Truy xuất thông tin bằng mô hình không gian vector và mô hình xác suất

CS419.M11.KHCL

Ngô Bảo Trân
18520173

1 Giới thiệu

Hiện nay, với sự phát triển của Internet và dữ liệu số, truy xuất thông tin không còn đơn thuần là hoạt động của một số ít người (thủ thư, nhà nghiên cứu...) mà đã trở thành hoạt động thường ngày của tất cả mọi người, thường thấy nhất là tìm kiếm thông qua các công cụ tìm kiếm trực tuyến ví dụ như Google. Không những thế, truy xuất thông tin cũng được ứng dụng trong các trang bán hàng, mạng xã hội, trong các dịch vụ nghe nhạc, xem video trực tuyến để giúp đề xuất những thông tin mới phù hợp với người dùng. Như vậy, truy xuất thông tin đóng vai trò cực kì quan trọng trong thế giới số hiện nay. Hiểu được điều đó, chúng tôi thực hiện đồ án này nhằm bước đầu tìm hiểu về truy xuất thông tin nói chung và một số mô hình truy xuất thông tin; từ đó xây dựng nền tảng để có thể tiếp tục nghiên cứu về lĩnh vực này.

Nội dung của báo cáo được tổ chức như sau. Ở phần 2, chúng tôi sẽ giới thiệu sơ lược về truy xuất thông tin nói chung, về khái niệm, ứng dụng, kiến trúc của một hệ thống truy xuất thông tin và các mô hình truy xuất thông tin. Ở phần 3 và phần 4, chúng tôi sẽ trình bày các vấn đề lý thuyết của mô hình không gian vector và mô hình xác suất mà chúng tôi thực hiện trong đồ án này. Ở phần 5, phần 6 và phần 7 là chi tiết về phương pháp phân tích tài liệu, quá trình lập, truy xuất chỉ mục và quá trình tính toán truy vấn. Phần 8 là kết quả thực nghiệm và phần cuối là phần kết luận.

2 Truy xuất thông tin

2.1 Khái niệm

Theo [6], truy xuất thông tin là hoạt động tìm kiếm thông tin (thường là tài liệu) không có cấu trúc (thường là văn bản) nhằm thỏa mãn một nhu cầu thông tin trên một tập lớn (thường được lưu trữ trên máy tính).

2.2 Ứng dụng

Ứng dụng phổ biến nhất của truy xuất thông tin là xây dựng các công cụ tìm kiếm (search engine). Công cụ tìm kiếm cho phép người dùng nhập vào một truy vấn và trả về các kết quả có liên quan. Truy vấn bằng công cụ tìm kiếm không chỉ thu nhỏ trong phạm vi văn bản mà còn mở rộng ra các dạng thông tin đa phương tiện khác như hình ảnh, âm thanh, video... Một số công cụ tìm kiếm phổ biến là Google, Bing... (web search), tìm kiếm trên desktop (desktop search), tìm kiếm trên mạng xã hội như Facebook, Twitter... (social search)...

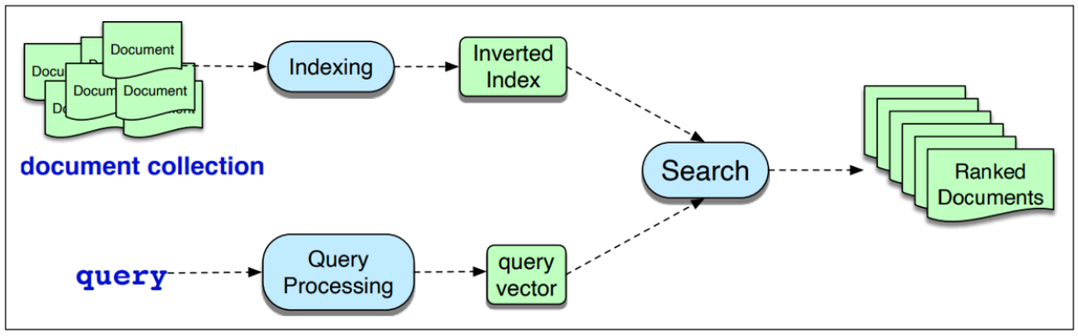
Các hệ thống gợi ý (recommendation system) cũng là một ứng dụng của truy xuất thông tin. Các hệ thống này thường được tích hợp trong các ứng dụng như ứng dụng bán hàng online (Amazon, Shopee...), mạng xã hội (Facebook, Twitter...), các trang nghe nhạc, xem video trực tuyến (Youtube, Spotify...) để giúp nâng cao trải nghiệm của người dùng bằng cách gợi ý đồ dùng, tin tức, bài đăng, bài hát, video... mà người dùng có thể hứng thú dựa trên thông tin của chính người dùng.

Truy xuất thông tin còn được ứng dụng để xây dựng các hệ thống hỏi đáp (question-answering system). Hệ thống hỏi đáp là một hệ thống cho phép người dùng nhập vào

một truy vấn và trả về câu trả lời chính xác cho truy vấn của người dùng. Như vậy, hệ thống hỏi đáp là một phiên bản nâng cao của hệ thống truy xuất thông tin thông thường: hệ thống không chỉ trả về các tài liệu có liên quan, mà còn có thể trả về câu trả lời chính xác từ trong tập tài liệu. Ứng dụng thường thấy của các hệ thống này là chatbot dùng để trả lời tự động các câu hỏi của người dùng trên các trang web.

2.3 Kiến trúc của một hệ thống truy xuất thông tin

Trong đồ án này, tác vụ truy xuất thông tin mà chúng tôi tập trung giải quyết được gọi là ad hoc retrieval. Một hệ thống ad hoc retrieval sẽ nhận một truy vấn từ người dùng và trả về danh sách xếp hạng các tài liệu liên quan từ một tập tài liệu nào đó. Một **tài liệu** (document) là bất cứ đơn vị văn bản nào mà hệ thống lập chỉ mục và truy vấn (trang web, bài báo khoa học, bảng tin hoặc những đoạn nhỏ hơn như đoạn văn). Một **tập tài liệu** (collection) là một tập hợp các tài liệu dùng để thỏa mãn nhu cầu thông tin của người dùng. Một **term** là một từ trong tập tài liệu, nhưng cũng có thể bao gồm cụm từ. Cuối cùng, một **truy vấn** (query) biểu diễn nhu cầu thông tin của người dùng bằng một tập các terms.



