Tìm kiếm thông tin

Chương 4. Trọng số từ và giải pháp không gian vec-tơ

Soạn bởi: TS. Nguyễn Bá Ngọc

2021

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Tần suất từ và độ tương đồng

- Xét một trường hợp đơn giản với câu truy vấn một từ:
  - Nếu từ truy vấn không xuất hiện trong văn bản thì độ tương đồng (của) văn bản (với) truy vấn thường được đánh giá = 0
  - Nếu ngược lại, nếu từ truy vấn xuất hiện càng nhiều lần trong văn bản thì độ phù hợp thường được đánh giá càng cao.

Trong bài giảng này chúng ta sẽ tìm hiểu về cách đánh giá ***độ tương đồng*** *văn bản-truy vấn dựa trên nền tảng* ***đại số*** *trong mô hình* ***không gian vec-tơ****.*

Thử nghiệm đầu tiên: Hệ số Jaccard

- Hệ số Jaccard thường được sử dụng để đánh giá độ tương đồng giữa các đối tượng có thể được biểu diễn như những tập thuộc tính:
  - Trong chương 2 chúng ta đã sử dụng hệ số Jaccard để tính độ tương đồng giữa các từ.
- Hệ số Jaccard về bản chất là tỉ lệ phần tử chung của 2 tập hợp. Hệ số Jaccard của hai tập X và Y là:

$$Jaccard(X, Y) = |X \cap Y|/|X \cup Y|$$

  - $Jaccard(X, Y) \in [0, 1]$:
  - $Jaccard(X, Y) = 1 \iff X = Y$,
  - $Jaccard(X, Y) = 0 \iff X \cap Y = \varnothing$

Liệu có thể đánh giá độ tương đồng của văn bản với truy vấn theo hệ số Jaccard?

Ví dụ 4.1. Hệ số Jaccard

Thử sử dụng hệ số Jaccard để tính độ tương đồng giữa truy vấn và các văn bản:

Truy vấn: Mùa xuân
Văn bản 1: Đào hồng khoe sắc chào xuân đến
Văn bản 2: Xuân đi, xuân đến, hãy còn xuân

Liệu có thể sử dụng hệ số Jaccard để tìm kiếm văn bản?

Một số đặc trưng tiêu biểu

- Số lần từ (truy vấn) xuất hiện trong văn bản
- Số lượng văn bản chứa từ.
- Độ dài văn bản (số lượng từ trong văn bản có ảnh hưởng tới 2 đặc trưng đầu tiên).

Hệ số Jaccard không tính đến các đặc trưng này. Chúng ta sẽ phát triển công thức xếp hạng khác.

Tần suất từ

- Tần suất từ - Term frequency.
- Chúng ta ký hiệu $tf_{t,d}$ là số lần từ t xuất hiện trong văn bản d
- Ví dụ cho văn bản d1 có nội dung:

Xuân đi, xuân đến, hãy còn xuân

Từ xuân xuất hiện 3 lần, vì vậy $tf_{xuân,\ d1} = 3$.

Sử dụng tần suất từ như thế nào để tính độ tương đồng?

Tần suất văn bản

- Tần suất văn bản - Document frequency
- Chúng ta ký hiệu df_t là số lượng văn bản trong bộ văn bản có chứa từ t.
- Ví dụ cho bộ văn bản gồm các văn bản sau:

  d1: Ngôn ngữ lập trình C++

  d2: Lập trình hướng đối tượng với C++

  d3: Ngôn ngữ lập trình C

  Từ C++ xuất hiện trong 2 văn bản, vì vậy $df_{C++} = 2$.

Sử dụng tần suất văn bản như thế nào để tính độ tương đồng?

Trọng số tf.idf

- Trọng số từ được sử dụng trong xếp hạng văn bản thể hiện tính đặc trưng của từ đối với văn bản.
- Trọng số tf.idf được tính dựa trên tần suất từ và tần suất văn bản theo một số quan sát cơ bản sau:
  - Từ xuất hiện trong văn bản càng nhiều lần thì càng có ý nghĩa đối với văn bản đó:
    - Tuy nhiên nếu số lần xuất hiện của từ trong văn bản tăng lên gấp 10 lần thì ý nghĩa của nó đối với văn bản không tăng theo 10 lần => trọng số từ không tăng tuyến tính cùng với tần suất từ.
    - Chúng ta có thể mô phỏng quan sát này bằng một đại lượng đồng biến với tần suất từ.
  - Càng nhiều văn bản chứa từ thì tính đặc trưng của từ đó đối với 1 văn bản cụ thể càng nhỏ:
    - Chúng ta có thể mô phỏng quan sát này bằng một đại lượng nghịch biến với tần suất văn bản.

Trọng số tf.idf: Công thức tiêu biểu

- Có nhiều cách mô phỏng quan hệ đồng biến (với tf) và cũng có nhiều cách mô phỏng quan hệ nghịch biến (với df) => Có nhiều phiên bản tf.idf khác nhau.
- Trong tìm kiếm thông tin thường sử dụng công thức:

$$w_{tf.idf}(t, d) = (1 + \log(tf_{t, d})) * \log(N/df_t),$$

trong đó N là số lượng văn bản trong bộ dữ liệu;

trọng số được quy ước = 0 nếu $tf_{t,d} = 0$.

- Ví dụ:
  - $tf_{t,d} = 1$, $N = 1000$, $df_t = 10$, $\log_{10}$ => $w_{tf.idf}(t, d) = 2$
  - $tf_{t,d} = 2$, $N = 1000$, $df_t = 10$, $\log_{10}$ => $w_{tf.idf}(t, d) = 2.6$
  - $tf_{t,d} = 3$, $N = 1000$, $df_t = 100$, $\log_{10}$ => $w_{tf.idf}(t, d) = 1.48$

Vì sao cơ số hàm log được để trống trong công thức?

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Biểu diễn văn bản như vec-tơ

- Coi mỗi từ chỉ mục là một trục của không gian vec-tơ
  - Không gian vec-tơ thu được có số chiều rất lớn
  - Số chiều M = |V| = bằng kích thước bộ từ vựng
    - M có thể lên tới hàng chục triệu với tập văn bản lớn.
    - (*Định luật Heap đã học trong chương 2*).
- Biểu diễn văn bản
  - Sử dụng trọng số từ đối với văn bản như giá trị tọa độ trên trục tọa độ tương ứng với từ, chúng ta có thể biểu diễn mỗi văn bản như một vec-tơ M chiều.
    - Các vec-tơ thu được thường rất thưa (hầu hết các thành phần = 0).
  - Đồng thời có thể coi mỗi văn bản như một điểm trong không gian vec-tơ

Mô hình không gian vec-tơ / Vector Space Model (VSM)

Biểu diễn truy vấn như vec-tơ

- Chúng ta cũng có thể biểu diễn câu truy vấn như vec-tơ trong cùng không gian biểu diễn văn bản
- Mục đích: Mô phỏng độ tương đồng giữa văn bản với truy vấn bằng các đại lượng có thể đo được trong không gian vec-tơ:
  - Khoảng cách giữa các điểm;
  - Góc tạo bởi các vec-tơ.
  - v.v..

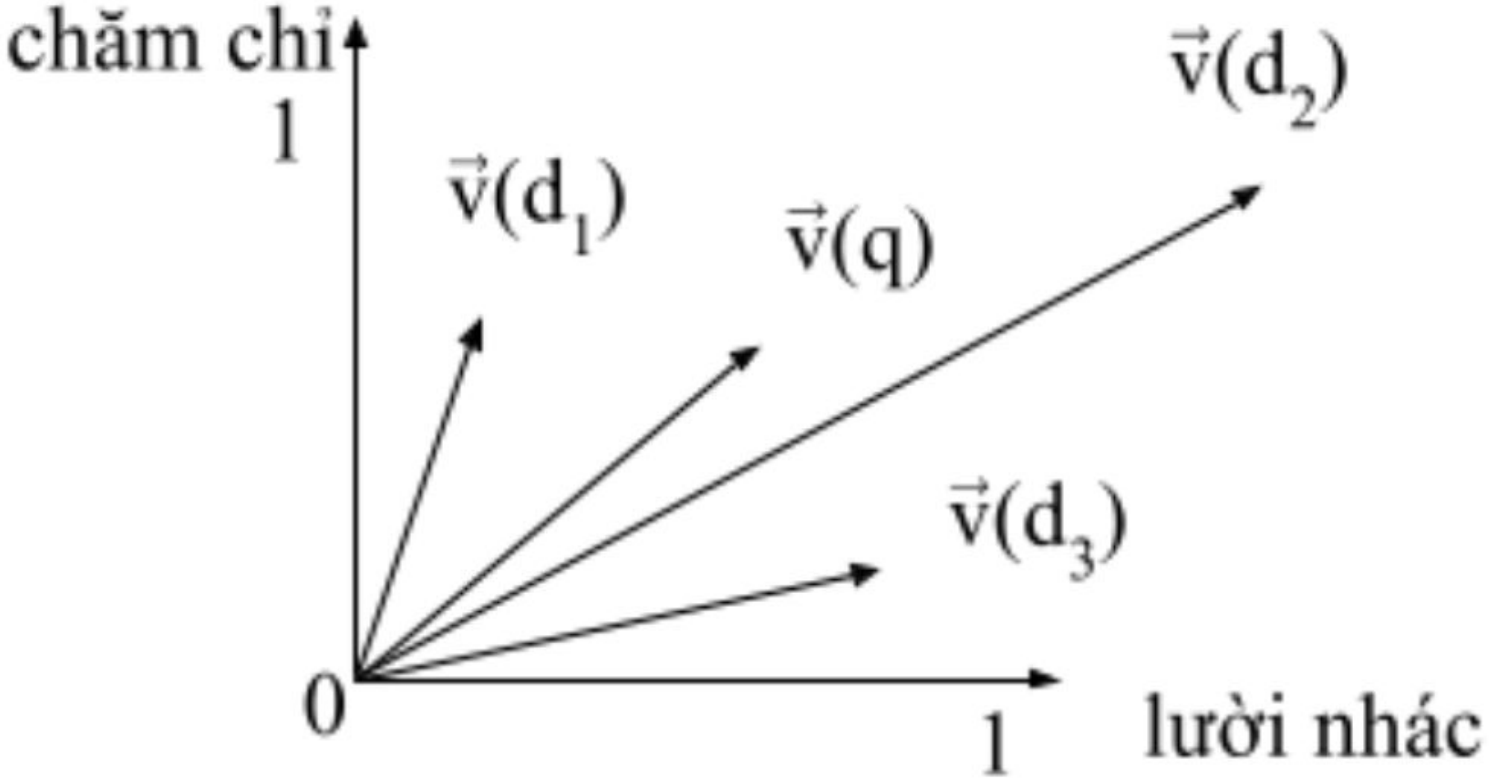
Các văn bản trong danh sách kết quả tìm kiếm được sắp xếp theo thứ tự giảm dần độ tương đồng với truy vấn.

