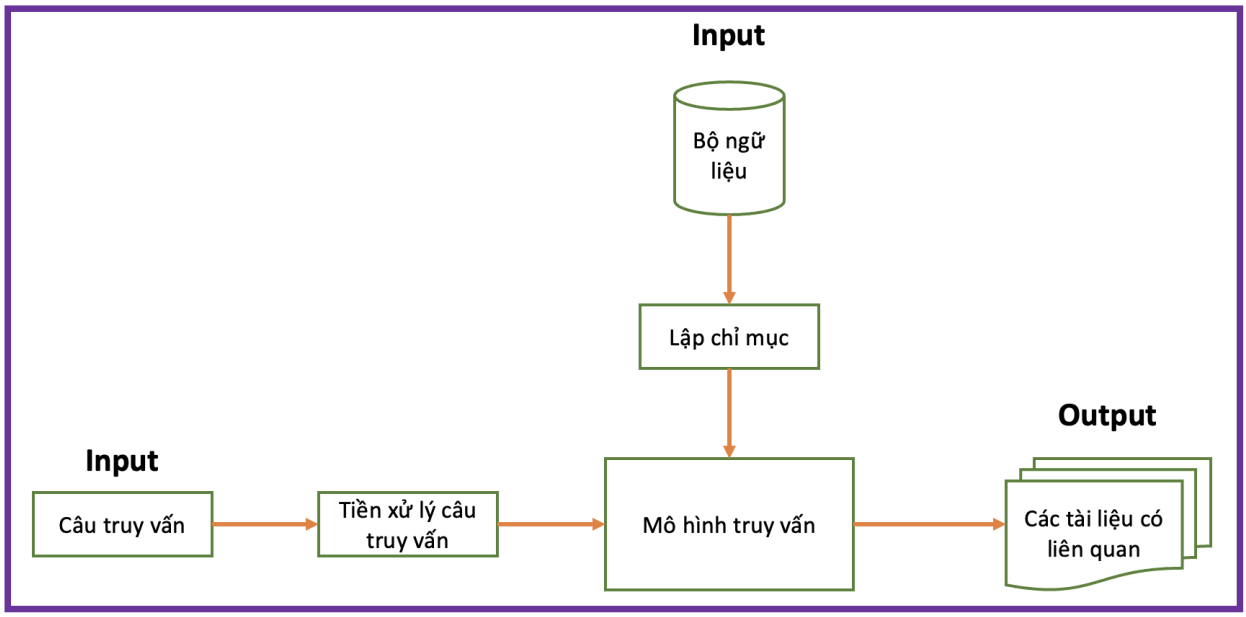
AI VIET NAM – COURSE 2022

Module 2 Project - Text Retrieval

Ngày 25 tháng 8 năm 2022

Truy vấn văn bản (Text Retrieval) là một bài toán thuộc lĩnh vực Truy vấn thông tin (Information Retrieval). Trong đó, nhiệm vụ của ta là xây dựng một chương trình trả về các tài liệu (Document) có liên quan đến câu truy vấn (Query) đầu vào và các tài liệu được lấy từ một bộ ngữ liệu (Corpus) cho trước, trong lĩnh vực xử lý ngôn ngữ tự nhiên, bộ ngữ liệu có thể hiểu là một bộ dữ liệu văn bản hay tập các tài liệu văn bản. Có rất nhiều cách thiết kế hệ thống truy vấn văn bản khác nhau, tuy nhiên về mặt tổng quát sẽ có một pipeline chung sau đây:



Hình 1: Pipeline tổng quan của một hệ thống Text Retrieval.

Dựa vào hình trên, có thể phát biểu Input/Output của một hệ thống truy vấn văn bản bao gồm:

- **Input:** Câu truy vấn $\mathbf{q}$ và bộ ngữ liệu $\mathbf{C}$.
- **Output:** Danh sách các tài liệu $\mathbf{c}$ ($\mathbf{c} \in \mathbf{C}$) có nội dung liên quan đến câu truy vấn.

