toán AP bằng cách lấy giá trị precision tại các điểm giao nhau trên đồ thị precision-recall, MAP nội suy dùng một phương pháp nội suy để tính toán precision tại các điểm recall bị thiếu. Phương pháp nội suy này cho phép tính toán giá trị precision của một mức recall bằng cách nội suy tuyến tính giữa các mức recall lân cận có giá trị precision đã được tính toán. Việc dùng MAP nội suy giúp đạt được độ chính xác cao hơn khi đánh giá hiệu quả của một hệ thống tìm kiếm trong truy xuất thông tin. MAP nội suy được dùng rộng rãi trong các cuộc thi và nghiên cứu trong lĩnh vực truy xuất thông tin để đánh giá hiệu quả của các phương pháp tìm kiếm khác nhau. Đối với bài toán lần này, vì đều dùng MAP và MAP nội suy nên việc tính toán MAP nội suy dựa vào TREC 11 điểm đồng thời dựa vào kết quả precision-recall đã được tính từ trước.

Để hiểu rõ hơn, ta có ví dụ sau

Ví dụ:

Cho tập tài liệu liên quan đến truy vấn là:

R = {3, 5, 9, 25, 39, 44, 56, 71, 89, 123}

Cho kết quả truy vấn gồm tập các tài liệu được xếp hạng như sau:

Query1: {**123**, 84, **56**, 6, 8, **9**, 511, 129, 187, **25**, 38, 48, 250, 113, **3**}

Query2: {**56**, **123**, 6, 8, 12, **25**, **3**, 129, **9**, 187, 30, **44**, 71, 220}

Tính MAP và MAP nội suy theo bộ TREC 11 điểm.

Bài làm:

Từ giả thiết trên ta có tập tài liệu liên quan gồm 10 tài liệu.

MAP không nội suy

n	Query1		Query2	
	recall	Precision	recall	precision
1	0.1 (1/10)	1 (1/1)	0.1	1 (1/1)
2	0.2 (2/10)	0.67 (2/3)	0.2	1 (2/2)
3	0.3 (3/10)	0.5 (3/6)	0.3	0.5 (3/6)
4	0.4 (4/10)	0.4 (4/10)	0.4	0.57 (4/7)
5	0.5 (5/10)	0.33 (5/15)	0.5	0.56 (5/9)
6			0.6	0.5 (6/12)
AP	0.58		0.69	
MAP	0.635			

MAP nội suy

TREC	Precision_query1	Precision_query2
0.1	0.67	1
0.2	0.67	1
0.3	0.5	0.57
0.4	0.4	0.57
0.5	0.33	0.56
0.6	0	0.5
0.7	0	0
0.8	0	0
0.9	0	0
1	0	0
AP nội suy	0.23	0.38
MAP nội suy	0.305	

5.2. Mục tiêu đánh giá

Nhóm đã đề xuất 2 mô hình truy xuất thông tin để giải quyết bài toán đặt ra tương ứng với 6 phương pháp đã giới thiệu ở trên. 2 phương pháp đó bao gồm:

STT	Tên phương pháp được đề xuất
1	BIM + PorterStemmer + stopword removal + log-odds
2	VSM + PorterStemmer + stopword removal + TF-IDF + Cosine similarity

Các phương pháp thiết kế mô hình truy xuất thông tin do nhóm đề xuất cài đặt.

Mục đích đánh giá của nhóm là tìm ra được phương pháp tối ưu trong 2 phương pháp sử dụng về độ chính xác cũng như về thời gian truy vấn. Để từ đó có thể tìm ra được những phương pháp tối ưu nhất so sánh với thư viện Whoosh khi sử dụng BM25F.

CHƯƠNG V: CÀI ĐẶT THỬ NGHIỆM

1. Thiết kế chương trình cài đặt

1.1. Môi trường lập trình

Jupyter Notebook là một ứng dụng web mã nguồn mở cho phép chạy Interactive Python (hay IPython), có thể gộp cả code Python và các thành phần văn bản phức tạp như hình ảnh, công thức, video, biểu thức… vào trong cùng một file giúp cho việc trình bày trở lên dễ hiểu, giống như một file trình chiếu nhưng lại có thể thực hiện chạy code tương tác trên đó, cốt lõi của việc này chính là Markdown. Các file "notebook" này có thể được chia sẻ với mọi người và có thể thực hiện lại các công đoạn một cách nhanh chóng và chính xác như những gì bạn đã làm trong quá trình tạo ra file. Nhóm chọn Jupyter notebook để có thể thử nghiệm code thuận tiện.