Hình 1. Kiến trúc của hệ thống ad hoc retrieval [4]

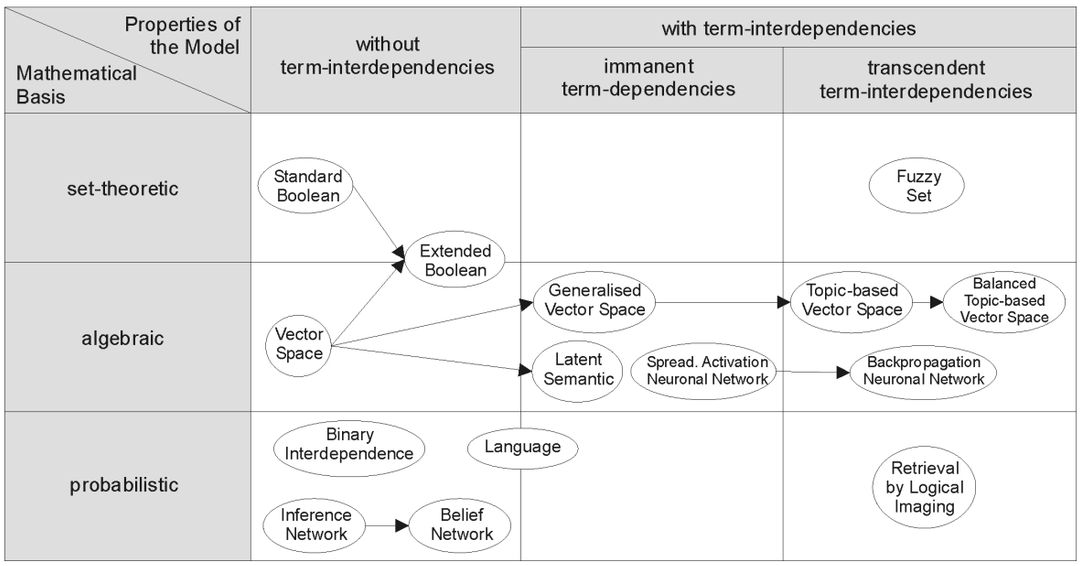
Kiến trúc của một hệ thống ad hoc retrieval được miêu tả ở hình 1. Đầu vào của hệ thống là một tập tài liệu và một truy vấn từ người dùng. Đối với tập tài liệu, hệ thống tiến hành lập chỉ mục (trong hình 1 là chỉ mục đảo ngược). Đối với truy vấn, hệ thống tiến hành xử lý truy vấn và biểu diễn truy vấn theo một cách biểu diễn nào đó (trong hình 1 là vector). Sau đó, hệ thống sẽ thực hiện tìm kiếm các tài liệu liên quan đến truy vấn thông qua chỉ mục bằng một mô hình truy xuất thông tin nào đó. Cuối cùng, hệ thống sẽ trả về danh sách xếp hạng các tài liệu liên quan, với tài liệu liên quan nhất được xếp đầu tiên và theo sau là các tài liệu có độ liên quan giảm dần. Danh sách này có thể khác nhau tùy vào mô hình truy xuất mà hệ thống sử dụng.

2.4 Các mô hình truy xuất thông tin

Theo [5], các mô hình truy xuất thông tin được phân chia dựa trên hai tiêu chí:

- Cơ sở toán học: dựa trên lý thuyết tập hợp, dựa trên đại số và dựa trên xác suất
- Dựa trên sự độc lập của các terms với nhau: độc lập và không độc lập. Ở nhóm các mô hình định nghĩa terms không độc lập với nhau có hai nhóm nhỏ: các mô hình tự học/tự định nghĩa sự độc lập (immanent) và các mô hình không tự định nghĩa sự độc lập mà dựa vào con người hoặc các thuật toán phức tạp (transcendent).

Như vậy, tổng cộng có 9 nhóm mô hình tất cả như ở hình 2.



Hình 2. Các mô hình truy xuất thông tin [5]

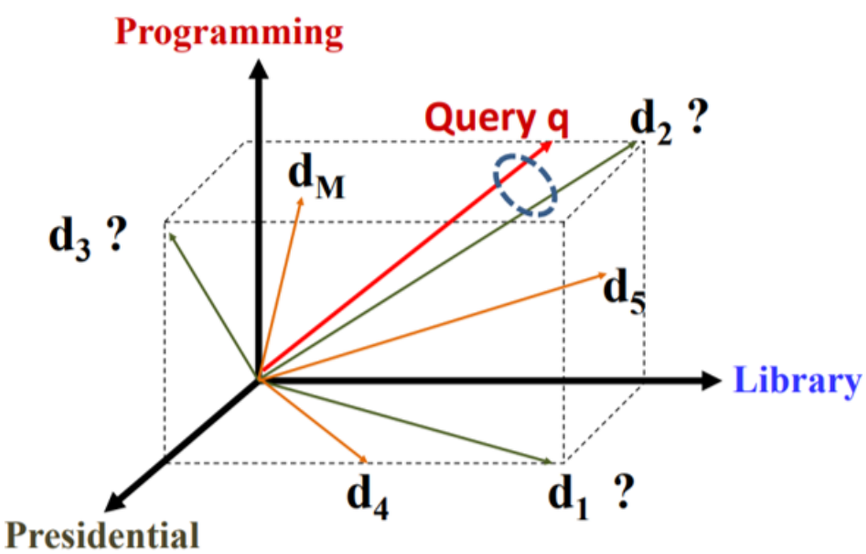
Trong đồ án này, chúng tôi sẽ tập trung tìm hiểu về mô hình không gian vector (dựa trên đại số và các terms độc lập) và mô hình ngôn ngữ (dựa trên xác suất và các terms độc lập/không độc lập). Mô hình ngôn ngữ chúng tôi lựa chọn tìm hiểu định nghĩa các terms là độc lập với nhau.

3 Mô hình không gian vector

3.1 Ý tưởng

Ý tưởng của mô hình không gian vector là biểu diễn tài liệu và xử lý truy vấn bằng các phép tính toán trong không gian vector. Ở hình 3 là minh họa cách biểu diễn vector tài liệu của một mô hình không gian vector cho các tài liệu d_1, d_2, d_3 với

- d_1 = `Presidential Library`
- d_2 = `Programming Library`
- d_3 = `Presidential Programming`



Hình 3. Minh họa cách biểu diễn tài liệu trong một mô hình không gian vector [7]

Để biểu diễn tài liệu thành các vector, đầu tiên ta phải định nghĩa không gian vector. Như hình 3, không gian vector cho các tài liệu này là một không gian ba chiều, mỗi chiều dùng để biểu diễn một từ trong tài liệu với trục hoành là từ `Library`, trục tung là từ `Presidential` và trục cao là từ `Programming`. Như vậy, với tài liệu d_1 gồm

hai từ `Presidential` và `Library`, đơn giản nhất thì d_1 được biểu diễn như một vector trong mặt phẳng Oxy. Dấu chấm hỏi bên cạnh d_1 để biểu thị rằng liệu biểu diễn vector tài liệu d_1 như thế này thì có đúng không. Tương tự với các tài liệu d_2 và d_3. Và với không gian vector này, ta có thể biểu diễn tất cả các tài liệu d_4, d_5, d_M, kể cả truy vấn q. Vấn đề tiếp theo là phải tính toán độ tương đồng giữa truy vấn và các tài liệu như thế nào. Như hình 3 thì có vẻ truy vấn q gần với tài liệu d_2 nhất.

Như vậy, để xây dựng một mô hình không gian vector, cần phải:

- Định nghĩa/chọn tập term để định nghĩa không gian vector
- Định nghĩa cách biểu diễn tài liệu và truy vấn trong không gian vector
- Định nghĩa cách tính độ tương đồng và xếp hạng cho các tài liệu.

Cách định nghĩa mô hình của chúng tôi trong đồ án này được trình bày chi tiết ở phần sau.

3.2 Mô hình

Đầu tiên, chúng tôi chọn tập term là tất cả những từ (token) có trong tài liệu theo mô hình Bag-of-Words (BOW). Qua bước này, chúng tôi định nghĩa được mỗi vector sẽ có T chiều với T là số lượng term có trong tập tài liệu.

