Đại học Quốc gia Thành phố Hồ Chí Minh
Trường Đại học Công nghệ Thông tin
Khoa Khoa học Máy tính

# Truy xuất Thông tin

Đồ án môn học

## Mô hình Truy vấn Vector Space Model và Neural Network based cho Cranfield và Dữ liệu Tuyển sinh UIT

Sinh viên:
Huỳnh Anh Dũng – 22520278
Hà Huy Hoàng – 22520460
Nguyễn Duy Hoàng – 22520467

Giảng viên hướng dẫn:
thầy Nguyễn Trọng Chỉnh

Ngày thực hiện: Ngày 24 tháng 6 năm 2025

## Tóm tắt nội dung

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

# Phân công

| STT | MSSV | Họ và tên | Phân công nhiệm vụ | Hoàn thành |
|---|---|---|---|---|
| 1 | 22520278 | Huỳnh Anh Dũng | ... | 100% |
| | | | ... | |
| | | | ... | |
| 2 | 22520460 | Hà Huy Hoàng | ... | 100% |
| | | | ... | |
| | | | ... | |
| 3 | 22520467 | Nguyễn Duy Hoàng | ... | 100% |
| | | | ... | |
| | | | ... | |

# Mục lục

# Chương 1

# Giới thiệu tổng quan

## 1.1 Bài toán thực tế

Trong bối cảnh dữ liệu số ngày càng gia tăng nhanh chóng, nhu cầu tìm kiếm và truy xuất thông tin một cách hiệu quả trở nên thiết yếu hơn bao giờ hết. Từ thực tiễn này, lĩnh vực Truy xuất Thông tin (Information Retrieval, về sau sẽ gọi là IR) đã hình thành và phát triển như một nhánh quan trọng của khoa học máy tính. Thay vì đơn thuần xử lý dữ liệu, truy xuất thông tin tập trung vào việc xác định, đánh giá và cung cấp những tài liệu liên quan đến nhu cầu của người dùng từ một kho dữ liệu lớn. Quá trình này bao gồm các bước như lập chỉ mục, truy vấn và xếp hạng kết quả, với sự hỗ trợ của nhiều thuật toán và mô hình tính toán hiện đại.

Thông qua việc mô hình hóa mối quan hệ giữa truy vấn và tài liệu, truy xuất thông tin cho phép cải thiện hiệu quả tiếp cận tri thức trong nhiều bối cảnh, từ công cụ tìm kiếm trên Internet, hệ thống đề xuất nội dung, đến các thư viện số và kho học liệu điện tử. Đặc biệt, với sự phát triển của trí tuệ nhân tạo và học máy, các hệ thống truy hồi ngày nay không chỉ dựa vào từ khóa đơn lẻ mà còn có khả năng hiểu ngữ nghĩa, phân tích ngữ cảnh và thậm chí dự đoán nhu cầu thông tin tiềm ẩn của người dùng. Điều này góp phần nâng cao trải nghiệm truy xuất và đảm bảo thông tin được cung cấp phù hợp hơn với mục đích sử dụng.

Bên cạnh đó, IR còn đóng vai trò quan trọng trong khai phá dữ liệu, xử lý ngôn

ngữ tự nhiên và phân tích dữ liệu lớn. Tính ứng dụng rộng rãi cùng khả năng thích nghi với sự biến động của công nghệ khiến lĩnh vực này trở thành một nền tảng không thể thiếu trong hệ sinh thái tri thức hiện đại.

## 1.2 Khái niệm Truy xuất Thông tin

IR giúp người dùng tìm ra được thông tin quan trọng trong một tập dữ liệu đồ sộ. Khái niệm có thể được hiểu như một chương trình phần mềm có khả năng tổ chức, lưu trữ, truy hồi và đánh giá thông tin trong một bộ ngữ liệu tương ứng. Truy xuất thông tin có thể được xem như một thủ thư, không trực tiếp trả lời câu hỏi của người dùng, mà chỉ dẫn người dùng đến với những đầu sách cần thiết cho câu trả lời. [2]

Khi ta đặt một truy vấn, mô hình IR giúp ta tìm ra những tài liệu liên quan nhất đến truy vấn và xếp hạng chúng. Mô hình IR thực hiện cơ chế này bằng cách ứng dụng một hàm đối chiếu để so sánh truy vấn của chúng ta so với tập ngữ liệu hiện có trong hệ thống. Hàm đối chiếu này đánh điểm lên những tài liệu đã được so sánh, làm tiền đề cho việc xếp hạng những tài nguyên thông tin liên quan nhất lên hàng đầu. [2]

## 1.3 Vai trò của Truy xuất Thông tin

Song song với sự bùng nổ của các phương tiện thông tin đại chúng, trong mọi lĩnh vực, vai trò của các hệ thống IR được đề cao hơn bao giờ hết, vì nhiều lý do:

- Hỗ trợ tìm kiếm thông tin nhanh chóng: nhờ đặc thù mạnh mẽ, nhanh chóng và chính xác, IR cho phép người dùng truy cập đến thông tin cần thiết theo thời gian thực mà không cần thỏa hiệp với độ chính xác của dữ kiện IR mang lại $\longrightarrow$ tiết kiệm thời gian, công sức của người dùng cùng lúc với tăng độ hiệu quả của công việc.
- Hỗ trợ quản lý tri thức: IR giúp quản lý đa hình thông tin, thường thấy như

tri thức, một cách hiệu quả ⟶ người dùng có thể tổ chức, tìm kiếm và truy cập thông tin một cách dễ dàng hơn.

- Hỗ trợ phát triển các ứng dụng khai thác thông tin: IR đóng vai trò cốt lõi trong việc phát triển các ứng dụng khai thác thông tin như phân tích dữ liệu, khai thác tri thức, trích xuất thông tin và tổng hợp thông tin ⟶ vận dụng triệt để cơ sở dữ liệu trong thời đại tràn lan thông tin số.

## 1.4 Quy trình truy xuất thông tin

Quy trình truy xuất thông tin được tổ chức thành nhiều bước lớn nhỏ khác nhau, nhưng tổng quát có thể chia thành ba giai đoạn chính:

- Giai đoạn tiền xử lý,
- Giai đoạn lập chỉ mục,
- Giai đoạn xử lý truy vấn, truy xuất chỉ mục và xếp hạng kết quả.

Trước khi bắt đầu quá trình truy xuất thông tin, điều đầu tiên cần làm là xác định rõ loại tài liệu sẽ được truy xuất, có thể là văn bản, hình ảnh, video hay âm thanh. Tiếp đến, ta lựa chọn thuật toán truy xuất phù hợp với loại tài liệu đó, đồng thời tiến hành các bước xử lý và chuyển đổi tài liệu gốc sao cho phù hợp với mô hình truy xuất đã chọn.

Sau đó, cần xây dựng hệ thống chỉ mục (indexing) nhằm tăng hiệu quả và tốc độ trong việc tìm kiếm thông tin. Khi hệ thống chỉ mục đã sẵn sàng, bước tiếp theo là xử lý truy vấn. Khi người dùng nhập vào một truy vấn – thể hiện nhu cầu thông tin của họ – truy vấn này sẽ được phân tích, chuyển đổi về định dạng tương thích với mô hình hệ thống.

Lúc này, hệ thống sẽ thực hiện việc so khớp truy vấn với chỉ mục đã tạo, tính toán mức độ liên quan, và truy xuất các tài liệu phù hợp. Cuối cùng, các tài liệu này sẽ được sắp xếp theo độ phù hợp (ranking) và hiển thị cho người dùng

### 1.4.1 Giai đoạn tiền xử lý

**Truy xuất thông tin** là hành trình khám phá tri thức, nơi hệ thống phải lựa chọn những mảnh ghép có liên quan nhất từ một biển dữ liệu khổng lồ. Trong quá trình đó, **tiền xử lý văn bản** nổi lên như một bước đi tất yếu và tinh tế, đóng vai trò là chiếc cầu nối giữa ngôn ngữ tự nhiên phức tạp và cấu trúc dữ liệu được chuẩn hóa – thứ mà các mô hình truy xuất có thể dễ dàng tiếp cận và xử lý.

Ở giai đoạn này, trước khi bất kỳ thuật toán nào có thể hoạt động hiệu quả, mỗi tài liệu đều phải trải qua một chuỗi các biến đổi có hệ thống. Từ việc **phân tích từ vựng**, **loại bỏ stopwords** cho đến thao tác **đưa từ về dạng gốc** như *stemming* hay *lemmatization*, mọi bước đều được thực hiện nhằm làm mềm hóa cấu trúc ngôn ngữ, chuyển hóa các biểu hiện ngữ nghĩa phong phú của văn bản thành dạng chuẩn hóa – một hình thức tinh gọn nhưng vẫn giữ được linh hồn thông tin cốt lõi.

Kết quả của quá trình này là một tập dữ liệu mới, nơi các từ khóa đã được chuyển đổi thành những đơn vị biểu diễn ngữ nghĩa thống nhất. Không còn sự rườm rà của cú pháp, không còn những nhiễu loạn đến từ các hình thức từ ngữ khác nhau, chỉ còn lại một nền tảng ổn định, tinh giản và giàu tiềm năng để mô hình truy xuất tiếp cận, so sánh và phân tích.

Quá trình **đánh chỉ mục**, diễn ra ngay sau tiền xử lý, sẽ sử dụng tập dữ liệu đã được làm sạch này để tạo ra một biểu diễn nội tại cho mỗi tài liệu. Nhờ đó, hệ thống truy xuất không còn phải dò tìm từng từ một trong mê cung văn bản ban đầu, mà có thể dựa vào một cấu trúc định vị hiệu quả – như một bản đồ rút gọn dẫn đến thông tin mục tiêu.

Lợi ích của tiền xử lý văn bản không chỉ dừng lại ở hiệu quả hệ thống, mà còn chạm đến bản chất của việc hiểu ngôn ngữ trong không gian số:

- **Tối ưu hóa tốc độ truy xuất**: khi văn bản được làm gọn và thống nhất, các thao tác tính toán trở nên nhẹ nhàng hơn, giảm bớt đáng kể thời gian truy vấn mà không làm mất đi giá trị thông tin.
- **Cải thiện độ chính xác**: những từ ngữ dư thừa, lặp lại, hay chỉ có tính chức

năng (như *a*, *the*, *is*, ...) được loại bỏ, nhường chỗ cho những từ khóa giàu ý nghĩa – qua đó giúp mô hình đưa ra kết quả sát hơn với ý định thực sự của người dùng.

- **Tăng tính nhất quán**: khi tất cả các biến thể của một từ được quy về cùng một gốc, hệ thống có thể xử lý các biểu hiện khác nhau của cùng một khái niệm theo một cách đồng bộ – điều kiện tiên quyết để đảm bảo sự công bằng và chính xác trong so sánh.

- **Giảm thiểu nhiễu**: những yếu tố không mang giá trị phân biệt – các ký tự đặc biệt, lỗi chính tả, từ vô nghĩa – đều bị loại bỏ, mang đến một dòng dữ liệu trong sạch hơn để phân tích.

- **Tăng khả năng tìm kiếm**: khi từ đồng nghĩa, từ viết tắt hoặc từ viết sai chính tả được chuẩn hóa, hệ thống truy xuất sẽ có cơ hội mở rộng phạm vi hiểu biết của mình, từ đó tăng khả năng thu thập đúng thông tin dù đầu vào không hoàn hảo.

Tựu trung lại, **tiền xử lý văn bản không chỉ là một bước kỹ thuật**, mà còn là sự tinh luyện ngôn ngữ – một quá trình chắt lọc, gọt giũa văn bản từ hình thức tự nhiên đầy biến hóa trở thành cấu trúc tinh giản và sẵn sàng phục vụ mục tiêu cuối cùng: **truy xuất thông tin nhanh chóng, chính xác và nhất quán**. Đó là tiền đề không thể thiếu cho mọi hệ thống truy vấn hiện đại – nơi hiệu quả kỹ thuật được xây dựng từ sự nhạy bén ngôn ngữ

### 1.4.2 Giai đoạn lập chỉ mục

**Lập chỉ mục** (*Indexing*) là một trong những trụ cột cấu thành nên hiệu quả hoạt động của hệ thống **truy xuất thông tin (Information Retrieval – IR)**. Trong bản chất của mình, đây không chỉ đơn thuần là một thao tác kỹ thuật, mà còn là quá trình thiết kế nên một cấu trúc logic tinh tế – nơi mà các yếu tố cốt lõi của ngôn ngữ được mã hóa, tổ chức và lưu trữ để phục vụ cho quá trình tìm kiếm diễn ra nhanh chóng và chính xác.

Quá trình lập chỉ mục khởi đầu bằng một bước tiền xử lý cẩn trọng. Tại đây, các văn bản thô sẽ được thanh lọc và chuẩn hóa thông qua các thao tác như loại bỏ **stopwords**, chuyển về chữ thường, xử lý dấu câu, tách từ và lấy gốc từ (stemming/lemmatization). Mục tiêu của bước này là loại bỏ đi những yếu tố gây nhiễu, đồng thời làm sáng tỏ những đơn vị thông tin có giá trị – vốn là những *term* (thuật ngữ chỉ mục) sẽ trở thành linh hồn của bộ chỉ mục sau này.

Khi văn bản đã được tiền xử lý, bước tiếp theo là **phân tích văn bản** nhằm xác định các thuật ngữ đại diện. Các *term* này được trích xuất như những đại biểu mang tính ngữ nghĩa cho nội dung văn bản, và từ đó trở thành chìa khóa để kết nối câu truy vấn của người dùng với các tài liệu phù hợp trong kho dữ liệu.

Tiếp theo là công đoạn **xây dựng bộ chỉ mục** – một cấu trúc dữ liệu phức hợp nhưng tối ưu, trong đó mỗi thuật ngữ sẽ được ánh xạ đến danh sách các tài liệu chứa nó, đi kèm với các thuộc tính bổ trợ như **vị trí xuất hiện**, **tần suất**, hoặc các chỉ số đánh giá mức độ quan trọng của thuật ngữ trong từng tài liệu cụ thể. Chính nhờ vào những chỉ số này mà hệ thống có thể thực hiện việc xếp hạng các tài liệu theo mức độ liên quan trong giai đoạn truy vấn.

Sau khi hoàn tất, **bộ chỉ mục được lưu trữ** như một bản đồ thu gọn của kho tri thức, đóng vai trò là nơi tham chiếu cho tất cả các hoạt động truy xuất sau này. Việc truy vấn không còn phải thực hiện trực tiếp trên toàn bộ văn bản – điều vốn tiêu tốn thời gian và tài nguyên – mà được tối ưu hóa thông qua việc dò tìm trên bộ chỉ mục đã được tổ chức chặt chẽ và hiệu quả.

Tựu trung, **quá trình lập chỉ mục không chỉ là một bước trung gian**, mà là một tầng nền tảng quan trọng trong toàn bộ hệ sinh thái truy xuất thông tin. Một bộ chỉ mục được thiết kế tốt sẽ không chỉ nâng cao **độ chính xác** và **tốc độ truy vấn**, mà còn góp phần mở rộng **khả năng bao phủ thông tin**, mang đến cho người dùng những kết quả tìm kiếm đầy đủ, có ý nghĩa và kịp thời. Trong thời đại dữ liệu bùng nổ, khi mà việc tìm đúng thông tin đúng lúc trở nên sống còn, thì vai trò của lập chỉ mục càng trở nên không thể thay thế – như một hệ thần kinh kết nối kho tàng tri thức với nhu cầu tức thời của con người

### 1.4.3 Giai đoạn truy vấn

#### Xử lý truy vấn

Khi người dùng phát sinh nhu cầu thông tin, hệ thống truy xuất sẽ tiếp nhận biểu hiện cụ thể của nhu cầu ấy thông qua một **câu truy vấn** (*query*). Câu truy vấn không chỉ đơn thuần là chuỗi ký tự, mà chính là hình ảnh nén lại của một mục tiêu tri thức – được hệ thống tiếp nhận như một tín hiệu định hướng để bắt đầu quá trình tìm kiếm. Giống như cách mà các tài liệu đã được xử lý trước đó, truy vấn cũng trải qua các bước tiền xử lý như chuẩn hóa, tách từ, loại bỏ từ dừng và chuyển hóa về dạng ngữ nghĩa thích hợp, nhằm tạo nên sự tương thích cấu trúc giữa truy vấn và kho dữ liệu lưu trữ.

#### Truy xuất chỉ mục

Sau khi truy vấn đã được chuẩn hóa, từng *term* (từ khóa) được sinh ra từ câu truy vấn sẽ đóng vai trò như những chìa khóa tìm kiếm, được đưa vào hệ thống để đối chiếu với tập *chỉ mục* đã được xây dựng. Đây là giai đoạn trung tâm của toàn bộ tiến trình truy xuất – nơi diễn ra sự tương tác giữa tri thức cần tìm và tri thức đang có. Quá trình này là một phép so sánh tinh vi giữa các từ khóa trong truy vấn và các từ khóa đã được lập chỉ mục trong các tài liệu, nhằm xác định xem tài liệu nào chứa đựng những nội dung có khả năng thỏa mãn nhu cầu thông tin.

#### Xếp hạng kết quả

Khi các tài liệu phù hợp được xác định, hệ thống sẽ không trả về kết quả một cách ngẫu nhiên hay theo thứ tự xuất hiện, mà thực hiện một bước sắp xếp có ý thức – gọi là **xếp hạng kết quả**. Mỗi tài liệu được gán một điểm số phản ánh mức độ liên quan giữa nội dung của nó và truy vấn đã đưa vào. Mức điểm này được tính toán dựa trên công thức riêng của từng mô hình truy xuất thông tin, như TF-IDF, BM25 hay các phương pháp học sâu hiện đại. Kết quả cuối cùng là một danh sách các tài liệu được sắp xếp theo thứ tự giảm dần của độ liên quan – một hành lang

tinh lọc mà ở đó, tri thức gần nhất với nhu cầu người dùng sẽ được đặt lên hàng đầu, dẫn lối cho hành trình tìm kiếm tri thức trở nên rõ ràng và hiệu quả hơn bao giờ hết.

# Chương 2

# Vector Space Model

## 2.1 Giới thiệu

Mô hình không gian vector (*Vector Space Model* – VSM) là một trong những nền tảng toán học được áp dụng rộng rãi và có ảnh hưởng sâu sắc trong lĩnh vực truy xuất thông tin hiện đại. Được xem như cầu nối giữa ngôn ngữ tự nhiên và thế giới hình học trừu tượng, mô hình này mang đến một cách tiếp cận mang tính đại số, cho phép biểu diễn các tài liệu và truy vấn dưới dạng các vector số trong một không gian nhiều chiều – nơi mỗi chiều tương ứng với một thuật ngữ trong tập chỉ mục.