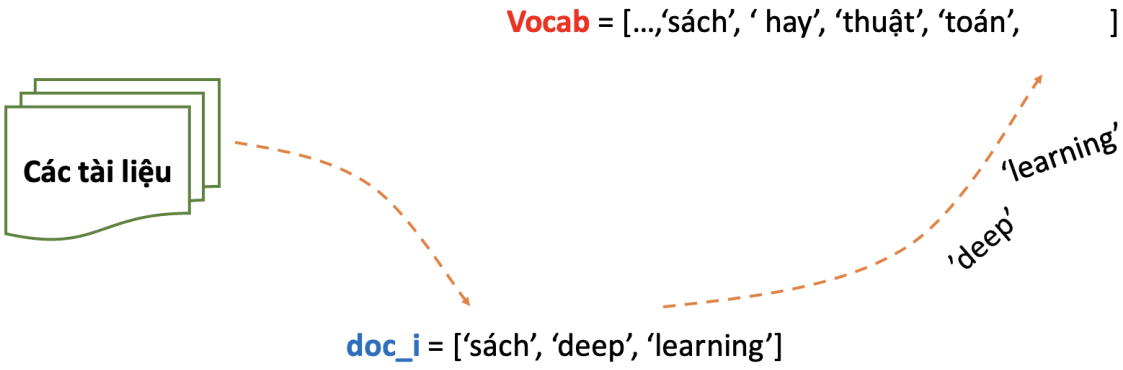
Hình 2: Demo về hệ thống truy vấn văn bản

Trong project này, chúng ta sẽ xây dựng một chương trình cho phép truy vấn văn bản báo chí Việt Nam sử dụng mô hình không gian vector (Vector Space Model), các bước xây dựng được mô tả như sau:

1. **Xây dựng hàm chuẩn hóa văn bản:** Với hầu hết các bài toán liên quan đến văn bản, việc chuẩn hóa văn bản là vô cùng quan trọng bởi nó giúp ta phần nào giảm được độ phức tạp trong việc biểu diễn văn bản. Trong project này, các bạn sẽ xây dựng một hàm **normalize_text()** với tham số đầu vào là một chuỗi **s**, sau đó trả về một chuỗi đã được chuẩn hóa. Các kỹ thuật chuẩn hóa văn bản các bạn có thể tham khảo ở Assignment 06 - Module 1 hoặc lần lượt áp dụng ba kỹ thuật bao gồm:
    - **Chuyển chữ viết thường (Lowercasing)**
    - **Xóa dấu câu (Punctuations Removal)**
    - **Xóa stopwords (Stopwords Removal)**

Hình 3: Input/Output của hàm chuẩn hóa văn bản

2. **Xây dựng bộ từ vựng (vocab):** Từ bộ ngữ liệu cho trước, ta duyệt qua nội dung của từng tài liệu, thực hiện chuẩn hóa văn bản và sử dụng kỹ thuật Tokenization để tách thành các token. Với mỗi token nhận được, kiểm tra nếu token hiện tại không tồn tại trong bộ từ vựng thì ta sẽ thêm nó vào.



Hình 4: Ảnh minh họa quá trình xây dựng bộ từ vựng.

3. **Xây dựng hàm vector hóa văn bản:** Vì máy tính không thể sử dụng chuỗi văn bản để thực hiện tính toán, ta cần biểu diễn lại các văn bản dưới dạng vector. Trong project này, ta xây dựng một hàm **vectorize()** với tham số đầu vào là một chuỗi văn bản **s** và bộ từ vựng **vocab**, sau đó thực hiện vector hóa văn bản sử dụng kỹ thuật bag-of-words (các bạn có thể tham khảo lý thuyết về bag-of-words tại đây) dựa trên bộ từ vựng cho trước, kết quả trả về sẽ là vector bag-of-words của văn bản đầu vào.