Tiếp theo chúng tôi chọn cách biểu diễn tài liệu và truy vấn bằng trọng số tf-idf [6]. Giá trị $\text{tf-idf}_{t,d}$ của một term t trong tài liệu d được tính bằng công thức:

$$\text{tf-idf}_{t,d} = \text{tf}_{t,d} * \text{idf}_t \tag{1}$$

trong đó:

- $\text{tf}_{t,d}$ là tần suất xuất hiện của term t trong tài liệu d, được tính bằng công thức:

  $$\text{tf}_{t,d} = \text{count}(t, d) \tag{2}$$

  Tuy nhiên, với công thức sử dụng số lần đếm thuần như trên thì mô hình sẽ thiên vị những tài liệu có độ dài lớn hơn vì tài liệu càng dài thì có khả năng số lần xuất hiện của từ cũng lớn hơn, nhưng thực tế thì không nhất thiết phải như vậy. Hơn nữa, trong cùng một tài liệu, một term xuất hiện 100 lần không nhất thiết phải quan trọng gấp 100 lần so với term chỉ xuất hiện 1 lần. Vì vậy, để giảm bớt sự ảnh hưởng của tần suất term, chúng tôi sử dụng scaling với phương pháp Sublinear TF scaling bằng công thức sau:

  $$\text{tf}_{t,d} = \begin{cases} 1 + \log_{10} \text{count}(t, d) & \text{nếu count}(t, d) > 0 \\ 0 & \text{ngược lại} \end{cases} \tag{3}$$

- idf_t là nghịch đảo tần suất tài liệu có chứa term t, được tính bằng công thức sau:

  $$\text{idf}_t = \log_{10} \frac{N}{\text{df}_t} = \log_{10} \frac{N}{\text{count}(d, t)} \tag{4}$$

  với N là số lượng tài liệu có trong tập tài liệu và df_t là số lượng tài liệu có chứa term t và $\text{df}_t = \text{count}(d, t)$.

Giá trị $\text{tf-idf}_{t,d}$ của term t trong tài liệu d sẽ:

- Cao nhất khi term t xuất hiện nhiều lần trong tài liệu d và trong một số lượng tài liệu nhỏ. Khi đó, ta nói term t giúp mô hình có khả năng phân biệt (discrimative) một số ít tài liệu so với các tài liệu khác (không chứa/chứa ít term t).
- Thấp hơn khi term t xuất hiện ít lần trong tài liệu d hoặc xuất hiện trong nhiều tài liệu khác. Khi đó, term t mang lại tín hiệu phân biệt ít rõ ràng hơn.
- Thấp nhất khi term t xuất hiện trong hầu hết các tài liệu.

Qua định nghĩa này, chúng tôi có được mỗi vector tài liệu/truy vấn sẽ có T chiều với mỗi phần tử là giá trị $\text{tf-idf}_{t,d}$ của term t trong tài liệu d.

Cuối cùng, chúng tôi sẽ tính độ tương đồng và xếp hạng tài liệu dựa trên độ đo cosine similarity. Độ tương đồng giữa tài liệu d và truy vấn q được tính bằng độ đo cosine similarity bằng công thức:

$$\text{score}(q, d) = \cos(q, d) = \frac{q \cdot d}{|q| \cdot |d|} = \frac{q}{|q|} \cdot \frac{d}{|d|} \tag{5}$$

Khai triển 5, ta có:

$$\text{score}(q, d) = \sum_{t \in q} \frac{\text{tf-idf}_{t,q}}{\sqrt{\sum_{q_i \in q} \text{tf-idf}^2_{q_i,q}}} \cdot \frac{\text{tf-idf}_{t,d}}{\sqrt{\sum_{d_i \in d} \text{tf-idf}^2_{d_i,d}}} \tag{6}$$

Trong thực tế, ta có thể bỏ đi phần tử $\frac{q}{|q|} = \frac{\text{tf-idf}_{t,q}}{\sqrt{\sum_{q_i \in q} \text{tf-idf}^2_{q_i,q}}}$ vì $|q|$ là giống nhau cho mọi tài d và độ dài của truy vấn thường ngắn, mỗi term thường chỉ xuất hiện một lần nên cũng có thể bỏ qua được. Vì vậy, công thức tính độ tương đồng của chúng tôi là:

$$\text{score}(q, d) = \sum_{t \in q} \frac{\text{tf-idf}_{t,d}}{\sqrt{\sum_{d_i \in d} \text{tf-idf}^2_{d_i,d}}} \tag{7}$$

Chúng tôi cũng xếp hạng các tài liệu dựa trên công thức 7 bằng cách tài liệu nào có giá trị $\text{score}(q, d)$ lớn hơn thì sẽ được xếp hạng cao hơn.

3.3 Ví dụ

Cho một truy vấn q và bốn tài liệu d_1, d_2, d_3, d_4 như sau:

Query:	sweet love
Doc 1:	sweet sweet nurse love
Doc 2:	sweet sorrow
Doc 3:	how sweet is love
Doc 4:	nurse

Ta sẽ thực hiện truy xuất thông tin cho truy vấn này trên tập dữ liệu như trên bằng mô hình không gian vector. Đầu tiên, theo phương pháp chọn term BOW, ta được một tập term gồm có 6 term là {`sweet, nurse, love, sorrow, how, is`}. Như vậy, mỗi vector tài liệu/truy vấn sẽ có 6 chiều.

Tiếp theo, tính các giá trị $\text{tf-idf}_{t,d}$ đối với từng term t trong tài liệu d theo các công thức 3, 4, 1. Đối với tập tài liệu này, số lượng tài liệu N là 4. Chi tiết kết quả tính toán được trình bày ở bảng 1.

Doc 1					
term	count	$\text{tf}_{t,d}$	df_t	idf_t	$\text{tf-idf}_{t,d}$
sweet	2	1.3	3	0.12	0.16
nurse	1	1	2	0.3	0.3
love	1	1	2	0.3	0.3
sorrow	0	0	1	0.6	0
how	0	0	1	0.6	0
is	0	0	1	0.6	0

(a) Tài liệu d_1

Doc 2					
term	count	$\text{tf}_{t,d}$	df_t	idf_t	$\text{tf-idf}_{t,d}$
sweet	1	1	3	0.12	0.12
nurse	0	0	2	0.3	0
love	0	0	2	0.3	0
sorrow	1	1	1	0.6	0.6
how	0	0	1	0.6	0
is	0	0	1	0.6	0

(b) Tài liệu d_2

Doc 3					
term	count	$\text{tf}_{t,d}$	df_t	idf_t	$\text{tf-idf}_{t,d}$
sweet	1	1	3	0.12	0.12
nurse	0	0	2	0.3	0
love	1	1	2	0.3	0.3
sorrow	0	0	1	0.6	0
how	1	1	1	0.6	0.6
is	1	1	1	0.6	0.6

(c) Tài liệu d_3

Doc 4					
term	count	$\text{tf}_{t,d}$	df_t	idf_t	$\text{tf-idf}_{t,d}$
sweet	0	0	3	0.12	0
nurse	1	1	2	0.3	0.3
love	0	0	2	0.3	0
sorrow	0	0	1	0.6	0
how	0	0	1	0.6	0
is	0	0	1	0.6	0

(d) Tài liệu d_4

Bảng 1. Kết quả tính toán vector biểu diễn tài liệu theo trọng số tf-idf của các tài liệu cho mô hình không gian vector

Với truy vấn q, nếu cũng tính toán và tạo vector để tính cosine similarity thì cũng tính tương tự như các tài liệu. Tuy nhiên, nếu tính theo công thức 2 thì không cần quan tâm đến các giá trị tf-idf của truy vấn mà chỉ cần tính cho các tài liệu là đủ. Chi tiết kết quả tính toán được trình bày ở bảng 2.