1.2. Giao diện chương trình cài đặt

Giao diện web do dùng Jupyter Notebook. Thiết kế theo hướng module hóa nên có thể thay thế hoặc bổ sung thêm các chức năng và các cách cài đặt hàm.

1.3. Cấu trúc chương trình cài đặt

Dưới đây là sơ đồ cấu trúc chương trình cài đặt.

Trong sơ đồ dưới đây đối với Binary Independence Model quy trình sẽ đi theo thứ tự là "Lập chỉ mục đảo ngược", "Tạo từ điển" và sau đó mới tới "Truy vấn". Còn đối với Vector Space Model quy trình sẽ đi từ "Tạo dictionary" tới "Truy vấn". Trong "Lập chỉ mục đảo ngược" các bước sẽ được thực hiện tuần tự từ trên xuống dưới và các nhánh còn lại của sơ đồ cũng tương tự như vậy.

Sơ đồ cấu trúc chương trình cài đặt

1.4. Quy trình thiết kế chương trình cài đặt

Với sơ đồ như trên, ở mỗi mô hình chúng ta sẽ viết các hàm, mỗi hàm chỉ nên thực hiện duy nhất nhiệm vụ của chúng để có thể sửa lỗi các hàm, thay đổi các hàm. Các bước thực hiện như sau: Bước đầu tiên chúng ta đưa tập Cranfield và tập TEST vào trong mô hình. Vì tập Cranfield và tập TEST đã được nén lại nên chúng ta phải giải nén nó. Tiếp theo cần phải đọc thư mục Cranfield (thư mục chứa các file .txt dùng làm document) và thư mục TEST (thư mục chứa các file .txt dùng để truy vấn) tạo một danh sách lưu các file đó lại, nên sẽ cần viết một hàm đọc thư mục cho bước này.

1.5. Binary Independence Model

Đầu tiên chúng ta sẽ lập chỉ mục cho mỗi tài liệu (mỗi file .txt trong tập Cranfield), bước này gồm các bước khác, bước này cần phải viết một hàm với mục đích liệt kê ra tất cả các term khác nhau có trong tài liệu đồng thời tính luôn tần số của mỗi term. Tiếp theo, lập chỉ mục cho toàn bộ tài liệu, nên sẽ cần viết một hàm để lập chỉ mục cho toàn bộ tài liệu. Từ danh sách words (terms), quantity và index_doc từ chỉ mục đã được lập, tạo một từ điển cho toàn bộ từ điển đồng thời tính trọng số cho các term có trong từ điển, bước này cần viết một hàm lấy các giá trị đó và tạo từ điển. Tiếp theo sẽ tới bước truy vấn, bước này cũng gồm các bước khác, bước này cần viết một hàm để thực hiện truy vấn cho mỗi truy vấn và trả về các kết quả. Bây giờ, lập chỉ mục cho câu truy vấn, ở bước này chỉ cần dùng lại hàm lập chỉ mục cho mỗi tài liệu ở trên, sau đó trích xuất ra danh sách words (terms), ở bước này cần viết một hàm để thực hiện các công việc trên. Tính độ liên quan sử dụng độ đo log-odds. Sau đó, chúng ta chỉ cần lưu lại danh sách các kết quả đã được xếp hạng. Sau khi có kết quả truy vấn, chúng ta cần trả về các kết quả đã được sắp xếp giảm dần theo độ liên quan, nên cần viết một hàm in ra các kết quả.

1.6. Vector Space Model

Đầu tiên chúng ta sẽ tạo từ điển, bước này gồm các bước khác, bước này cần phải viết một hàm tạo từ điển cho các tài liệu. Tiếp theo, gán docID cho mỗi file, nên sẽ cần viết một hàm gán docID. Tạo danh sách token-id cho toàn bộ tài liệu (file), nên sẽ cần viết một hàm