Hình 5: Input/Output của hàm vector hóa văn bản. Hàm này sẽ thực hiện chuẩn hóa văn bản sau đó tính bag-of-words với bộ từ vựng có sẵn.

4. **Xây dựng ma trận document-term:** Với các tài liệu trong bộ ngữ liệu, ta cần lưu trữ dạng biểu diễn vector của chúng trong một cấu trúc được gọi là ma trận document-term. Trong ma trận này, mỗi hàng sẽ đại diện cho một tài liệu và mỗi cột sẽ đại diện cho mỗi từ (term) trong bộ từ vựng. Như vậy, giả sử với một bộ ngữ liệu gồm có ba tài liệu văn bản có nội dung như sau:

   - **doc1** = "Học sách học AI!"
   - **doc2** = "Sách Học Máy."
   - **doc3** = "Người ấy là ai?"

   Sau khi thực hiện chuẩn hóa văn bản và tokenization, ta thu thập các token độc nhất trong toàn bộ bộ ngữ liệu để lập thành bộ từ vựng gồm ['học', 'sách', 'ai', 'máy', 'người', 'ấy', 'là']. Dựa vào danh sách từ vựng có được, tính tần suất xuất hiện của mỗi từ trong bộ từ vựng với từng

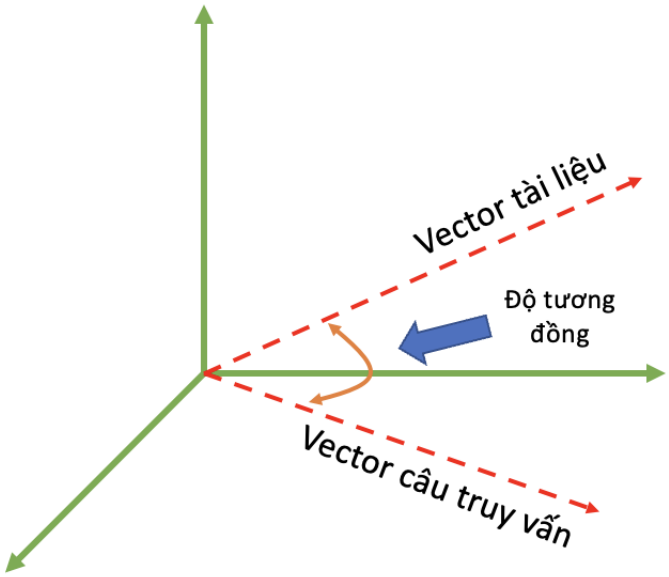
văn bản đã được tokenize để từ đó có được dạng vector biểu diễn của chúng. Trong ma trận document-term, ba vector này chính là 3 hàng của ma trận tượng trưng cho 3 tài liệu và các giá trị trong vector sẽ tượng trưng cho các cột đại diện cho các từ trong bộ từ vựng.

doc1 = ['học', 'sách', 'học', 'ai'] →
doc2 = ['sách', 'học', 'máy'] →
doc3 = ['người', 'ấy', 'là', 'ai'] →

	học	sách	ai	máy	người	ấy	là
doc1	2	1	1	0	0	0	0
doc2	1	1	0	1	0	0	0
doc3	0	0	1	0	1	1	1

Hình 6: Ma trận document-term. Mỗi hàng đại diện cho chỉ mục mỗi tài liệu và mỗi cột đại diện cho các term trong bộ từng vựng.

5. **Xây dựng hàm tính độ tương đồng giữa hai vector:** Để có thể tìm được các tài liệu có liên quan đến câu truy vấn, ta có thể sử dụng các công thức được dùng để đo sự tương đồng giữa hai vector (ở đây sẽ hiểu là vector tài liệu và vector câu truy vấn), từ đó xây dựng một hàm **distance()** nhận đầu vào là hai vector có cùng kích thước, sau đó trả về một giá trị là điểm đại diện cho độ tương đồng của vector này.



