ĐẠI HỌC QUỐC GIA TP.HỒ CHÍ MINH

TRƯỜNG ĐẠI HỌC CÔNG NGHỆ THÔNG TIN

KHOA:KHOA HỌC MÁY TÍNH

MÔN HỌC: TRUY VẤN THÔNG TIN ĐA PHƯƠNG TIỆN

MÔ HÌNH SEARCH VỚI BOOLEAN VÀ VECTOR SPACE

Khóa: KHTN- 2014

Lớp: CS336.H21

GVLT: Nguyễn Vinh Tiệp

GVTH: Đỗ Văn Tiến

Sinh Viên: Lâm Hàn Vương - 14521106

Vũ Thế Dũng - 14520205

Nguyễn Cao Minh - 14520529

TP.Hồ Chí Minh, ngày 11 tháng 6 năm 2017

Mục Lục

Mục Lục Hình

I. Các khái niệm cơ bản

- Tài liệu – document là sản phẩm của quá trình tìm kiếm, chứa thông tin cần thiết
- Tập compound word: tập từ vựng gồm các từ có hai hoặc ba từ ghép lại có nghĩa
- Tập Stop-word: là tập từ vựng gồm các từ thường sử dụng trong tiếng anh nhưng ít bao hàm nghĩa cho cả đoạn văn, các từ thường hay bổ trợ cho các từ.
- Phép toán BitWise: là thực hiên phép toán AND, OR, NOT theo phép toán dịch Bit 0, 1 kết quả sẽ có giá trị là dãy Bit, phép toán BitWise chủ yếu thực hiện cho mô hình Boolean
- Token: là tập các từ trong văn bản chưa được normalize
- Term: là tập các từ trong văn bản đã được normalize
- Thuật ngữ Term: mỗi tài liệu được biểu điễn một cách lo-gic như một tập hợp các thuật ngữ (term), một tập tài liệu tương ứng với tập hợp các term

II. Search Engine[1] – công cụ tìm kiếm

- Ra đời vào những nằm 1993, Search Engine đã trở thành công cụ trong thế giới sô, và không ngừng được cải thiện về chất lượng và chất lượng, được biết đến là những công cụ tìm kiếm dựa trên những nội dung mà người dùng nhập vào để có thể trả về nội dung, hình ảnh, video, văn bản có liên quan đến những thông tin truy vấn
- Các kết quả trả về được sắp xếp theo thứ tự nhất định bằng các thuật toán của search engine, mối search engine cho lĩnh vực hay nội dung khác nhau thì sẽ có những thuật toán khác nhau

1. Cấu tạo của Search Engine

- Crawler[2]: được biết đến với tên gọi là Spider hay Bot là công cụ giúp Search Engine thu thập dữ liệu của một trang web bất kì, bao gồm thông tin nội dung văn bản, hình ảnh, video về trang web
- Công cụ lập chỉ mục – index[3]: là công cụ thực hiện quá trình lưu trử nội dung tối ưu toàn bộ dữ liệu đã thu thập được, để có thể lưu trử dưới dạng dung lượng thấp nhưng truy cập nhanh, dữ liệu sẽ được mã hóa và lưu lại trong cơ sở dữ liệu gốc, giúp Search Engine không cần phải tìm nội dung trang web mà có thể dựa vào quá trình Index để giảm tối thiểu thời gian truy xuất và trả về kết quả nhanh nhất cho người dùng
- Công cụ trích xuất kết quả tìm kiếm: khi có một truy vấn của người dùng hệ thống sẽ tìm các tập văn bản thông qua Index để tìm được tất cả các kết quả công cụ truy xuất kết quả giúp lọc truy vấn nhằm giúp công cụ tìm kiếm đánh giá rỏ hơn về thông tin tìm kiếm và thông tin của người dùng. Từ đó tìm được kết quả sao cho phù hợp nhất đối với truy vấn

2. Mô hình IR – information retreival

a. Mô hình Boolean[1]

Mô hình biểu diễn vector với hàm f cho ra giá trị rời rạc và duy nhất hai giá trị đúng và sai (true và false, hoặc 0 và 1) gọi là mô hình boolean.

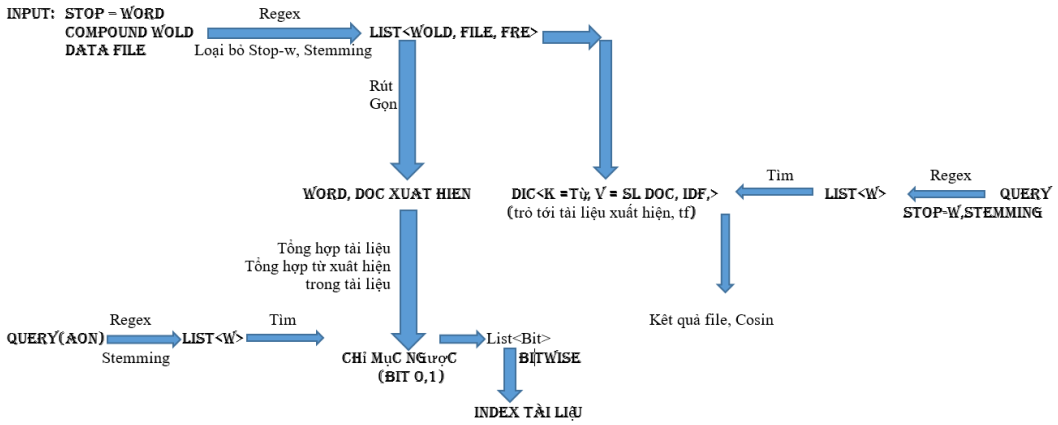
Mô hình Boolean được xác định như sau: giả sử có một cơ sở dữ liệu gồm m văn bản, D = $\{d_1, d_2, d_3 ....\}$ mỗi văn bản được biểu diện dưới dạng một vector gồm n từ khóa T =$\{t_1, t_2, t_n\}$. Gọi $W = \{w_{ij}\}$ là ma trận trọng số, trong đó w_{ij} là giá trị trọng số của từ khóa t_i trong văn bản d_i

b. Mô hình Vector Space[4]