tạo danh sách token-id. Chúng ta cần phải tiền xử lý mỗi tài liệu trước khi có thể lập danh sách token-id, nên cần phải viết một hàm tiền xử lý. Tạo từ điển cho các tài liệu và tạo danh sách các vector tài liệu từ danh sách token-id (để thuận tiện cho việc chuẩn hóa các vector và khi áp dụng Euclidean distance), ở đây cần một hàm tạo từ điển. Tính trọng số tf, idf, tf-idf thì chúng ta sẽ phải chia ra viết 3 hàm tính trọng số riêng biệt. Chuẩn hóa các vector, nên bước này cần một hàm chuẩn hóa vector. Tiếp theo sẽ tới bước truy vấn, bước này cũng gồm các bước khác, bước này cần viết một hàm để thực hiện truy vấn cho mỗi truy vấn và trả về các kết quả. Bây giờ, chúng ta sẽ biểu diễn query thành một cấu trúc từ điển để có thể dùng lại các hàm đã viết ở bước "Tạo từ điển", tương tự, bước này sẽ gồm các bước khác, bước này cần viết một hàm tạo từ điển cho truy vấn. Vì truy vấn chỉ có một nên chúng ta sẽ tiến thẳng tới bước tạo danh sách token-id, vì ở trên đã viết hàm tạo danh sách token-id rồi nên chúng ta chỉ cần dùng lại. Do vì chỉ có mỗi một truy vấn nên không cần có docID cho truy vấn, nhưng vì chúng ta dùng lại hàm ở trên nên cần phải gán cho truy vấn một docID đặc biệt, không trùng với các docID của các tài liệu. Chúng ta cũng cần tiền xử lý truy vấn trước khi tạo danh sách token-id, vì ở chương 2 đã nói rõ tại sao cần phải áp dụng cùng một phương pháp tiền xử lý với cả các tài liệu và truy vấn, do đó chúng ta cũng sẽ dùng lại hàm tiền xử lý ở trên. Tiếp theo là bước tạo từ điển cho truy vấn và tạo vector truy vấn, chúng ta cũng chỉ cần dùng lại hàm tạo từ điển ở trên. Tính trọng số tf, idf và tf-idf thì chúng ta chỉ có thể dùng lại 2 hàm tính trọng số tf, tf-idf, còn idf là do lấy từ từ điển của các tài liệu nên cần phải viết hàm lấy các giá trị idf từ từ điển của các tài liệu cho truy vấn. Chuẩn hóa vector truy vấn thì chỉ cần dùng lại hàm chuẩn hóa ở trên. Tính độ liên quan sử dụng Cosine similarity. Sau đó, chúng ta chỉ cần lưu lại danh sách các kết quả đã được xếp hạng. Bây giờ là bước cuối cùng của việc truy vấn, chúng ta cần trả về các kết quả đã được sắp xếp giảm dần theo độ liên quan, nên cần viết một hàm in ra các kết quả.

2. Thiết kế quy trình đánh giá các mô hình được cài đặt thử nghiệm

2.1. Tiêu chí đánh giá

Với việc thực nghiệm để đánh giá mô hình truy xuất của nhóm, nhóm đã đề ra tiêu chí đánh giá của 2 mô hình như các phương pháp được nói ở phần 4. Nhóm cũng đã thống kê thực

nghiệm bằng cách tính các giá trị precision và recall của mỗi mô hình. Sau đó lưu các giá trị precision và recall để thực hiện việc tính toán còn lại và so sánh các mô hình, ngoài việc tính toán trên thì nhóm đã tính thời gian truy vấn của mỗi mô hình và lưu lại để so sánh. Nhắc lại về các phương pháp đánh giá: MAP tính toán trung bình của các giá trị Average Precision (AP) cho tất cả các truy vấn trong tập dữ liệu. MAP nội suy là một cách để tính toán MAP bằng cách dùng nội suy tuyến tính trên đồ thị precision-recall. Thay vì tính toán AP bằng cách lấy giá trị precision tại các điểm giao nhau trên đồ thị precision-recall, MAP nội suy dùng một phương pháp nội suy để tính toán precision tại các điểm recall bị thiếu. Phương pháp nội suy này cho phép tính toán giá trị precision của một mức recall bằng cách nội suy tuyến tính giữa các mức recall lân cận có giá trị precision đã được tính toán.