Mô phỏng độ tương đồng trong VSM

- Thử nghiệm đầu tiên: Đo khoảng cách giữa các điểm
- Khoảng cách Euclid?
  - Các vec-tơ có độ dài khác nhau thường có khoảng cách Euclid lớn;
    - Trong thực tế độ dài của vec-tơ truy vấn thường ngắn hơn nhiều so với văn bản
  - Nếu lặp nội dung văn bản nhiều lần, thì độ dài vec-tơ tăng lên nhưng nội dung vẫn không thay đổi

Khoảng cách Euclide ít được sử dụng trong tìm kiếm và có thể cho kết quả không tốt như trong trường hợp các văn bản có khác biệt về độ dài.

Ví dụ 4.2. Khoảng cách Euclid



Văn bản d_2 có phân bố từ vựng giống với q nhất nhưng xa truy vấn q nhất, khoảng cách Euclid giữa $\vec{d}_2$ và $\vec{q}$ lớn nhất vì vậy d_2 bị xếp hạng thấp nhất.

Tuy nhiên có thể thấy góc giữa $\vec{d}_2$ và $\vec{q}$ là nhỏ nhất.

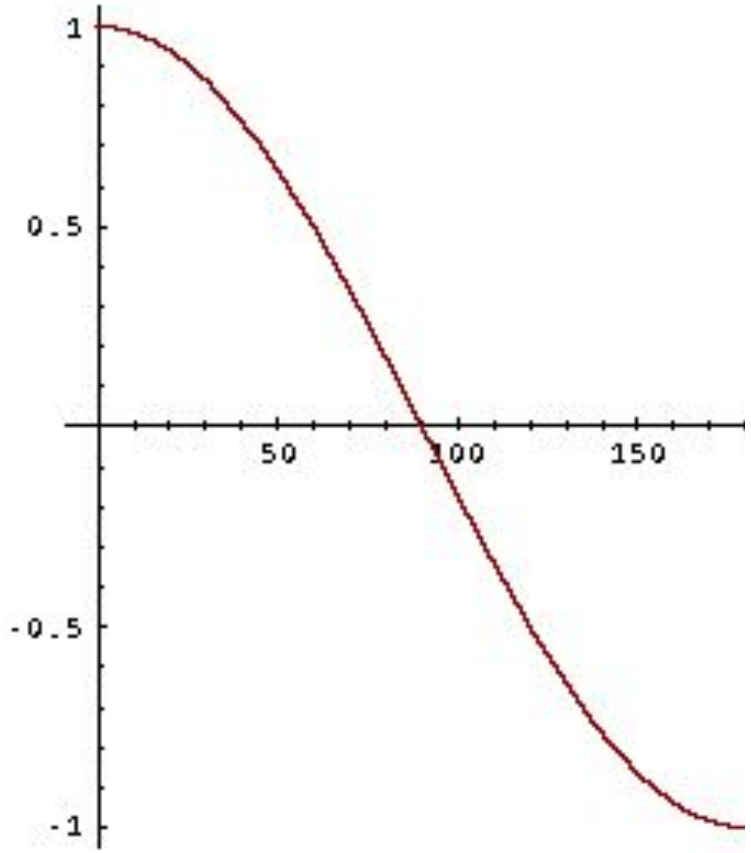
Thử nghiệm 2: Khoảng cách góc

- Thử lấy một văn bản d và lặp nội dung của nó nhiều lần, chúng ta ký hiệu văn bản thu được là d'
- Về mặt ý nghĩa thì d và d' tương đương
  - Khoảng cách Euclid giữa $\vec{d}$ và $\vec{d}'$ có thể rất lớn.
  - Nhưng góc giữa hai vec-tơ $\vec{d}$ và $\vec{d}'$ bằng 0 - tương ứng với khoảng cách tối thiểu.

Khoảng cách góc có thể mô phỏng độ tương đồng tốt hơn khoảng cách Euclide trong VSM

Góc và cosine

- Cosine là hàm đơn điệu giảm trong khoảng [0º, 180º]
- Vì vậy hai lựa chọn sau là tương đương:
  - Sắp xếp các văn bản theo thứ tự tăng dần góc tạo bởi các biểu diễn vec-tơ của văn bản truy vấn
  - Sắp xếp các văn bản theo thứ tự giảm dần cosine góc tạo bởi các biểu diễn vec-tơ của văn bản và truy vấn

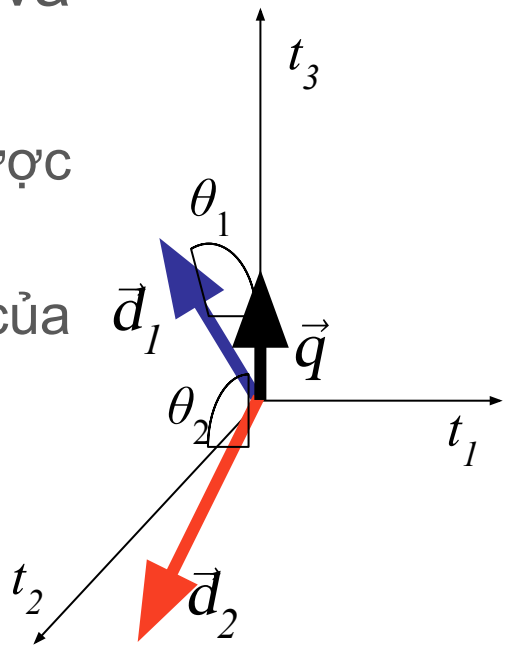


Tuy nhiên tính cosine thuận tiện và đơn giản hơn tính góc

Độ tương đồng cosine

- Cosine của góc tạo bởi vec-tơ văn bản và vec-tơ truy vấn:
  - Nghịch biến so với khoảng cách góc và được gọi là độ tương đồng cosine
  - Bằng tích vô hướng chia cho tích độ dài của các vec-tơ

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



Ví dụ:

$\vec{d}_1 = 2\vec{t}_1 + 3\vec{t}_2 + 5\vec{t}_3$ $\quad Sim_{\cos}(\vec{d}_1, \vec{q}) = 10 / \sqrt{(4+9+25)(0+0+4)} = 0.81$

$\vec{d}_2 = 3\vec{t}_1 + 7\vec{t}_2 + 1\vec{t}_3$ $\quad Sim_{\cos}(\vec{d}_2, \vec{q}) = 2 / \sqrt{(9+49+1)(0+0+4)} = 0.13$

$\vec{q} = 0\vec{t}_1 + 0\vec{t}_2 + 2\vec{t}_3$

=> d_1 phù hợp với truy vấn q hơn so với d_2.