Vector Space Là mô hình đại số thể hiện thông tin văn bản như một vector, các phần tử của vector này thể hiện mức độ quan trọng của một từ và cả sự xuất hiện hay không xuất hiện của nó trong một tài liệu

Mô hình này biểu diễn văn bản như những điểm không gian Eclid n – chiều, mỗi chiều tương ứng với một từ trong tập hợp các từ. Phần tử thứ i trong vector cho biết số lần từ thứ I xuất hiện trong văn bản đó. Sự tương đồng hai văn bản được định nghĩa là khoảng cách giữa các điểm, hoặc là góc giữa các vector trong không gian

c. Mô hình chương trình



Hình 1: Mô hình chương trình

III. Crawler – Spider – Bot

1. Khái niệm

- Crawler, Spider hay Bot là thuật ngữ chỉ đến các phần mềm hay công cụ dùng để thu thập dữ liệu cho các công cụ tìm kiếm, phần mềm này có thể thông nhập vào website một cách có hệ thống, nhằm thu thập thông tin
- Giống như một virus nhằm đánh cắp dữ liệu từ các trang web từ những nội dung hiển thị trên nền web.

2. Hoạt động của Crawler

- Web Crawler dùng để khám phá và tìm hiểu thông tin trên các trang website công khai hiện nay có trên trên mạng WWW. Các công cụ thu thập thông tin này sẽ lần lượt truy cập vào các trang web và dò theo từng liên kết trên các trang đó, giống như việc chúng ta duyệt từng nội dung trên trang. Bằng việc lần lượt đi từ liên kết này tới liên kết khác, chúng thu thập dữ liệu trên các trang và đem các dữ liệu đó về cho máy chủ Search Engine.
- Web Crawler cũng đồng thời xác định được những trang web nào cần thu thập thông tin, tần suất cũng như số lượng trang cần tìm nạp từ mỗi trang web. Chúng hoạt động tự động và ít chịu sự can thiệp bởi con người. Sau khi thu thập đầy đủ dữ liệu của trang, các Crawler sẽ tổng hợp lại những dữ liệu đó với những dữ liệu ngoài trang như số lượng backlink trỏ đến website, lượng visits,… và gửi chúng về ngân hàng dữ liệu để tiến hành xét duyệt trước khi bắt đầu được index.

IV. Indexing - truy xuất kết quả tìm kiếm

1. Lập chỉ mục – chỉ mục ngược[5]

- Để lập chỉ mục cho một dataset, mỗi tập tài liệu đều trải qua một quá trình theo trình tự:
    - Tách các từ theo khoảng trắng, dấu câu. Việc tách từ văn phạm phải duyệt hết văn phạm, mỗi từ phân biệt mới nhau bằng khoảng trắng hoặc dấu câu.
    - Quá trính loại bỏ stopword trong văn phạm: stopword là tập từ vựng gồm những từ vựng được thu thập có điểm chung là thường xuất hiện trong các tập tài liệu, nhưng dùng để bổ trợ hoặc từ nối giữa các từ trong văn phạm, việc loại bỏ stopword giúp loại bỏ một số văn phạm trả về không cần thiết trong quá trình truy vấn. Loại bỏ stopword chỉ cần so sánh văn phạm với tập stopword, nếu văn phạm có từ chứa stopword chỉ cần loại bỏ khỏi văn phạm.
    - Sau khi loại bỏ stopword ta được tập tài liệu chỉ chứa những từ vựng mô tả cho tài liêu, nhưng lại có những từ giống nhau nhưng khác về tráng thái ví như: go - goes, study – studying – studied …, để giảm trường hợp các từ này sử dụng một tập luật nhằm loại bỏ các End word cuối mỗi từ, việc này thực hiện theo mô hình Heuristic sử dụng tập luật có sẵn để loại bỏ. Quá trình này còn gọi stemming
    - Sau khi thực hiện các bước Regex, loại bỏ stopword, stemming ta thu được tập văn phạm chuẩn hóa và thực hiện quá trình lập chỉ mục cho văn phạm
- Lập chỉ mục cho tài liệu phương pháp thực hiện rút trích các từ, cụm từ (term) trên các file văn bản cũng như các thông tin đi kèm với mỗi term là tần số, độ quan trong của term trong tài liệu các thông tin này sẽ được tổ chức lưu trử riêng gọi là chỉ mục. Lúc nào thao tác tìm kiếm sẽ được tiến hành dựa trên chi mục thay vì quá trình tìm kiếm trực tiếp trên tập dữ liệu. Một giá trị của chỉ mục thường có thông tin mô tả về từ như: <từ khóa – Chỉ số tài liệu – Tần số của từ khóa trong tài liệu>, Thay vì lưu tập tài liệu bằng văn phạm, thì chỉ mục ngược giúp tối ưu hóa văn phạm thành danh sách, giúp cho quá trình tìm kiêm và so khớp nhanh hơn