Doc	$\|d\|$	tf-idf$_{\texttt{sweet}}$	tf-idf$_{\texttt{love}}$	score(q, d)
1	0.45	0.16	0.3	1.02
2	0.61	0.12	0	0.2
3	0.91	0.12	0.3	0.46
4	0.3	0	0	0

Bảng 2. Kết quả tính toán độ tương đồng của truy vấn và các tài liệu

Cuối cùng, ta so sánh các giá trị score và xếp hạng tài liệu. Với kết quả ở bảng 2, ta có danh sách xếp hạng là d_1, d_3, d_2, d_4.

4 Mô hình xác suất

4.1 Ý tưởng

Khác với mô hình không gian vector, các mô hình xác suất tính toán độ tương đồng và xếp hạng các tài liệu dựa trên xác suất một tài liệu d và truy vấn q có liên quan với nhau hay:

$$p(R = 1|d, q), R \in \{0, 1\}$$

trong đó R là biến ngẫu nhiên nhị phân biểu thị sự liên quan của tài liệu d và truy vấn q. R có hai giá trị:

- $R = 0$: tài liệu d và truy vấn q không liên quan với nhau
- $R = 1$: tài liệu d và truy vấn q có liên quan với nhau

Câu hỏi đặt ra là như thế nào được gọi là có liên quan với nhau. Ở [7] đã đề ra một hệ thống trong đó người dùng sẽ đưa vào các truy vấn và hệ thống sẽ trả về một danh sách các tài liệu. Tài liệu và truy vấn được tính là có liên quan với nhau khi người dùng nhấn (click) vào tài liệu để xem. Một bảng thống kê đơn giản được minh họa ở hình 4.

Query *q*	Document *d*	Relevant? *R*
q1	d1	1
q1	d2	1
q1	d3	0
d1	d4	0
q1	d5	1
⋮		
q1	d1	0
d1	d2	1
q1	d3	0
q2	d3	1
q3	d1	1
q4	d2	1
q4	d3	0

Hình 4. Minh họa một bảng thống kê sự liên quan của tài liệu và truy vấn với hệ thống ở [7]

Với hệ thống này, có thể tính $p(R = 1|d, q)$ bằng công thức sau:

$$p(R = 1|d, q) = \frac{\text{count}(q, d, R = 1)}{\text{count}(q, d)}$$

trong đó:

- $\text{count}(q, d, R = 1)$ là số lần người dùng nhập vào truy vấn q và nhấn vào tài liệu d mà hệ thống trả về để xem
- $\text{count}(q, d)$ là số lần người dùng nhập vào truy vấn q và hệ thống trả về tài liệu d

Một số kết quả ví dụ theo bảng thống kê trên như sau:

$$p(R = 1|q1, d1) = \frac{1}{2} = 0.5$$
$$p(R = 1|q1, d2) = \frac{2}{2} = 1$$
$$p(R = 1|q1, d3) = \frac{0}{2} = 0$$

Đây là một cách để thống kê và tính toán $p(R = 1|d, q)$ khá đơn giản. Tuy nhiên, hệ thống này có các vấn đề như sau:

- Unseen document: khi truy vấn và tìm kiếm kết quả, thì người dùng thường chỉ nhấn vào các tài liệu đầu tiên và đã có thể có được câu trả lời. Như vậy, các tài liệu nằm sau mặc dù có liên quan nhưng vì người dùng không nhấn vào nên được thống kê là không liên quan. Điều này dẫn đến sai lệch trong kết quả thống kê. Để thống kê đúng thì người dùng phải đánh giá toàn bộ các tài liệu có trong tập tài liệu, điều này là rất khó và tiêu tốn rất nhiều thời gian và nhân lực.
- Unseen query: số lượng truy vấn mà người dùng có thể nhập vào rất đa dạng, vì vậy cũng không thể thống kê được tất cả các truy vấn được.

Vì vậy, để xây dựng một hệ thống giống như vậy ngoài thực tế là một điều không thể, hoặc nếu có thể thì sẽ tiêu tốn rất nhiều tài nguyên và thời gian để thống kê và tính toán. Thay vào đó, ta sẽ tính xác suất tài liệu d có liên quan đến truy vấn q khi cho trước truy vấn q hay:

$$p(d|q)$$

Triển khai công thức trên bằng Bayes' rule, ta được:

$$p(d|q) = \frac{p(d)p(q|d)}{p(q)}$$

trong đó:

- $p(d)$ là xác suất có tài liệu d
- $p(q)$ là xác suất có truy vấn q
- $p(q|d)$ là xác suất của truy vấn q khi có tài liệu d

Vì $p(q)$ là giống nhau đối với tất cả các tài liệu, và $p(d)$ thường là giống nhau với tất cả các tài liệu có trong tập tài liệu, vì vậy ta có thể đơn giản hai phần tử này đi. Vậy cuối cùng, công thức tính xác suất của mô hình xác suất là:

$$p(d|q) = p(q|d) \tag{8}$$

Để tính $p(q|d)$, chúng tôi đã sử dụng mô hình Query Likelihood. Đây là một mô hình thuộc nhóm mô hình ngôn ngữ (là một nhóm nhỏ trong mô hình xác suất). Ý tưởng của mô hình này là xây dựng một mô hình ngôn ngữ M_d cho từng tài liệu d và xếp hạng tài liệu bằng xác suất mô hình ngôn ngữ M_d có thể sinh ra truy vấn q hay:

$$p(q|d) = p(q|M_d) \tag{9}$$

Để xây dựng mô hình Query Likelihood, cần phải:

- Định nghĩa cách thức xây dựng mô hình ngôn ngữ
- Định nghĩa cách tính toán của $p(q|M_d)$

Cách định nghĩa mô hình của chúng tôi trong đồ án này được trình bày chi tiết ở phần sau.

4.2 Mô hình

Để xây dựng mô hình Query Likelihood, đầu tiên, chúng tôi lựa chọn cách xây dựng mô hình ngôn ngữ unigram. Các mô hình ngôn ngữ nói chung đều nhắm đến mục tiêu tính toán xác suất một chuỗi q được tạo thành từ các term $q_1q_2...q_m$ có khả năng xảy ra dựa trên một tài liệu nào đó hay:

$$p(q) = p(q_1q_2...q_m) = p(q_1)p(q_2|q_1)p(q_3|q_1q_2)...p(q_m|q_1...q_{m-1}) \tag{10}$$

trong đó xác suất của một token nào đó sẽ phụ thuộc vào các token đứng trước nó. Tuy nhiên, mô hình ngôn ngữ unigram định nghĩa rằng xác suất để một term xảy ra là độc lập so với những term khác hay:

$$p(q) = p(q_1q_2...q_m) = p(q_1)p(q_2)p(q_3)...p(q_m) \tag{11}$$

Để tính xác suất của term t trong mô hình ngôn ngữ M_d của tài liệu d, chúng tôi sử dụng công thức ước lượng Maximum Likelihood Estimation như sau:

$$p(t|M_d) \approx \hat{p}_{MLE}(t|M_d) = \frac{\text{count}(t,d)}{L_d} \tag{12}$$

với L_d là độ dài của tài liệu d (hay số lượng token trong tài liệu d).