Chuẩn hóa cosine

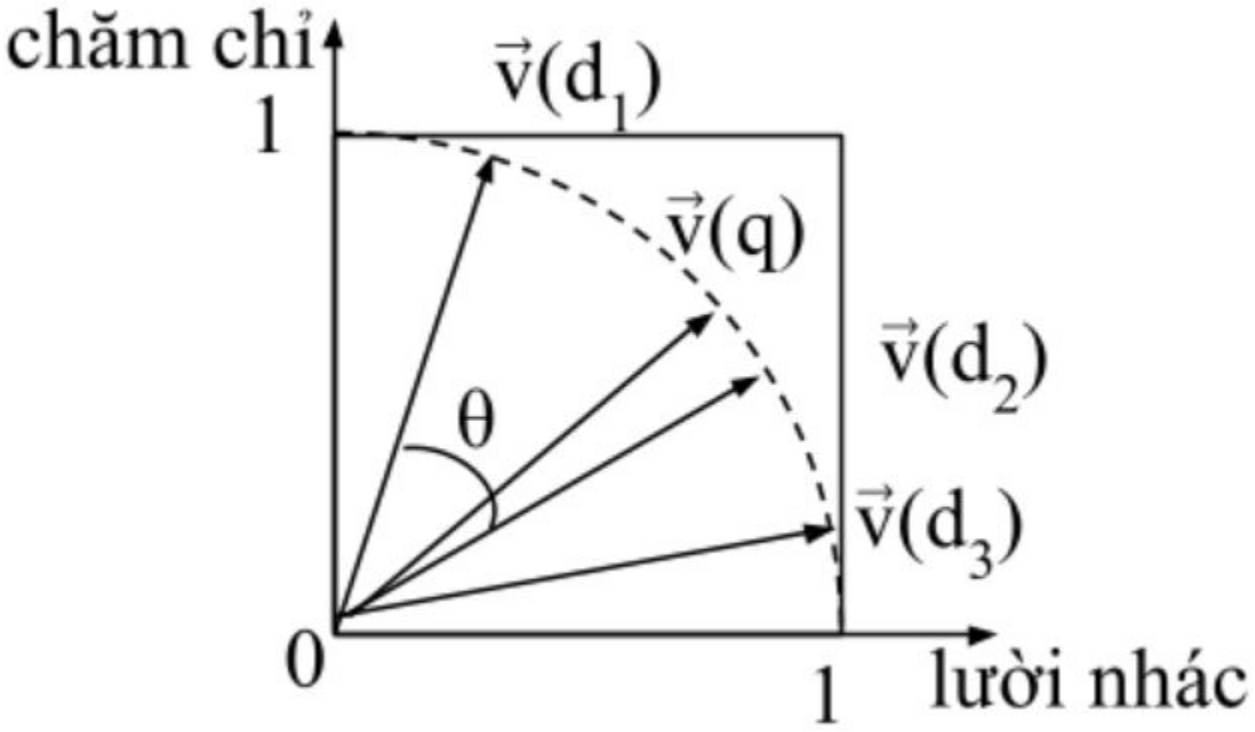
- Chuẩn hóa vec-tơ bằng cách chia vec-tơ cho độ dài của nó. Đại lượng chuẩn hóa cosine:

$$1/||\vec{x}|| = 1/\sqrt{\Sigma_i x_i^2}$$

- Các hệ quả:
  - Độ dài vec-tơ sau chuẩn hóa cosine = 1. Mỗi văn bản có thể được coi như một điểm trên bề mặt siêu cầu có bán kính = 1 (sau khi chuẩn hóa biểu diễn vec-tơ);
  - Cosine góc của các vec-tơ bằng tích vô hướng của các vec-tơ đó sau khi chuẩn hóa cosine.
  - => Cosine là một trường hợp đặc biệt của tích vô hướng.

Chuẩn hóa cosine làm giảm sự khác biệt khoảng cách Euclide của các văn bản ngắn và văn bản dài tới truy vấn.

Ví dụ 4.3. Chuẩn hóa cosine



Đối với các vec-tơ đã chuẩn hóa: Thứ tự xếp hạng theo khoảng cách góc tương đương với thứ tự xếp hạng theo khoảng cách Euclid.

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Hệ ký hiệu SMART

- SMART – System for the Mechanical Analysis and Retrieval of Text - là một dự án nghiên cứu TKTT dựa trên nền tảng đại số
  - Thử nghiệm nhiều công thức tính độ tương đồng

Term frequency		Document frequency		Normalization	
n (natural)	$\mathrm{tf}_{t,d}$	n (no)	1	n (none)	1
l (logarithm)	$1+\log(\mathrm{tf}_{t,d})$	t (idf)	$\log \frac{N}{\mathrm{df}_t}$	c (cosine)	$\frac{1}{\sqrt{w_1^2+w_2^2+\ldots+w_M^2}}$
a (augmented)	$0.5+\frac{0.5\times\mathrm{tf}_{t,d}}{\max_t(\mathrm{tf}_{t,d})}$	p (prob idf)	$\max\{0, \log\frac{N-\mathrm{df}_t}{\mathrm{df}_t}\}$	u (pivoted unique)	$1/u$
b (boolean)	$\begin{cases}1 & \text{if } \mathrm{tf}_{t,d}>0\\ 0 & \text{otherwise}\end{cases}$			b (byte size)	$1/CharLength^{\alpha}$, $\alpha<1$
L (log ave)	$\frac{1+\log(\mathrm{tf}_{t,d})}{1+\log(\mathrm{ave}_{t\in d}(\mathrm{tf}_{t,d}))}$				

Các chữ cái trong bảng là các ký hiệu cho các công thức được sử dụng để tính độ tương đồng.

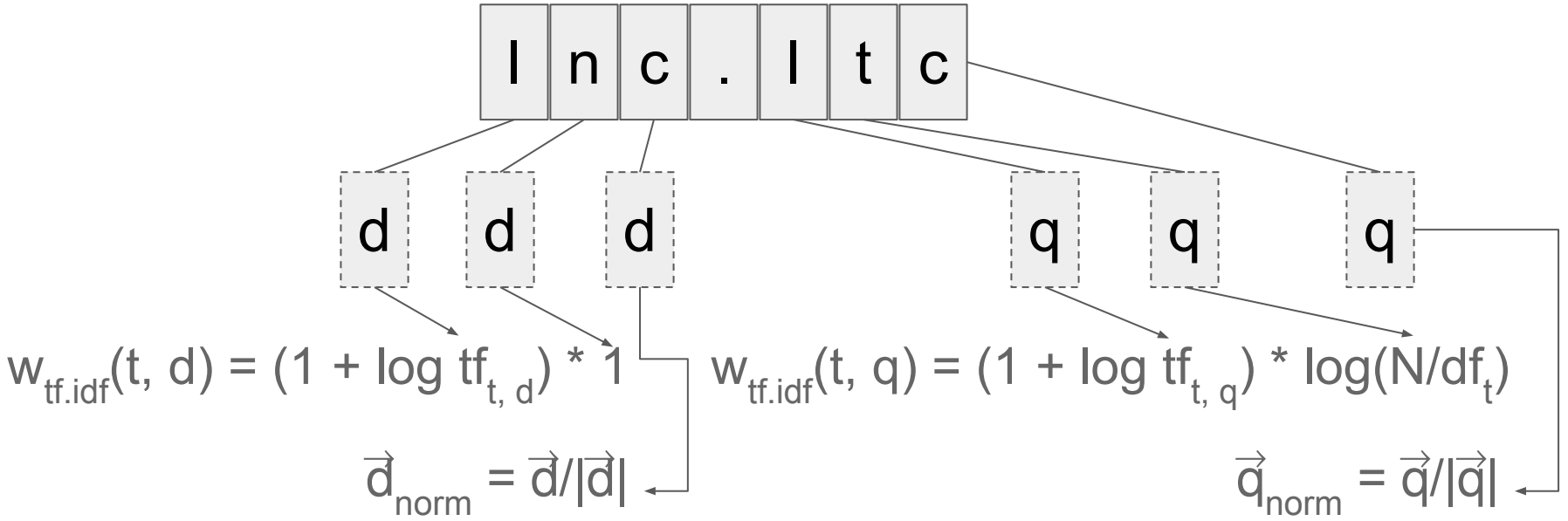
Vì sao cơ số hàm log được để trống?

Tính độ tương đồng trong hệ SMART

- Trong hệ ký hiệu SMART, một công thức tính độ tương đồng được mô tả bằng một bộ 6 ký tự theo định dạng ddd.qqq
  - (*Các chữ cái d, q trong mô tả chỉ là đại diện cho các ký tự tương ứng với các công thức sẽ được áp dụng cho văn bản và truy vấn*).
  - Ba ký tự đầu tiên mô tả các công thức được áp dụng cho văn bản (để tính và chuẩn hóa biểu diễn vec-tơ).
  - Ba ký tự tiếp theo mô tả các công thức được áp dụng cho truy vấn.
  - Độ tương đồng được tính bằng tích vô hướng của các vec-tơ thu được sau khi áp dụng các công thức.

Ví dụ 4.4. Công thức tính độ tương đồng

- Xét công thức tiêu biểu là lnc.ltc:
  - Đối với văn bản:
    - l - Lấy log tf (ý nghĩa của chữ l đầu tiên),
    - n - không sử dụng idf (= 1 cho tất cả các từ).
    - c - chuẩn hóa cosine
  - Đối với truy vấn: sử dụng log tf, idf, và chuẩn hóa cosine



Ví dụ 4.5. Độ tương đồng theo **lnc.ltc**

Sử dụng các dữ liệu:

- Văn bản: Bảo hiểm ô tô bảo hiểm xe máy
- Truy vấn: Bảo hiểm ô tô tốt nhất

Từ chỉ mục	Văn bản				Truy vấn						Tích
	tf	w_{tf}	w	norm	tf	w_{tf}	df	idf	w	norm	
xe máy	1				0		5000	2.3			
tốt nhất	0				1		50000	1.3			
ô tô	1				1		10000	2.0			
bảo hiểm	2				1		1000	3.0			

Dựa trên các giá trị df và idf hãy xác định N = ?
Hãy điền đủ các giá trị vào các cột đang còn trống?