- Chỉ mục được lập bởi các tài liệu ứng với những từ đó:

	Từ Khóa	Chỉ số tài liệu	Tần số
▶	acrothermoelast	12	1
	addit	8	1
	aerelast	12	1
	aerodynam	1	1
	aerodynam	5	1
	aerodynam	11	1
	aeroelast	12	1

Hình 2: Bảng lập chỉ mục

- Các thuật ngữ được tách trong các file văn bản dựa vào khoảng trắng giữa các từ, ngoài ra có những từ ghép bởi hai hoặc nhiều từ nhưng có một nghĩa, sẽ được định nghĩ tạo thành tập hợp danh sách các từ vựng, dựa vào đó có thể tách các từ ngữ theo nghĩa cố định không làm mất đi hay thay đổi nội dung của văn bản như những từ: bottled water, cell phone, civil right, crossword puzzle….
- Ngoài ra cũng có thể biểu diễn chỉ mục dưới dạng mã nhị phân theo cấu trúc:

	t_1	t_2	t_3	t_4			t_m
d_1	1	1	0	0			1
…	0	0	0	1			0
d_n	1	0	0	0			0

Hình 3: Chỉ mục theo Matrix

- Quá trình lập chỉ mục nhằm tối ưu lưu trử nội dung văn bản ngoài ra còn đẩy tốc độ của chương trình và phục vụ quá trình truy vấn một cách nhân nhất

2. **Chỉ mục ngược – Ma trận chỉ mục term – Document**[6]

- Một tập n văn bản được biểu diễn bởi m term được tổng hợp từ n văn bản sẽ được vector hóa tạo thành 1 danh sách vector nhằm mô tả từ có hoặc không trong văn bản. trong đó n văn bản được biểu diễn cho đại diện giá trị của mỗi cột, và mỗi hàng sẽ được biểu diễn bởi m term, Phần tử d_{ij} là giá trị của từ đó trong tài liệu. đó gọi là ma trận đảo ngược hay chỉ mục ngược

t_1	d_1	d_3	d_{51}	d_{151}	d_{2011}		
t_2	d_2	d_{10}	d_{61}				
...							
t_m	d_{100}	d_{1001}	d_{3000}	d_{3001}	d_{5001}		

Hình 4: Ma trận chỉ mục ngược

- Dựa vào số lần xuất hiện trong tài liệu, ta có thể tính ra được tần số từ trong tài liệu đó gọi là tf (term frequency), giá trị tf_i ứng với tần số của từ I trong tài liệu đó
- Tần số nghịch đảo idf (inverse document frequency), tần sô nghịch của một từ trong văn bản, idf để giảm giá trị của những từ phổ biến trong tập các văn bản, mỗi từ chỉ có một giá trị duy nhất trong tập các văn bản. Giá trị càng lớn thì thuật ngữ càng không quan trọng trong tập tài liệu đang xét

3. **Từ Vựng và Posting List**[6]

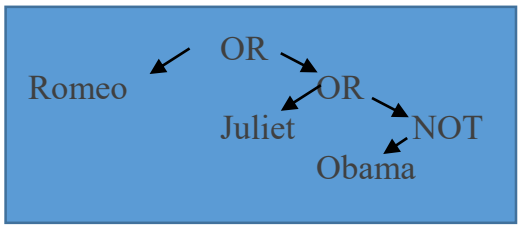
- Từ vựng: là tập thu thập dựa trên bản Chỉ mục nhằm giảm độ lưu trử, trùng lắp từ trong chỉ mục, mỗi phần tử trong tập từ vựng gồm các giá trị < Từ - Tổng số tài Liệu xuất hiện – Tổng tần số từ - IDF – pointer >, mỗi giá trị pointer trỏ tới một tập dữ liệu mô tả cho từ đó, mỗi tập dữ liệu gọi là posting gồm các giá trị <Tài Liệu xuất hiện – tần số từ trong tài liệu đó – TF >
- IDF: thể hiện cho sự quan trọng của từ trong tập dataset, được tính bằng tần số khả nghịch của tài liệu xuất hiện từ trên cho tổng tài liệu, giá trị IDF càng thấp thì độ tin cậy của từ đó càng thấp, nếu không loại bỏ các stopword thì đa số từ được xem là stopword sẽ có giá trị bằng 0, bởi vị stopword đa số xuất hiện trong mỗi tập văn bản. việc loại bỏ stopword sẽ giảm được quá trình tính toán. Để giảm độ lớn của IDF người ta thường sử dụng log để giảm kích thước IDF nhưng vẫn thể hiện được độ quan trọng của từ đối với tập văn phạm
- TF: là tần số của từ trong mỗi tập tài liệu, được tính bằng thương số của số lần xuất hiện của từ trên cho tổng số từ trong văn bản, TF thể hiện độ quan trong của từ trong văn bản, giá trị TF càng cao thì mức độ quan trong đối với tập văn bản càng lớn

Từ Khóa	Chỉ số tài liệu	Tần số	TF	IDF	TF * IDF
a	4	4		1	
	1	1	0.007142857		0.007142857182...
	2	1	0.005076142		0.005076142027...
	3	1	0.04		0.039999999105...
	4	1	0.01298701		0.012987012974...
aerodynam	1	1		2.386294361119...	
	1	1	0.007142857		0.017044959817...
after	1	1		2.386294361119...	
	1	1	0.007142857		0.017044959817...
again	1	1		2.386294361119...	
	2	1	0.005076142		0.012113169097...
agre	1	1		2.386294361119...	
	1	1	0.007142857		0.017044959817...
also	1	1		2.386294361119...	
	4	1	0.01298701		0.030990835829...
an	2	2		1.693147180559...	
	1	1	0.007142857		0.012093908500...
	2	1	0.005076142		0.008594655562...
and	3	3		1.287682072451...	
	1	1	0.007142857		0.009197729140...
	2	1	0.005076142		0.006536457086...
	4	1	0.01298701		0.016723143782...
angl	1	1		2.386294361119...	