Ra đời từ hệ thống truy xuất thông tin SMART – một trong những hệ thống tiên phong trong lĩnh vực này – mô hình không gian vector đã chứng minh được tính ứng dụng rộng rãi của mình trong các bài toán truy vấn, lọc thông tin, lập chỉ mục và đặc biệt là xếp hạng độ phù hợp giữa truy vấn và tài liệu.

Trong khuôn khổ của mô hình này, mỗi tài liệu cũng như mỗi câu truy vấn đều được ánh xạ thành một vector, mà thành phần của vector chính là các trọng số đại diện cho mức độ quan trọng của từng thuật ngữ trong văn bản. Những trọng số này thường được tính toán dựa trên tần suất xuất hiện của từ khóa trong tài liệu (term frequency – *TF*), được điều chỉnh bởi mức độ phổ biến của từ đó trong toàn bộ tập hợp tài liệu (inverse document frequency – *IDF*), tạo thành công thức trọng số nổi tiếng TF-IDF. Trọng số càng cao, từ khóa càng mang tính đặc trưng và có khả năng

phân biệt tài liệu đó với các tài liệu khác.

Khi các tài liệu và truy vấn đã được mã hóa dưới dạng các vector trong cùng một không gian, bài toán xác định mức độ liên quan giữa chúng trở thành một phép toán hình học: khoảng cách – hay chính xác hơn, góc – giữa hai vector. Phép đo thường được sử dụng nhất là *độ đo cosine* (cosine similarity), phản ánh mức độ "hướng cùng chiều" giữa truy vấn và tài liệu trong không gian đa chiều. Giá trị cosine càng lớn thì mức độ tương đồng càng cao, đồng nghĩa với việc tài liệu đó càng phù hợp với nội dung truy vấn.

Thông qua cơ chế xếp hạng dựa trên độ tương đồng, mô hình không gian vector cho phép hệ thống truy xuất đưa ra một danh sách các tài liệu sắp xếp theo thứ tự giảm dần của mức độ liên quan – từ đó mang lại cho người dùng những kết quả tìm kiếm chính xác và có ý nghĩa nhất. Không dừng lại ở việc xử lý ngôn ngữ dưới dạng dữ liệu, mô hình này còn mở ra một góc nhìn đầy hình tượng: nơi những dòng chữ được chuyển hóa thành hình khối toán học, và hành trình đi tìm tri thức trở thành một phép đo khoảng cách giữa hai điểm trong không gian tư duy.

## 2.2 Kí hiệu

Trong mô hình không gian vector – một phương pháp mang tính hình học và đại số trong lĩnh vực truy xuất thông tin – việc xây dựng một hệ thống biểu diễn trừu tượng là điều kiện tiên quyết để có thể xử lý, đo lường và so sánh các thực thể ngôn ngữ một cách hệ thống và hiệu quả. Để làm được điều này, một tập hợp các ký hiệu và định nghĩa chuẩn tắc được sử dụng nhằm mô hình hóa các yếu tố chính tham gia vào quá trình truy vấn và truy xuất.

Trước hết, khái niệm **vocabulary** (tập từ vựng) được định nghĩa là tập hợp tất cả các thuật ngữ (*term*) riêng biệt xuất hiện trong toàn bộ tập tài liệu. Ta ký hiệu tập từ vựng này là $V = \{t_1, t_2, t_3, \ldots, t_N\}$, trong đó $N$ là tổng số lượng từ khóa khác nhau được trích xuất từ toàn bộ corpus. Tập từ vựng này chính là không gian hình học của mô hình, nơi mỗi chiều trong không gian đại diện cho một thuật ngữ duy nhất.

Tiếp theo, một **document** (tài liệu) bất kỳ được biểu diễn như một vector trong không gian từ vựng, có dạng:

$$d_i = \{d_{i1}, d_{i2}, d_{i3}, \ldots, d_{iN}\} \tag{2.1}$$

Trong đó, mỗi thành phần $d_{ij}$ đại diện cho trọng số của thuật ngữ $t_j$ trong tài liệu $d_i$. Tập hợp tất cả các tài liệu tạo thành **collection** – ký hiệu là $C = \{d_1, d_2, d_3, \ldots, d_M\}$, trong đó $M$ là tổng số lượng tài liệu trong hệ thống.

Tương tự như tài liệu, một **query** (câu truy vấn) cũng được ánh xạ vào không gian vector với cấu trúc:

$$q = \{q_1, q_2, q_3, \ldots, q_N\} \tag{2.2}$$

Mỗi thành phần $q_j$ là trọng số biểu diễn mức độ quan trọng của từ khóa $t_j$ trong truy vấn. Việc biểu diễn đồng nhất này cho phép truy vấn và tài liệu được so sánh trực tiếp thông qua các phép toán tuyến tính.

Để đo lường mức độ tương đồng giữa một tài liệu và một truy vấn, ta sử dụng hàm **Rel(q, d)** – biểu diễn *độ liên quan* giữa tài liệu $d$ và truy vấn $q$. Mức độ này thường được tính bằng các hàm tương đồng như cosine similarity hoặc các hàm khoảng cách.

Để thực hiện phép tính này, trước hết ta cần các hàm biểu diễn cho truy vấn và tài liệu, ký hiệu lần lượt là **Rep(q)** và **Rep(d)**. Hai hàm này chính là cơ chế ánh xạ thông tin ngôn ngữ sang biểu diễn toán học – từ hình thức biểu đạt bằng chữ sang dạng số có thể xử lý được bởi hệ thống máy tính.

Cuối cùng, một thành phần quan trọng khác trong hạ tầng của mô hình là **dictionary** (từ điển). Đây là một cấu trúc dữ liệu chuyên biệt – thường được xây dựng dưới dạng bảng băm hoặc cây tìm kiếm – nhằm tổ chức, lưu trữ và truy xuất nhanh chóng các thuật ngữ và thông tin liên quan của chúng. Dictionary đóng vai trò xương sống trong quá trình lập chỉ mục và tìm kiếm, giúp cải thiện đáng kể hiệu

suất hệ thống cả về thời gian xử lý lẫn tài nguyên sử dụng.

Tựu trung, việc quy ước và chuẩn hóa các khái niệm trong mô hình không gian vector không chỉ giúp mô hình hóa rõ ràng các yếu tố ngôn ngữ, mà còn mở đường cho việc xử lý truy vấn một cách chính xác và có hệ thống – nơi ngôn ngữ được "giải nghĩa" bằng hình học, và tri thức được "đo lường" bằng các đại lượng số học tinh tế.

## 2.3 Ý tưởng

Mô hình không gian vector là một tiếp cận kinh điển nhưng vẫn đầy sức mạnh trong lĩnh vực truy xuất thông tin, nổi bật bởi khả năng ánh xạ ngôn ngữ tự nhiên vào không gian hình học trừu tượng, nơi mỗi thực thể ngôn ngữ – bất kể là tài liệu hay truy vấn – đều được biểu diễn như một **vector trọng số** trong một không gian nhiều chiều.

Trong không gian này, mỗi chiều tương ứng với một *term* từ tập từ vựng đã được xây dựng trước đó. Nhờ đó, toàn bộ tập hợp tài liệu và truy vấn có thể được số hóa và đồng nhất hóa về mặt biểu diễn, tạo điều kiện thuận lợi cho việc so sánh, đối chiếu và xếp hạng. Cụ thể, ta có thể mô tả biểu diễn của một tài liệu $d$ và truy vấn $q$ thông qua các vector trọng số:

$$\text{Rep}(d) = (w_{d1}, w_{d2}, \ldots, w_{dN}), \quad \text{Rep}(q) = (w_{q1}, w_{q2}, \ldots, w_{qN}) \tag{2.3}$$

trong đó mỗi trọng số $w_{ij}$ phản ánh **mức độ quan trọng** của từ khóa thứ $j$ trong văn bản hoặc câu truy vấn tương ứng. Trọng số này thường được tính theo các công thức như TF, TF-IDF hoặc các biến thể hiện đại hơn nhằm phản ánh độ đặc trưng và tính phân biệt của từ khóa.

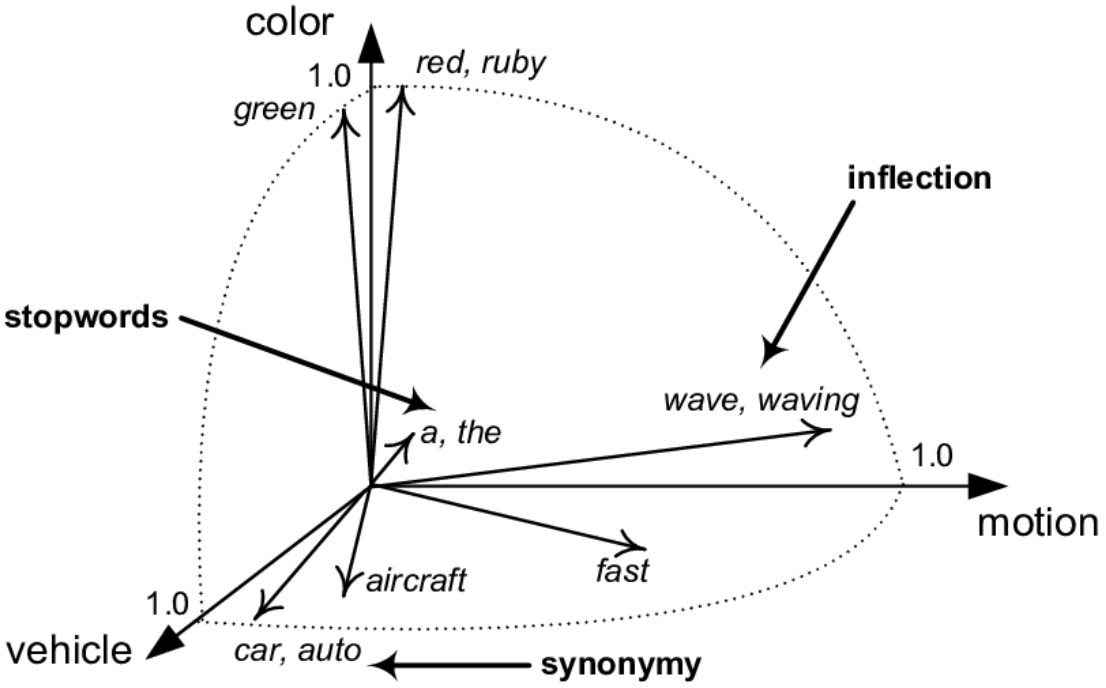
Tại thời điểm truy vấn, hệ thống sẽ tính toán **độ tương đồng** giữa truy vấn và từng tài liệu trong tập hợp, sử dụng các độ đo hình học như **khoảng cách Euclid** hoặc phổ biến hơn là **độ đo cosine** – một phép đo góc giữa hai vector nhằm xác

định mức độ định hướng tương đồng giữa chúng trong không gian từ vựng. Kết quả của phép tính này là một giá trị đại diện cho mức độ phù hợp giữa tài liệu và truy vấn, được sử dụng để **xếp hạng** tài liệu theo thứ tự giảm dần về độ liên quan.

$$\text{similarity}(\text{Rep}(d), \text{Rep}(q)) \rightarrow \text{ranking} \tag{2.4}$$

Minh họa dưới đây cho ta cái nhìn trực quan về cách mà truy vấn và tài liệu được biểu diễn trong không gian hình học. Trong ví dụ, ba thuật ngữ: *color*, *vehicle*, *motion* tạo thành không gian ba chiều, và tập hợp các từ được ánh xạ thành các vector tương ứng trong không gian đó:

Hình 2.1: Minh họa mô hình không gian vector




Mô hình không gian vector, với khả năng chuyển hóa văn bản thành hình khối đại số, đã khẳng định vai trò không thể thiếu trong nền tảng các hệ thống tìm kiếm hiện đại – nơi sự phù hợp không chỉ nằm ở từ khóa chung, mà còn ở sự cộng hưởng định hướng giữa các biểu diễn toán học của ngôn ngữ.

## 2.4 Phân tích tài liệu và lập chỉ mục

### 2.4.1 Phân tích tài liệu

Phân tích tài liệu là bước khởi đầu có tính nền tảng trong quá trình truy xuất thông tin – một giai đoạn được xem như cánh cửa đầu tiên dẫn ngôn ngữ tự nhiên đến thế giới hình thức và toán học. Về bản chất, đây là quá trình **tiền xử lý văn bản** (text pre-processing), nhằm chuyển hóa dữ liệu ngôn ngữ thô – vốn thường không có cấu trúc và mang nhiều yếu tố nhiễu – thành dạng thức chuẩn hóa, dễ dàng cho việc mô hình hóa và tính toán về sau.

Trong thế giới của dữ liệu văn bản, tồn tại rất nhiều từ ngữ có tần suất xuất hiện cao nhưng lại **ít mang giá trị ngữ nghĩa phân biệt**, điển hình như các **stopwords** (ví dụ: *a*, *the*, *in*, *on*, ...), các biểu thức cú pháp lặp lại, hay các ký hiệu đặc biệt gây nhiễu. Nếu không được xử lý triệt để, những thành phần này sẽ làm tăng kích thước không cần thiết của mô hình, gây suy giảm hiệu suất và độ chính xác trong quá trình truy vấn.

Bên cạnh đó, văn bản ngôn ngữ tự nhiên còn thường xuyên chứa các biến thể về mặt hình thức, dù mang cùng một ý nghĩa. Chẳng hạn, nếu trong tập từ vựng $V$ của hệ thống chỉ tồn tại từ "USA", nhưng người dùng lại nhập truy vấn dưới dạng "U.S.A", thì nếu không có bước chuẩn hóa – như loại bỏ dấu chấm và ghép từ – thì hệ thống sẽ không thể nhận ra hai biểu thức này là tương đương về ngữ nghĩa. Hệ quả là tài liệu có nội dung liên quan sẽ không được truy xuất, dẫn đến việc đánh mất thông tin giá trị.

Vì vậy, **phân tích và xử lý văn bản không chỉ là một bước kỹ thuật, mà còn là một quá trình "làm sạch" và "gột rửa" ngôn ngữ**, đưa nó về dạng tinh giản và thuần túy hơn, để từ đó, những biểu diễn toán học phía sau có thể hoạt động hiệu quả, chính xác và phản ánh đúng nhu cầu thông tin của người dùng. Đây chính là bước đi đầu tiên nhưng mang ý nghĩa quyết định trong việc xây dựng một hệ thống truy xuất thông tin thông minh, mạnh mẽ và giàu tính ngữ nghĩa.

### Tokenization

**Tokenization**, hay còn gọi là quá trình tách từ, là một bước cơ bản nhưng thiết yếu trong chuỗi tiền xử lý văn bản, đóng vai trò như cánh cổng đầu tiên đưa ngôn ngữ tự nhiên bước vào thế giới hình thức của truy xuất thông tin. Về nguyên lý, đây là hành động phân chia một văn bản thành những đơn vị nhỏ hơn – gọi là **token** – thường được xác định dựa trên các dấu phân cách như khoảng trắng, dấu câu hoặc ký tự đặc biệt. Trong tiếng Anh, việc phân tách này chủ yếu dựa theo dấu cách giữa các từ.

Một **token** được hiểu là biểu hiện cụ thể của một chuỗi ký tự – ví dụ như "to", "dream", hay "$10" – và nó được xem là một đơn vị có ý nghĩa trong quá trình xử lý ngôn ngữ. Khi nhiều token có cùng chuỗi ký tự, chúng được gom vào cùng một lớp trừu tượng gọi là **type**. Trên cơ sở đó, khi một type được hệ thống chấp nhận và lưu trữ trong **dictionary**, nó trở thành một **term** – tức là một từ khóa chính thức sẽ được dùng trong lập chỉ mục và truy vấn.

Lấy ví dụ với câu văn "to sleep perchance to dream": khi thực hiện tách từ, ta thu được 5 token, bao gồm hai lần xuất hiện của từ "to". Tuy nhiên, về mặt type thì chỉ có 4 loại khác nhau, và cũng chỉ có 4 term được thêm vào từ điển. Nếu hệ thống loại bỏ "to" vì đây là một stopword, thì kết quả cuối cùng chỉ còn lại 3 term mang giá trị ngữ nghĩa thực sự: *sleep*, *perchance* và *dream*.

Một ví dụ khác có thể thấy rõ tính chất của quá trình này là với câu: "There are only $10 for two tickets". Khi áp dụng tokenization đơn giản dựa trên khoảng trắng, ta thu được các token: **There**, **are**, **only**, **$10**, **for**, **two**, **tickets**, **(.)**. Mỗi token này sau đó sẽ trải qua các bước xử lý tiếp theo như loại bỏ stopword, chuẩn hóa ký tự hoặc gán trọng số để trở thành những thành phần hữu dụng trong hệ thống truy xuất thông tin.

Qua đó có thể thấy, tokenization không chỉ là một thao tác phân tách thuần túy, mà là bước đầu tiên để định hình ngữ nghĩa, thiết lập cấu trúc cho dữ liệu văn bản, và tạo tiền đề cho toàn bộ tiến trình phân tích và truy xuất phía sau.

### Lowercase

Một bước quan trọng trong tiến trình tiền xử lý văn bản là chuyển đổi tất cả ký tự về chữ thường (lowercasing), nhằm loại bỏ sự phân biệt giữa chữ hoa và chữ thường – vốn là đặc trưng không cần thiết trong hầu hết các hệ thống truy xuất thông tin. Việc này giúp đảm bảo rằng những từ có hình thức khác nhau nhưng mang cùng một ý nghĩa sẽ được xử lý thống nhất. Chẳng hạn, hai biểu thức "Vietnam" và "vietnam" nếu không được chuẩn hóa sẽ bị xem là hai term hoàn toàn khác biệt, dù về mặt ngữ nghĩa, chúng là một.

Trong một ví dụ minh họa đơn giản, ta có một tài liệu gồm các token sau khi thực hiện bước tách từ (tokenization): There, are, only, $, 10, for, two, tickets, (.). Sau khi áp dụng phép chuyển đổi về chữ thường, văn bản được chuẩn hóa thành: there, are, only, $, 10, for, two, tickets, (.).