Ví dụ 4.5. Độ tương đồng theo **lnc.ltc**(2)

Sử dụng các dữ liệu:

- Văn bản: Bảo hiểm ô tô bảo hiểm xe máy
- Truy vấn: Bảo hiểm ô tô tốt nhất

Từ chỉ mục	Văn bản				Truy vấn						Tích
	tf	w_{tf}	w	norm	tf	w_{tf}	df	idf	w	norm	
xe máy	1	1	1	0.52	0	0	5000	2.3	0	0	0
tốt nhất	0	0	0	0	1	1	50000	1.3	1.3	0.34	0
ô tô	1	1	1	0.52	1	1	10000	2.0	2.0	0.52	0.27
bảo hiểm	2	1.3	1.3	0.68	1	1	1000	3.0	3.0	0.78	0.53

$|\vec{d}| = \sqrt{1^2 + 0^2 + 1^2 + 1.3^2} \approx 1{,}92$ $|\vec{q}| = \sqrt{1{,}3^2 + 0^2 + 2{,}0^2 + 3.0^2} \approx 3{,}83$

$RSV(d, q) = \vec{d}_{norm} \cdot \vec{q}_{norm} = 0+0+0.27+0.53 = 0.8$

Ví dụ 4.6. Độ tương đồng theo **Inc.ltn**

Sử dụng các dữ liệu:

- Văn bản: Bảo hiểm ô tô bảo hiểm xe máy tốt nhất
- Truy vấn: Bảo hiểm ô tô tốt nhất

Từ chỉ mục	Văn bản				Truy vấn						Tích
	tf	w_{tf}	w	norm	tf	w_{tf}	df	idf	w	norm	
xe máy	1				0		5000	2.3			
tốt nhất	1				1		50000	1.3			
ô tô	1				1		10000	2.0			
bảo hiểm	2				1		1000	3.0			
										Tổng:	

Hãy điền đủ các giá trị vào các ô đang còn trống?

Nhược điểm của chuẩn hóa cosine

- Chuẩn hóa cosine có xu hướng cho các văn bản ngắn điểm quá cao và cho các văn bản dài điểm quá thấp.
  - Có thể dẫn đến các lỗi trả về các văn bản ngắn không phù hợp truy vấn
  - và các lỗi bỏ qua các văn bản dài phù hợp với truy vấn.

Khắc phục các vấn đề này bằng cách nào?

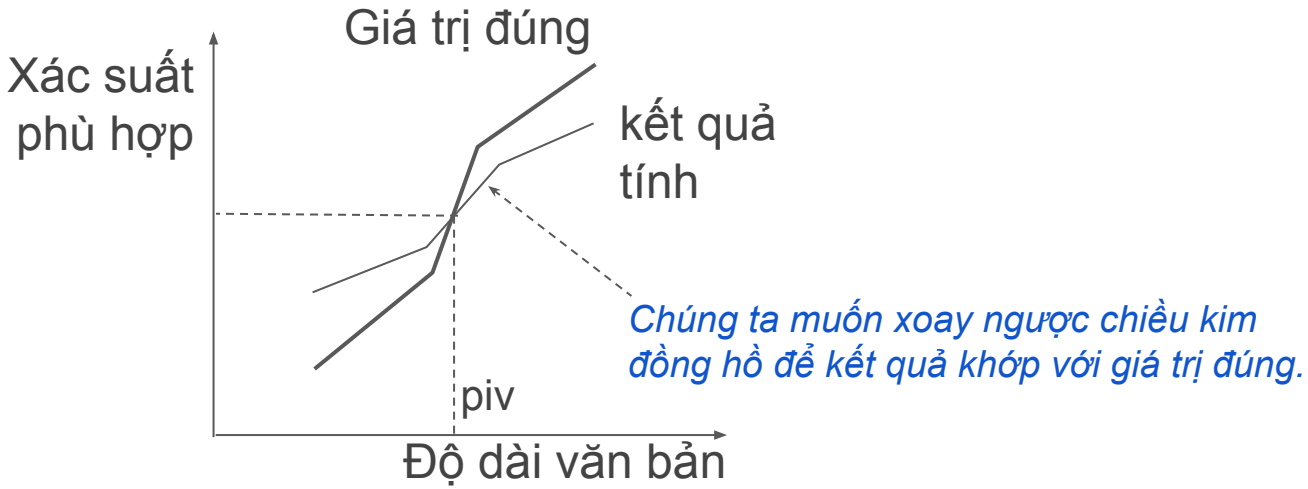
Chuẩn hóa xoay chốt

- Ý tưởng: Hiệu chỉnh đại lượng chuẩn hóa cosine theo một công thức tuyến tính: "Xoay đại lượng chuẩn hóa quanh chốt"
- Mục đích:
  - Giảm độ tương đồng với truy vấn cho các văn bản ngắn;
  - Tăng độ tương đồng với truy vấn cho các văn bản dài.

=> *Kiểm soát lợi thế ưu tiên của các văn bản ngắn so với các văn bản dài.*

Chuẩn hóa độ dài và xác suất phù hợp

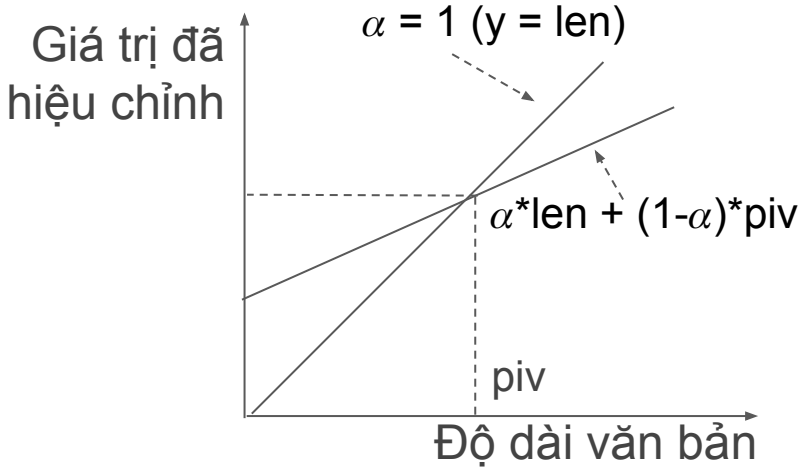
- Sử dụng một bộ dữ liệu kiểm thử (bao gồm tập văn bản và tập truy vấn đã được đánh giá tính phù hợp theo cặp).
- Gom nhóm các văn bản thành khối theo độ dài và tính xác suất phù hợp cho khối = tỉ lệ văn bản phù hợp trong khối.



Có thể quan sát được vấn đề với chuẩn hóa cosine trên biểu đồ: Xác suất phù hợp của các khối văn bản ngắn cao hơn giá trị đúng, và xác suất phù hợp của các khối văn bản dài thấp hơn giá trị đúng.

Hiệu chỉnh đại lượng chuẩn hóa độ dài

- Hiệu chỉnh đại lượng chuẩn hóa bằng một hàm tuyến tính. Đối với chuẩn hóa cosine: $\alpha||\vec{d}||+(1-\alpha)*piv$