Như vậy, bằng mô hình ngôn ngữ unigram và công thức ước lượng Maximum Likelihood Estimation, ta có xác suất tài liệu d có liên quan đến truy vấn q được tính như sau:

$$p(d|q) = p(q|d) = p(q|M_d) \approx \prod_{t\in q} \hat{p}_{MLE}(t|M_d) = \prod_{t\in q} \frac{\text{count}(t,d)}{L_d} \tag{13}$$

Ví dụ, ta có một truy vấn $q =$ `sweet love` và một tài liệu $d =$ `sweet sweet nurse love`. Xác suất $p(d|q)$ sẽ được bằng công thức 13 như sau:

$$p(q|M_d) = p(\texttt{sweet love}|M_d) = p(\texttt{sweet}|M_d)p(\texttt{love}|M_d) = \frac{\text{c}(\texttt{sweet}|d)}{L_d}\cdot\frac{\text{c}(\texttt{love}|d)}{L_d} = \frac{1}{8}$$

với $\text{c}(\texttt{sweet}|d) = 2$, $\text{c}(\texttt{love}|d)$ và $L_d = 4$.

Đây là cách tính xác suất của mô hình Query Likelihood. Tuy nhiên, với những truy vấn ví dụ như $q =$ `nurse love sorrow` thì kết quả trả về sẽ bằng 0 vì term `sorrow` trong truy vấn không xuất hiện trong tài liệu mặc dù trong tài liệu có chứa hai term còn lại. Tệ hơn nữa là ta không thể phân biệt được $q_1 =$ `nurse love sorrow` và $q_2 =$ `information retrieval` vì cả hai giá trị xác suất này đều bằng 0 trong khi rõ ràng rằng tài liệu sẽ có liên quan đến q_1 nhiều hơn q_2. Để giải quyết vấn đề này, chúng tôi đã sử dụng smoothing. Mục tiêu chung của smoothing là tránh trường hợp xác suất bằng 0 do trong truy vấn xuất hiện những term không có trong tài liệu hoặc trong tập từ vựng. Trong đồ án này, chúng tôi sử dụng phương pháp Linear Interpolation smoothing bằng công thức như sau:

$$p(t|M_d) = \lambda\hat{p}_{MLE}(t|M_d) + (1-\lambda)\hat{p}_{MLE}(t|M_C) \tag{14}$$

với

- M_C là mô hình ngôn ngữ trên toàn bộ tập tài liệu
- λ là hệ số tự định nghĩa có giá trị trong khoảng (0,1). Việc chọn giá trị λ sẽ ảnh hưởng đến kết quả của mô hình.

Như vậy, kết hợp với smoothing thì xác suất tài liệu d có liên quan đến truy vấn q được tính như sau:

$$\hat{p}(q|M_d) = \prod_{t\in q} (\lambda\hat{p}_{MLE}(t|M_d) + (1-\lambda)\hat{p}_{MLE}(t|M_C)) \tag{15}$$

$$= \prod_{t\in q} \left(\lambda\frac{\text{count}(t,d)}{L_d} + (1-\lambda)\frac{\text{count}(t,C)}{L_C}\right) \tag{16}$$

Vì các giá trị xác suất có giá trị trong khoảng (0,1) nên khi nhân lại với nhau sẽ có hiện tượng underflow vì số quá nhỏ. Cho nên chúng tôi sẽ có thêm một bước chuẩn hóa log như sau:

$$\hat{p}(q|M_d) = \sum_{t\in q} \log_{10}\left(\lambda\frac{\text{count}(t,d)}{L_d} + (1-\lambda)\frac{\text{count}(t,C)}{L_C}\right) \tag{17}$$

và công thức này cũng chính là công thức dùng để xếp hạng các tài liệu. Tài liệu nào có giá trị xác suất lớn hơn sẽ được xếp hạng cao hơn.

4.3 Ví dụ

Cho một truy vấn q và bốn tài liệu d_1, d_2, d_3, d_4 như sau:

Query:	sweet love
Doc 1:	sweet sweet nurse love
Doc 2:	sweet sorrow
Doc 3:	how sweet is love
Doc 4:	nurse

Ta sẽ thực hiện truy xuất thông tin cho truy vấn này trên tập dữ liệu như trên bằng mô hình Query Likelihood. Với mô hình này, ta cũng lấy tập term theo phương pháp BOW như mô hình không gian vector. Vậy, tập term sẽ có 6 term là {`sweet, nurse, love, sorrow, how, is`}. Đầu tiên, ta sẽ xây dựng mô hình ngôn ngữ cho từng tài liệu. Chi tiết kết quả tính toán được trình bày ở bảng 3.

Doc 1			
term	count	L_d	$\hat{p}_{MLE}(t\|M_d)$
sweet	2	4	0.5
nurse	1	4	0.25
love	1	4	0.25
sorrow	0	4	0
how	0	4	0
is	0	4	0

(a) Tài liệu d_1

Doc 2			
term	count	L_d	$\hat{p}_{MLE}(t\|M_d)$
sweet	1	2	0.5
nurse	0	2	0
love	0	2	0
sorrow	1	2	0.5
how	0	2	0
is	0	2	0

(b) Tài liệu d_2

Doc 3			
term	count	L_d	$\hat{p}_{MLE}(t\|M_d)$
sweet	1	4	0.25
nurse	0	4	0
love	1	4	0.25
sorrow	0	4	0
how	1	4	0.25
is	1	4	0.25

(c) Tài liệu d_3

Doc 4			
term	count	L_d	$\hat{p}_{MLE}(t\|M_d)$
sweet	0	1	0
nurse	1	1	1
love	0	1	0
sorrow	0	1	0
how	0	1	0
is	0	1	0

(d) Tài liệu d_4

Bảng 3. Kết quả tính toán mô hình ngôn ngữ cho từng tài liệu của mô hình Query Likelihood

Tiếp theo ta sẽ tính mô hình ngôn ngữ cho toàn bộ tập tài liệu. Kết quả tính toán được trình bảy ở bảng 4.

term	count	L_C	$\hat{p}_{MLE}(t\|M_C)$
sweet	4	11	0.37
nurse	2	11	0.18
love	2	11	0.18
sorrow	1	11	0.1
how	1	11	0.1
is	1	11	0.1

Bảng 4. Kết quả tính toán mô hình ngôn ngữ cho toàn bộ tập tài liệu

Tiếp theo, ta tính xác suất tài liệu d có liên quan đến truy vấn q bằng công thức 17 với $\lambda = 0.5$. Kết quả tính toán được trình bày ở bảng 5.