Qua đó có thể thấy, thao tác hạ chuẩn chữ viết không làm thay đổi nội dung thông tin, nhưng lại mang ý nghĩa lớn về mặt hình thức, giúp mô hình truy xuất tránh được những sai lệch không cần thiết trong quá trình so khớp và tính toán. Đây là một bước nhỏ về mặt kỹ thuật, nhưng lại là một bước đi dài trong hành trình đưa ngôn ngữ về trạng thái chuẩn mực và tối ưu cho truy xuất.

### Loại bỏ punctuations

**Punctuation removal** – tức việc loại bỏ các ký tự dấu câu – là một thao tác tiền xử lý có vai trò tinh lọc văn bản, giúp loại bỏ những thành phần ký hiệu không mang lại giá trị đáng kể trong quá trình truy xuất thông tin. Tập hợp các dấu câu thường gặp bao gồm: `! , . ? : ; - – ( ) [ ] { } < > ' " … / \@ # $ % & * + = _ ` ~`, vốn được sử dụng để định hình cấu trúc câu trong văn bản tự nhiên, nhưng trong bối cảnh truy vấn, chúng lại trở thành rào cản đối với sự chuẩn hóa và đối sánh ngữ nghĩa.

Thật vậy, dấu câu không chỉ hiếm khi đóng góp ý nghĩa nội dung rõ ràng, mà còn có thể gây ra sự phân mảnh không mong muốn giữa các từ. Một ví dụ tiêu biểu là sự khác biệt giữa hai biểu thức "math" và "math." – nếu dấu chấm không được

loại bỏ, hệ thống sẽ xem đây là hai term hoàn toàn riêng biệt, từ đó làm giảm độ chính xác và khả năng truy xuất hiệu quả.

Trong minh họa sau, một văn bản ban đầu với các token như:

**there**, **are**, **only**, **$**, **10**, **for**, **two**, **tickets**, **.**

Sau khi áp dụng thao tác loại bỏ dấu câu, sẽ được chuẩn hóa thành:

**there**, **are**, **only**, **10**, **for**, **two**, **tickets**

Từ đó, có thể thấy rằng việc loại bỏ dấu câu không chỉ làm đơn giản hóa biểu diễn văn bản mà còn mang lại sự chính xác cần thiết trong việc khớp truy vấn và tài liệu. Đây là một bước đi tinh tế nhưng không thể thiếu trong quá trình biến ngôn ngữ tự nhiên thành dữ liệu có cấu trúc, giúp các mô hình truy xuất vận hành mượt mà và hiệu quả hơn

### Loại bỏ stopwords

**Stopword removal**, hay còn gọi là việc loại bỏ các từ dừng, là một thao tác quan trọng trong chuỗi tiền xử lý văn bản, nhằm tinh gọn dữ liệu ngôn ngữ và loại trừ những thành phần không mang giá trị phân biệt cao trong quá trình truy xuất thông tin. Stopwords là những từ xuất hiện với tần suất dày đặc trong hầu hết các tài liệu – như *a, an, is, in, of, the*, …– nhưng lại hiếm khi góp phần đáng kể vào việc xác định chủ đề hay nội dung cốt lõi của văn bản.

Việc giữ lại các từ như vậy không chỉ làm tăng kích thước không cần thiết của các vectơ biểu diễn, mà còn gây tốn kém về mặt tính toán do làm phình to không gian đặc trưng một cách vô nghĩa. Trong các mô hình biểu diễn toán học như ma trận term-document hoặc không gian vector, sự hiện diện của các stopword làm giảm độ sắc nét trong việc phân tách các khái niệm, và đôi khi còn gây nhiễu trong quá trình truy vấn. Do đó, loại bỏ stopword không đơn thuần là một hành vi cắt giảm, mà là một hành động thanh lọc – giữ lại những gì mang tính bản chất và loại trừ những yếu tố rườm rà.

Tuy nhiên, việc xác định stopword không phải lúc nào cũng mang tính tuyệt đối. Tùy vào ngữ cảnh sử dụng, lĩnh vực chuyên môn hay mục tiêu phân tích, tập hợp các từ cần loại bỏ có thể thay đổi linh hoạt. Một từ có thể vô nghĩa trong tài liệu này, nhưng lại mang tính chất quan trọng trong tài liệu khác.

Trong ví dụ trên, sau khi thực hiện thao tác loại bỏ stopwords, ta thu được kết quả sau: **10, two, tickets**.

Kết quả này phản ánh rõ ràng nội dung chính của câu, trong khi những từ còn lại – dù đóng vai trò ngữ pháp – không còn cần thiết cho mục tiêu truy xuất. Vậy nên, stopword removal chính là nghệ thuật của sự tinh lọc – nơi ngôn ngữ được bóc tách, gạn lọc để lại những hạt nhân ngữ nghĩa giàu giá trị cho quá trình phân tích và truy vấn.

### Stemming

**Stemming**, hay còn gọi là quá trình "lấy gốc từ", là một thao tác tiền xử lý mang tính nền tảng trong truy xuất thông tin, với mục tiêu chuẩn hóa các từ ngữ về dạng cơ bản nhất của chúng. Về bản chất, quá trình này tìm cách loại bỏ những biến thể ngữ pháp không cần thiết như hậu tố hoặc tiền tố, nhằm thu về một dạng từ thống nhất – thường được gọi là **stem**. Điều này không chỉ giúp giảm thiểu kích thước từ vựng mà còn tăng cường khả năng tổng quát hóa trong quá trình truy vấn, cho phép người dùng không cần nắm rõ mọi biến thể ngôn ngữ vẫn có thể tiếp cận đúng tài liệu họ cần.

Thật vậy, trong kho dữ liệu văn bản tự nhiên, sự phong phú về hình thái từ có thể tạo ra hàng loạt phiên bản khác nhau của cùng một ý nghĩa. Các động từ như *walked, walking, walks* dù mang hình thức khác nhau về ngữ pháp, nhưng tựu trung lại đều biểu đạt hành động "đi bộ". Với kỹ thuật stemming, những biến thể này sẽ được đưa về dạng gốc duy nhất là *walk* – thông qua thao tác cắt bỏ các hậu tố như *-ed*, *-ing*, *-s*. Các công cụ thực hiện quá trình này được gọi là **stemmer**, và dù phương pháp này mang tính quy tắc cứng nhắc, nó mang lại hiệu suất xử lý nhanh và phù hợp với những hệ thống yêu cầu tốc độ cao hơn độ chính xác hình thái.

Tuy nhiên, nếu nhu cầu về độ chính xác được đặt lên hàng đầu, người ta thường sử dụng một kỹ thuật tinh vi hơn mang tên **lemmatization**. Khác với stemming vốn chỉ dựa vào quy tắc ký tự đơn giản, lemmatization sử dụng từ điển hoặc tập ontology để xác định dạng gốc thực sự của một từ – được gọi là **lemma**. Nhờ đó, không chỉ các biến thể như *goes*, *going*, *went* được đưa về *go*, mà ngay cả các cặp danh từ bất quy tắc như *mouse* và *mice* cũng có thể được đồng nhất hóa. Quá trình này yêu cầu tra cứu từ vựng phức tạp hơn, đồng nghĩa với chi phí thời gian lớn hơn, nhưng đổi lại là độ chính xác vượt trội trong biểu diễn ngôn ngữ.

Tựu trung lại, việc chuẩn hóa hình thái từ ngữ không chỉ mang lại lợi ích rõ rệt trong việc giảm tải số lượng từ khóa cần lập chỉ mục, mà còn tăng cường tính nhất quán trong toàn bộ quá trình truy xuất. Khi số lượng từ được rút gọn và quy về dạng gốc, quá trình xây dựng chỉ mục trở nên nhẹ nhàng hơn, đồng thời tốc độ truy vấn cũng được cải thiện đáng kể.

Vẫn xét ví dụ như trên, kết quả xử lý lúc này sẽ là: **10**, **two**, **ticket**.

Mỗi từ còn lại đều mang trọng lượng ngữ nghĩa rõ rệt, phản ánh trực tiếp nội dung của văn bản mà không bị phân tán bởi các yếu tố ngữ pháp không cần thiết. Đây chính là minh chứng cho vai trò thiết yếu của stemming trong việc tinh lọc ngôn ngữ và tối ưu hóa hiệu suất của hệ thống truy xuất thông tin hiện đại.

### 2.4.2 Lập chỉ mục

#### Chỉ mục

**Lập chỉ mục** là một trong những bước cốt lõi và không thể thiếu trong quy trình xây dựng hệ thống truy xuất thông tin. Về bản chất, đây là quá trình thực hiện một phép quét duy nhất trên toàn bộ tập văn bản, với mục đích nhận diện và ghi nhận lại các **thuật ngữ** – bao gồm từ hoặc cụm từ – xuất hiện trong từng tài liệu. Mỗi thuật ngữ sau khi được phát hiện sẽ được gắn kèm theo các thông tin phụ trợ như **vị trí xuất hiện**, **tần suất lặp lại**, hay thậm chí là **độ quan trọng** của nó trong ngữ cảnh tài liệu tương ứng. Những thông tin quý giá này sau đó được tổ chức, cấu trúc và lưu trữ trong một thực thể dữ liệu đặc biệt – đó chính là **chỉ mục** (*index*).

Có thể hình dung đơn giản rằng, lập chỉ mục chính là hành động tạo nên một **cấu trúc dữ liệu trung gian**, nơi chứa đựng những thông tin thiết yếu được trích xuất từ tài liệu gốc, nhằm phục vụ trực tiếp cho quá trình truy vấn về sau. Nhờ vào cấu trúc này, thay vì phải quét lại toàn bộ tập hợp văn bản mỗi lần có yêu cầu tìm kiếm, hệ thống chỉ cần thao tác trên chỉ mục – một tập hợp đã được nén gọn, tổ chức hợp lý và tối ưu cho việc tra cứu.

Bảng 2.1: Ví dụ về chỉ mục

| | $t_1$ | $t_2$ | ... | $t_m$ |
|---|---|---|---|---|
| $d_1$ | 1 | 0 | ... | 1 |
| ... | ... | ... | ... | 1 |
| $d_n$ | 1 | 1 | ... | 1 |

Quá trình lập chỉ mục thường diễn ra với tốc độ nhanh chóng, bởi các thuật ngữ được ghi nhận theo cách **gia tăng tuyến tính**, tức là mỗi khi một từ khóa mới được phát hiện, nó sẽ ngay lập tức được thêm vào chỉ mục với các thông tin đi kèm được cập nhật đồng thời. Tuy nhiên, chính sự phong phú và không đồng đều của ngôn ngữ tự nhiên lại tạo nên một thách thức đáng kể – đó là **kích thước chỉ mục có thể phình to nhanh chóng**, đặc biệt khi làm việc với kho dữ liệu lớn. Kết quả là, trong khi việc xây dựng chỉ mục có thể được hoàn tất nhanh chóng, thì quá trình tìm kiếm thông tin – nếu không có thêm biện pháp tối ưu – có thể gặp trở ngại do chi phí xử lý gia tăng theo độ lớn của cấu trúc chỉ mục.

Tuy nhiên, dù có những hạn chế nhất định về mặt tài nguyên, lập chỉ mục vẫn là một bước đi mang tính chiến lược – như việc xây dựng một bản đồ tường minh cho vùng dữ liệu ngôn ngữ, nơi mà mỗi bước truy vấn sau này đều có thể lần theo dấu vết của từ ngữ một cách rõ ràng, chính xác và hiệu quả.

### Chỉ mục đảo (Inverted Index)

**Chỉ mục đảo** (*Inverted Index*) là một hình thức tổ chức dữ liệu đặc biệt, được thiết kế để tối ưu hóa khả năng truy vấn trong các hệ thống tìm kiếm thông tin. Khác với cách tiếp cận tuyến tính thông thường, trong đó ta phải quét qua từng tài liệu để kiểm tra sự hiện diện của một thuật ngữ, thì với chỉ mục ngược, mỗi từ khóa sẽ được **ánh xạ ngược trở lại** – tức là liên kết trực tiếp đến danh sách các tài liệu mà nó xuất hiện. Đây chính là một cấu trúc dữ liệu có khả năng trả lời cho câu hỏi: *"Từ khóa này nằm ở đâu trong kho dữ liệu?"*, thay vì *"Kho dữ liệu này chứa những gì?"*.

Cấu trúc này hoạt động như một chiếc bản đồ hai chiều, trong đó mỗi thuật ngữ giữ vai trò như một cổng chỉ dẫn, mở ra một danh sách các tài liệu liên quan – một cách tiếp cận vừa tinh gọn, vừa giàu hiệu quả, đặc biệt trong những kho văn bản có quy mô lớn và yêu cầu truy xuất nhanh chóng, chính xác.

Quy trình xây dựng một chỉ mục đảo được tiến hành theo từng bước rõ ràng và chặt chẽ. Đầu tiên, hệ thống sẽ **quét toàn bộ tập tài liệu**, tiến hành xử lý văn bản để thu thập ra danh sách các từ đơn lẻ, đã được chuẩn hóa. Tại bước này, những thuật ngữ đã từng được xét và đã được đưa vào chỉ mục sẽ được **loại bỏ**, tránh việc lặp lại không cần thiết, qua đó đảm bảo tính hiệu quả trong lưu trữ và truy xuất.

Tiếp theo, với danh sách các từ khóa còn lại – tức là các thuật ngữ duy nhất – hệ thống sẽ **tạo lập một bảng ánh xạ**, trong đó mỗi từ khóa sẽ liên kết đến một tập hợp các tài liệu mà nó xuất hiện. Đây chính là phần cốt lõi của chỉ mục ngược: *mỗi từ khóa là một nút trung tâm, được nối với các tài liệu liên quan như những nhánh cây lan tỏa trong rừng dữ liệu.*

Quy trình này sẽ được **lặp lại tuần tự** cho đến khi toàn bộ tài liệu trong kho dữ liệu được xử lý và mọi thuật ngữ tiềm năng đều đã được ánh xạ một cách đầy đủ. Kết quả cuối cùng là một cấu trúc chỉ mục vững chắc, cho phép các thao tác truy vấn về sau diễn ra không chỉ nhanh mà còn có tính định hướng cao, như thể ta đang tra cứu một từ điển chuyên biệt mà mỗi mục từ lại chính là cánh cổng dẫn tới tri thức tiềm ẩn trong vô vàn trang văn bản.

## 2.5 Truy xuất chỉ mục

Trong quá trình truy xuất chỉ mục, những thứ ta cần quan tâm sẽ bao gồm: concept vectors và trọng số (weight) cho các vectors đó.

### 2.5.1 Concept Vectors

**Concept Vectors** là một phương pháp biểu diễn trừu tượng, mang tính hình thức hóa cao, được sử dụng để mã hóa cả **tài liệu (document)** lẫn **truy vấn (query)** dưới dạng các vector trong một không gian ngữ nghĩa đa chiều. Trong không gian này, mỗi chiều không còn đơn thuần đại diện cho một từ khóa (term) cụ thể như trong mô hình không gian vector cổ điển, mà thay vào đó là một **khái niệm (concept)** – một đơn vị thông tin mang tính tổng quát và trừu tượng hơn, phản ánh sâu sắc bản chất ngữ nghĩa của nội dung. [1]

Việc **truy xuất chỉ mục** trong mô hình này không còn dựa trực tiếp vào từ khóa bề mặt, mà được thực hiện thông qua sự so sánh giữa các **vector khái niệm** (concept vectors) – vốn là những đại diện ngữ nghĩa sâu của cả tài liệu và câu truy vấn. Mỗi concept vector được cấu trúc như một vector trong không gian nhiều chiều, trong đó **mỗi chiều tượng trưng cho một khái niệm**, và các **trọng số (weights)** tương ứng với từng chiều biểu thị mức độ hiện diện hoặc tầm quan trọng của khái niệm đó trong nội dung được xét.

Minh họa rõ ràng cho cách biểu diễn này có thể thấy trong 2.2, nơi các tài liệu – cụ thể là danh sách các cuốn sách giáo khoa môn Toán do Bộ Giáo dục và Đào tạo Việt Nam ban hành – được mã hóa thành các concept vector. Hàng đầu tiên trong bảng biểu diễn là tập hợp các sách, trong khi cột đầu tiên liệt kê các từ hoặc khái niệm then chốt. Các ô giao giữa hàng và cột được sử dụng để biểu thị sự hiện diện hay vắng mặt của từng khái niệm trong mỗi cuốn sách – từ đó tạo nên các vector ngữ nghĩa đặc trưng cho từng tài liệu.

Bảng 2.2: Ví dụ về Concept Vectors với sách giáo khoa

| | Toán 12 | Toán 11 | Toán 10 | Toán 9 | Toán 8 | Toán 7 |
|---|---|---|---|---|---|---|
| Hình học không gian | 1 | 1 | 1 | 0 | 0 | 0 |
| Đệ quy | 1 | 1 | 0 | 0 | 0 | 0 |
| Đạo hàm | 1 | 0 | 0 | 0 | 0 | 0 |
| Ma trận | 1 | 0 | 1 | 1 | 0 | 0 |
| Lượng giác | 0 | 1 | 1 | 0 | 1 | 1 |

Thông qua biểu diễn như vậy, mỗi tài liệu trở thành một điểm trong không gian khái niệm, và phép đo độ tương đồng giữa truy vấn và tài liệu được chuyển hóa thành bài toán hình học – nơi mà các khái niệm không còn bị giới hạn bởi sự khác biệt bề mặt về từ vựng, mà được liên kết thông qua ý nghĩa sâu xa hơn. **Concept Vector**, do đó, không chỉ là một công cụ biểu diễn, mà còn là một cầu nối ngữ nghĩa giữa câu truy vấn của con người và thế giới tri thức ẩn sâu bên trong các tài liệu.

### 2.5.2 Trọng số (weight) cho các vectors

Trong không gian truy xuất thông tin, **việc xác định và gán trọng số (weights) cho các thành phần trong vector biểu diễn tài liệu và truy vấn** không chỉ đơn thuần là một thao tác kỹ thuật, mà còn là yếu tố cốt lõi, chi phối sâu sắc đến **độ chính xác, độ nhạy** và **khả năng phản ánh ngữ nghĩa thực tế** của hệ thống xếp hạng tài liệu. Thực tiễn cho thấy, không phải mọi từ trong ngôn ngữ đều sở hữu **mức độ quan trọng như nhau**, và do đó, việc sử dụng một cách cơ học **tần suất xuất hiện (term frequency)** của từ như là một trọng số duy nhất sẽ không đủ để phản ánh chiều sâu thông tin ẩn chứa trong mỗi văn bản.