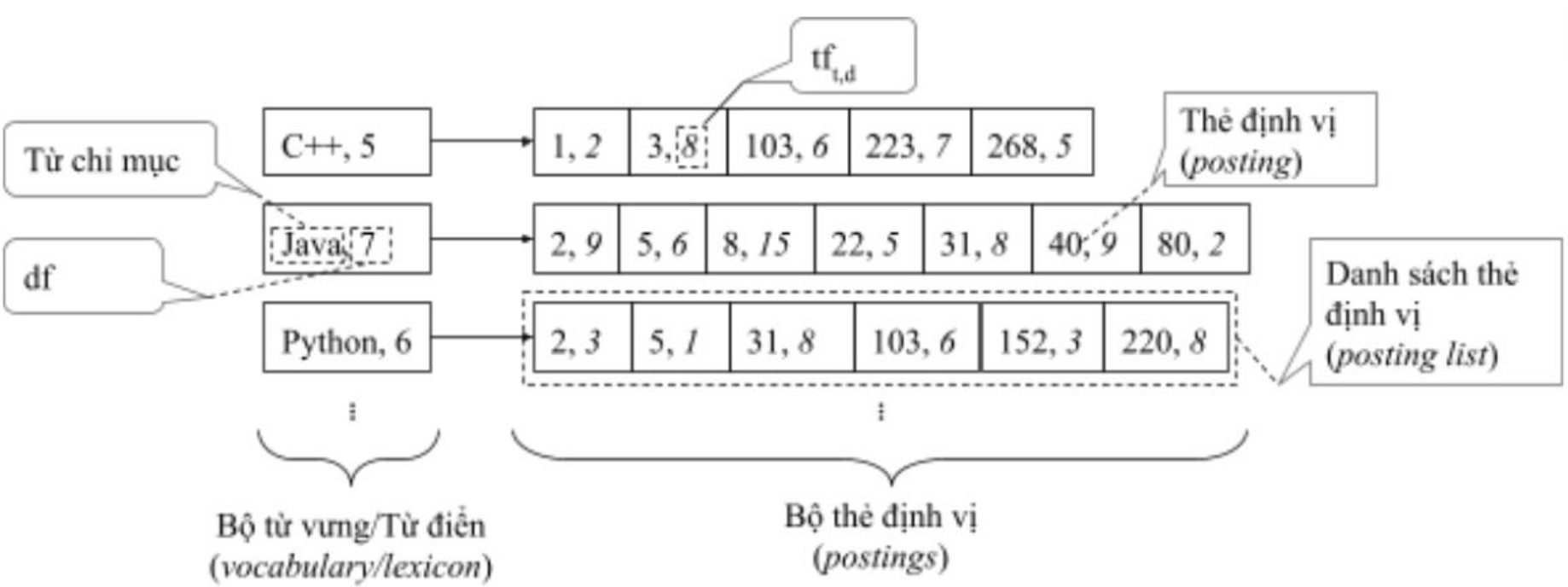


- Có những nghiên cứu cho thấy chuẩn hóa xoay chốt cho kết quả tìm kiếm tốt hơn so với chuẩn hóa cosine (trong một số trường hợp)
- Đại lượng chuẩn hóa $\alpha u_d + (1 - \alpha) * piv$, trong đó u_d là số lượng từ duy nhất trong d có hiệu ứng tương tự như sử dụng đại lượng chuẩn hóa cosine

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Trọng số từ và chỉ mục ngược



- Để tính độ tương đồng trong VSM chúng ta cần lưu thêm tf trong chỉ mục ngược:
  - Lưu tf trong thẻ định vị cùng với docId, kích thước chỉ mục có tf sau khi nén có tăng lên nhưng không đáng kể.
  - Không nên lưu giá trị trọng số (tf.idf):
    - Khó nén số thực hơn so với số nguyên;
    - Không đáp ứng được yêu cầu thay đổi công thức tính độ tương đồng.

Tính độ tương đồng

- Cách đơn giản nhất là xử lý từng từ truy vấn:
  - Cấp cho mỗi docId một biến tích lũy;
  - Duyệt các danh sách thẻ định vị gắn với các từ truy vấn:
    - Tính thành phần tích vô hướng tương ứng với thẻ định vị
    - Cộng vào biến tích lũy cho văn bản
  - Chuẩn hóa các giá trị biến tích lũy
    - Các đại lượng chuẩn hóa như độ dài vec-tơ, số lượng ký tự trong văn bản, v.v... được tính ở thời điểm tạo chỉ mục và lưu riêng bên ngoài chỉ mục ngược.
  - Tuy nhiên tốn thời gian đối với các danh sách dài.
- Trong trường hợp trật tự thẻ định vị trong các danh sách là ổn định chúng ta có thể xử lý từng văn bản:
  - Tương tự giải thuật lấy giao đồng thời nhiều danh sách.

Giải thuật tính độ tương đồng với chỉ mục

+ Các phần tử trong mảng scores có giá trị tăng dần vì vậy được gọi là các biến tích lũy
+ Độ dài truy vấn là cố định => Có thể bỏ qua thành phần độ dài truy vấn mà không ảnh hưởng tới kết quả tìm kiếm.

N có thể rất lớn

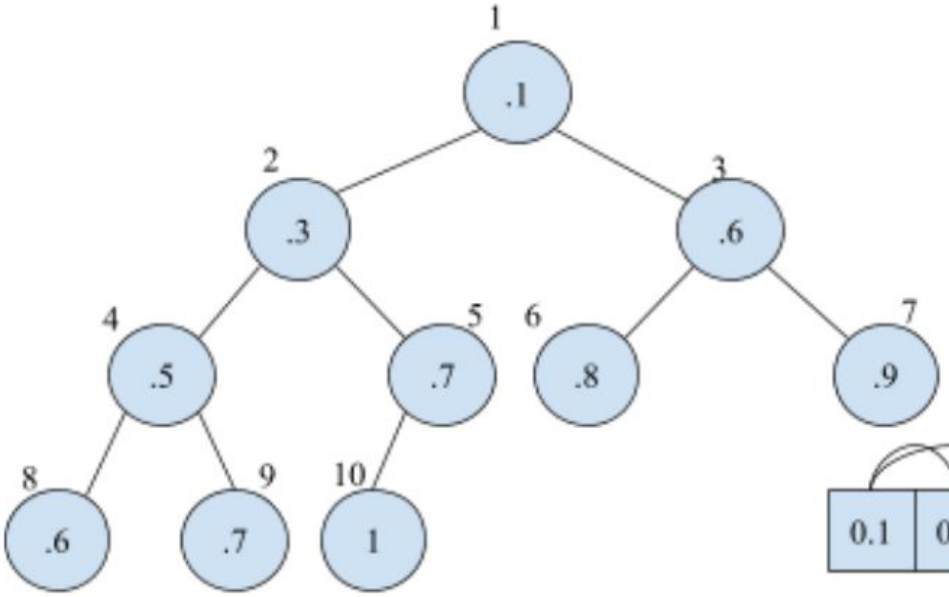
```
SimilarityScore(q, K)
1     float scores[N] = 0
2     Khởi tạo mảng độ dài văn bản length[N]
3     for t in q
4           Tính w_{t,q}
            p = postings_list(t)
5           for (d, tf_{t,d}) in p
6                 tính w_{t,d}
7                 scores[d] += w_{t,d} * w_{t,q}
8     for each d
9           scores[d] /= length[d]
10    return top K of scores
```

Chọn top K giá trị lớn nhất

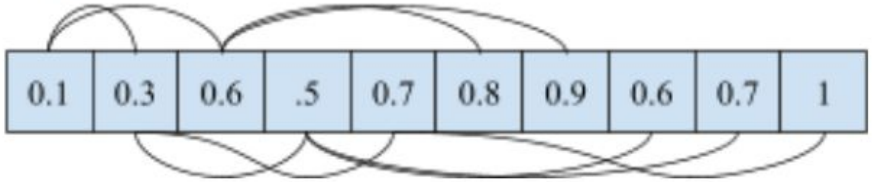
- Thông thường top K văn bản có độ tương đồng lớn nhất được trả về.
  - Đặt J là số lượng văn bản được tính độ tương đồng.
  - Chúng ta cần tìm K văn bản có cosine lớn nhất trong số J văn bản
- Không có giải thuật với độ phức tạp tuyến tính để lấy ra top-K từ một mảng chưa được sắp xếp.
  - => Cần hạn chế phạm vi tìm top K để giảm thời gian xử lý truy vấn.

Làm sao để lấy ra top-K nhanh mà không sắp xếp?

Sử dụng Min Heap để lọc top K



Lưu K giá trị trong Min Heap, nếu văn bản có độ tương đồng nhỏ hơn giá trị ở gốc thì bỏ qua, nếu ngược lại thì cập nhật gốc bằng độ tương đồng của văn bản.



- Min Heap là cây nhị phân (đầy đủ) trong đó tất cả các nút đều có giá trị < các giá trị của các nút con (thường được triển khai bằng mảng).
- Chi phí lấy top K bằng Min Heap nhỏ hơn nhiều so với chi phí sắp xếp.

Tất nhiên cũng có thể tạo Max Heap cho mảng tích lũy rồi lấy ra top K.

Xử lý từng văn bản

- Đối với các truy vấn nhiều từ: Nếu các thẻ định vị được sắp xếp theo thứ tự tăng dần mã văn bản, thì chúng ta có thể tính điểm hoàn thiện cho từng văn bản bằng cách duyệt đồng thời nhiều danh sách thẻ định vị.
  - Xử lý tương tự như với giải thuật lấy giao hai danh sách.
  - Trong trường hợp tổng quát: Cần duy trì thứ tự ổn định giữa các văn bản để có thể tính độ tương đồng hoàn thiện cho từng văn bản.
- Nếu các văn bản không được lưu theo một thứ tự ổn định, thì độ tương đồng cho các văn bản chỉ có thể được tính đầy đủ sau khi duyệt hết các danh sách.

Nếu có thể xử lý từng văn bản thì dễ xác định khả năng văn bản nằm trong top K hay không hơn => Có thể giảm số lượng biến tích lũy cần lưu.

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Vấn đề thời gian xử lý truy vấn

- Các danh sách thẻ định vị có thể rất dài
  - Làm sao để giữ tốc độ phản hồi nhanh?
  - Liệu có thể bỏ qua (không tính cosine cho) một số văn bản?
- Bỏ qua văn bản kéo theo khả năng sai lệch top K:
  - Những văn bản nằm trong top K có thể không phải là những văn bản có độ tương đồng lớn nhất.
- Sai lệch top K có làm giảm chất lượng tìm kiếm?
  - Độ tương đồng chỉ thể hiện khả năng đáp ứng nhu cầu thông tin của người dùng - văn bản có độ tương đồng cao có thể là văn bản người dùng đang tìm kiếm hoặc không.
  - Văn bản có độ tương đồng nhỏ hơn vẫn có thể là kết quả phù hợp.
  - => Chất lượng tìm kiếm có thể giảm nhưng không quá nhiều … Bù lại tốc độ phản hồi nhanh đem lại trải nghiệm tích cực.