Doc	$\hat{p}_{MLE}(\texttt{sweet}\|M_d)$	$\hat{p}_{MLE}(\texttt{sweet}\|M_C)$	$\hat{p}_{MLE}(\texttt{love}\|M_d)$	$\hat{p}_{MLE}(\texttt{love}\|M_C)$	$\hat{p}(q\|M_d)$
1	0.5	0.37	0.25	0.18	-1.03
2	0.5	0.37	0	0.18	-1.41
3	0.25	0.37	0.25	0.18	-1.18
4	0	0.37	0	0.18	-1.78

Bảng 5. Kết quả $\hat{p}(q|M_d)$ cho từng tài liệu

Cuối cùng, ta so sánh các giá trị score và xếp hạng tài liệu. Với kết quả ở bảng 5, ta có danh sách xếp hạng là $d1, d3, d2, d4$.

5 Phương pháp phân tích tài liệu

Với phương pháp chọn term bằng BOW, phải lưu trữ tất cả các từ có xuất hiện trong toàn bộ tài liệu, gồm cả những từ không quan trọng: stop words, dấu câu, các thể khác nhau của một từ... Cách này là không tối ưu và hiệu quả vì phải tốn thêm bộ nhớ để lưu trữ những từ lặp lại hoặc những từ không quan trọng và cũng làm cho thời gian truy xuất tăng lên. Vì vậy, phải có một phương pháp phân tích tài liệu nhằm giảm bớt tập terms, từ đó giúp giảm bộ nhớ lưu trữ và thời gian tính toán.

Một số phương pháp phân tích tài liệu phổ biến là:

- Loại bỏ stop words (mạo từ, giới từ, trợ từ...) vì các stop words xuất hiện hầu hết trong mọi tài liệu và không mang nhiều ngữ nghĩa và không giúp phân biệt các tài liệu với nhau. Hình 5 là danh sách 25 stop words phổ biến trong dataset Reuters-RCV1 [6].
- Đưa từ về dạng nguyên bản bằng stemming và lemmatization. Stemming là phương pháp sử dụng heuristic và bắt bớt các ký tự cuối của từ nhằm đưa từ về dạng gốc. Còn lemmatization trước tiên sẽ xem xét đến ngữ nghĩa và cách dùng của từ trong câu/tài liệu rồi mới thực hiện cắt bỏ. Vì vậy stemming sẽ làm giảm số lượng terms nhiều hơn lemmatization, nhưng có thể tạo ra những từ không có thực. Hình 6 là minh họa kết quả của stemming và lemmatization trong tiếng Anh [6].

a	an	and	are	as	at	be	by	for	from
has	he	in	is	it	its	of	on	that	the
to	was	were	will	with					

Hình 5. Danh sách 25 stop words phổ biến trong dataset Reuters-RCV1 [6]

$$\text{am, are, is} \Rightarrow \text{be}$$
$$\text{car, cars, car's, cars'} \Rightarrow \text{car}$$

Hình 6. Minh họa kết quả của stemming và lemmatization trong tiếng Anh [6]

Trong đồ án này, chúng tôi sử dụng phương pháp phân tích tài liệu gồm các bước như sau:

- Bước 1: Chuẩn hóa các ký tự về dạng thường, loại bỏ ký tự số, dấu câu, ký tự xuống dòng.
- Bước 2: Loại bỏ stop words.
- Bước 3: Stemming bằng Porter stemmer. Đây là mô hình stemming tiếng Anh phổ biến nhất. Mô hình sử dụng thuật toán Porter gồm 5 giai đoạn; ở mỗi giai đoạn, thuật toán sẽ lựa chọn một số luật phù hợp để rút ngắn từ. Hình 7 là minh họa một luật trong thuật toán Porter và hình 8 là kết quả stemming của Porter stemmer và một số stemmer khác [6].
- Bước 4: Loại bỏ các từ ngắn chỉ gồm 1, 2 ký tự.

Rule			**Example**		
SSES	→	SS	caresses	→	caress
IES	→	I	ponies	→	poni
SS	→	SS	caress	→	caress
S	→		cats	→	cat

Hình 7. Minh họa một luật (đưa một dạng số nhiều thành dạng nguyên gôc số ít) trong thuật toán Porter [6]

Sample text: Such an analysis can reveal features that are not easily visible from the variations in the individual genes and can lead to a picture of expression that is more biologically transparent and accessible to interpretation

Lovins stemmer: such an analys can reve featur that ar not eas vis from th vari in th individu gen and can lead to a pictur of expres that is mor biolog transpar and acces to interpres

Porter stemmer: such an analysi can reveal featur that ar not easili visibl from the variat in the individu gene and can lead to a pictur of express that is more biolog transpar and access to interpret

Paice stemmer: such an analys can rev feat that are not easy vis from the vary in the individ gen and can lead to a pict of express that is mor biolog transp and access to interpret

Hình 8. Kết quả stemming của Porter stemmer và một số stemmer khác [6]

Phương pháp trên đã giúp giảm số lượng term và từ đó giúp cải thiện bộ nhớ lưu trữ và thời gian truy xuất của mô hình. Một số ví dụ cụ thể ở hai tập dữ liệu Cranfield và NFCorpus-test được minh họa ở hình 12.

Org (Cranfield doc 3)	the boundary layer in simple shear flow past a flat plate . the boundary-layer equations are presented for steady incompressible flow with no pressure gradient .
Processed	boundari layer simpl shear flow past flat plate boundari layer equat present steadi incompress flow pressur gradient
Org (NF-Corpus test MED-722)	understanding excessive intestinal gas pubmed ncbi abstract complaints excessive gas patients common difficult impossible physician document review addresses pathophysiology management complaints sources routes elimination excessive eructation bloating distention addition common flatulence problems summarized including excessive flatus volume noxious flatus
Processed	understand excess intestin pubm ncbi abstract complaint excess patient common difficult imposs physician document review address pathophysiolog manag complaint sourc rout elimin excess eruct bloat distent addit common flatul problem summar includ excess flatu volum noxiou flatu

Hình 9. Minh họa tài liệu gốc và tài liệu đã qua xử lý ở hai tập dữ liệu Cranfield và NfCorpus test

6 Lập và truy xuất chỉ mục

Khi truy xuất thông tin, phải xét toàn bộ tập tài liệu trong khi ta chỉ cần quan tâm đến những tài liệu có liên quan đến truy vấn. Cách tính toán này là không hiệu quả và làm hiệu suất truy xuất thông tin giảm đi đáng kể. Vì vậy, cần phải lập chỉ mục để giảm đi số lượng tài liệu cần xét, từ đó giúp giảm bộ nhớ lưu trữ và thời gian tính toán.

Một cấu trúc chỉ mục phổ biến là chỉ mục đảo ngược (inverted index). Ta có thể cho rằng những tài liệu không chứa term nào trong truy vấn thì khả năng cao không liên quan đến truy vấn. Vì vậy chỉ mục này cho phép mô hình truy xuất đến ngay những tài liệu có chứa term t trong truy vấn q và chỉ tính toán trên những tài liệu có chứa ít nhất một term trong truy vấn. Những tài liệu không chứa term nào thì sẽ không được truy xuất đến. Nhờ vậy, thời gian truy xuất sẽ giảm đi đáng kể. Hơn nữa, trong chỉ mục có thể lưu trữ tích hợp thêm những thông tin cần thiết để tính toán, nhờ vậy, quá trình tính toán có thể rút ngắn đi rất nhiều.

Trong đồ án này, chúng tôi lập hai chỉ mục riêng biệt cho hai mô hình như trình bày ở hai mục nhỏ bên dưới.