Hình 5: kết quả quá trình Xây dựng Từ vựng và Posting List

4. **Mô hình Boolean**[1]

- Xem mỗi tài liệu là một tập từ, mỗi query là tập các từ có giá từ nối với nhau bằng các ký hiệu AND, OR, NOT.
- Truy vấn giữa tài liệu và từ là quá trình kiểm tra từ trong query thuộc tài liệu nào và sử dụng phép toán bitwise để thực hiện. Có nhiều các để tính giá trị query:
    - Sử dụng thuật toán De – Morgan để giải quyết câu truy vấn
      Ví dụ: Romeo OR Juliet OR NOT Obama

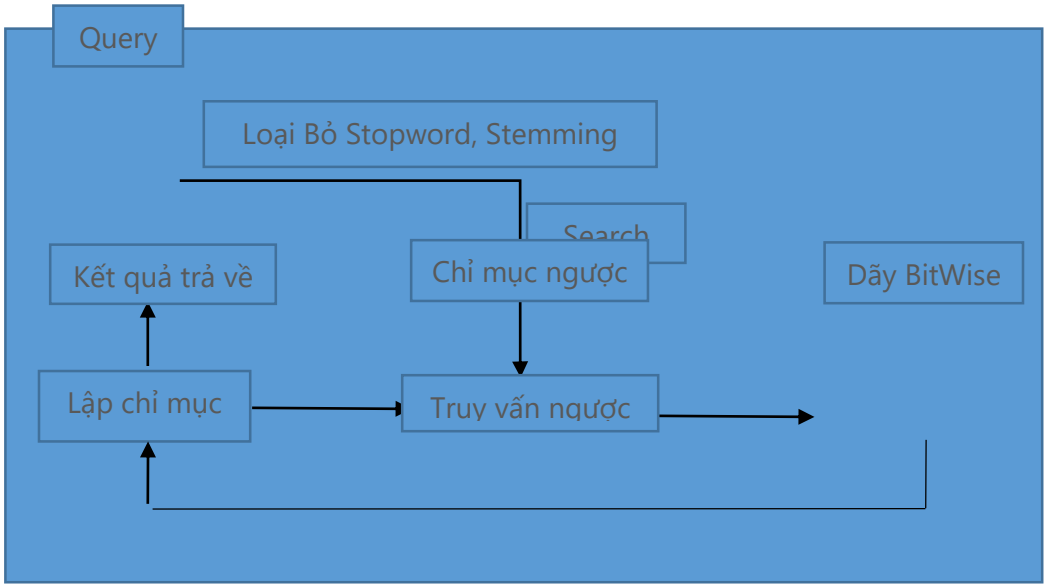


Hình 6: Mô phỏng luật theo mô hình Boolean

Mỗi Node trong cây sẽ là phép toán, các node lá sẽ làm term cần truy vấn, quá trình thực hiện từ dưới lên trên, mỗi Node khi thực hiện sẽ có giá trị là số tài liệu, với phép OR là phép lấy tài liệu có cả 2 term, còn phép AND là lấy tài liệu chung của 2 term.

- Sử dụng thuật toán theo BitWise: thây từ bằng giá tị 1 0, thành dãy cách index, vị trí index xuất hiện thì từ đó trong query xuất hiện trong tập index, ngược lại
- Ví dụ: D_1 có các từ: Romeo, Obama, Clinton
  D_2 có các từ: Juliet, Franky, Newton
  Với query: Romeo OR Juliet OR NOT Obama
  Thì sẽ biểu diễn Query với dạng mã nhị phân:
  Query: 10 OR 01 OR NOT 10 → 10 OR 01 OR 01 = 11
  Kết quả truy vấn trả về 2 tài liệu trên