Ở phương diện phân tích tài liệu, có những từ – mặc dù xuất hiện ít – lại mang nặng hàm lượng ngữ nghĩa, đóng vai trò như những **chìa khóa ngữ nghĩa (semantic cues)**, giúp phân định rõ ràng nội dung hoặc chủ đề cốt lõi của văn bản. Trái lại, những từ thường xuyên xuất hiện trong hầu hết các tài liệu – dù mang tính phổ quát – lại thường không đóng góp nhiều vào việc phân biệt hay nhận diện sự

khác biệt giữa các nội dung. Do đó, **việc đánh giá trọng số cần được thực hiện một cách tinh tế**, trong đó phải cân nhắc đồng thời **mức độ xuất hiện trong từng tài liệu riêng biệt**, cũng như **tần suất lan tỏa của từ đó trong toàn bộ tập hợp tài liệu (corpus)**.

Một ví dụ điển hình có thể minh họa cho vấn đề này là trong một tập hợp các tài liệu chuyên ngành về công nghiệp ô-tô, từ "ô-tô" – dù có xuất hiện với tần suất cao – lại trở nên kém giá trị trong việc phân biệt các tài liệu, bởi nó mang tính nền tảng và phổ biến đến mức không còn giúp ích nhiều cho việc nhận diện sự khác biệt giữa các văn bản. Chính vì thế, **các kỹ thuật gán trọng số hiện đại như TF-IDF (Term Frequency-Inverse Document Frequency)** đã được đề xuất, nhằm điều chỉnh và hiệu chỉnh lại ảnh hưởng của những từ như vậy. Ý tưởng trung tâm của phương pháp này là **giảm trọng số của các thuật ngữ có tần suất xuất hiện cao trên toàn bộ tập tài liệu**, từ đó làm nổi bật các từ hiếm – những thành phần mang dấu ấn cá nhân và chiều sâu chuyên biệt của từng tài liệu.

### Term Frequency (TF)

Trong lĩnh vực xử lý ngôn ngữ tự nhiên và khai thác dữ liệu văn bản, **tần số thuật ngữ (Term Frequency – TF)** là một đại lượng cơ bản nhưng mang giá trị cốt lõi trong việc định lượng mức độ xuất hiện của một từ hoặc cụm từ trong một tài liệu nhất định. Khái niệm này đóng vai trò như một chỉ số phản ánh **mức độ nổi bật nội tại** của thuật ngữ trong bối cảnh của tài liệu đó, từ đó góp phần xác định tầm quan trọng tương đối của nó so với các thuật ngữ khác cùng tồn tại trong văn bản.

Về mặt hình thức, term frequency được tính bằng cách **lấy số lần xuất hiện thực tế của một từ khóa** trong tài liệu, sau đó **chia cho tổng số từ** trong tài liệu đó. Tỷ lệ này mang ý nghĩa thống kê rõ rệt: một từ xuất hiện với tần suất cao trong một văn bản cụ thể nhiều khả năng sẽ đóng vai trò then chốt trong việc phản ánh nội dung hoặc chủ đề mà tài liệu đề cập tới. Ngược lại, những từ hiếm gặp – nếu xuất hiện – sẽ chỉ chiếm một phần rất nhỏ trong tổng thể và do đó có chỉ số TF thấp hơn. [3]

Công thức thường thấy cho TF [3]:

$$tf(t,d) = \frac{f_{t,d}}{\sum_{t'} f_{t',d}} \tag{2.5}$$

Chính vì đặc tính tỷ lệ này, TF không chỉ đơn thuần là một con số, mà là một **dạng lượng hóa ý nghĩa ngôn ngữ**, cho phép mô hình hóa văn bản như một cấu trúc số liệu mà ở đó các đơn vị từ vựng được định vị và định lượng một cách khách quan. Trong thực tế, TF là thành phần quan trọng trong nhiều hệ thống đánh giá thông tin, đặc biệt là khi kết hợp với các cơ chế khác nhằm hiệu chỉnh ảnh hưởng lan rộng của từ trong toàn bộ tập tài liệu.

### Inverse Document Frequency

Trong lĩnh vực xử lý ngôn ngữ tự nhiên và khai phá dữ liệu văn bản, **Inverse Document Frequency (IDF)** là một đại lượng thống kê mang tính nền tảng, được thiết kế để định lượng **độ hiếm tương đối** của một từ hoặc cụm từ trong toàn bộ tập hợp tài liệu. Nếu như **Term Frequency (TF)** phản ánh mức độ phổ biến của một thuật ngữ trong nội bộ một tài liệu cụ thể, thì **IDF** lại đo lường **khả năng phân biệt** của thuật ngữ đó trên bình diện toàn cục – tức là, trong phạm vi toàn bộ tập tài liệu đang xét.

Ý tưởng trung tâm của IDF nằm ở chỗ: những từ ngữ xuất hiện trong rất nhiều tài liệu - như các từ chức năng hoặc từ phổ thông – thường không cung cấp nhiều giá trị thông tin cho quá trình phân loại hay truy xuất. Ngược lại, một từ xuất hiện **hiếm hoi nhưng đều đặn** chỉ trong một số ít tài liệu sẽ mang hàm lượng thông tin cao hơn, vì nó có thể giúp **phân biệt rõ ràng** các chủ đề hoặc nội dung khác nhau.

Về mặt hình thức, IDF của một từ được tính bằng **logarithm** của tỉ số giữa tổng số tài liệu trong tập dữ liệu và số lượng tài liệu trong đó từ đó xuất hiện ít nhất một lần:

$$\text{IDF}(t) = \log\left(\frac{N}{n}\right) \tag{2.6}$$

trong đó:

- $N$ là tổng số tài liệu trong tập dữ liệu,
- $n$ là số tài liệu mà từ $t$ xuất hiện ít nhất một lần.

Giá trị của hàm logarit trong công thức trên không chỉ giúp làm trơn phân phối tần suất mà còn bảo đảm rằng IDF không tăng quá nhanh đối với những từ cực hiếm – giữ cho thang đo luôn trong giới hạn kiểm soát được. Khi được kết hợp cùng TF trong công thức **TF-IDF**, IDF góp phần làm nổi bật các từ ngữ không chỉ xuất hiện thường xuyên trong một tài liệu, mà còn đồng thời **ít phổ biến trong toàn bộ tập hợp** – từ đó nâng cao độ chính xác và sắc thái ngữ nghĩa trong các mô hình truy xuất thông tin hiện đại.

### Term Frequency – Inverse Document Frequency (TF-IDF)

**TF-IDF** là một kỹ thuật thống kê phổ biến và hiệu quả trong lĩnh vực xử lý ngôn ngữ tự nhiên và khai thác dữ liệu văn bản, nhằm đo lường mức độ **quan trọng** của một thuật ngữ đối với một tài liệu trong toàn bộ bộ sưu tập dữ liệu (corpus).

Phương pháp này kết hợp hai thành phần trọng yếu:

- **Term Frequency (TF)**, biểu thị tần suất xuất hiện của một từ hoặc cụm từ trong một tài liệu cụ thể.
- **Inverse Document Frequency (IDF)**, phản ánh mức độ **hiếm** của từ đó trong toàn bộ tập tài liệu, qua việc sử dụng hàm logarit của tỷ lệ giữa tổng số tài liệu và số tài liệu chứa từ đó.

Công thức chung được xây dựng dưới dạng:

$$\text{TF-IDF}(t, d, D) = \text{TF}(t, d) \cdot \text{IDF}(t, D) \tag{2.7}$$

trong đó:

- $\text{TF}(t, d)$ là tỷ lệ giữa số lần từ $t$ xuất hiện trong tài liệu $d$ và tổng số từ trong $d$;
- $\text{IDF}(t, D) = \log \frac{N}{n_t}$, với $N$ là tổng số tài liệu trong tập dữ liệu $D$, và $n_t$ là số tài liệu chứa ít nhất một lần từ $t$.

Về mặt ngữ nghĩa, TF-IDF giúp **làm sáng** những từ khóa có tần suất cao trong từng tài liệu cụ thể (qua TF), đồng thời **hạ thấp vai trò của những từ phổ thông**, xuất hiện rộng khắp trong corpus (qua IDF). Kết quả thu được là một đại lượng trọng số, cho biết **độ quan trọng tổng hòa** của thuật ngữ đó trong ngữ cảnh của tài liệu lẫn toàn bộ tập hợp văn bản.

Nhờ khả năng cân bằng giữa tần suất nội dung và mức độ phân biệt khắp bộ dữ liệu, TF-IDF trở thành một công cụ đắc lực được ứng dụng trong các hệ thống tìm kiếm, phân loại văn bản, trích xuất đặc trưng từ dữ liệu văn bản, và nhiều ứng dụng NLP khác, giúp nâng cao độ chính xác và mức độ thẩm thấu sâu sắc của các mô hình truy xuất thông tin.

### Vector hóa tài liệu và truy vấn

Một điểm cần đặc biệt lưu ý trong toàn bộ quy trình tiền xử lý và biểu diễn dữ liệu là cả tài liệu (*document*) và câu truy vấn (*query*) đều phải được biến đổi và ánh xạ về dạng vector theo cùng một phương pháp thống nhất. Điều này xuất phát từ yêu cầu tất yếu của tính đồng nhất trong không gian biểu diễn: nếu hai thực thể không được xử lý đồng nhất, các thuật ngữ (terms) xuất hiện trong truy vấn có thể không tương thích hoặc không trùng khớp với các thuật ngữ trong tài liệu, dẫn đến việc suy giảm nghiêm trọng hiệu quả truy xuất thông tin. Việc chuẩn hóa biểu diễn không chỉ đảm bảo sự tương thích giữa truy vấn và tài liệu, mà còn nâng cao độ chính xác trong việc tính toán độ tương đồng và xếp hạng kết quả.

Xét một ví dụ minh họa để làm rõ tầm quan trọng của việc áp dụng quy trình tiền xử lý một cách nhất quán giữa tài liệu và truy vấn. Giả sử ta có hai tài liệu và một câu truy vấn, đồng thời sử dụng một tập từ dừng (*stopwords*) được rút gọn từ danh sách tiếng Anh trong thư viện NLTK.

Cụ thể, hai tài liệu được giữ nguyên mà không loại bỏ từ dừng:

- *doc1*: *what have i done and why who s this*
- *doc2*: *what have you done*

Trong khi đó, câu truy vấn có hai phiên bản:

- *query1* (chưa loại bỏ stopwords): *what have i done*
- *query2* (đã loại bỏ stopwords): *done*

Với *query1*, hệ thống có thể xếp hạng *doc1* cao hơn *doc2* vì toàn bộ các term trong truy vấn đều được tìm thấy đầy đủ trong *doc1*, tạo ra một mức độ tương đồng ngữ nghĩa cao hơn. Ngược lại, với *query2*, khi chỉ còn lại từ khóa "*done*", cả hai tài liệu đều chứa từ này, dẫn đến việc hệ thống đánh giá chúng ngang nhau về mặt liên quan – làm giảm độ chính xác trong việc lựa chọn tài liệu phù hợp nhất.

Tình huống này cho thấy nếu quá trình xử lý được áp dụng không đồng đều giữa truy vấn và tài liệu – trong trường hợp này là chỉ loại bỏ stopwords ở truy vấn – thì hiệu quả truy xuất thông tin có thể bị suy giảm đáng kể. Do đó, việc đảm bảo một quy trình xử lý thống nhất giữa hai thực thể này là điều kiện tiên quyết để duy trì tính nhất quán và độ chính xác trong các mô hình truy xuất thông tin hiện đại.

Dưới đây là một ví dụ điển hình về việc gán trọng số TF-IDF cho một tập tài liệu và một truy vấn cụ thể bằng tiếng Anh:

Giả sử chúng ta có ba tài liệu trong corpus:

Bảng 2.3: Ví dụ tài liệu cần gán trọng số

| **Tài liệu** | **Nội dung** |
|---|---|
| $d_1$ | The cat sat on the mat |
| $d_2$ | My dog and cat are the best |
| $d_3$ | The locals are playing |

Và một truy vấn đơn giản:

**"The cat"**

Bước đầu tiên, chúng ta biểu diễn mỗi tài liệu và truy vấn dưới dạng vector TF-IDF. Ví dụ, tính TF-IDF cho các từ trong truy vấn như sau:

- TF("cat", $d_1$) = $\frac{1}{6}$; IDF("cat") = $\log\left(\frac{3}{2}\right) \approx 0.18$, nên TF-IDF $\approx 0.03$
- TF("cat", $d_2$) = $\frac{1}{7}$; IDF("cat") = 0.18; TF-IDF $\approx 0.025$
- TF("cat", $d_3$) = 0; nên TF-IDF = 0

Trong khi đó, từ "the" có IDF = $\log(\frac{3}{3}) = 0$, dẫn đến TF-IDF luôn bằng 0, bất kể tài liệu nào chứa từ này.

Sau khi tính toán, chúng ta được hai vector TF-IDF:

Bảng 2.4: Giá trị TF-IDF lập theo term cho từng tài liệu

| **Term** | $d_1$ **TF-IDF** | $d_2$ **TF-IDF** | $d_3$ **TF-IDF** | **Query TF-IDF** |
|---|---|---|---|---|
| **the** | 0 | 0 | 0 | 0 |
| **cat** | 0.0306 | 0.0252 | 0 | 0.18 |

Tiếp theo, biểu diễn truy vấn "The cat" cũng dưới dạng TF-IDF vector — ở đây, "cat" nhận trọng số 0.18, còn "the" vẫn là 0.

Cuối cùng, hệ thống tính **độ tương đồng cosine** giữa vector truy vấn và từng vector tài liệu, rồi xếp hạng dựa trên giá trị đó:

1. $d_1$ – có độ tương đồng cao nhất (vì chứa đầy đủ cả hai term),
2. $d_2$ – là lựa chọn thứ hai,
3. $d_3$ – đứng cuối cùng.

Thông qua ví dụ này, chúng ta thấy rõ cách TF-IDF không chỉ phản ánh sự hiện diện của từ trong tài liệu, mà còn điều chỉnh theo mức độ phân bố của từ đó trong toàn bộ tập dữ liệu. Kết quả là, những thuật ngữ đặc trưng hơn cho một chủ đề sẽ được hệ thống ưu tiên đánh trọng số – và từ đó, truy vấn "The cat" cũng dẫn đến kết quả chính xác và có ý nghĩa hơn.

### Chuẩn hóa trọng số

Chuẩn hóa trọng số, hay còn gọi là *normalize*, là một bước tinh tế nhưng thiết yếu trong tiến trình biểu diễn văn bản dưới dạng vector trọng số, nhằm đảm bảo rằng mọi tài liệu, bất kể độ dài, đều được định lượng một cách công bằng và đồng nhất trong không gian truy xuất. Khi biểu diễn các tài liệu hoặc truy vấn bằng vector, nếu không có bước chuẩn hóa, các tài liệu dài với nhiều lần lặp lại của các term sẽ có xu hướng chiếm ưu thế không công bằng so với các tài liệu ngắn hơn, mặc dù có thể mức độ liên quan của chúng lại không thực sự cao hơn. Bằng cách đưa các trọng số về cùng một miền giá trị chuẩn hóa – thường là đoạn $[0, 1]$ – ta không những làm giảm ảnh hưởng không mong muốn của độ dài tài liệu, mà còn biến đổi các vector thành các đại diện đơn vị, có độ dài bằng 1 và cùng phương hướng trong không gian vector chuẩn. Điều này tạo điều kiện thuận lợi cho việc áp dụng các phép đo như cosine similarity, vì mọi phép đo khi đó chỉ còn phụ thuộc vào hướng của vector – tức là nội dung ngữ nghĩa – thay vì độ lớn của nó. Về mặt kỹ thuật, một trong những công thức chuẩn hóa trọng số thường dùng là:

$$w_{ik} = \frac{tf_{ik} \cdot idf_{ik}}{\sqrt{\sum_{k=1}^{N} (tf_{ik} \cdot idf_{ik})^2}} \tag{2.8}$$

trong đó $w_{ik}$ là trọng số chuẩn hóa của term $k$ trong tài liệu $i$, đảm bảo rằng tổng bình phương các trọng số của mọi term trong một document sẽ luôn bằng 1. Ngoài ra, trong thực tiễn triển khai, người ta còn sử dụng những phương pháp chuẩn hóa khác như hệ thống S.M.A.R.T., trong đó trọng số được xác định theo công thức:

$$w_{kd} = collect_k \cdot fred_{kd} \cdot \frac{1}{norm} \tag{2.9}$$

với mỗi biến số phản ánh các yếu tố đặc trưng trong mối quan hệ giữa term, tài liệu và tập hợp tài liệu. Những phương pháp như vậy giúp điều chỉnh độ quan trọng của từ khóa không chỉ dựa vào tần suất xuất hiện, mà còn kết hợp với các yếu tố về ngữ cảnh và phân bố để nâng cao tính chính xác và công bằng trong hệ thống truy xuất thông tin.

## 2.6 Tính độ tương đồng và xếp hạng

Khi các tài liệu (documents) và câu truy vấn (queries) đã được ánh xạ thành các vectơ trọng số trong không gian |V| chiều, nơi |V| đại diện cho kích thước của từ vựng (vocabulary) hay chính là số lượng các thuật ngữ (terms) riêng biệt, mỗi vectơ trong không gian này không đơn thuần chỉ là một chuỗi số liệu trừu tượng mà là một điểm mang giá trị hình học cụ thể, có vị trí, có hướng, và có độ lớn. Mỗi điểm được định danh bởi tập hợp các trọng số tương ứng với các term trong V, tuy nhiên, để tạo thành một vectơ hoàn chỉnh – một đối tượng có phương và độ dài – thì ta cần thêm một điểm mốc để định hướng chuyển động. Điểm mốc đó không gì khác chính là gốc tọa độ $\mathbf{0} = (0, 0, \ldots, 0)$ trong không gian |V| chiều, đóng vai trò như điểm khởi đầu của mọi vectơ biểu diễn document hoặc query. Nhờ có điểm gốc này, mỗi document hoặc query được hình dung như một vectơ kéo dài từ điểm $\mathbf{0}$ đến vị trí tương ứng với tập trọng số của chính nó trong không gian.