2.2. Thực nghiệm đánh giá

Trong phần thực nghiệm của đề tài được nhóm tổng hợp, thống kê và phân tích, ghi nhận những số liệu thực nghiệm. Các kết quả thực nghiệm do nhóm cài đặt trên máy tính đã được nhóm ghi lại rõ ràng và đầy đủ, từ đó để thuận tiện quan sát, nhằm làm cơ sở để đưa ra những kết luận về việc so sánh, từ đó lựa chọn ra được đâu là phương pháp tách từ hay nhất, mô hình gán nhãn hay nhất, hiệu quả nhất cho bài toán mà nhóm xử lý. Qua quy trình đánh giá đã giúp cho một số thành viên trong nhóm chúng em hiểu hơn về cách thức, các bước tiến hành khi đánh giá một phương pháp thực nghiệm nào đó, điều này giúp nâng cao kiến thức của từng thành viên trong nhóm.

3. Kết quả thực nghiệm

Khi tiến hành chạy thực nghiệm chương trình do nhóm cài đặt trên máy tính có cấu hình nêu trên, chúng em thu được kết quả thực nghiệm như dưới đây. Chúng em thống kê kết quả theo từng độ đo đánh giá để dễ dàng quan sát và đối chiếu các phương pháp đã được đề xuất, từ đó có thể so sánh và rút ra kết luận về các phương pháp. Sau đây là các biểu đồ thống kê các độ đo đánh giá của các phương pháp.

3.1. So sánh hai mô hình Binary Independence và Vector Space

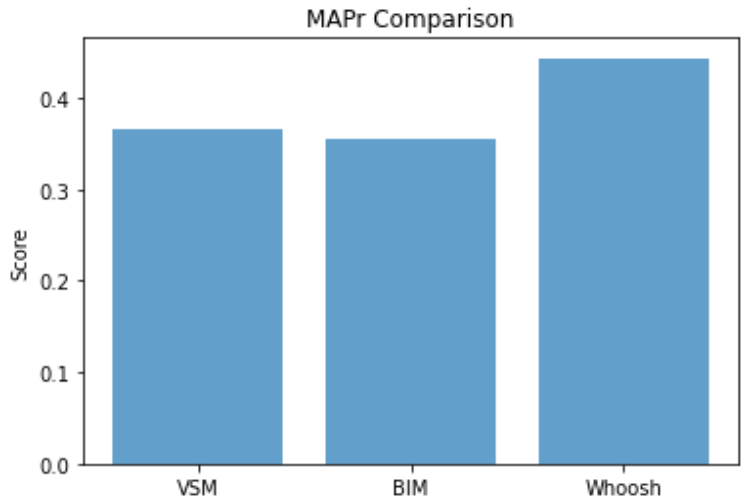
	Vector Space Model	Binary Independence Model
Precision	0.0096	0.008664
Recall	0.9608	0.959173
MAPr	0.366	0.356051

Nhận xét:

Có thể thấy không có quá nhiều sự thay đổi hay khác biệt trong cả hai mô hình. Mô hình Vector Space có phần cao hơn mô hình Binary Independence một chút. Nhìn chung cả hai không thể hiện quá tốt trong độ chính xác khi đây cũng chỉ là những mô hình khá cơ bản.

3.2. So sánh hai mô hình Binary Independence, Vector Space và thư viện Whoosh

	Vector Space Model	Binary Independence Model	Whoosh
Precision	0.0096	0.008664	0.008758
Recall	0.9608	0.959173	0.958084
MAPr	0.366	0.356051	0.443878



Nhận xét:

Có thể thư viện Whoosh có phần thể hiện tốt nhất trong cả 3, đặc biệt ở chỉ số MAPr sau khi đã được nhóm chúng em tuỳ chỉnh.

analyzer = RegexTokenizer() | LowercaseFilter() | StopFilter(stoplist=stoplist) | StemFilter()

RegexTokenizer(): sử dụng để tách từ

LowercaseFilter(): sử dụng để đưa chữ hoa về chữ thường

StopFilter(): sử dụng để loại bỏ stopwords

a, an, and, are, as, at, be, but, by, for, if, in, into, is, it, no, not, of, on, or, such, that, the, their, then, there, these, they, this, to, was, will, with