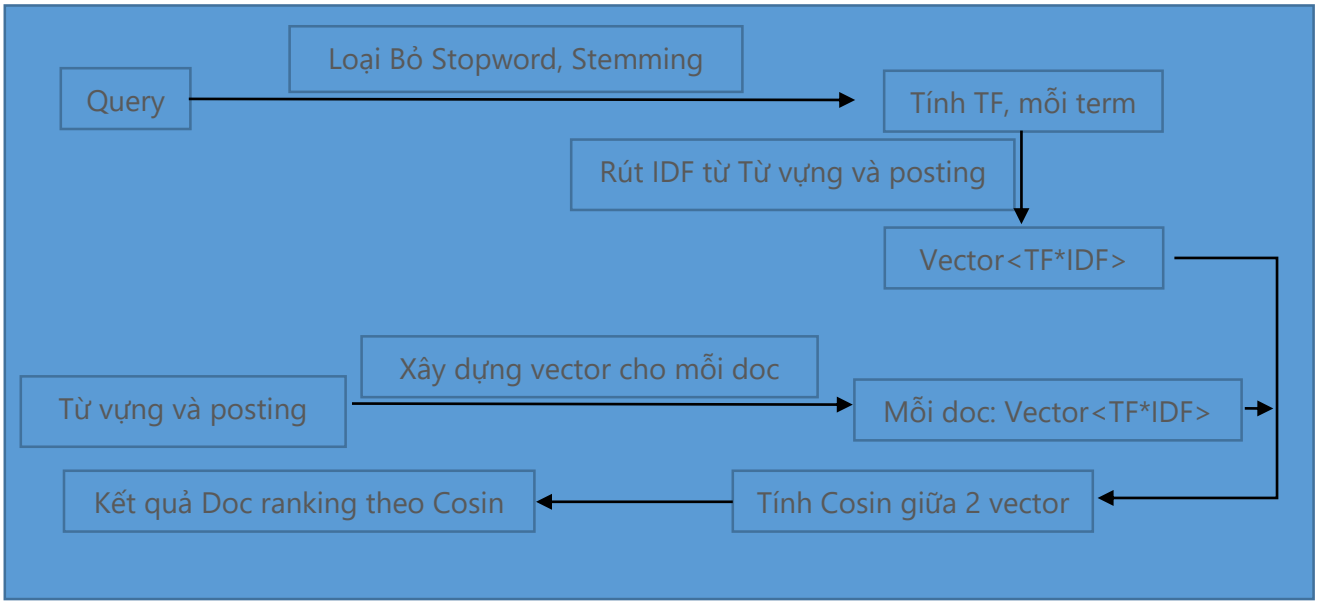
- Ưu nhược điểm mô hình Boolean:
    - Ưu điểm:
        - Đơn giản và dễ sử dụng.
    - Nhược điểm:
        - Chuyển câu truy vấn sang dạng boolean là không ñơn giản;
        - Văn bản trả về không quan tâm ñến thứ tự quan hệ với câu truy vấn



Hình 7: Mô hình Search với Boolean

5. **Mô hình truy vấn theo vector space model** [4]

- Vector space model xem mỗi tập tài liệu sẽ được biểu diễn dưới dạng vector, trong đó mỗi phần tử trong vector là giá trị TF*IDF của từ có trong tài liệu, độ dài của vector phụ thuộc vào số lượng từ vựng trong một tập tài liệu
- Query trong vector space cũng qua các bước xử lý tương tự và được xem như là một vector, giá trị là TF*IDF của mỗi từ trong query
- Sự tương đồng của query và văn phạm phụ thuộc vào giá trị của mỗi thành phần trong vector, có thể tính dựa trên nhiều công thức toán học như: cosin, euclid, L1…



Hình 8: Mô hình search với Vector Space

- Tính Cosin giữa hai vector: mỗi vector trong công thức là độ lớn của các từ giống nhau query so với document, các từ không giống nhau không được sử dụng

- Giả sử Query có vector của các từ Query =(A, B, G), và Document có vector của các từ Document = (A, C, D, H, G, Z, T) thì Cosin giữa Query và Document được tính:

$$Cosin(Q, D) = \frac{Q_A * D_A + Q_G * D_G}{\sqrt{Q_A^2 + Q_G^2}\sqrt{D_A^2 + D_G^2}}$$

- Sau quá tình tinh cho mỗi tài liệu sẽ sắp xếp kết quả theo chiều giảm dần của cosin, những tài liệu có giá trị càng lơn thì độ tương đồng càng cao và ngược lại
- Ưu nhược điểm mô hình Vector Space:
  - Ưu điểm
    - Các tài liệu trả lại có thể được sắp xếp theo mức độ liên quan đến nội dung yêu cầu do trong phép thử mỗi tài liệu đều trả lại chỉ số đánh giá độ liên quan của nó đến nội dung.
    - Việc đưa ra các câu hỏi tìm kiếm là dễ dàng và không yêu cầu người tìm kiếm có trình độ chuyên môn cao về vấn đề đó.
    - Tiến hành lưu trữ và tìm kiếm đơn giản hơn phương pháp Logic.
  - Nhược điểm
    - Việc tìm kiếm tiến hành chậm khi hệ thống các từ vựng là lớn do phải tính toán trên toàn bộ các vector của tài liệu.
    - Khi biểu diễn các vector với các hệ số là số tự nhiên sẽ làm tăng mức độ chính xác của việc tìm kiếm nhưng làm tốc độ tính toán giảm đi rất nhiều do các phép nhân vector phải tiến hành trên các số tự nhiên hoặc số thực, hơn nữa việc lưu trữ các vector sẽ tốn kém và phức tạp.
    - Hệ thống không linh hoạt khi lưu trữ các từ khóa. Chỉ cần một thay đổi rất nhỏ trong bảng từ vựng sẽ kéo theo hoặc là vector hoá lại toàn bộ các tài liệu lưu trữ, hoặc là sẽ bỏ qua các từ có nghĩa bổ sung trong các tài liệu được mã hóa trước đó.
    - Một nhược điểm nữa, chiều của mỗi Vector theo cách biểu diễn này là rất lớn, bởi vì chiều của nó được xác định bằng số lượng các từ

khác nhau trong tập hợp văn bản. Ví dụ số lượng các từ có thể có từ 10^3 đến 10^5 trong tập hợp các văn bản nhỏ, còn trong tập ợp các văn bản lớn thì số lượng sẽ nhiều hơn, đặc biệt trong môi trường Web.