Sau khi đã hoàn tất giai đoạn gán trọng số cho từng thành phần trong các vectơ tài liệu và truy vấn – tức là sau khi đã xây dựng được biểu diễn đại số của chúng – bài toán tiếp theo cần giải quyết là xác định mức độ liên quan (relevance) giữa mỗi tài liệu và truy vấn. Vấn đề đặt ra ở đây là: làm thế nào để xếp hạng các tài liệu

trong không gian |V| chiều theo độ phù hợp với truy vấn đã cho? Câu trả lời nằm chính trong cách thức ta biểu diễn các tài liệu và truy vấn như những vectơ trong cùng một không gian hình học. Bởi vì tất cả các vectơ đều được định nghĩa cùng trong một hệ tọa độ và cùng dựa trên cùng một từ vựng, ta có thể tiến hành so sánh sự gần nhau – hay còn gọi là *proximity* – giữa chúng bằng các phương pháp định lượng. Trong ngữ cảnh này, proximity chính là một dạng của similarity (độ tương đồng), phản ánh qua khoảng cách hình học hoặc góc giữa các vectơ. Sự tương đồng càng cao, tức là vectơ truy vấn càng gần với vectơ tài liệu trong không gian |V| chiều, thì mức độ liên quan của tài liệu đó đối với truy vấn càng lớn. Từ đây, việc xếp hạng các tài liệu trở thành một quá trình đo đạc và sắp xếp độ gần giữa các vectơ tài liệu và vectơ truy vấn, dựa trên các hàm đo cụ thể như cosine similarity hay các chuẩn hình học khác. Bước kế tiếp trong tiến trình truy xuất chính là định lượng sự gần gũi ấy một cách chính xác, để xây dựng nên một thứ tự truy xuất có ý nghĩa về mặt ngữ nghĩa và hình học

### 2.6.1 Độ tương đồng cosine

Độ đo tương đồng cosine, hay còn gọi là *cosine similarity*, là một phương pháp hình học được áp dụng phổ biến trong các mô hình không gian vector nhằm định lượng mức độ tương đồng giữa hai vector trong một không gian nhiều chiều. Trong lĩnh vực xử lý ngôn ngữ tự nhiên, phương pháp này đặc biệt hữu ích trong việc đánh giá mức độ tương quan về ngữ nghĩa giữa hai văn bản, nhờ khả năng phản ánh định hướng của các vector hơn là độ lớn tuyệt đối của chúng.

Cosine similarity được định nghĩa dựa trên góc giữa hai vector trong không gian: khi hai vector cùng hướng, góc giữa chúng nhỏ, và độ đo cosine tiến gần đến 1; khi hai vector vuông góc, góc là 90 độ, cosine bằng 0, cho thấy sự không tương đồng; và nếu chúng ngược hướng, cosine đạt giá trị -1, phản ánh một mức độ tương phản hoàn toàn. Công thức tính cosine similarity giữa hai vector A và B được viết dưới dạng:

$$\text{cosine\_similarity}(A, B) = \frac{A \cdot B}{\|A\| \times \|B\|} \tag{2.10}$$

trong đó $A \cdot B$ là tích vô hướng giữa hai vector, còn $|A|$ và $|B|$ là độ dài (chuẩn Euclid) của hai vector tương ứng. Giá trị kết quả luôn nằm trong đoạn $[-1, 1]$, phản ánh trực quan mối quan hệ định hướng giữa hai thực thể trong không gian biểu diễn.

Chính vì khả năng bỏ qua sự khác biệt về độ dài và tập trung vào hướng chung của các vector, cosine similarity đã trở thành một công cụ trọng yếu trong các bài toán như tìm kiếm thông tin, phân loại văn bản, gợi ý nội dung và các ứng dụng khác trong khai phá dữ liệu văn bản.

## 2.7 Nhận xét

Dưới ánh sáng của quá trình khảo cứu toàn diện về mô hình Không gian Vector (*Vector Space Model*), ta có thể đi đến một cái nhìn tổng thể về những khả năng cốt lõi mà mô hình này mang lại cho nhiệm vụ truy vấn thông tin trong các hệ thống hiện đại. Trước hết, mô hình cho phép thực hiện việc **xếp hạng các kết quả truy vấn** một cách linh hoạt và định lượng – nghĩa là không chỉ xác định xem một tài liệu có liên quan hay không, mà còn cho phép đo lường mức độ liên quan ấy để sắp xếp tài liệu theo trật tự ưu tiên. Đây là một điểm mạnh nổi bật so với các mô hình nhị phân cổ điển.

Mặt khác, cả **tài liệu và câu truy vấn đều được mô hình hóa dưới dạng các vector số thực** trong một không gian nhiều chiều, trong đó mỗi chiều đại diện cho một *term* (thuật ngữ chỉ mục) trong tập từ vựng đã được chuẩn hóa và thống nhất. Việc biểu diễn như vậy không chỉ giúp tiêu chuẩn hóa cách thức hệ thống nhìn nhận các thực thể ngôn ngữ, mà còn mở đường cho các phép toán đại số tuyến tính có thể được áp dụng một cách hiệu quả để phân tích mối quan hệ giữa các văn bản.

**Mỗi chiều của vector đại diện cho một chỉ mục cụ thể**, và giá trị trên chiều ấy – hay còn gọi là trọng số – là một số thực phản ánh **mức độ quan trọng của thuật ngữ đó trong ngữ cảnh của văn bản hoặc câu truy vấn**. Những trọng số này, thường được tính bằng các công thức như TF, IDF hay TF-IDF, là nền tảng để tạo ra các biểu diễn giàu thông tin.

Từ những biểu diễn này, mô hình sử dụng **độ đo cosine** để xác định **độ tương đồng giữa hai vector**, chính là giữa truy vấn và tài liệu. Phép đo này phản ánh mức độ định hướng tương đồng giữa hai thực thể trong không gian ngữ nghĩa, cho phép suy ra mức độ liên quan một cách mượt mà và hình học.

Không dừng lại ở đó, vì mọi thực thể đều được quy đổi về dạng vector – dù là tài liệu, truy vấn, hay thậm chí là một câu văn đơn lẻ – nên **mọi vector trong hệ thống đều có thể được đem ra so sánh trực tiếp với nhau về mặt ngữ nghĩa**. Điều này mở ra một không gian thao tác phong phú, nơi các thực thể ngôn ngữ được phân tích như các điểm hình học, giúp hệ thống truy xuất vận hành theo lối tư duy toán học chính xác và nhất quán.

Qua toàn bộ tiến trình nghiên cứu, có thể thấy rõ rằng mô hình Không gian Vector không chỉ đơn thuần là một kỹ thuật biểu diễn, mà còn là một triết lý mô hình hóa ngôn ngữ – nơi văn bản được hiểu như những vector sống động, sẵn sàng được so sánh, xếp hạng và suy diễn trong một vũ trụ hình học đầy tiềm năng. Những ưu điểm và hạn chế cụ thể của mô hình sẽ được trình bày chi tiết hơn ở phần sau, nhằm tiếp tục làm rõ vị thế của mô hình này trong hệ sinh thái truy xuất thông tin hiện đại

### 2.7.1 Ưu điểm

Mô hình Không gian Vector (*Vector Space Model*) sở hữu nhiều ưu điểm nổi bật, khiến nó trở thành một trong những phương pháp được chấp nhận rộng rãi và ứng dụng phổ biến trong lĩnh vực truy xuất thông tin. Trước hết, đây là một mô hình đã được nghiên cứu sâu rộng trong suốt nhiều thập kỷ, từ đó hình thành nên một nền tảng lý thuyết vững chắc và được củng cố qua vô số thực nghiệm. Tính lịch sử này không chỉ mang lại độ tin cậy cao mà còn tạo điều kiện thuận lợi cho việc tiếp cận, học tập và áp dụng mô hình trong thực tiễn.

Bên cạnh đó, mô hình cũng được đánh giá cao nhờ vào **cấu trúc đơn giản, dễ hiểu và dễ triển khai**, giúp người thực hiện có thể nhanh chóng hiện thực hóa các nguyên lý lý thuyết thành hệ thống hoạt động hiệu quả. Một điểm mạnh then chốt khác là khả năng **xếp hạng các tài liệu truy xuất được dựa trên mức**

**độ liên quan định lượng** với truy vấn đầu vào – một khả năng vượt trội so với các mô hình nhị phân vốn chỉ cho phép phân loại rạch ròi tài liệu là liên quan hoặc không liên quan.

Cuối cùng, **tính linh hoạt trong việc gán trọng số cho các thuật ngữ** cũng là một ưu điểm lớn của mô hình này. Với nhiều phương pháp như TF, IDF hay TF-IDF, người thiết kế hệ thống có thể tùy chỉnh các cách đánh giá tầm quan trọng của từ vựng trong tài liệu và truy vấn, từ đó nâng cao chất lượng và độ chính xác của kết quả truy xuất. Chính nhờ sự kết hợp hài hòa giữa tính chặt chẽ về lý thuyết và sự mềm dẻo trong thực tiễn, mô hình Không gian Vector đã và đang giữ một vai trò then chốt trong nhiều hệ thống truy xuất thông tin hiện đại

### 2.7.2 Nhược điểm

Bên cạnh những ưu điểm vượt trội, mô hình Không gian Vector (*Vector Space Model*) cũng bộc lộ một số hạn chế cố hữu, xuất phát từ chính bản chất hình học giản lược của nó trong việc biểu diễn ngôn ngữ tự nhiên. Trước hết, mô hình này **bỏ qua hoàn toàn thứ tự xuất hiện của các từ khóa** trong tài liệu và truy vấn. Điều đó đồng nghĩa với việc mọi khía cạnh liên quan đến ngữ pháp, cú pháp hay cấu trúc của câu – vốn có vai trò quan trọng trong việc truyền tải nghĩa – đều không được phản ánh trong không gian biểu diễn.

Thêm vào đó, **mỗi chiều trong không gian vector tương ứng với một từ khóa đơn lẻ**, song không hề có bất kỳ sự ràng buộc hay liên hệ nào về mặt ngữ nghĩa giữa các chiều đó. Điều này khiến mô hình không thể nắm bắt được các mối quan hệ ngữ nghĩa sâu sắc giữa từ ngữ, chẳng hạn như đồng nghĩa, trái nghĩa hay tính chất phân cấp trong từ vựng. Không gian mà các vector tồn tại cũng chỉ được giới hạn **trong phần dương**, do các trọng số như TF hay TF-IDF không thể nhận giá trị âm. Sự hạn chế này làm giảm tính biểu cảm của mô hình trong những tình huống cần biểu diễn mối quan hệ phủ định hoặc đối lập giữa các khái niệm.

Một trong những điểm yếu then chốt của mô hình là **cơ chế so khớp từ khóa cứng nhắc**. Nếu truy vấn và tài liệu không có bất kỳ từ khóa chung nào – dù nội dung thực tế có thể tương đồng về mặt ý tưởng – thì **độ tương đồng được tính**

**toán sẽ bằng không tuyệt đối**, phản ánh một khoảng cách triệt để không thực sự hợp lý trong nhiều trường hợp ngữ nghĩa. Chính vì những hạn chế này, mô hình Không gian Vector tuy mạnh mẽ trong các tình huống đơn giản, nhưng cũng cần được bổ sung hoặc cải tiến khi áp dụng cho các bài toán truy xuất thông tin phức tạp và giàu ngữ nghĩa hơn

# Chương 3

# Neural Network-based Model

## 3.1 Giới thiệu

Sự xuất hiện của các mô hình mạng nơ-ron trong lĩnh vực truy xuất thông tin (IR) đã khơi mở một chương mới đầy triển vọng, được ví như làn sóng thứ ba trong tiến trình phát triển của học máy, tiếp nối chuỗi thành tựu vang dội trong nhận dạng giọng nói, thị giác máy và xử lý ngôn ngữ tự nhiên. Trong bối cảnh đó, *Neural IR* – tức việc ứng dụng trực tiếp các kiến trúc mạng nơ-ron nông hoặc sâu vào các bài toán IR – đã nhanh chóng khẳng định vai trò trung tâm trong việc kiến tạo nên những mô hình có khả năng học biểu diễn dữ liệu từ văn bản thô một cách hiệu quả, vượt xa giới hạn của các phương pháp dựa trên đặc trưng thủ công truyền thống.

Động lực cho sự trỗi dậy của Neural IR đến từ một tập hợp hài hòa giữa các yếu tố then chốt: tiến bộ trong thiết kế kiến trúc mạng nơ-ron như CNNs, RNNs và đặc biệt là Transformer; khả năng tiếp cận các tập dữ liệu lớn được gán nhãn đầy đủ – yếu tố thiết yếu cho sự học hiệu quả của các mô hình sâu; cùng với sức mạnh tính toán ngày càng gia tăng, đặc biệt từ các GPU chuyên dụng. Điểm nổi bật của các mô hình nơ-ron hiện đại là khả năng chấp nhận đầu vào là toàn văn truy vấn và tài liệu, từ đó tự học các biểu diễn phù hợp cho việc xếp hạng hoặc truy vấn mà không cần thiết kế thủ công các đặc trưng như trước.

Ban đầu, các mô hình này thường được sử dụng như lớp xếp hạng lại trong cấu

hình nhiều tầng, xử lý N tài liệu ứng viên được truy xuất bởi các phương pháp truyền thống. Tuy nhiên, phạm vi ứng dụng đã dần mở rộng sang các ngữ cảnh truy xuất đa dạng hơn như tìm kiếm ngữ nghĩa, gợi ý nội dung, tìm kiếm đa phương tiện và thậm chí là các hệ thống hội thoại sinh phản hồi tự nhiên. Dẫu vậy, điểm chung xuyên suốt là năng lực biểu diễn ngữ nghĩa sâu sắc từ văn bản – một năng lực không thể đạt được nếu chỉ dừng lại ở các chỉ số thống kê bề mặt.

Bằng cách tận dụng sức mạnh biểu diễn của các mô hình mạng nơ-ron hiện đại, nghiên cứu này lựa chọn sử dụng kiến trúc *BERT* làm nền tảng chính cho các thí nghiệm truy xuất thông tin phía sau – như một minh chứng sống động cho tiềm năng mà Neural IR mang lại trong việc tái định nghĩa cách con người truy tìm tri thức trong kho tàng văn bản khổng lồ.

## 3.2 Kiến trúc BERT

Kiến trúc của BERT – Bidirectional Encoder Representations from Transformers – là một minh chứng điển hình cho sự kết hợp hài hòa giữa chiều sâu hình thức toán học và trực giác ngôn ngữ học. Là một mô hình **encoder-only** được phát triển trên nền tảng của Transformer, BERT được thiết kế để hấp thụ ngữ nghĩa không tuyến tính của ngôn ngữ tự nhiên bằng cách cho phép các biểu diễn ngữ cảnh hóa toàn diện theo cả hai chiều: từ trái sang phải và từ phải sang trái. Chính đặc tính này – **bidirectionality** – đã mở ra một kỷ nguyên mới trong xử lý ngôn ngữ, vượt khỏi giới hạn của các mô hình ngôn ngữ truyền thống vốn mang tính đơn phương.

BERT gồm nhiều tầng encoder được sắp xếp tuần tự, trong đó mỗi tầng là một sự hòa phối của hai thành phần cơ bản: **multi-head self-attention** và **feed-forward neural network (FFN)**. Bên trong mỗi tầng attention, thông tin từ mỗi token không chỉ được truyền qua một luồng đơn, mà được phân phối qua nhiều "đầu" (head) chú ý song song, mỗi head học cách tập trung vào một khía cạnh khác nhau của ngữ cảnh – có thể là cú pháp, đồng tham chiếu, hoặc các mối liên hệ ngữ nghĩa trừu tượng hơn. Cơ chế attention này bắt đầu bằng việc tạo ba vector con từ mỗi token: **Query (Q)**, **Key (K)** và **Value (V)**, thông qua các biến đổi tuyến tính. Điểm nổi bật nằm ở phép nhân vô hướng giữa Q và K, được chuẩn hóa theo căn

bậc hai của kích thước chiều không gian $d_k$, sau đó đi qua hàm softmax để tạo ra một phân phối chú ý có tổng bằng 1. Kết quả cuối cùng là tổ hợp có trọng số của các vector V, tạo thành sự phản hồi ngữ nghĩa từ toàn bộ chuỗi đầu vào cho mỗi token riêng lẻ.

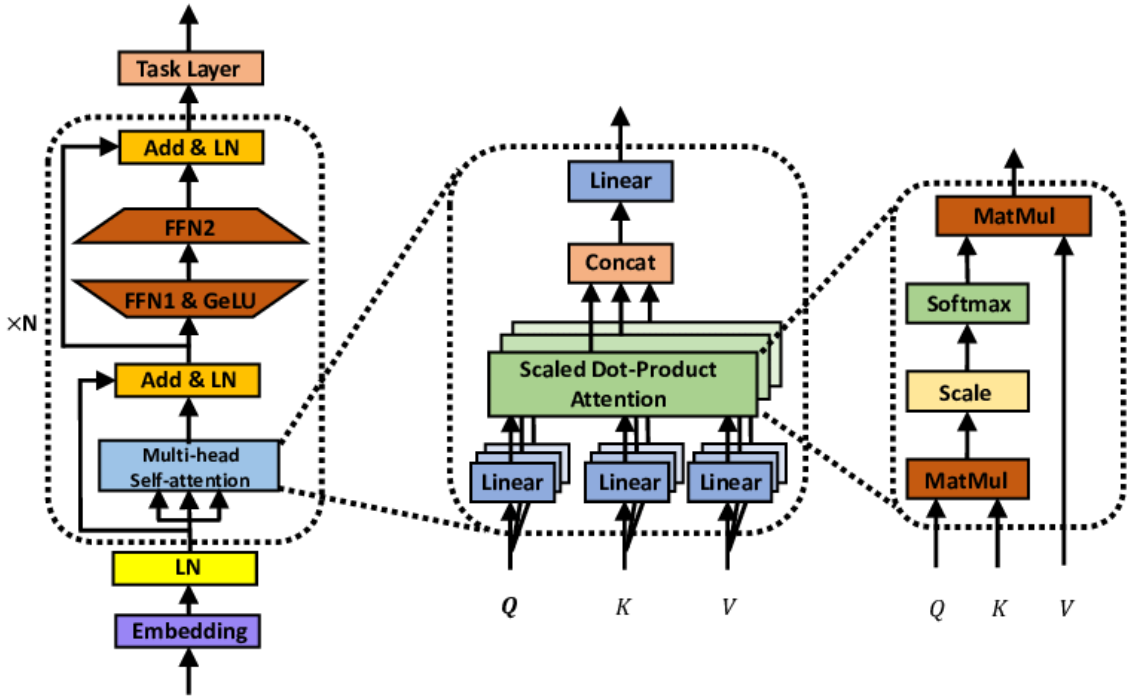
$$Attention(Q, K, V) = softmax\left(\frac{QK^T}{\sqrt{d_k}}\right) \cdot V \tag{3.1}$$

Sau đó, các output của các head attention được liên kết lại (concatenated) và đưa qua một tầng biến đổi tuyến tính cuối cùng để tạo ra biểu diễn tổng hợp. Mỗi tầng attention đều được bao bọc bởi một **residual connection** và **layer normalization**, nhằm ổn định gradient và thúc đẩy quá trình học. Phía sau attention là một mạng FFN với hai lớp tuyến tính, trong đó lớp ẩn thường có kích thước lớn gấp bốn lần kích thước ẩn của mô hình – ví dụ, từ 768 đến 3072 trong BERT base – và sử dụng hàm kích hoạt phi tuyến như ReLU để tăng cường khả năng biểu diễn phi tuyến.