6.1 Lập và truy xuất chỉ mục cho mô hình không gian vector

Chỉ mục đảo ngược của mô hình không gian vector được lưu trữ bằng một từ điển với key là term và value là một từ điển có key là id của tài liệu chứa term đó và giá trị $\frac{\text{tf-idf}_{t,d}}{|d|}$:

$$\text{inverted index vector} = \{\text{term: } \{\text{doc-id: } \frac{\text{tf-idf}_{t,d}}{|d|}\}\}$$

Tiếp theo là một ví dụ cụ thể minh họa cho quá trình truy xuất chỉ mục đối với mô hình không gian vector. Cho một truy vấn q và bốn tài liệu d_1, d_2, d_3, d_4 như sau:

Query:	sweet love
Doc 1:	sweet sweet nurse love
Doc 2:	sweet sorrow
Doc 3:	how sweet is love
Doc 4:	nurse

Với các kết quả đã tính ở mục 3.3, ta tạo chỉ mục như sau:

`sweet` :	$\{1 : 0.36, 2 : 0.2, 3 : 0.13\}$
`nurse` :	$\{1 : 0.67, 4 : 1\}$
`love` :	$\{1 : 0.67, 3 : 0.33\}$
`sorrow` :	$\{2 : 0.98\}$
`how` :	$\{3 : 0.67\}$
`is` :	$\{3 : 0.67\}$

Tiếp theo, tạo một từ điển rỗng để lưu danh sách các tài liệu có liên quan đến q. Từ điển này sẽ có key id của doc và value là $\text{score}_{q,d}$. Lần lượt duyệt qua từng term trong truy vấn và thêm các tài liệu chứa term vào danh sách kết quả hoặc cộng dồn score cho tài liệu. Với $q =$ `sweet love`, ta tính như sau:

- Khởi tạo `rst`={}
- Với term `sweet`, ta xét chỉ mục của term. Vì các tài liệu có id 1, 2, 3 đều chưa có trong danh sách nên ta thêm các id cũng như giá trị $\frac{\text{tf-idf}_{t,d}}{|d|}$ này vào `rst`. Sau bước này, `rst` được cập nhật: `rst`={1: 0.36, 2:0.2, 3:0.13}
- Với term `love`, ta xét chỉ mục tương tự như `sweet`. Vì cả tài liệu có id 1 và 3 đã có trong danh sách kết quả, ta cộng dồn giá trị $\frac{\text{tf-idf}_{t,d}}{|d|}$. Vậy `rst`={1: 1.03, 2:0.2, 3:0.46}

Sau cùng, ta xếp hạng tài liệu dựa trên giá trị $\text{score}_{q,d}$ trong từ điển kết quả. Như vậy, ta có danh sách xếp hạng là d_1, d_3, d_2. Vì d_4 không chứa term nào trong truy vấn nên không phải xét. Như vậy, chỉ mục đã giúp chúng ta loại bỏ những tài liệu không liên quan, từ đó, làm giảm thời gian tính toán. Không những thế, với cấu trúc chỉ mục có lưu trữ thêm những thông tin cần thiết như tf-idf, việc tính toán không cần lặp lại quá nhiều và từ đó giúp giảm chi phí tính toán đáng kể.

slipstream	{'1': 0.31428074778100523, '1064': 0.24726391629414696, '1089': 0.19998017615999716, '1090': 0.24916971528527188, '1091': 0.1823776162165622, '1092': 0.12605270295450452, '1094': 0.21230460725772424, '1095': 0.1298287320848947, '1144': 0.19929002175259944, '1164': 0.12925163438182743, '1165': 0.1422757959459282, '1166': 0.12682279532846974, '409': 0.23632538110557663, '453': 0.23718530411118838, '484': 0.2533679007487949}

Hình 10. Minh họa một term trong chỉ mục của mô hình không gian vector với bộ dữ liệu Cranfield

6.2 Lập và truy xuất chỉ mục cho mô hình xác suất

Chỉ mục đảo ngược của mô hình xác suất Query Likelihood được lưu trữ bằng một từ điển với key là term và value là một từ điển con có key là id của các tài liệu có chứa term đó và value là giá trị xác suất của term trên mô hình ngôn ngữ của tài liệu $\hat{p}_{MLE}(t|M_d) = \frac{\text{count}(t,d)}{L_d}$. Đặc biệt, từ điển con này có thêm một key là 'all' với value là xác suất của term trên mô hình ngôn ngữ của tập tài liệu $\hat{p}_{MLE}(t|M_C) = \frac{\text{count}(t,C)}{L_C}$. Cấu trúc của chỉ mục như sau:

$$\text{inverted index prob} = \{\text{term: } \{\text{doc-id: } \hat{p}_{MLE}(t|M_d)\}$$

Tiếp theo là một ví dụ cụ thể minh họa cho quá trình truy xuất chỉ mục đối với mô hình xác suất Query Likelihood. Cho một truy vấn q và bốn tài liệu d_1, d_2, d_3, d_4 như sau:

Query:	sweet love
Doc 1:	sweet sweet nurse love
Doc 2:	sweet sorrow
Doc 3:	how sweet is love
Doc 4:	nurse

Với các kết quả đã tính ở mục 4.3, ta tạo chỉ mục như sau:

`sweet` :	$\{1 : 0.5, 2 : 0.5, 3 : 0.25, all : 0.37\}$
`nurse` :	$\{1 : 0.25, 4 : 1, all : 0.18\}$
`love` :	$\{1 : 0.25, 3 : 0.25, all : 0.18\}$
`sorrow` :	$\{2 : 0.5, all : 0.1\}$
`how` :	$\{3 : 0.25, all : 0.1\}$
`is` :	$\{3 : 0.25, all : 0.1\}$

Tiếp theo, tạo một từ điển rỗng để lưu danh sách các tài liệu có liên quan đến q. Từ điển này sẽ có key id của doc và value là $\hat{p}(q|M_d)$. Đầu tiên, ta sẽ duyệt tất cả các term trong truy vấn và thêm tất cả các tài liệu có chứa ít nhất một term từ chỉ mục. Sau đó lần lượt duyệt qua từng tài liệu và tính giá trị xác suất cho tài liệu với công thức 17. Với $q =$ `sweet love` và $\lambda = 0.5$, ta tính như sau:

- Khởi tạo `rst`={}
- Duyệt qua chỉ mục của các term trong truy vấn. Với truy vấn này, ta được ba tài liệu là 1, 2, 3.
- Tính giá trị $\hat{p}(q|M_d)$ theo công thức 17. Những giá trị này đều đã có trong chỉ mục nên ta chỉ cần truy xuất vào chỉ mục là đủ. Sau bước này, ta được tập `rst`={1: -1.03, 2: -1.41, 3: -1.18}

Sau cùng, ta xếp hạng tài liệu dựa trên giá trị $\hat{p}(q|M_d)$ trong từ điển kết quả. Như vậy, ta có danh sách xếp hạng là d_1, d_3, d_2. Thực tế, ta cũng phải tính toán giá trị $\hat{p}(q|M_d)$ của tài liệu 4 vì mặc dù không chứa term nào nhưng theo công thức 17 thì $\hat{p}(q|M_d)$ của tài liệu 4 vẫn khác 0. Tuy nhiên, giá trị này chắc chắn sẽ thấp hơn so với các tài liệu có chứa ít nhất một term trong truy vấn như tài liệu 1, 2 và 3 cho nên ở đây chúng tôi sẽ bỏ luôn những tài liệu không chứa term trong truy vấn.