Bài toán thu hẹp phạm vi tìm kiếm

Mục tiêu:

- Tìm tập $A \subset D$, sao cho $K < |A| << N$
  - và A chứa nhiều văn bản trong top K (không cần tất cả)
- Thực hiện tìm kiếm trong A và trả về top K.
  - mong đợi kết quả tìm kiếm vẫn có chất lượng tốt.

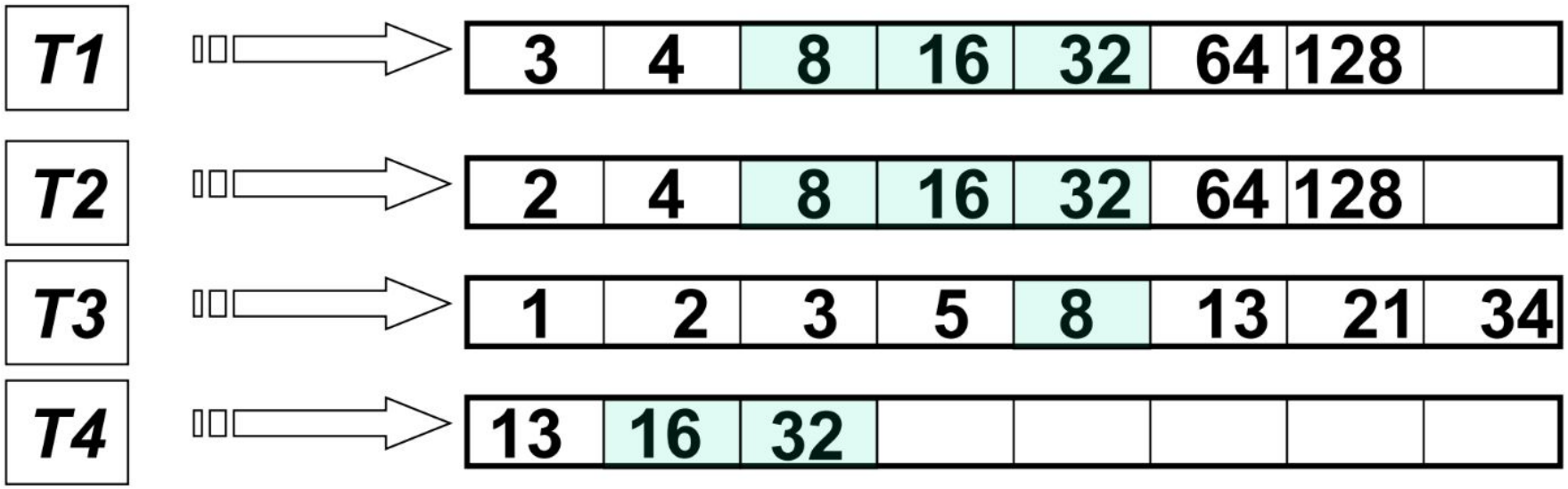
Tiếp theo chúng ta sẽ tìm hiểu một số phương pháp tiêu biểu để thu hẹp phạm vi tìm kiếm

Lọc từ truy vấn

Một số lựa chọn:

- Chỉ xét các từ truy vấn có idf cao
  - Cơ sở: Từ có idf nhỏ ít ảnh hưởng tới độ tượng đồng;
  - Lợi ích: Từ có idf nhỏ có danh sách thẻ định vị dài, loại bỏ từ có idf nhỏ giúp giảm đáng kể thời gian xử lý truy vấn.
- Chỉ xét các văn bản có chứa nhiều từ truy vấn
  - Cơ sở: Văn bản có chứa nhiều từ truy vấn có nhiều khả năng là văn bản phù hợp.
  - Điều kiện áp dụng: Trong điều kiện có thể xử lý từng văn bản và truy vấn chứa nhiều từ.
  - Lợi ích: Giảm được số trường hợp cần tính độ tương đồng.

Ví dụ 4.7. Lọc từ truy vấn



Nếu chỉ xét các văn bản có chứa tối thiểu 70% số lượng từ truy vấn, thì chúng ta chỉ cần tính độ tương đồng cho các văn bản 8, 16 và 32.

Các đại lượng tĩnh

- Tính phù hợp có thể được tính theo VSM cho văn bản theo từng truy vấn.
- Bên cạnh đó cũng có những đại lượng thể hiện chất lượng văn bản độc lập với truy vấn, chúng ta gọi đó là các đại lượng tĩnh.
- Ví dụ một số đại lượng tĩnh:
  - Có liên kết từ Wikipedia
  - Là bài viết thuộc tạp chí uy tín
  - Có nhiều trích dẫn
  - PageRank

  } Độc lập với truy vấn.
- Giả sử chúng ta có thể tính được đại lượng tĩnh thể hiện chất lượng của các văn bản, ký hiệu là g(d).

Kết hợp các thành phần

- Có thể kết hợp độ tương đồng và đại lượng chất lượng tĩnh bằng một công thức tuyến tính đơn giản:

  RSV(d, q) = net-score(d, q) = g(d) + sim(q, d),

  trong đó g(d) và sim(q, d) được chuẩn hóa về cùng miền giá trị.

- Cả hai đại lượng đều thể hiện khả năng đáp ứng nhu cầu thông tin của người dùng

Tất nhiên chúng ta có thể sử dụng bất kỳ hàm kết hợp tuyến tính nào khác.

Thu hẹp phạm vi tìm kiếm dựa trên g(d)

- Ý tưởng: Sắp xếp các thẻ định vị theo g(d)
  - Vẫn có thể đảm bảo trật tự ổn định cho cả thẻ định vị và thực hiện xử lý từng văn bản.
- Lợi ích: Các văn bản có điểm xếp hạng cao có thể ở đầu danh sách => Có thể dừng xử lý sớm
  - Ví dụ, giới hạn xử lý trong 50 ms.

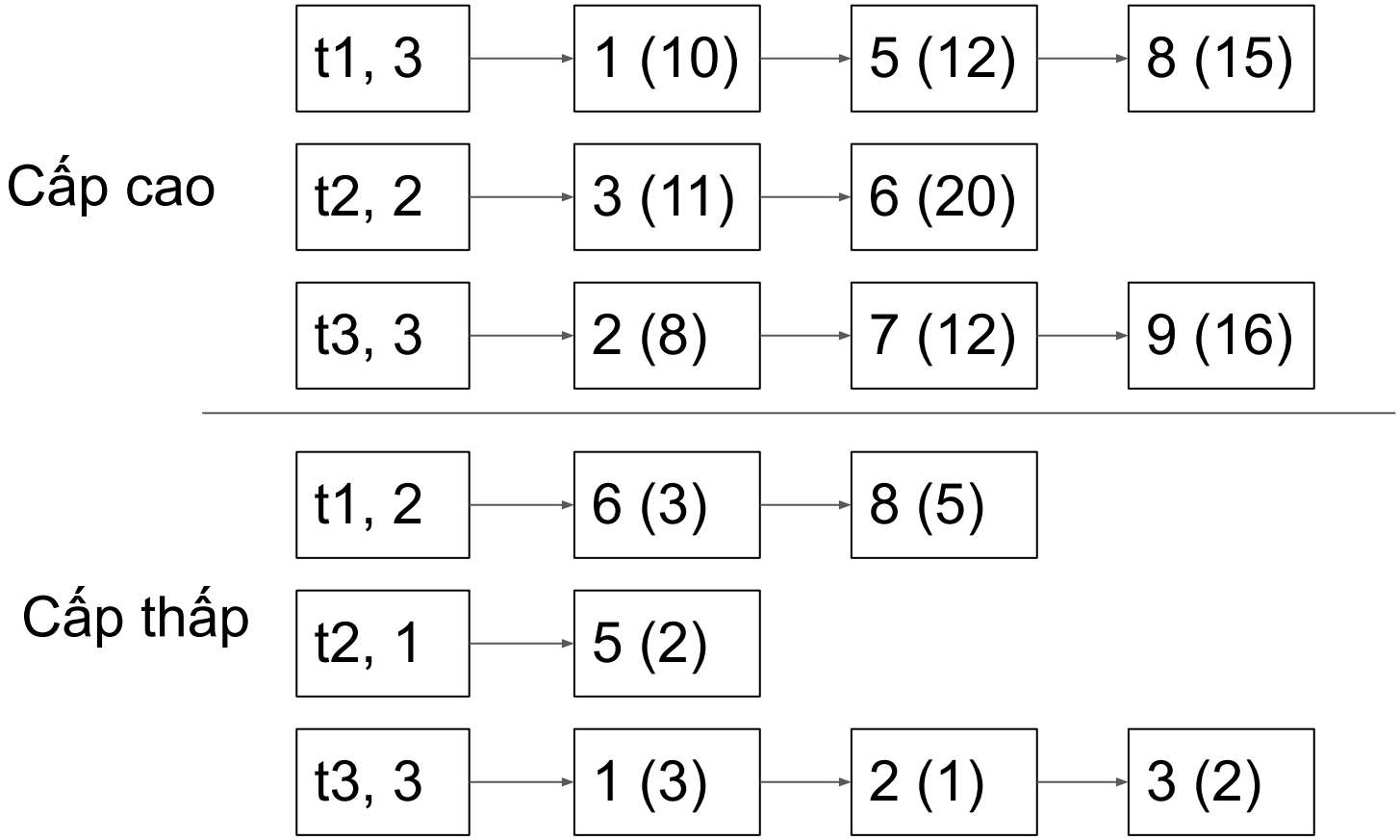
Thử: Viết giải thuật (mã giả) tính cosine với điều kiện các thẻ định vị được sắp xếp theo g(d)