Ngoài ra, đầu vào của BERT là tổng của ba loại embedding: **token embedding** từ từ điển WordPiece (30.000 mục từ), **segment embedding** để phân biệt giữa các câu trong tác vụ NSP (Next Sentence Prediction), và **position embedding** để giữ thông tin về vị trí tương đối của token trong chuỗi. Tổng thể này tạo nên một biểu diễn đầu vào dày đặc, giàu ngữ cảnh và đầy đủ cấu trúc.

Khác với transformer nguyên bản, BERT loại bỏ cơ chế attention che mặt (masked attention) ở phần decoder, thay vào đó cho phép attention mở rộng toàn phần trong encoder, từ đó học các biểu diễn ngữ nghĩa dựa trên toàn bộ chuỗi đầu vào. Đây là điểm then chốt tạo nên sức mạnh đặc biệt của BERT trong các tác vụ như truy xuất thông tin, phân loại văn bản và trả lời câu hỏi, khi mà ngữ nghĩa của một từ chỉ thực sự hiện hình trong mối tương quan với những từ vây quanh nó – trước và sau.

Hình 3.1: Minh họa kiến trúc BERT




Như một bản giao hưởng cấu trúc giữa hình học và ngôn ngữ học, kiến trúc của BERT không chỉ đơn thuần là một mạng học sâu, mà là một không gian ngữ nghĩa nơi mỗi token được tái định nghĩa theo cách nhìn toàn cục. Với khả năng học từ văn bản thô thông qua cơ chế attention đa chiều, BERT đã đặt nền móng cho hàng loạt ứng dụng downstream, khẳng định mình như một trong những trụ cột vững chắc của thời đại học biểu diễn trong xử lý ngôn ngữ tự nhiên.

## 3.3 Ý tưởng

Trong một hệ thống truy xuất thông tin hiện đại vận hành theo định hướng mạng nơ-ron, việc tích hợp BERT làm thành phần lõi cho quy trình tạo và khai thác nhúng không chỉ là lựa chọn kỹ thuật mà còn là bước chuyển mình mang tính chiến lược nhằm khai thác chiều sâu ngữ nghĩa từ văn bản tự nhiên. Workflow của hệ thống này được kiến tạo thành một chuỗi các bước có tổ chức chặt chẽ, nơi mỗi thành tố đóng vai trò quyết định trong việc đảm bảo rằng thông tin truy xuất được thực sự phản ánh đúng mối liên hệ ngữ nghĩa giữa truy vấn và tài liệu.

Bắt đầu từ bước chuẩn bị và tiền xử lý, dữ liệu đầu vào—thường là văn bản thô từ nhiều nguồn – được xử lý để loại bỏ nhiễu, chuẩn hóa biểu đạt, và duy trì tính toàn vẹn ngữ cảnh. Việc này không chỉ là hành động làm sạch dữ liệu đơn thuần mà còn là giai đoạn đầu tiên trong việc định hình ngữ nghĩa mà hệ thống sẽ tiếp nhận. Ngay sau đó, các văn bản được phân đoạn thành những đơn vị nhỏ hơn, thường là các câu hoặc đoạn văn, phù hợp với giới hạn dung lượng mà BERT có thể xử lý hiệu quả trong một lượt biểu diễn. Phân đoạn không chỉ giúp tiết kiệm tài nguyên tính toán, mà còn giúp nâng cao độ chính xác của việc ánh xạ ngữ nghĩa bằng cách tránh hiện tượng loãng thông tin xảy ra ở các văn bản dài.

Ở bước kế tiếp, từng đoạn văn bản đã qua xử lý được đưa vào mô hình BERT để sinh ra nhúng. Quá trình này bắt đầu bằng việc token hóa văn bản thành các đơn vị ngữ nghĩa nhỏ hơn, sau đó các token được ánh xạ lên không gian vector thông qua các tầng encoder của mô hình. Các nhúng thu được, thường có kích thước cao (chẳng hạn 768 chiều đối với BERT cơ sở), chính là biểu diễn toán học cô đọng của nội dung ngữ nghĩa – các vector này giữ vai trò nền tảng cho toàn bộ quá trình truy xuất phía sau. Để đảm bảo khả năng truy cập nhanh và hiệu quả ở quy mô lớn, tất cả các vector được lưu trữ trong một hệ thống cơ sở dữ liệu vector chuyên biệt như FAISS, nơi cung cấp khả năng tìm kiếm theo mức độ tương đồng gần như tức thời.

Khi một truy vấn được người dùng nhập vào, hệ thống tiến hành xử lý nó qua cùng pipeline như với tài liệu: từ làm sạch, phân đoạn, token hóa, đến biểu diễn bằng BERT. Việc tái sử dụng cùng một mô hình BERT bảo đảm rằng vector biểu diễn của truy vấn và của tài liệu cùng tồn tại trong một không gian ngữ nghĩa nhất quán, từ đó làm cơ sở cho việc đo lường mức độ tương đồng một cách chính xác. Nhúng truy vấn sau đó được sử dụng để tìm kiếm trong kho vector, nơi các phép đo như cosine similarity được áp dụng để xếp hạng các tài liệu dựa trên độ gần ngữ nghĩa so với truy vấn.

Quá trình này có thể được tinh chỉnh thêm với các bước giảm chiều như PCA hoặc t-SNE nhằm tối ưu hóa lưu trữ và cải thiện hiệu suất xử lý ở giai đoạn truy xuất, mà không làm mất đi các đặc trưng ngữ nghĩa cốt lõi. Tuy nhiên, điều đáng nhấn mạnh là thành công của hệ thống không chỉ đến từ mô hình BERT như một khối xây dựng độc lập, mà là sự tổng hòa của toàn bộ kiến trúc – từ pipeline tiền

xử lý, cơ chế tạo nhúng, cấu trúc lưu trữ vector, đến kỹ thuật truy xuất hiệu quả.

Do đó, trong thực tiễn triển khai, hệ thống IR xoay quanh BERT là một thực thể phức hợp, nơi mỗi bước trong workflow không chỉ thực hiện một chức năng riêng biệt, mà còn gắn kết chặt chẽ với nhau để hình thành nên một dòng chảy ngữ nghĩa xuyên suốt, từ dữ liệu đầu vào đến kết quả truy xuất đầu ra. Đây chính là biểu hiện rõ nét của việc đưa học sâu vào trung tâm của truy xuất thông tin, không chỉ như một công cụ kỹ thuật, mà như một khung nhận thức mới cho việc hiểu và tái cấu trúc tri thức từ văn bản.

## 3.4 Lập chỉ mục

Trong một hệ thống truy xuất thông tin hiện đại dựa trên mô hình mạng nơ-ron, quá trình lập chỉ mục đóng vai trò như một nghi lễ khởi tạo tinh vi, nơi toàn bộ tập tài liệu $\{d_1, d_2, d_3, \ldots, d_n\}$ được chuyển hoá từ những biểu đạt ngôn ngữ tự nhiên sang hình thức hình học có thể thao tác trong không gian vector dày đặc. Trái tim của quá trình này là mô hình BERT, hoạt động như một kiến trúc mã hóa ngữ nghĩa, cho phép hệ thống chưng cất tinh túy của từng đơn vị văn bản thành những vector có cấu trúc, mang theo ngữ nghĩa ngữ cảnh đã được nội tại hoá.

Bắt đầu từ giai đoạn đầu tiên, mỗi tài liệu $d_i$ được trải qua chuỗi tiền xử lý bao gồm làm sạch cú pháp, chuẩn hoá ký tự, và phân đoạn nếu tài liệu vượt quá chiều dài giới hạn mà BERT có thể tiếp nhận. Các tài liệu dài sẽ được cắt thành những đơn vị nhỏ hơn – thường là đoạn văn hoặc câu—sao cho mỗi đoạn phản ánh đầy đủ các khía cạnh ngữ nghĩa riêng biệt, đồng thời duy trì được tính nhất quán khi ánh xạ vào không gian biểu diễn. Tập các đoạn văn thu được từ mỗi $d_i$ lúc này được xem như tập con $\{p_1^i, p_2^i, \ldots, p_k^i\}$, nơi từng $p_j^i$ đại diện cho một phần nhỏ nhưng mang tính phân giải cao của tài liệu $d_i$.

Tiếp theo, từng đoạn $p_j^i$ được đưa qua mô hình BERT hoặc Sentence-BERT, vốn đã được huấn luyện để tạo ra các biểu diễn dense, ngữ cảnh hoá sâu sắc. Vector embedding $\mathbf{v}_j^i$ thu được cho mỗi đoạn là một điểm trong không gian $\mathbb{R}^d$, với $d$ là số chiều không gian vector mà mô hình BERT quy định. Tập hợp các vector

$\{\mathbf{v}_1^i, \mathbf{v}_2^i, \ldots, \mathbf{v}_k^i\}$ của mỗi tài liệu $d_i$ lúc này chính là hình ảnh vector hoá của tài liệu đó trong không gian ngữ nghĩa được học máy.

Sau khi tất cả các đoạn trong tập tài liệu được ánh xạ thành vector, hệ thống tiến hành lưu trữ các vector embedding cùng với siêu dữ liệu (metadata) liên kết như ID tài liệu, chỉ số đoạn, hoặc thông tin ngữ cảnh mở rộng. Toàn bộ tập hợp này được cài đặt vào một cơ sở dữ liệu vector như FAISS hoặc HNSWlib, nơi cho phép tìm kiếm tương đồng hiệu quả với độ trễ thấp. Chính nhờ kho vector này, hệ thống có thể truy cập nhanh chóng tới những phần tài liệu có ý nghĩa ngữ nghĩa gần nhất với truy vấn, tạo tiền đề cho việc xếp hạng lại hoặc kết hợp với các chỉ mục sparse khác để gia tăng độ chính xác.

Quá trình lập chỉ mục với BERT như vậy là một quá trình chuyển hoá không chỉ về mặt hình thức, mà còn về bản chất biểu đạt: từ dòng văn bản tuyến tính sang những khối kiến thức ngữ nghĩa được sắp xếp trong một không gian hình học mạch lạc, nơi việc truy xuất không còn phụ thuộc vào sự trùng khớp bề mặt, mà được điều hướng bởi sự cộng hưởng ngữ nghĩa sâu xa giữa truy vấn và tài liệu.

# Chương 4

# Cài đặt và đánh giá

## 4.1 Dữ liệu và độ đo

Trong quá trình thực nghiệm, nhóm sử dụng tập dữ liệu **Cranfield** từ nguồn `https://ir-datasets.com/cranfield.html`. Đây là một tập dữ liệu kinh điển trong lĩnh vực truy hồi thông tin, được xây dựng nhằm phục vụ việc đánh giá các mô hình truy xuất tài liệu. Cấu trúc của tập dữ liệu bao gồm:

- **Số lượng văn bản (documents)**: 1400 tài liệu, mỗi tài liệu có một mã `doc_id` duy nhất có độ dài tối đa 4 ký tự.
- **Số lượng truy vấn (queries)**: 225 truy vấn được xây dựng trước.
- **Tập phản hồi chuẩn (qrels)**: 1837 cặp truy vấn–tài liệu có gán mức độ liên quan (relevance judgments) từ người đánh giá.

Các mức độ liên quan trong `qrels` được phân loại theo thang điểm như sau:

- `4`: rất liên quan (highly relevant) — 363 trường hợp
- `3`: liên quan (relevant) — 734 trường hợp
- `2`: hơi liên quan (somewhat relevant) — 387 trường hợp

- `1`: ít liên quan — 128 trường hợp
- `-1`: không liên quan — 225 trường hợp

Các độ đo được định nghĩa như sau:

- **Precision** (Độ chính xác): Tính tỷ lệ giữa số tài liệu liên quan được truy xuất so với tổng số tài liệu được truy xuất.

$$\text{Precision} = \frac{|\text{Relevant Documents} \cap \text{Retrieved Documents}|}{|\text{Retrieved Documents}|}$$

- **Recall** (Độ bao phủ): Tính tỷ lệ giữa số tài liệu liên quan được truy xuất so với tổng số tài liệu liên quan trong tập dữ liệu.

$$\text{Recall} = \frac{|\text{Relevant Documents} \cap \text{Retrieved Documents}|}{|\text{Relevant Documents}|}$$

- **Precision@k** và **Recall@k**: Là độ chính xác và độ bao phủ khi chỉ xét $k$ tài liệu đầu tiên được truy xuất (ví dụ $\boldsymbol{k} = 10$).

$$\text{Precision@}\boldsymbol{k} = \frac{\text{Số tài liệu liên quan trong top } \boldsymbol{k}}{\boldsymbol{k}};$$

$$\text{Recall@}\boldsymbol{k} = \frac{\text{Số tài liệu liên quan trong top } \boldsymbol{k}}{\text{Tổng số tài liệu liên quan}}$$

- **11-point Interpolated Precision (TREC-style)**: Precision tại 11 mức recall cố định $\{0.0, 0.1, \ldots, 1.0\}$ được nội suy:

$$P_{\text{interp}}(\boldsymbol{r}) = \max_{\tilde{r} \geq r} P(\tilde{r})$$

  Từ đó, ta lấy trung bình trên toàn bộ 11 điểm để thu được kết quả đánh giá.

- **Average Precision (AP)**: Tính trung bình độ chính xác tại mỗi vị trí trong danh sách truy xuất mà tài liệu tại đó là liên quan.

$$\text{AP} = \frac{1}{R} \sum_{\boldsymbol{k}=1}^{n} P(\boldsymbol{k}) \cdot \text{rel}(\boldsymbol{k})$$

  trong đó:

- $R$ là tổng số tài liệu liên quan cho truy vấn đó.
- $P(k)$ là độ chính xác tại vị trí thứ $k$.
- rel($k$) là hàm chỉ số, bằng 1 nếu tài liệu ở vị trí $k$ là liên quan, ngược lại bằng 0.

- **Mean Average Precision (MAP)**: Là trung bình của AP trên toàn bộ tập truy vấn.

$$\text{MAP} = \frac{1}{Q} \sum_{q=1}^{Q} \text{AP}_q$$

với $Q$ là tổng số truy vấn.

## 4.2 Vector Space Model

Hệ thống khởi đầu bằng việc tiếp nhận tập hợp các tài liệu văn bản đầu vào. Đây có thể là các file văn bản độc lập hoặc một tập dữ liệu lớn như Cranfield. Các tài liệu này sẽ trải qua bước tiền xử lý nhằm chuẩn hóa nội dung, bao gồm các thao tác như chuyển về chữ thường, loại bỏ dấu câu và ký tự đặc biệt, tách từ, loại bỏ từ dừng và có thể áp dụng kỹ thuật stemming để rút gọn từ về gốc.

Sau khi tiền xử lý xong, hệ thống tiến hành xây dựng tập từ vựng (vocab) – tức là danh sách tất cả các từ duy nhất xuất hiện trong tập tài liệu sau tiền xử lý. Mỗi từ trong từ vựng được ánh xạ với một mã số định danh duy nhất (term_id). Từ đó, hệ thống xây dựng tập posting, trong đó mỗi từ được liên kết với danh sách các tài liệu mà nó xuất hiện, kèm theo tần suất hoặc trọng số (thường là TF-IDF). Tập hợp thông tin này được tổ chức thành một cấu trúc gọi là chỉ mục đảo (inverted index), cho phép tra cứu nhanh các tài liệu chứa một từ cụ thể.

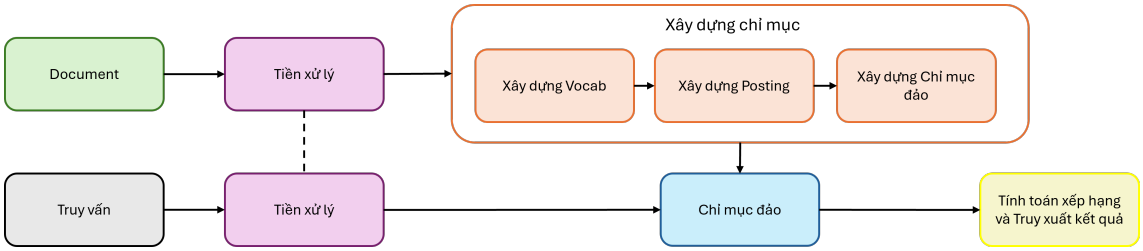
Trong khi đó, truy vấn của người dùng – dưới dạng một câu hỏi hoặc cụm từ – cũng được đưa vào hệ thống. Truy vấn này sẽ được tiền xử lý theo quy trình tương tự như tài liệu, nhằm đảm bảo tính thống nhất trong biểu diễn. Sau tiền xử lý, truy vấn được chuyển thành vector trong cùng không gian với các tài liệu đã lập chỉ mục.

Khi đó, chỉ mục đảo được sử dụng như một cầu nối giữa truy vấn và các tài liệu. Dựa vào các từ khóa trong truy vấn, hệ thống truy xuất các tài liệu liên quan từ

chỉ mục đảo, sau đó tiến hành tính toán mức độ tương đồng giữa vector truy vấn và vector tài liệu. Độ đo thường được sử dụng là độ tương đồng cosine, cho phép đánh giá mức độ gần nhau giữa các vector.

Cuối cùng, hệ thống xếp hạng các tài liệu theo độ tương đồng giảm dần và trả về danh sách các tài liệu phù hợp nhất với truy vấn. Đây là đầu ra của hệ thống – tập hợp các văn bản được sắp xếp theo mức độ liên quan, giúp người dùng nhanh chóng tiếp cận thông tin cần thiết.