V. Xây dựng ứng dụng search tài liệu

1. Tách từ và xây dựng tập Compound Word

- Được thu thập từ các trang báo điện tử tập hợp các từ được định nghĩa bởi hai hoặc ba từ khi ghép lại bằng khoảng trắng có nghĩa như: bottled water, cell phone, civil right, crossword puzzle, dump truck, French fry…
- Tập Compound Word giúp tăng khả năng hiểu về từ cho dataset và query, giúp tránh trường hợp hiểu sai nghĩa về mặt từ vựng, giúp tăng độ chính xác khi truy vấn
- Trong cấu trúc văn phạm, mỗi từ cách nhau bằng khoảng trắng để thể hiện sự ưu tiên các từ trong tập Compound word so sánh mỗi từ trong dữ liệu với tập từ điển có sẵn, sau đó nếu không còn từ nào sẽ tách theo khoảng trắng và dấu câu.
- Để tách từ sử dụng Regex với cú pháp:

```
Regex = @"((^|)(" + string.Join("|", wordCompound) + @"|[A-Za-z\-]+))+";
```

  - Mỗi từ trong compoundWord sẽ được ngăn cách bằng dấu | thể hiện phép hoặc
    - Ví Dụ: ((^|)(“bottled water| cell phone | civil right |[A-Za-z\-]+))+”
    - Phép toán này sẽ ưu tiên tách các từ: bottled water, cell phone, civil right sau đó tách các từ bằng khoảng trắng.
  - Với mỗi tập tài liệu sẽ xây dựng thành một danh sách các từ bằng cách áp dụng luật Regex cho tài liệu đó:

```
var valueEnumerable = Regex.Matches(content.ToLower(), regex);
```

    - Sau khi áp dụng được tập từ vựng ứng với mỗi tài liệu
    - Với content.ToLower() là tập nội dung của văn bản được chuẩn hóa về dạng kí tự thường.

2. Loại bỏ thành phần stopword

- Quá trình loại bỏ stopword là quá trình duyệt tất cả các ký tự trong tập dữ liệu đã Regex và so sánh xem có tồn tại trong stopword không, nếu có thì xóa khỏi tập từ vựng của văn bản

```
ListWorld = valueEnumerable.Cast<Match>().Select(match => match.Value).
ToList().Except(StopWords).OrderBy(a => a).ToList();
```

3. **Lập chỉ mục**[6]

- Quá trình lập chỉ mục tổng hợp các từ vựng có trong dataset đã được normalize thành các term, gồm có ba thành phần: từ vựng, DocID và tần số.

```
var vobu = ( from variable in _listpPages from iteam in variable.ListWorld
            select new Tuple<string, document, int>(iteam, variable,
            1)).ToList();
vobu = SortVoBu(vobu.ToList());
```

  - Với _listPages là tập hợp các document – mỗi document đã thực hiện qa normalize thành các term 1 document được mô phỏng gồm các thành phần

```
public string Title;
public string Content;
public IEnumerable<string> ListWorld;
```

    - Với title là DocID
    - Content là nội dung document
    - ListWorld là tập hợp các từ đã được stemming
  - Mỗi Tuple sẽ gồm giá trị là từ vựng, docId và tần số mặt định, sau khi thực hiện sẽ gom lại theo từ giống nhau và docId giống nhau tầng số sẽ tăng lên
- Kết quả của quá trịnh lập chỉ mục là thu được danh sách các DocID theo từ và tầng số nhằm phục vụ cho quá trình lập tập từ vựng và posting List

4. Xây Dựng Từ Vựng và Posting List[6]

- Để xây dựng tập từ vựng sử dụng cấu trúc Dictionary với Key là từ vựng values gồm các tần số, tài liệu, IDF và danh sách posting tương ứng với tài liệu

```
public  void CreateBoardDictionary()
    {
        var tup = new Dictionary<string, Tuple<int, int, double, List<Tuple<string,
int,float,double>>>>();
        for (var i = 0; i < Vobu.Count; i++)
        {
            var lst = new List<Tuple<string, int,float>>();
            lst = (from p in Vobu where p.Item1 == Vobu[i].Item1 select
                (new Tuple<string, int, float>(p.Item2.Title,
p.Item3,float.Parse(p.Item3.ToString())/float.Parse(p.Item2.count.ToString()))))
                .ToList();
            var count = lst.Sum(p => p.Item2);
            var list = new List<Tuple<string, int, float, double>>();
            // tinh idf = log_10(tong so page/ Tong page chua tu)
            var idf =1.0+ Math.Log((float)_listpPages.Count / (float)count);
            foreach (var item in lst)
            {
                var value = new Tuple<string, int, float,
double>(item.Item1,item.Item2,item.Item3,idf * item.Item3);
                list.Add(value);
            }
            var tupsub = new Tuple<int, int,double, List<Tuple<string, int,
float,double>>>(lst.Count, count,idf, list);
            _dictionList.Add(Vobu[i].Item1, tupsub);
            i = i + tupsub.Item4.Count - 1;
        }
    }
```

- Ứng với mỗi từ trong tập Từ vựng đều phải tính IDF để giúp cho việc đánh giá và tìm kiếm theo mô hình vector space. IDF tính dựa trên tổng số pages và số lượng pages chứa từ IDF

```
var idf =1.0+ Math.Log((float)_listpPages.Count / (float)count);
```

  - Với _listpPages.Count là số lượng pages, count là số lượng pages chứa từ cần tính IDF