Thay bằng stopwords của nltk

StemFilter(): sử dụng để stemming với PorterStemmer

Okapi BM25F sử dụng để xếp hạng

CHƯƠNG VI: KẾT LUẬN

1. Nhận xét mô hình

Mô hình Binary Independence là một mô hình truy vấn chỉ dựa vào việc term có xuất hiện hay không xuất hiện trong tập tài liệu truy vấn và trong câu truy vấn chứ hoàn toàn không phụ thuộc vào tần số của term trong tài liệu cũng như trong câu truy vấn. Điều này có thể dẫn đến việc một tài liệu liên quan hơn lại bị xếp sau một tài liệu ít liên quan hơn do có chỉ số relevant thấp hơn. Mặc dù mô hình đạt hiệu quả cũng khá cao nhưng thời gian chạy chương trình còn khá là chậm. Mô hình Vector Space là một mô hình dựa vào tần số xuất hiện của term có trong tập tài liệu và trong câu truy vấn. Điều này của mô hình Vector Space đã giải quyết được một phần nào đó về vấn đề độ liên quan của tài liệu ở mô hình Binary Independence. Mô hình cho kết quả MAP và MAP nội suy cũng khá tốt đồng thời thời gian thực hiện chương trình cũng khá nhanh. Nhưng mô hình Vector Space chỉ dựa vào term có sẵn trong chỉ mục để truy vấn nên có thể không truy vấn được theo nhu cầu thông tin. Ngoài ra VSM cũng tồn tại những nhược điểm như có rất nhiều các công thức tính trọng số, cách chuẩn hóa (normalize) vector. Vì vậy khá khó khăn trong việc chọn ra công thức phù hợp nhất để xây dựng chương trình truy xuất thông tin.

2. Bài học kinh nghiệm

Sau khi hoàn thành xong đề tài này, các thành viên trong nhóm chúng em đã đều có khả năng trả lời các câu hỏi liên quan tới mô hình Binary Independence và mô hình Vector Space. Bài báo cáo đi sát với toàn bộ nội dung thành viên đã thực hiện cũng như sưu tầm được những kiến thức cần phải phân tích rõ ràng. Trong quá trình nghiên cứu và tìm hiểu, chúng em đã thu được những kết quả thực nghiệm hữu ích thông qua việc áp dụng các phương pháp do nhóm đề xuất. Qua việc đánh giá các phương pháp, nhóm cũng đã thu được một số kinh nghiệm quý báu từ mô hình, tùy thuộc vào đặc điểm và yêu cầu của từng bài toán cụ thể mà lựa chọn phương pháp sao cho hợp lý. Mục đích của nhóm khi thực hiện thử nghiệm các trường hợp khác nhau khi cài đặt mô hình là để tìm ra phương pháp/mô hình tối ưu nhất với độ chính xác cao nhất có thể và đáp ứng được các yêu cầu đề ra. Ngoài

ra, để có thể củng cố cho việc nghiên cứu, phát triển, chúng em sẽ cố gắng thu tập những thông tin cần thiết, tạo ra nhiều tính năng mới, đồng thời để khắc phục những hạn chế còn tồn tại trong đề tài này và nghiên cứu thêm những cái mới, hay để có thể hoàn thiện đề tài nghiên cứu về sau.

Qua đó cũng có thể đánh giá rằng, đồ án do nhóm thực hiện không chỉ là một đồ án dừng lại ở mức độ chuyên ngành dành cho các thành viên trong nhóm, mà còn là cơ hội giúp các thành viên trong nhóm chúng em trau dồi thêm nhiều kiến thức và kỹ năng bổ ích, giúp mỗi thành viên trong nhóm nâng cao khả năng làm việc tập thể, nâng cao được tính tương tác giữa các thành viên, từ đó giúp bản thân mỗi thành viên trong nhóm qua quá trình học lập, làm việc nhóm rèn luyện thêm cho mình những kĩ năng cần thiết cho công việc sau này.

TÀI LIỆU THAM KHẢO

[1] An Introduction to Information Retrieval, https://www.academia.edu/27076940/An_Introduction_to_Information_Retrieval

[2] Information retrieval, https://www.slideshare.net/phuongtt2a/information-retrieval-4665369

[3] Tf-idf algorithm, https://viblo.asia/p/tf-idf-algorithm-text-retrieval-and-search-engines-1Je5EmGY5nL

[4] Vector Space Model, https://blog.duyet.net/2019/08/ir-vector-space-model.html

[5] Porter Stemming Algorithm Basic-Intro, https://vijinimallawaarachchi.com/2017/05/09/porter-stemming-algorithm/

[6] NLTK Lemmatizer, https://www.educba.com/nltk-lemmatizer/