Hình 4.1: Sơ đồ hệ thống của mô hình VSM




Để đánh giá mức độ hiệu quả của mô hình, nhóm đã thực nghiệm trên tập dữ liệu **Cranfield** — một tập văn bản tiêu chuẩn thường được sử dụng trong các bài toán truy hồi thông tin. Trong quá trình tiền xử lý, nhóm đã thực hiện các bước làm sạch văn bản nhằm chuẩn hóa dữ liệu đầu vào và nâng cao chất lượng truy xuất. Cụ thể:

- **Loại bỏ ký tự đặc biệt và dấu câu** nhằm loại bỏ nhiễu không cần thiết.
- **Loại bỏ stopwords** sử dụng tập từ dừng tiếng Anh chuẩn từ thư viện `nltk`, thông qua hàm `stopwords.words('english')`.
- **Stemming** được thực hiện bằng công cụ `SnowballStemmer` với ngôn ngữ tiếng Anh (`SnowballStemmer("english")`).

Không giống như các phương pháp stemming đơn giản hơn như `PorterStemmer`, công cụ `SnowballStemmer` (còn gọi là *Porter2*) là một trình gốc hóa từ nâng cao được thiết kế để cân bằng giữa hiệu quả và độ chính xác. Ngoài chức năng chính là *rút gọn từ về gốc*, `SnowballStemmer` còn có khả năng xử lý linh hoạt hơn đối với

các quy tắc hậu tố phức tạp trong tiếng Anh, giúp quy các từ có chung gốc nghĩa như “running”, “ran”, “runner” về một biểu diễn thống nhất. Điều này góp phần làm giảm phân mảnh từ vựng và nâng cao độ chính xác của truy vấn.

Sau khi hoàn tất bước tiền xử lý và xây dựng chỉ mục theo mô hình Vector Space Model (VSM), nhóm đã thu được các thống kê tổng quát như sau:

- **Tổng số văn bản (documents)**: 1400
- **Kích thước từ vựng (vocabulary size)**: 4672
- **Tổng số postings (số lượng cặp term–document)**: 86,680
- **Số postings trung bình trên mỗi từ**: 18.55

Những thống kê này cho thấy hệ thống chỉ mục đã được xây dựng hiệu quả, với một từ trung bình xuất hiện trong khoảng 18 tài liệu, phản ánh mức độ phổ biến của các từ trong tập dữ liệu và ảnh hưởng trực tiếp đến khả năng truy hồi tài liệu của hệ thống.

Sau khi xây dựng chỉ mục và triển khai mô hình Vector Space Model (VSM), nhóm đã thực nghiệm truy vấn trên tập dữ liệu Cranfield nhằm đánh giá hiệu quả của hệ thống truy xuất. Các chỉ số đánh giá được sử dụng bao gồm Precision@10, Recall@10, Mean Average Precision (MAP) và đường cong Precision-Recall theo chuẩn TREC.

- **Mean Precision@10**: 0.2271
- **Mean Recall@10**: 0.3922
- **Mean Average Precision (MAP)**: 0.2685

Những chỉ số này cho thấy mô hình VSM có khả năng truy xuất ở mức độ tương đối, với độ chính xác trung bình ở top 10 kết quả là khoảng 22.71% và độ bao phủ (recall) ở mức 39.22%. Giá trị MAP phản ánh hiệu quả tổng thể của mô hình trên toàn bộ tập truy vấn.

## 4.3 Neural Network-based Model

Hệ thống bắt đầu bằng việc tiếp nhận tập hợp các tài liệu văn bản đầu vào, chẳng hạn như các bài báo, mô tả sản phẩm hoặc đoạn văn bản trong một cơ sở dữ liệu lớn. Các tài liệu này sẽ được chuẩn bị để đưa vào mô hình học sâu, thường là một mô hình như BERT (Bidirectional Encoder Representations from Transformers) hoặc một mô hình tương tự.

Thay vì chỉ thực hiện các bước tiền xử lý đơn giản như trong VSM và các bước đó có thể ảnh hưởng đến ngữ nghĩa của toàn câu nên văn bản ở đây sẽ được mã hóa bằng tokenizer chuyên dụng của mô hình BERT, tạo ra chuỗi token được ánh xạ sang các chỉ số từ vựng đã được huấn luyện trước. Sau đó, mô hình BERT xử lý văn bản để tạo ra biểu diễn ngữ nghĩa dưới dạng vector – còn gọi là document embedding vector. Vector này là một biểu diễn dày đặc (dense representation), mang thông tin ngữ cảnh sâu và mối liên kết giữa các từ trong tài liệu.

Các vector biểu diễn tài liệu này được lưu trữ và lập chỉ mục trong một hệ thống tìm kiếm dựa trên vector như FAISS hoặc Annoy. Việc lập chỉ mục vector cho phép thực hiện truy vấn gần đúng (approximate nearest neighbor search), giúp tăng tốc đáng kể quá trình truy vấn trong không gian nhiều chiều.

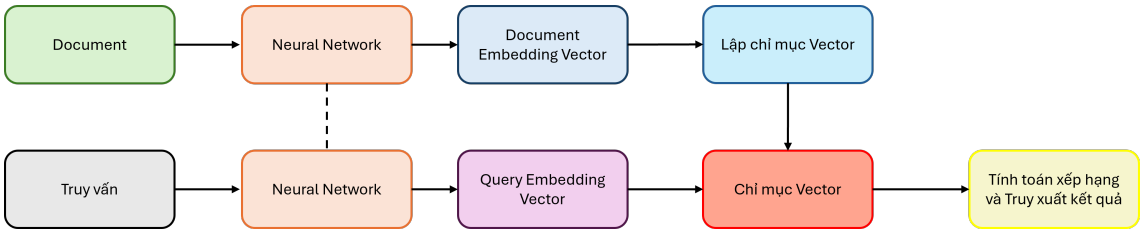
Tương tự như tài liệu, truy vấn từ người dùng cũng được đưa qua tokenizer và mô hình BERT để mã hóa thành một vector embedding. Truy vấn có thể là một câu hỏi ngắn hoặc một cụm từ tìm kiếm, và được xử lý theo cùng pipeline như văn bản.

Khi vector embedding của truy vấn đã được tạo, hệ thống sử dụng chỉ mục vector đã lập từ trước để tìm ra các vector tài liệu gần nhất với vector truy vấn – thường được đo bằng độ tương đồng cosine hoặc tích vô hướng. Các tài liệu có vector càng gần với vector truy vấn được xem là càng phù hợp.

Cuối cùng, hệ thống xếp hạng các tài liệu theo mức độ tương đồng với truy vấn và trả về danh sách các tài liệu liên quan nhất. Đây chính là đầu ra của hệ thống – danh sách kết quả truy xuất được sắp xếp theo độ liên quan cao đến thấp, mang lại trải nghiệm tìm kiếm hiệu quả và thông minh hơn so với các phương pháp dựa

trên thống kê truyền thống.

Hình 4.2: Sơ đồ hệ thống của Neural Network-based Model




Nhóm cũng thực hiện thực nghiệm trên tập dữ liệu Cranfield bằng cách sử dụng mô hình Neural Network để cải thiện chất lượng truy xuất. Cụ thể, nhóm sử dụng mô hình `sentence-transformers/all-MiniLM-L6-v2` để biểu diễn các câu truy vấn và văn bản tài liệu dưới dạng vector embedding. Đây là một mô hình được huấn luyện dựa trên kiến trúc **MiniLM**, tối ưu hóa cho bài toán *semantic textual similarity*, đặc biệt hiệu quả trong các tác vụ truy hồi văn bản nhờ vào khả năng học được biểu diễn ngữ nghĩa giàu thông tin ở dạng vector chiều thấp.

**Mô hình `all-MiniLM-L6-v2`** là một trong những phiên bản nhẹ và nhanh của họ mô hình MiniLM. Mặc dù chỉ sử dụng 6 tầng Transformer (so với 12 tầng trong `all-MiniLM-L12-v2`), mô hình này được huấn luyện bằng kỹ thuật distillation từ các mô hình lớn hơn, giúp duy trì hiệu suất cao trong khi giảm đáng kể thời gian tính toán và bộ nhớ. Trên thực tế, nhiều đánh giá cho thấy `MiniLM-L6-v2` đạt chất lượng tương đương hoặc thậm chí tốt hơn `MiniLM-L12-v2` trong các tác vụ truy hồi, nhờ việc cân bằng tốt giữa độ sâu mô hình và khả năng tổng quát hóa. Ngoài ra, do mô hình nhẹ hơn, embedding sinh ra từ `MiniLM-L6-v2` có xu hướng ít nhiễu và ổn định hơn đối với tập dữ liệu nhỏ như Cranfield.

Sau khi biểu diễn các tài liệu và truy vấn dưới dạng vector, nhóm sử dụng thư viện **FAISS (Facebook AI Similarity Search)** để xây dựng chỉ mục truy xuất nhanh. FAISS là một thư viện tối ưu cao được phát triển bởi Facebook AI Research, chuyên dùng để tìm kiếm gần đúng các vector trong không gian chiều cao (Approximate Nearest Neighbor Search). FAISS hỗ trợ nhiều cấu trúc dữ liệu như flat index, HNSW, IVF để đảm bảo cân bằng giữa tốc độ truy vấn và độ chính xác.

Trong thử nghiệm này, nhóm sử dụng FAISS để lập chỉ mục các vector văn bản và thực hiện truy vấn bằng cách tìm kiếm $k$ vector gần nhất (top-$k$) với vector truy vấn trong không gian embedding. Vì tập Cranfield có kích thước nhỏ (1400 tài liệu), nên nhóm sử dụng `IndexFlatIP` (Inner Product) — một phương pháp đơn giản nhưng hiệu quả trong không gian embedding được chuẩn hóa.

Cách tiếp cận này giúp hệ thống truy xuất không còn phụ thuộc vào từ khóa trực tiếp mà thay vào đó tận dụng mối quan hệ ngữ nghĩa giữa các câu, dẫn đến kết quả truy vấn chính xác hơn trong nhiều trường hợp, đặc biệt là với các truy vấn ngắn hoặc có ngữ nghĩa rộng.

Sau khi xây dựng hệ thống truy xuất dựa trên mô hình embedding ngữ nghĩa kết hợp với FAISS index, nhóm tiến hành đánh giá chất lượng truy xuất bằng cùng bộ truy vấn như đối với mô hình VSM. Các chỉ số đo lường được báo cáo như sau:

- **Mean Precision@10**: 0.2489
- **Mean Recall@10**: 0.4177
- **Mean Average Precision (MAP)**: 0.3264

Các chỉ số trên cho thấy mô hình học sâu kết hợp với biểu diễn embedding ngữ nghĩa đã đạt được kết quả tốt hơn so với mô hình VSM truyền thống. Giá trị Precision@10 và Recall@10 đều tăng lên, cho thấy khả năng truy xuất chính xác và bao phủ cao hơn ở top kết quả đầu tiên. Đặc biệt, chỉ số MAP tăng rõ rệt, phản ánh sự cải thiện tổng thể trong chất lượng truy xuất.

## 4.4 Đánh giá

Kết quả thực nghiệm cho thấy mô hình học sâu dựa trên embedding ngữ nghĩa tỏ ra vượt trội so với mô hình truyền thống VSM trong hầu hết các chỉ số đánh giá. Cụ thể:

- **Precision@10** tăng từ **0.2271** (VSM) lên **0.2489** (Neural Model) – tương

ứng với mức tăng gần 10%, phản ánh số lượng tài liệu liên quan trong top 10 kết quả truy xuất được cải thiện rõ rệt.

- **Recall@10** tăng từ **0.3922** lên **0.4177**, cho thấy hệ thống sử dụng embedding có khả năng bao phủ nhiều tài liệu liên quan hơn trong tập kết quả đầu ra.
- Đáng chú ý nhất là chỉ số **Mean Average Precision (MAP)** – một thước đo quan trọng phản ánh độ chính xác tổng thể trên toàn bộ truy vấn – đã tăng từ **0.2685** lên **0.3264**, tương ứng mức cải thiện hơn **21%**. Điều này chứng minh rằng mô hình embedding không chỉ tốt hơn ở các kết quả đầu tiên mà còn duy trì hiệu quả truy xuất ổn định trên toàn bộ danh sách kết quả.

Sự khác biệt này đến từ cách biểu diễn dữ liệu: trong khi VSM sử dụng biểu diễn rời rạc dựa trên từ vựng và trọng số thống kê (như TF-IDF), thì mô hình học sâu tạo ra các vector biểu diễn ngữ nghĩa đậm đặc (dense embeddings), có khả năng học được mối liên hệ ngữ cảnh giữa các từ và câu. Nhờ vậy, hệ thống có thể nhận diện các văn bản liên quan ngay cả khi không chứa trực tiếp các từ khóa trong truy vấn, điều mà VSM khó làm được do tính chất từ khóa trực tiếp (keyword-matching).

Bên cạnh đó, khả năng tổng quát hóa của mô hình neural network cũng được cải thiện nhờ vào việc mô hình được huấn luyện trên các tập dữ liệu lớn trước đó (pretrained), giúp nắm bắt các mối quan hệ ngữ nghĩa phức tạp giữa các từ. Điều này đặc biệt hữu ích khi làm việc với các truy vấn ngắn, mơ hồ hoặc mang tính khái quát – vốn là điểm yếu của các mô hình thống kê truyền thống như VSM.

Tuy nhiên, cần lưu ý rằng chi phí tính toán và triển khai của mô hình học sâu cao hơn đáng kể so với VSM. Trong khi VSM chỉ yêu cầu các phép tính tuyến tính đơn giản trên ma trận TF-IDF, thì mô hình neural đòi hỏi tài nguyên lớn hơn cho việc mã hóa văn bản và xây dựng chỉ mục vector. Do đó, lựa chọn mô hình phù hợp còn tùy thuộc vào bối cảnh ứng dụng: nếu yêu cầu hiệu suất cao, đặc biệt là trong các hệ thống tìm kiếm ngữ nghĩa, mô hình học sâu là lựa chọn ưu việt; còn nếu hệ thống giới hạn về tài nguyên hoặc cần xử lý nhanh các truy vấn đơn giản, VSM vẫn là một giải pháp hiệu quả và dễ triển khai.

# Chương 5

# Tổng kết và mở rộng

Qua quá trình xây dựng và thực nghiệm hai mô hình truy xuất thông tin – Vector Space Model (VSM) truyền thống và mô hình Neural Network – nhóm đã tiến hành đánh giá toàn diện về hiệu suất, khả năng tổng quát hóa, cũng như tiềm năng ứng dụng của từng phương pháp.

Kết quả cho thấy rằng mô hình neural network-based kết hợp biểu diễn ngữ nghĩa (semantic embeddings) mang lại độ chính xác và khả năng bao phủ tài liệu tốt hơn, thể hiện qua các chỉ số như Precision@10, Recall@10 và Mean Average Precision (MAP). Điều này khẳng định ưu thế rõ rệt của các phương pháp học sâu trong việc nắm bắt ngữ nghĩa và xử lý các truy vấn mơ hồ hoặc có ngữ cảnh phức tạp – điều mà các mô hình dựa trên biểu diễn rời rạc như VSM khó có thể đạt được.

Tuy nhiên, VSM vẫn có những ưu điểm nhất định về tính đơn giản, dễ triển khai, và tốc độ xử lý khi làm việc với tập dữ liệu nhỏ hoặc hệ thống có tài nguyên hạn chế. Do đó, thay vì loại bỏ hoàn toàn VSM, một hướng mở rộng tiềm năng là tăng cường khả năng biểu diễn của VSM thông qua các chiến lược tiền xử lý nâng cao. Cụ thể:

- **Áp dụng lemmatization thay vì stemming**: Khác với stemming vốn chỉ rút gọn từ về gốc hình thức, lemmatization giúp đưa từ về dạng cơ sở có nghĩa, đảm bảo giữ lại ngữ nghĩa tốt hơn, từ đó giảm hiện tượng phân mảnh từ vựng và cải thiện chất lượng truy xuất.

- **Sử dụng bigram/trigram**: Việc mở rộng biểu diễn từ đơn sang cụm từ (n-grams) có thể giúp hệ thống nắm bắt tốt hơn các đơn vị ngữ nghĩa phức tạp, ví dụ như "information retrieval" hay "machine learning", thay vì xem chúng là hai từ rời rạc.
- **Lọc đặc trưng bằng ngưỡng TF-IDF**: Một số từ có thể xuất hiện rất thường xuyên nhưng lại không mang ý nghĩa phân biệt. Do đó, việc sử dụng chỉ số TF-IDF không chỉ phục vụ cho việc biểu diễn vector mà còn có thể dùng như một tiêu chí lọc đặc trưng: loại bỏ các từ có giá trị TF-IDF quá thấp (không mang tính phân biệt), hoặc quá cao (xuất hiện trong quá ít tài liệu). Điều này giúp giảm nhiễu trong không gian đặc trưng và cải thiện hiệu suất truy xuất.
- **Áp dụng weighting scheme cải tiến**: Thay vì sử dụng TF-IDF truyền thống, có thể thử các biến thể như BM25 hoặc lược đồ weighting theo Entropy nhằm đánh giá mức độ quan trọng của từ trong ngữ cảnh cụ thể hơn.

Về phía mô hình học sâu, mặc dù hiệu năng vượt trội nhưng vẫn tồn tại các thách thức về chi phí tính toán và khả năng mở rộng trong môi trường tài nguyên hạn chế. Do đó, một số hướng cải tiến trong tương lai có thể bao gồm:

- **Tối ưu hóa pipeline truy xuất**: Kết hợp các chiến lược nén embedding, giảm chiều vector hoặc sử dụng các kỹ thuật indexing hiệu quả hơn như HNSW trong FAISS để tăng tốc độ và tiết kiệm bộ nhớ.
- **Fine-tune mô hình trên tập liệu miền hẹp**: Việc huấn luyện lại (fine-tuning) mô hình embedding như MiniLM trên tập dữ liệu Cranfield có thể giúp mô hình học tốt hơn các đặc điểm ngữ nghĩa chuyên biệt trong miền, từ đó nâng cao chất lượng truy xuất.
- **Kết hợp hai mô hình (Hybrid Retrieval)**: Một hướng tiếp cận đầy tiềm năng là xây dựng hệ thống truy xuất hai giai đoạn. Giai đoạn đầu sử dụng VSM để lọc nhanh tập tài liệu ban đầu (candidate documents), sau đó giai đoạn hai dùng mô hình học sâu để tái xếp hạng (re-ranking) theo mức độ liên quan ngữ nghĩa. Mô hình lai này tận dụng ưu điểm tốc độ của VSM và độ chính xác của embedding.