Phân cấp chỉ mục

- Với mỗi từ chỉ mục, chúng ta duy trì hai danh sách thẻ định vị, gọi là danh sách cao và danh sách thấp
  - Tách các danh sách theo ngưỡng tf.
  - Danh sách cao chứa những thẻ định vị có tf lớn.
  - Danh sách thấp chứa những thẻ định vị còn lại.
  - Như vậy chúng ta sẽ có 2 chỉ mục.
  - *Có nên chia thành nhiều cấp hơn hay không?*
- Khi xử lý truy vấn danh sách cao được duyệt trước:
  - Nếu đã tìm đủ top-K văn bản, thì dừng tiến trình duyệt
  - Nếu ngược lại thì tiếp tục tìm kiếm trong danh sách thấp

Chúng ta cũng có thể tách danh sách theo số lượng thẻ định vị

Ví dụ 4.8. Phân cấp chỉ mục



Thử tìm top 3 cho truy vấn t1 t3 trong 2 trường hợp:

- Giới hạn tìm kiếm trong chỉ mục cấp cao.
- Tìm kiếm trong toàn bộ chỉ mục (Không giới hạn).

Sắp xếp thẻ định vị theo ảnh hưởng

- Chúng ta cũng có thể sắp xếp thẻ định vị theo trọng số, ví dụ $tf_{t,d}$:
  - Lợi ích: Các văn bản có trọng số cao đứng đầu danh sách => Có thể chủ động ngắt sớm
    - Ví dụ: có thể ngắt sau khi hết giới hạn thời gian, hoặc trọng số giảm xuống nhỏ hơn ngưỡng.
  - Tuy nhiên: Không có trật tự ổn định giữa các thẻ định vị => Không thể xử lý từng văn bản:
    - Độ tương đồng của văn bản khi ngắt sớm có thể là giá trị gần đúng.
    - Cần duy trì giá trị tích lũy của các văn bản trong tiến trình xử lý.

Giản lược tìm kiếm dựa trên chia cụm

Tổ chức chỉ mục:

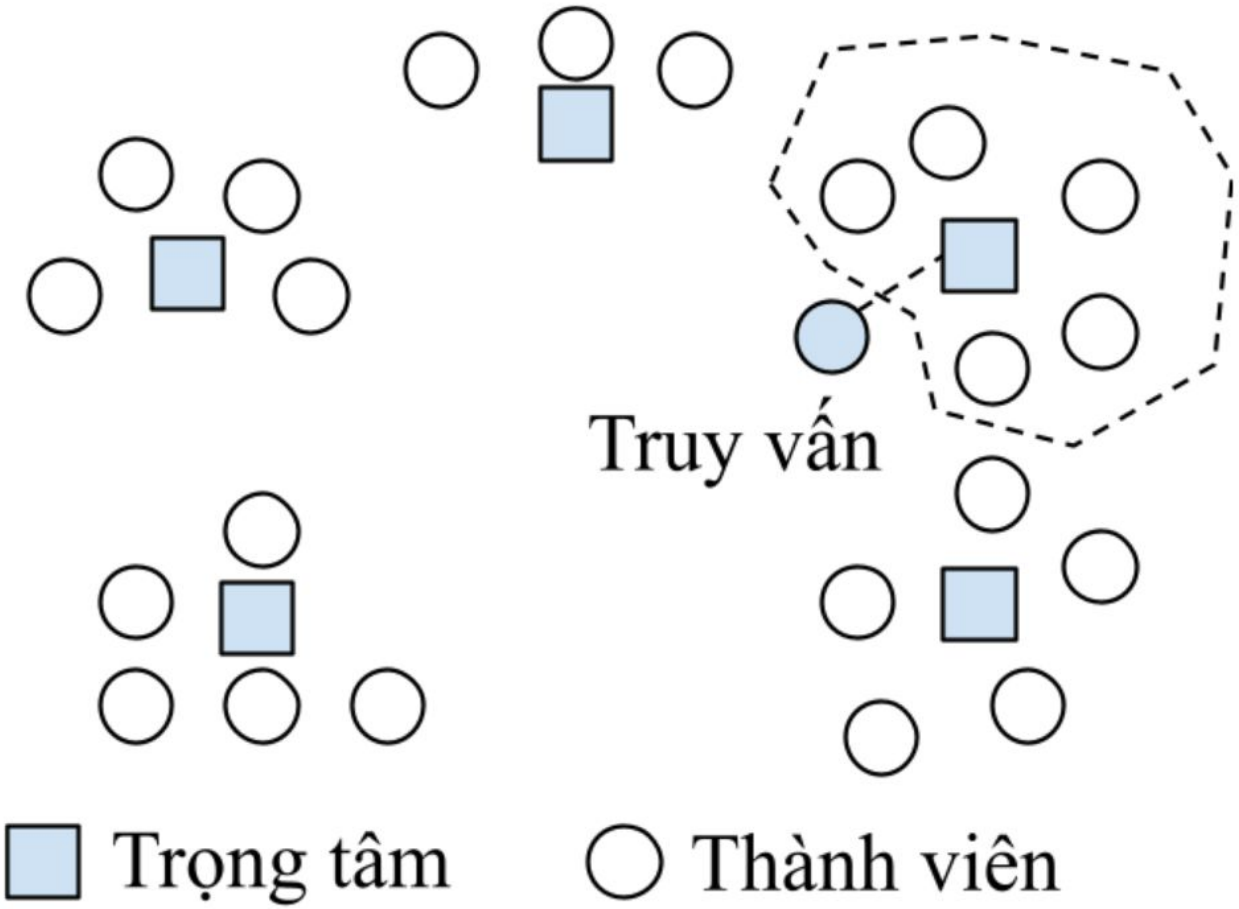
- Chia bộ văn bản được sử dụng để tìm kiếm thành b1 cụm và xác định trọng tâm cho các cụm.
  - *(Giải thuật chia cụm văn bản sẽ được học sau)*
- Tạo chỉ mục riêng cho từng cụm => Chúng ta thu được cấu trúc chỉ mục phân tán theo văn bản.

Xử lý truy vấn:

- Cho truy vấn q, tìm b2 trọng tâm gần nhất.
- Tìm top K trong phạm vi các cụm tương ứng với b2 trọng tâm tìm được.

Giới hạn tìm kiếm dựa trên phân cụm vẫn có thể bỏ qua kết quả phù hợp; b1 và b2 là các tham số tùy chỉnh.

Ví dụ 4.9. Tìm kiếm trong phạm vi cụm

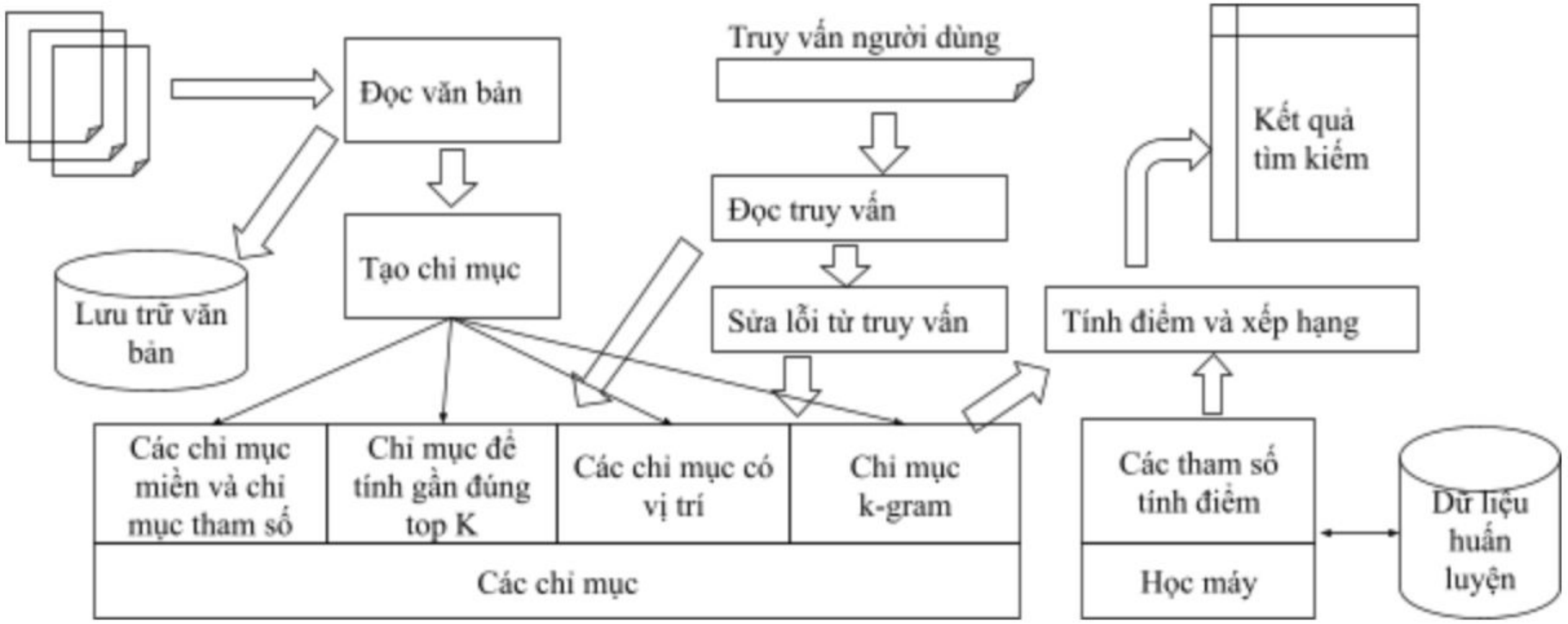


Lấy tâm cụm ngẫu nhiên: Nhanh và phản ánh được phân bố dữ liệu.