5. Truy vấn theo mô hình Boolean[1]

- Truy xuất theo mô hình boolean sử dung phép toán BitWise thực hiện các câu truy vấn bằng cách chuyển đổi query dưới dạng ký tự sang những mã Bit việc xử lý này có khuyết điểm độ lớn của phép toán tương đương với độ dài câu truy vấn và tập dataset.
- Để chuyển đối cấu query thành mã Bit, phải so sánh câu truy vấn với tập index, giá trị từ trong câu truy vấn bằng 1 khi tập tài liệu chứa từ đó và ngược lại
    - Ví dụ: với phép truy vấn: *Brutus AND Caesar AND NOT Calpurnia*

	Antony and Cleopatra	Julius Caesar	The Tempest	Hamlet	Othello	Macbeth
Antony	1	1	0	0	0	1
Brutus	1	1	0	1	0	0
Caesar	1	1	0	1	1	1
Calpurnia	0	1	0	0	0	0
Cleopatra	1	0	0	0	0	0
mercy	1	0	1	1	1	1
worser	1	0	1	1	1	0

Hình 9: Ví dụ cho truy vấn với Boolean

        - Mã Bit được chuyển qua mã Bit:

          110100 AND 110111 AND NOT 010000
        - Kết quả thực hiện: 100100 tương đương với tập tin: Antony And Cleopatra và Hamlet
- Thứ tự thực hiện trong phép toán theo cấp phép toán:
    - Thực hiện các phép toán trong ngoặc trước
    - Thực hiện phép toán NOT
    - Thực hiện phép toán AND OR theo thứ tự trừ trái sang phải

6. Truy vấn theo mô hình Vector Space[4]

- Tính TF*IDF cửa vector query:

```
        foreach (var item in lst)
        {
            if (!dic.ContainsKey(item.Item1)) continue;
            Item = dic.First(a => a.Key == item.Item1).Value;
            lstQuery.Add(new Tuple<string,double>(item.Item1, item.Item2*
dic[item.Item1].Item3));
            listItem.Add(item.Item1, Item);
        }
```

  - Mỗi từ trong query sẽ được kiểm tra trong dic (dictionary, nếu tồn tại từ đó thì giá trị TF * IDF bằng tích tầng số từ và IDF trong dictionary)

- Tính độ tương đồng giữa query và tài liễu theo cosin

```
        foreach (var item in dicFile)
        {
            var doDaiTaiLieu = Math.Sqrt(item.Value.Sum(a => Math.Pow(a.Item2, 2)));
            var file = item.Value.ToList();
            var d = 0.0;
            var x = item.Value.Sum(subitem => lstQuery.FirstOrDefault(a => a.Item1 ==
subitem.Item1).Item2 * subitem.Item2);
            var doDaiQuery = Math.Sqrt(lstQuery.Sum(e =>Math.Pow(e.Item2,2)));
            var c = (float)((float)x / ((float)doDaiTaiLieu * (float)doDaiQuery));
            docTfidf.Add(item.Key,(double)c);
        }
```

7. Xây dựng crawler[2]

- Xây dựng các poster truy xuất thông tin trang web BBC nhằm rút trích các thông tin nội dung bài báo phục vụ cho quá trình truy vấn
- Các trang web BBC có nội dung theo div với class = “story-body__inner” hoặc với id = “story-page”
- Để rút trích nội dung trang web truy cập tới các div với class và id và lưu lại tập dữ liệu để sử dụng cho quá trình truy vấn
- Rút trích 1 div nội dung trong trang web và loại bỏ các kí tự không liên quan thay các kí tự bằng khoảng trắng

```
for htag in soup.find_all("div", {"class": "story-body__inner"}):
    for atag in htag.find_all('p'):
        acontent = atag.text.replace("."," . ").replace(","," , ").replace("?", " ?
").replace("\'s","").replace("u'","").replace("\n" ," ")
        f = f+ acontent
```

VI. Đánh giá kết quả tìm kiếm

1. Tại sao phải đánh giá:

- Nhằm tìm kiếm và đánh giá các hệ thống search engine tốt và phù hợp các kết quả trả về, thông thường các kết quả trả về của hệ thống sẽ so sánh với tập ground truth – tập dữ liệu ứng với mỗi query mà người dùng sẽ cho nội dung nào phù hợp với query. Để đánh giá hệ thống sẽ đánh giá thông qua nhiều tập query và sao sánh các kết quả trả về. kết quả trả về có các tập tin

2. Đánh giá dữ liệu tìm kiếm:

- Có nhiều cách đánh giá hệ thống như sử dụng F-measure[7], TREC[8], R-precision[9]… mỗi cách đánh giá sẽ cho ra kết quả khác nhau:
    - Với F-measure: sẽ đánh giá rựa trên precision và recall

$$\boldsymbol{F = \frac{2PR}{P + R}}$$

        - Với $P = \frac{Tập\ tin\ đúng\ trả\ về}{tất\ cả\ tập\ tin\ trả\ về}$, $R = \frac{Tập\ tin\ đúng\ trả\ về}{tập\ ground\ truth}$
        - Giá trị F càng cao hệ thống càng chính xác, F phụ thuộc P và R với P càng lớn tức là kết quả trả về gần xác với kết quả đúng, R càng lớn tức là giá trị trả về càng gần đúng với grouth truth
    - Với TREC: đánh giá dự trên Recall tại 11 điểm bắc đầu từ: 0 đên 1. Và trung bình của các giá trị chính xác tại 11 điểm.

$$\boldsymbol{P = \frac{1}{11}\sum_{i=1}^{11} P_i}$$

        - Với P_i là giá trị của recall tại R_i
    - Với R-precision: xác định dựa trên từ 0 đến số kết quả ground-truth

$$\boldsymbol{R - Precision = \frac{Số\ kết\ quả\ đúng\ lấy\ số\ tập\ tin\ bằng\ số\ ground\ truth}{Số\ kết\ quả\ của\ ground\ truth}}$$