finland	{'all': 4.34636155208571e-05, 'MED-10': 0.011695906432748537, 'MED-1201': 0.008264462809917356, 'MED-1242': 0.004878048780487805, 'MED-1276': 0.03225806451612903, 'MED-2582': 0.014184397163120567, 'MED-3856': 0.010309278350515464, 'MED-3976': 0.006802721088435374, 'MED-3981': 0.006578947368421052, 'MED-4070': 0.007142857142857143, 'MED-4727': 0.018018018018018018, 'MED-5192': 0.006060606060606061, 'MED-5331': 0.015384615384615385}

Hình 11. Minh họa một term trong chỉ mục của mô hình xác suất Query Likelihood với bộ dữ liệu NFCorpus test

7 Quá trình tính toán truy vấn

Đối với truy vấn, chúng tôi tiền xử lý giống như tài liệu gồm 4 bước đã trình bày ở mục 5. Sau đó, chúng tôi sẽ tokenize truy vấn đã xử lý, với mỗi token:

- Nếu token không có trong tập term, bỏ qua token này
- Ngược lại, truy xuất vào chỉ mục và tính toán như đã trình bày ở mục 6

Org	what similarity laws must be obeyed when constructing aeroelastic models of heated high speed aircraft .
Processed	similar law must obey construct aerolast model heat high speed aircraft

Hình 12. Minh họa truy vấn gốc và truy vấn đã qua xử lý ở tập dữ liệu Cranfield

8 Kết quả thực nghiệm

Chúng tôi đã thực nghiệm trên hai bộ dữ liệu là Cranfield [1] và NFCorpus test [3] [2]. Quá trình thực nghiệm gồm có đầy đủ các bước tiền xử lý dữ liệu, chọn terms, lập chỉ mục, xử lý truy vấn và truy vấn như đã trình bày ở các phần trước. Chúng tôi đã đo đạc một số kết quả thực nghiệm như sau.

Đầu tiên, chúng tôi đo thời gian lập chỉ mục cho hai mô hình trên hai bộ dữ liệu. Số lượng term và số lượng tài liệu của bộ dữ liệu Cranfield là 4129 terms và 1400 tài liệu. Với bộ dữ liệu NFCorpus test là 14829 terms và 3162 tài liệu. Kết quả ở bảng 6 cho thấy thời gian lập chỉ mục của hai mô hình gần như tương đương nhau. Vì có số lượng term và tài liệu nhiều hơn nên việc lập chỉ mục cho bộ dữ liệu NFCorpus test tốn nhiều thời gian hơn.

	Thời gian lập chỉ mục	
	Mô hình không gian vector	Mô hình Query Likelihood
Cranfield	1.26s	1.34s
NFCorpus test	4.07s	4.03s

Bảng 6. Thời gian lập chỉ mục của hai mô hình trên hai bộ dữ liệu

Tiếp theo, chúng tôi đo thời gian truy vấn của hai mô hình trên hai bộ dữ liệu. Thời gian truy vấn bao gồm thời gian đọc chỉ mục, xử lý truy vấn, tính toán truy vấn và ghi kết quả vào file. Số lượng truy vấn của tập dữ liệu Cranfield và NFCorpus test lần lượt là 225 truy vấn và 325 truy vấn. Thời gian truy vấn này là tổng thời gian bao gồm thời gian đọc chỉ mục (một lần duy nhất cho từng bộ dữ liệu), xử lý từng truy vấn,

tính toán từng truy vấn và ghi kết quả vào từng file. Ở cả hai bộ dữ liệu, thời gian truy vấn của mô hình Query Likelihood là cao hơn so với mô hình không gian vector. Với bộ dữ liệu NFCorpus test và 325 truy vấn (từ file `test.all.queries`), vì có nhiều câu truy vấn hơn và các câu truy vấn cũng dài hơn nên thời gian xử lý và tính toán truy vấn tốn thời gian hơn nhiều so với bộ dữ liệu Cranfield. Đặc biệt, với mô hình Query Likelihood trên NFCorpus test, thời gian để xử lý toàn bộ câu truy vấn lên đến gần 30 phút.

	Thời gian truy vấn	
	Mô hình không gian vector	Mô hình Query Likelihood
Cranfield	1.55s	4.7s
NFCorpus test	32.62s	1735.15s

Bảng 7. Thời gian truy vấn của hai mô hình trên hai bộ dữ liệu

Cuối cùng, chúng tôi tính toán độ chính xác của hai mô hình trên hai bộ dữ liệu bằng độ đo MAP. Với mô hình Query Likelihood, chúng tôi chọn giá trị $\lambda = 0.5$ cho cả hai bộ dữ liệu. Ở hai bộ dữ liệu, mô hình Query Likelihood đều cho kết quả cao hơn so với mô hình không gian vector.

	MAP	
	Mô hình không gian vector	Mô hình Query Likelihood ($\lambda = 0.5$)
Cranfield	0.39	0.41
NFCorpus test	0.2	0.22

Bảng 8. MAP của hai mô hình trên hai bộ dữ liệu

9 Kết luận và hướng phát triển

Trong đồ án này, chúng tôi đã tìm hiểu hai mô hình truy xuất thông tin và đã thực nghiệm được hai mô hình này trên hai bộ dữ liệu Cranfield và NFCorpus test. Ngoài ra, chúng tôi cũng hiểu được về tầm quan trọng của việc chọn term và lập chỉ mục trong quá trình tối ưu hiệu suất của hệ thống truy xuất thông tin.

Hai mô hình này cũng mới chỉ là những mô hình cơ bản nhất của truy xuất thông tin. Vì vậy, để phát triển đồ án này, chúng tôi sẽ tìm hiểu thêm những mô hình khác, cũng như tìm cách để tăng độ chính xác và thời gian truy vấn của các mô hình sẵn có.

Tài liệu tham khảo

[1] URL: `http://ir.dcs.gla.ac.uk/resources/test_collections/cran/`.

[2] URL: `https://www.cl.uni-heidelberg.de/statnlpgroup/nfcorpus/`.

[3] Vera Boteva et al. "A Full-Text Learning to Rank Dataset for Medical Information Retrieval". In: 2016. URL: `http://www.cl.uni-heidelberg.de/~riezler/publications/papers/ECIR2016.pdf`.

[4] Daniel Jurafsky and James H. Martin. *Speech and Language Processing (3rd Edition draft)*. 2021. URL: `https://web.stanford.edu/~jurafsky/slp3/`.

[5] Dominik Kuropka. "Modelle zur Repräsentation natürlichsprachlicher Dokumente: Ontologie-basiertes Information-Filtering-und-Retrieval mit relationalen Datenbanken". In: 2004.

[6] Christopher D. Manning, Prabhakar Raghavan, and Hinrich Schütze. *Introduction to Information Retrieval*. USA: Cambridge University Press, 2008. ISBN: 0521865719.

[7] ChengXiang Zhai and Sean Massung. *Text Data Management and Analysis: A Practical Introduction to Information Retrieval and Text Mining*. Association for Computing Machinery and Morgan amp; Claypool, 2016. ISBN: 9781970001174.