Hình 7: Độ tương đồng giữa hai vector câu truy vấn và tài liệu theo phép cosine similarity.

6. **Thực hiện truy vấn:** Cuối cùng, dựa trên các thành phần đã xây dựng ở các bước trên, ta sẽ bắt đầu thực hiện tìm kiếm các tài liệu có liên quan đến một chuỗi văn bản câu truy vấn cho trước, các bước thực hiện như sau:

(a) **Vector hóa câu truy vấn:** Với một chuỗi văn bản truy vấn cho trước, sử dụng hàm **vectorize()** đã khai báo trước đó để tìm được dạng biểu vector của câu truy vấn. Giả sử, với câu truy vấn có nội dung **q = "Ai là người máy AI?"**, khi đưa vào **vectorize()**, chuỗi câu truy vấn sẽ được chuẩn hóa và tính vector bag-of-words trên bộ vocab đã tìm được trước đó, cuối cùng sẽ được kết quả là {'học': 0, 'sách': 0, 'ai': 2, 'máy': 1, 'người': 1, 'ấy': 0, 'là': 1} hay $[0, 0, 2, 1, 1, 0, 1]$.

(b) **Tính độ tương đồng:** Với từng tài liệu trong ma trận document-term, thực hiện tính độ tương đồng giữa vector câu truy vấn với vector tài liệu (có thể lưu kết quả này vào một list).

(c) **Thực hiện xếp hạng:** Sau khi có danh sách các điểm tương đồng, thực hiện sắp xếp theo thứ tự giảm dần về điểm tương đồng. Cuối cùng, các tài liệu nằm ở đầu danh sách đã sắp xếp này sẽ có thể được coi là có liên quan nhất đến câu truy vấn.

	học	sách	ai	máy	người	ấy	là
doc1	2	1	1	0	0	0	0
doc2	1	1	0	1	0	0	0
doc3	0	0	1	0	1	1	1
query	0	0	2	1	1	0	1

distance(q, d)

Tài liệu	Điểm tương đồng
d3	0.756
d1	0.308
d2	0.218

Hình 8: Thực hiện truy vấn và lập bảng xếp hạng các tài liệu có liên quan theo điểm độ tương đồng tính được.

Như vậy, trong project này các bạn sẽ thực hiện các công việc như sau:

1. Code hoàn thiện một chương trình truy vấn văn bản dựa trên các bước trong mô tả với bộ ngữ liệu tải tại đây. Hàm tính điểm tương đồng sử dụng phép cosine similarity với công thức:

$$cosine\_similarity(\vec{a}, \vec{b}) = \frac{a \cdot b}{\|a\|\|b\|} = \frac{\sum_{i=1}^{N} a_i b_i}{\sqrt{\sum_{i=1}^{N} a_i^2}\sqrt{\sum_{i=1}^{N} b_i^2}}$$

Lưu ý rằng:

- Cấu trúc mỗi tài liệu trong bộ ngữ liệu cho trước sẽ bao gồm tiêu đề bài báo ở hàng đầu tiên, hai hàng tiếp theo là nội dung bài báo và dòng cuối cùng là tên tác giả nếu có.
- Ngoài nội dung bài báo, các bạn cũng cần lưu trữ tiêu đề bài báo để làm nhận diện cho tài liệu tương ứng trong ma trận document-term.
- Các bạn cần sử dụng token từ nội dung bài báo để làm bộ từ vựng.

2. Thực hiện xếp hạng sử dụng điểm tương đồng là L1 Norm với công thức:

$$l1\_norm(\vec{a}, \vec{b}) = \sum_{i=1}^{N} |a - b|$$

3. Thực hiện xếp hạng sử dụng điểm tương đồng là L2 Norm với công thức:

$$l2\_norm(\vec{a}, \vec{b}) = \sqrt{\sum_{i=1}^{N} (a-b)^2}$$

4. Thay đổi hàm vector hóa văn bản sử dụng binary bag-of-words.

- Hết -