        - Kết quả của R-Precision chủ yếu đánh giá kết quả xếp hạng và độ tập trung của chương trình

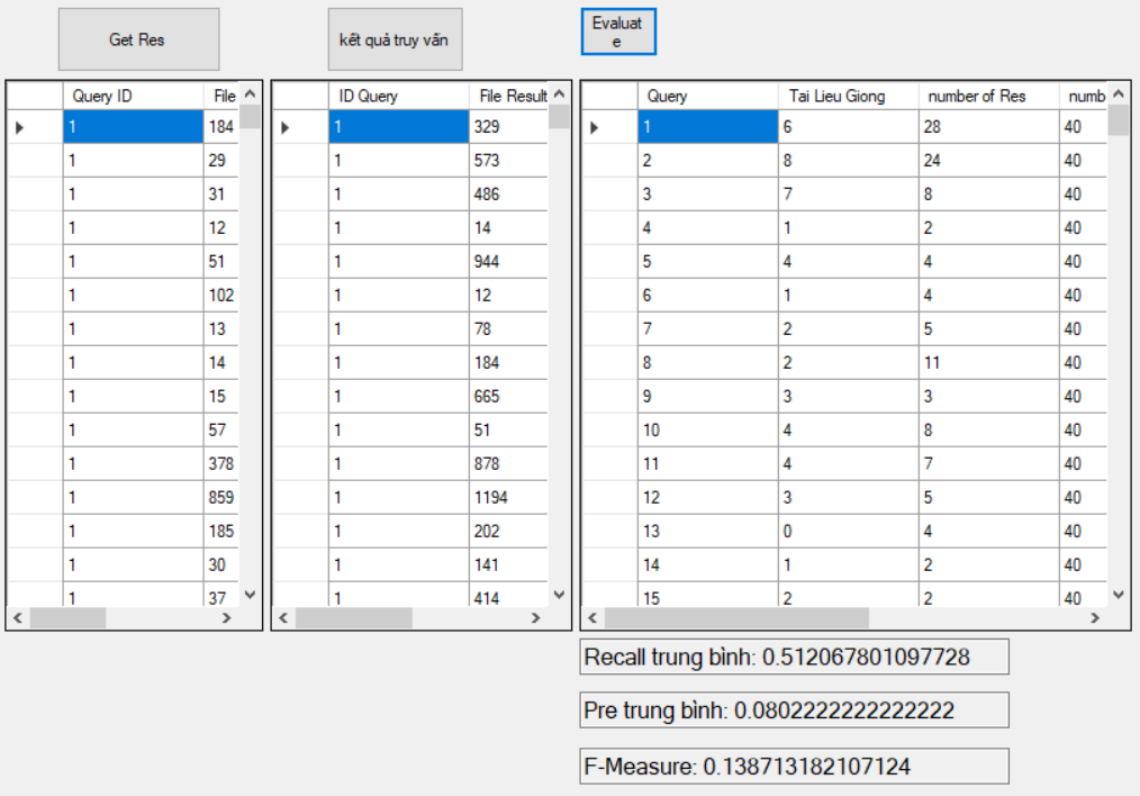
3. Đánh giá hệ thống chương trình:

a. Mô hình Boolean

- Kết qủa truy vấn theo mô hình Boolean: chưa đánh giá, Chỉ truy vấn với câu truy vấn AND OR NOT, không truy vấn với các query dùng ngôn ngữ tự nhiên, đánh giá hệ thống Boolean cũng giống như vector space so sánh với grouth truth, 2 model cùng dùng chung 1 hệ thống đánh giá.

b. Mô hình Vector Space

- Kết quả mô hình xét trên tập Cranfield trả về 40 kết quả xếp hạng cao nhất



Hình 10: kết quả đánh giá chương trình

VII. Tài liệu tham khảo

[1] W. Quick, R. Guide, and U. Terms, “Searching with Boolean Terms and Connectors,” no. Figure 1, pp. 1–4.

[2] C. Olston and M. Najork, “Web Crawling,” *Found. Trends® Inf. Retr.*, vol. 4, no. 3, pp. 175–246, 2010.

[3] J. S. Beis and D. G. Lowe, “Shape Indexing Using Approximate Nearest-Neighbour Search in High-Dimensional Spaces,” in *Proc. {CVPR}*, 1997.

[4] V. Space, S. Engine, and C. Coordinates, “Basic Vector Space Search Engine Theory,” pp. 1–6, 2004.

[5] K. G. Derpanis, “Mean Shift Clustering,” *Computer (Long. Beach. Calif).*, vol. 1, no. x, pp. 1–3, 2005.

[6] C. D. Manning, P. Raghavan, and H. Schütze, “Inverted Indexing for Text Retrieval,” *Introd. to Inf. Retr.*, 2008.

[7] Y. Sasaki, “The truth of the F-measure,” *Teach Tutor mater*, pp. 1–5, 2007.

[8] E. Yilmaz, E. Kanoulas, M. Verma, B. Carterette, N. Craswell, and R. Mehrotra, “Overview of the TREC 2015 Tasks Track,” pp. 1–7, 2015.

[9] S. Shalev-Shwartz, Y. Singer, N. Srebro, and A. Cotter, “Pegasos: {P}rimal {E}stimated sub-{G}r{A}dient {SO}lver for {SVM},” *MBP*, 2010.