Tổng thể, sự chuyển dịch từ các mô hình thống kê truyền thống sang các mô hình ngữ nghĩa học sâu là xu hướng tất yếu trong lĩnh vực truy hồi thông tin hiện đại. Tuy nhiên, việc tận dụng tối ưu từng mô hình cần được cân nhắc kỹ lưỡng dựa trên đặc điểm bài toán, tập dữ liệu và hạn chế tài nguyên thực tế. Trong tương lai, các hướng nghiên cứu kết hợp giữa biểu diễn ngữ nghĩa và khai thác cấu trúc truy vấn, hoặc áp dụng mô hình đa phương thức (kết hợp văn bản và hình ảnh) hứa hẹn sẽ tiếp tục mở rộng tiềm năng của các hệ thống tìm kiếm thông minh.

# Chương 6

# Ứng dụng thực tế

## 6.1 Giới thiệu chung về hệ thống

### 6.1.1 Mục tiêu thực hiện

Dự án hướng đến việc xây dựng một hệ thống chatbot tuyển sinh thông minh nhằm hỗ trợ thí sinh và phụ huynh tiếp cận thông tin tuyển sinh một cách chủ động, nhanh chóng và chính xác, mà không cần phải gửi email, gọi điện hoặc nhắn tin qua fanpage của nhà trường để được phản hồi thủ công.

### 6.1.2 Hướng dẫn sử dụng hệ thống

Hệ thống chatbot được triển khai dưới dạng giao diện web tương tác, cho phép người dùng đặt câu hỏi trực tiếp mà không cần đăng nhập tài khoản. Mỗi phiên làm việc được khởi tạo mới khi tải lại trang, hệ thống có tích hợp lưu trữ lịch sử hội thoại trong phiên hiện tại để hiểu và xử lý ngữ cảnh liên tục.

Sau khi người dùng gửi truy vấn, hệ thống thực hiện các bước:

- Truy vấn được mã hóa thành vector ngữ nghĩa và so khớp với các đoạn văn bản trong cơ sở dữ liệu.

- Tính điểm và lọc các kết quả không đủ ngưỡng tin cậy.
- Truyền các tài liệu vào mô hình ngôn ngữ lớn (LLM) để mô hình tạo ra phản hồi với các tri thức có được từ các tài liệu.

Hình 6.1: Giao diện hội thoại chatbot truy xuất thông tin tuyển sinh


### 6.1.3 Công nghệ sử dụng

Hệ thống được xây dựng với kiến trúc client-server, sử dụng các công nghệ hiện đại và phù hợp với các bài toán xử lý ngôn ngữ tự nhiên trong môi trường web. Cụ thể:

- **Ngôn ngữ và nền tảng:** Toàn bộ hệ thống phía máy chủ được phát triển bằng Python, sử dụng **Django REST Framework** để xây dựng các dịch vụ API. Phía giao diện người dùng sử dụng **React với TypeScript**, triển khai theo hướng component hóa, hỗ trợ thiết kế phản hồi (responsive) và tương tác thời gian thực.
- **Cơ sở dữ liệu:** Hệ thống sử dụng **PostgreSQL (Neon)** cho dữ liệu quan hệ và **Qdrant** làm cơ sở dữ liệu vector chuyên biệt cho truy xuất ngữ nghĩa. Qdrant hỗ trợ tìm kiếm nhanh dựa trên khoảng cách giữa các embedding vector.

- **Xử lý ngôn ngữ và truy xuất:** Dữ liệu được biểu diễn bằng mô hình embedding tiếng Việt `AITeamVN/Vietnamese_Embedding`. Việc phân đoạn tài liệu và quản lý ngữ cảnh được thực hiện thông qua thư viện **LangChain**.
- **Sinh phản hồi:** Hệ thống tích hợp **Google Cloud Generative AI** để sinh câu trả lời hội thoại dựa trên ngữ cảnh đã truy xuất.
- **Kết nối và giao tiếp:** Frontend sử dụng thư viện **Axios** để tương tác với các dịch vụ phía backend thông qua các endpoint RESTful.

## 6.2 Thu thập và xử lý dữ liệu

### 6.2.1 Web Crawling

Quá trình thu thập dữ liệu được thực hiện thông qua module `crawler.py`, cho phép tự động thu thập toàn bộ nội dung liên quan trên trang tuyển sinh của UIT chỉ với một tham số đầu vào là đường dẫn gốc:

```
base_url = `https://tuyensinh.uit.edu.vn/'
```

Thuật toán thực hiện việc truy cập vào địa chỉ chính, trích xuất các liên kết có trong trang, và lần lượt tiếp tục thu thập theo hướng duyệt theo chiều rộng trong cùng miền. Tại mỗi liên kết, nội dung chính được xác định thông qua các bộ chọn cấu trúc HTML, sau đó được chuyển đổi sang định dạng Markdown phục vụ cho quá trình xử lý tiếp theo.

Quy trình crawling bao gồm:

- Truy xuất nội dung HTML của từng trang.
- Trích xuất vùng nội dung chính, loại bỏ các thành phần không liên quan như điều hướng và chân trang.
- Chuyển đổi nội dung sang định dạng Markdown để lưu trữ.

- Tiếp tục thu thập các liên kết mới được tìm thấy trong trang và lặp lại quy trình.

Việc tự động hóa quy trình này giúp đảm bảo toàn bộ thông tin trên website tuyển sinh được thu thập một cách đầy đủ và dễ dàng mở rộng cho các website khác trong tương lai chỉ bằng cách thay đổi giá trị `base_url`.

### 6.2.2 Tiền xử lý dữ liệu

Sau quá trình thu thập, dữ liệu văn bản ở định dạng Markdown được xử lý trước khi đưa vào mô hình embedding. Quá trình tiền xử lý bao gồm hai giai đoạn chính: làm sạch nội dung và phân chia tài liệu thành các đoạn nhỏ (chunk) phù hợp cho truy xuất ngữ nghĩa.

#### Làm sạch Markdown

Module `clean_markdown.py` thực hiện các thao tác làm sạch định dạng Markdown nhằm chuẩn hoá nội dung và cấu trúc của tài liệu. Các bước xử lý bao gồm:

- Loại bỏ các ký tự không mong muốn (whitespace thừa, ký hiệu ẩn, HTML tags).
- Chuẩn hoá hệ thống tiêu đề theo cấp độ (heading hierarchy) bằng cách ánh xạ các mục La Mã, mục đánh số sang cú pháp Markdown chuẩn.

#### Phân đoạn tài liệu (Document Chunking)

Sau khi nội dung được làm sạch, hệ thống thực hiện phân chia tài liệu thành các đoạn nhỏ bằng module `chunking.py`. Thuật toán sử dụng kỹ thuật Semantic Chunking từ thư viện LangChain để xác định ranh giới ngữ nghĩa giữa các đoạn. Cấu trúc tổng thể như sau:

- Tài liệu đầu vào được phân tích theo các tiêu đề con (heading levels) để tạo thành các phần nội dung chính.
- Mỗi phần tiếp tục được chia nhỏ theo ngữ nghĩa sử dụng công cụ `Semantic Chunker` kết hợp với mô hình embedding tiếng Việt `AITeamVN/Vietnamese_Embedding`.
- Mỗi đoạn nhỏ (chunk) được gắn kèm thông tin tiêu đề, nguồn, từ khóa và nhãn phân loại chủ đề.

#### Phân loại theo chủ đề

Hệ thống tích hợp cơ chế gán nhãn chủ đề tự động cho mỗi đoạn dựa trên tập từ khóa xác định trước. Một số chủ đề chính bao gồm:

- **Ngành học:** như An toàn thông tin, Khoa học dữ liệu, Trí tuệ nhân tạo, etc.
- **Tuyển sinh:** điểm chuẩn, phương thức xét tuyển, chỉ tiêu, điều kiện.
- **Học bổng:** các loại học bổng, tiêu chí, quy trình xét duyệt.
- **Thông tin trường:** giới thiệu giảng viên, cơ sở vật chất, hoạt động sinh viên.

Việc gán nhãn được thực hiện bằng cách so khớp từ khóa theo cả dạng có dấu và không dấu, kết hợp với ưu tiên các từ xuất hiện sớm trong văn bản để tăng tính chính xác trong phân loại.

## 6.3 Hệ thống Tìm kiếm Kết hợp (Hybrid Search)

Hệ thống truy xuất thông tin trong dự án được thiết kế theo hướng kết hợp (hybrid), tích hợp đồng thời hai phương pháp: truy xuất ngữ nghĩa dựa trên vector embedding và truy xuất truyền thống dựa trên mô hình BM25. Cách tiếp cận này cho phép tận dụng được điểm mạnh của từng phương pháp: khả năng hiểu ngữ cảnh sâu của mô hình học sâu và tính chính xác của từ khóa trong các truy vấn cụ thể.

### 6.3.1 Truy xuất ngữ nghĩa (Semantic Search)

Các đoạn văn bản sau khi được xử lý sẽ được ánh xạ thành các vector ngữ nghĩa sử dụng mô hình `AITeamVN/Vietnamese_Embedding`. Việc ánh xạ được thực hiện thông qua lớp `DocEmbedder`, trong đó mỗi đoạn được tổ hợp từ tiêu đề, tiêu đề mục và nội dung thân nhằm tăng độ bao phủ ngữ cảnh.

Các vector sau khi sinh được lưu trữ trong cơ sở dữ liệu vector Qdrant, sử dụng thuật toán lập chỉ mục `HNSW (Hierarchical Navigable Small World Graph)`. HNSW là một cấu trúc đồ thị đa tầng cho phép tìm kiếm gần đúng với độ chính xác cao, bằng cách duyệt từ tầng cao nhất (ít điểm, khoảng cách xa) xuống tầng thấp hơn (nhiều điểm, khoảng cách gần hơn). Việc tìm kiếm được thực hiện bằng phép đo `cosine similarity` giữa vector truy vấn và các vector tài liệu.

Mỗi vector được lưu kèm một *payload* chứa metadata như tiêu đề, ngành học, năm, và nguồn gốc nội dung. Ngoài chỉ mục vector, Qdrant còn hỗ trợ các chỉ mục phụ (payload index) để lọc theo điều kiện cụ thể, ví dụ: chỉ lấy các đoạn liên quan đến "tuyển sinh năm 2024".

### 6.3.2 Truy xuất theo mô hình BM25

Song song với truy xuất ngữ nghĩa, hệ thống cũng tích hợp mô hình `BM25` - một cải tiến của TF-IDF

Quá trình xử lý văn bản cho BM25 bao gồm:

- Chuẩn hóa văn bản: loại bỏ ký tự đặc biệt và stopwords, chuyển về chữ thường.
- Token hóa tiếng Việt bằng thư viện `pyvi`.
- Sinh n-gram ngữ cảnh để tăng độ bao phủ truy vấn (với kích thước cửa sổ ngữ cảnh là 2).

### 6.3.3 Cơ chế kết hợp điểm số

Hai điểm số truy xuất từ mô hình ngữ nghĩa và mô hình BM25 được kết hợp theo công thức sau:

$$\text{final\_score} = \alpha \cdot \text{semantic\_score} + (1 - \alpha) \cdot \text{bm25\_score\_normalized}$$

Người dùng hoặc hệ thống có thể điều chỉnh tham số $\alpha$ nhằm ưu tiên một phương pháp hơn trong các ngữ cảnh truy vấn khác nhau. Ví dụ, nếu $\alpha = 1$ thì hệ thống sẽ thực hiện truy xuất hoàn toàn theo hướng ngữ nghĩa; ngược lại, nếu $\alpha = 0$, chỉ có điểm BM25 được sử dụng.

Việc kết hợp hai hướng truy xuất này giúp tăng độ linh hoạt và độ chính xác của hệ thống.

### 6.3.4 Các tính năng khác

#### Lọc theo ngưỡng độ tin cậy (Confidence Thresholding)

Để nâng cao chất lượng kết quả truy xuất, hệ thống áp dụng cơ chế lọc theo ngưỡng độ tin cậy kép. Cụ thể, mỗi kết quả truy xuất chỉ được giữ lại nếu đồng thời thỏa mãn hai điều kiện:

- Điểm truy xuất ngữ nghĩa (semantic score) phải lớn hơn hoặc bằng một ngưỡng xác định.
- Điểm BM25 cũng phải lớn hơn hoặc bằng một ngưỡng tối thiểu.

Cơ chế này giúp loại bỏ các đoạn văn bản có điểm ngữ nghĩa cao nhưng không chứa từ khóa rõ ràng liên quan, hoặc ngược lại — các đoạn chứa từ khóa nhưng không phù hợp ngữ cảnh truy vấn. Việc kết hợp cả hai tiêu chí giúp tăng độ chính xác và phù hợp của kết quả đầu ra, đặc biệt quan trọng trong các truy vấn tuyển sinh có tính đặc thù và yêu cầu chính xác cao.

**Phân tích từ khóa (Keyword Analysis)**

Ngoài truy xuất, hệ thống còn hỗ trợ chức năng phân tích và trích xuất các từ khóa quan trọng nhất trong mỗi đoạn văn bản (chunk). Dựa trên mô hình BM25, hệ thống tính toán mức độ quan trọng của từng từ khóa trong mỗi đoạn và liệt kê các từ có trọng số cao nhất. Việc hiển thị từ khóa có trọng số cao nhất theo từng đoạn (chunk) cũng có thể được sử dụng trong giai đoạn đánh giá hệ thống hoặc tinh chỉnh ngưỡng điểm để cải thiện hiệu quả truy xuất.

## 6.4 Kết quả và đánh giá

### 6.4.1 Ví dụ truy vấn và kết quả sinh phản hồi

Truy vấn người dùng: *"Giới thiệu về ngành học Trí tuệ nhân tạo"*

**Các đoạn văn bản được truy xuất (top-5):**

- **Tài liệu 1** (Score: 0.6405): "*Với sự phát triển như vũ bão của Cách mạng công nghệ 4.0, Trí tuệ nhân tạo (AI) trở thành ngành 'hot'...*"
- **Tài liệu 2** (Score: 0.6162): "*Nguồn: https://tuyensinh.uit.edu.vn/tong-quan-nganh-tri-tue-nhan-tao...*"
- **Tài liệu 3** (Score: 0.6038): "*Bộ môn có nhiệm vụ nghiên cứu khoa học và giảng dạy các môn học theo định hướng Trí tuệ nhân tạo...*"
- **Tài liệu 4** (Score: 0.5921): "*Chương trình đào tạo ngành Trí tuệ nhân tạo hướng đến mục tiêu đào tạo nguồn nhân lực chất lượng cao...*"
- **Tài liệu 5** (Score: 0.5909): "*Sau khi tốt nghiệp, sinh viên có thể làm việc: Chuyên gia lập trình Trí tuệ nhân tạo...*"

**Câu trả lời:**

Ngành Trí tuệ Nhân tạo (AI) là một trong những ngành trọng điểm hiện nay, đóng vai trò then chốt trong Cách mạng công nghiệp 4.0. Trường Đại học Công nghệ Thông tin (UIT) chú trọng đào tạo đội ngũ nhân lực chất lượng cao trong lĩnh vực này.

Thông tin chung: Tên ngành: Trí tuệ nhân tạo (Artificial Intelligence); Mã ngành: 7480107; Hệ đào tạo: chính quy; Thời gian: 3.5 năm.

Mục tiêu đào tạo: Cung cấp nền tảng vững chắc về khoa học máy tính và chuyên sâu về AI; hình thành năng lực thiết kế, phát triển hệ thống thông minh; có thể tiếp tục học lên cao.

Cơ hội nghề nghiệp: Làm việc trong lĩnh vực phát triển phần mềm AI, phân tích dữ liệu, nghiên cứu tại các trung tâm R&D hoặc tiếp tục học cao học chuyên ngành AI.

Lĩnh vực đào tạo chuyên sâu: Máy học, biểu diễn tri thức và suy luận, dữ liệu lớn, thuật toán tiến hóa, v.v.

**Nhận xét:** Hệ thống đã truy xuất đúng các đoạn có liên quan ngữ nghĩa cao, kết hợp nội dung tốt để tạo phản hồi đầy đủ, đúng trọng tâm.

### 6.4.2 Nhận xét và định hướng phát triển

Mặc dù hệ thống đã hoạt động ổn định và sinh phản hồi tương đối chính xác cho các câu hỏi trong miền chủ đề tuyển sinh, vẫn tồn tại một số hạn chế cần khắc phục:

- Hệ thống hiện tại vẫn truy xuất và đưa vào mô hình sinh phản hồi ngay cả khi truy vấn không liên quan đến chủ đề tuyển sinh. Việc đánh giá mức độ liên quan chủ yếu do mô hình ngôn ngữ lớn (LLM) đảm nhiệm, dẫn đến tình trạng lãng phí tài nguyên truy xuất và có thể tạo ra phản hồi sai nếu ngữ cảnh không phù hợp. Định hướng sắp tới là tích hợp thêm một mô hình học máy nhẹ để phân loại truy vấn ngay từ đầu, chỉ thực hiện truy xuất khi truy vấn thực sự liên quan, từ đó tiết kiệm tài nguyên và nâng cao độ chính xác.
- Mở rộng dữ liệu: Tìm thêm các trang base_url để có thể có thêm các thông tin liên quan đến tuyển sinh hoặc các thông tin về trường...

- Nâng cao chiến lược lọc nội dung: Sử dụng các công cụ như `KeyBERT` để trích xuất từ khóa chủ đề và áp dụng cơ chế lọc cứng (hard filtering) trước khi truy xuất, giúp loại bỏ các đoạn không liên quan.
- Tối ưu hiệu năng: Giảm thời gian truy xuất và sinh phản hồi bằng cách tinh chỉnh vector index và giảm số lượng đoạn văn cần đưa vào LLM.

Hướng đi này sẽ giúp hệ thống trở nên chính xác hơn, phù hợp hơn với mục tiêu hỗ trợ tuyển sinh thực tế và nâng cao trải nghiệm người dùng trong môi trường web tương tác.

# Tài liệu tham khảo


[1] Karam Abdulahhad. "Concept Embedding for Information Retrieval." In: *Advances in Information Retrieval*. Springer International Publishing, 2018, pp. 563–569. ISBN: 9783319769417. DOI: 10.1007/978-3-319-76941-7_45. URL: http://dx.doi.org/10.1007/978-3-319-76941-7_45.

[2] GeeksforGeeks. *What is Information Retrieval?*

[3] C.D. Manning, P. Raghavan, and H. Schütze. *Introduction to Information Retrieval*. An Introduction to Information Retrieval. Cambridge University Press, 2008. ISBN: 9780521865715. URL: https://books.google.com.vn/books?id=GNvtngEACAAJ.