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Kiến trúc tổng quan của máy tìm kiếm



Các thành phần đã được giới thiệu

- Đọc văn bản - Tách nội dung văn bản từ tài liệu, tách từ và các xử lý NNTN
- Tạo chỉ mục - Các giải thuật tạo chỉ mục
- Sửa lỗi từ truy vấn - Tính khoảng cách soạn thảo, tính độ tương đồng, gợi ý từ truy vấn.
- Tính điểm và xếp hạng - Xử lý theo từng văn bản (hoặc từng từ).
- Các chỉ mục - Cần nhiều chỉ mục ngược khác nhau theo lô-gic truy vấn

Các thành phần chưa được nhắc đến

- Lưu trữ văn bản - Chúng ta cần nội dung văn bản để tạo các trích đoạn kết quả tìm kiếm (trích đoạn động)
  - Có thể được lưu trong các CSDL
  - Với số lượng văn bản rất lớn như quy mô Web cần đến các công nghệ lưu trữ dữ liệu lớn.
- Các hàm xếp hạng dựa trên học máy.
- Đọc truy vấn - Chuyển đổi các truy vấn dạng văn bản được cung cấp bởi người dùng thành các biểu diễn truy vấn bên trong hệ thống.

Mô-đun đọc truy vấn

- Truy vấn dạng văn bản do người dùng cung cấp có thể kéo theo một hoặc nhiều truy vấn được xử lý bên trong hệ thống, ví dụ, truy vấn: Tăng trưởng kinh tế có thể kéo theo:
  - Thực hiện truy vấn như truy vấn nguyên câu
  - Nếu có ít hơn K văn bản có chứa nguyên đoạn "Tăng trưởng kinh tế" thì chạy hai truy vấn: Tăng trưởng và Kinh tế
  - Xếp hạng các văn bản tìm được theo VSM
  - v.v…
- Các kết quả thực hiện các truy vấn thành phần sau đó được tổng hợp lại thành kết quả thực hiện truy vấn ban đầu.

Tổng hợp điểm và học máy

- Các hàm tính điểm có thể kết hợp các thành phần cosine, đại lượng tĩnh, v.v.., và sử dụng các tham số.
- Làm sao để xác định các giá trị tham số tốt nhất?
  - Trong một trường hợp được thiết lập bởi chuyên gia
  - … Trong (có thể là phần lớn) các trường hợp còn lại được thiết lập bằng các phương pháp học máy

Mô-đun học máy cung cấp các bộ tham số được sử dụng để tính điểm và xếp hạng.

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Xếp hạng dựa trên trọng số miền

- Để minh họa phương pháp thiết lập trọng số bằng học máy, trước tiên chúng ta xét một mô hình xếp hạng đơn giản dựa trên các miền.
- Giả sử các văn bản có cấu trúc bao gồm Z miền
  - Đặt g_1, g_2, ..., g_Z là các trọng số của các miền;
    - Các trọng số có giá trị trong khoảng [0, 1] và $g_1 + ... + g_Z = 1$
  - Đặt s_i là kết quả tính toán tính phù hợp của miền thứ i với truy vấn q (ví dụ theo mô hình Boolean): $s_i = 1$ nếu phù hợp, và $s_i = 0$ nếu ngược lại.
- Chúng ta sẽ sử dụng công thức đơn giản để xếp hạng các văn bản theo truy vấn q như sau:

$$RSV(d, q) = \Sigma_{i=1..Z} g_i * s_i$$

Phát biểu bài toán

- Xét một trường hợp đơn giản nhất với các văn bản được chia thành 2 miền: Tiêu đề (T) và nội dung (B)
  - Đặt $g \in [0, 1]$ là trọng số của miền tiêu đề;
  - $RSV(d, q) = g * s_T(d, q) + (1 - g) * s_B(d, q)$.
- Dữ liệu huấn luyện được biểu diễn như bộ ba
  - $\Phi_j=(d_j, q_j, r(d_j, q_j))$, trong đó $r(d_j, q_j) = 1$ nếu d_j phù hợp với q_j và $r(d_j, q_j) = 0$ nếu ngược lại; $r(d_j, q_j)$ được đánh giá bởi người (*chi tiết hơn được cung cấp sau trong phần đánh giá*).
- Hàm lỗi:
  - Giá trị lỗi cho 1 mẫu Φ_j: $\varepsilon(g,\Phi_j)=(r(d_j,q_j)-RSV(d_j,q_j))^2$;
  - giá trị lỗi cho cả bộ dữ liệu huấn luyện Φ:
  - $\varepsilon(g,\Phi) = \varepsilon(g,\Phi_1) + \varepsilon(g,\Phi_2) + ... + \varepsilon(g,\Phi_N)$
- Bài toán học máy: Tìm g sao cho ε đạt cực tiểu trên bộ dữ liệu huấn luyện Φ

Kết quả tính toán và dữ liệu huấn luyện

Thực hiện một thống kê đơn giản cho các trường hợp:

STT	$s_T(d_j, q_j)$	$s_B(d_j, q_j)$	$RSV(d_j, q_j)$	Số mẫu	
				phù hợp	không phù hợp
1	0	0	0	n_{00R}	n_{00N}
2	0	1	1 - g	n_{01R}	n_{01N}
3	1	0	g	n_{10R}	n_{10N}
4	1	1	1	n_{11R}	n_{11N}

Chúng ta có giá trị lỗi trên toàn bộ dữ liệu huấn luyện là:

$$(n_{01R} + n_{10N}) * g^2 + (n_{01N} + n_{10R}) * (1 - g)^2 + n_{00R} + n_{11N}$$

Đạt cực tiểu tại: $g = (n_{10R} + n_{01N})/(n_{10R}+n_{10N}+n_{01R}+n_{01N})$

Giá trị tham số cần tìm

Nội dung

1. Trọng số từ
2. Mô hình không gian vec-tơ
3. Các phiên bản của trọng số tf-idf
4. Xử lý truy vấn trong mô hình không gian vec-tơ
5. Các kỹ thuật thu hẹp phạm vi tìm kiếm
6. Các thành phần của hệ thống tìm kiếm thông tin
7. Minh họa thiết lập tham số bằng học máy
8. So sánh các hình thức truy vấn đã học

Một số câu hỏi mở

- Làm sao để kết hợp truy vấn nguyên câu với VSM?
  - Có thể tính df và idf cho các câu hay không?
- Làm sao để kết hợp các toán tử Boolean với VSM?
- Làm sao để sử dụng các ký tự đại diện trong VSM?
- Làm sao để đưa khoảng cách giữa các từ vào tính điểm xếp hạng?
  - Người dùng ưa thích các văn bản có từ truy vấn xuất hiện gần nhau.

Xử lý các truy vấn như trong mô hình Boolean rồi sau đó xếp hạng các kết quả theo VSM?

Truy vấn trong VSM và mô hình Boolean

- Trong VSM nhu cầu thông tin được diễn đạt như văn bản, không chứa các toán tử lô-gic, vì vậy đơn giản hơn (đối với người dùng) so với truy vấn Boolean
  - Truy vấn dạng văn bản (như trong VSM) hiện nay cũng là hình thức truy vấn được sử dụng phổ biến nhất.
  - Truy vấn nguyên câu và các liên kết lô-gic vẫn được hỗ trợ cùng với truy vấn dạng văn bản, nhưng ít phổ biến hơn.
- Trong VSM câu truy vấn được tách thành tập từ truy vấn, các văn bản chứa nhiều từ truy vấn có thể có độ tương đồng cao hơn các văn bản có chứa ít từ truy vấn vì vậy có nhiều khả năng được trả về hơn => Tương tự như liên kết AND được ngầm định và ưu tiên cao hơn liên kết OR.

Bài tập 4.1

Khoảng cách Euclide (hoặc khoảng cách L2) giữa hai vec-tơ $\vec{x}$ và $\vec{y}$ được xác định như sau:

$$\|\vec{x} - \vec{y}\| = \sqrt{\sum_{i=1}^{M} (x_i - y_i)^2}$$

Cho truy vấn q và các văn bản d_1, d_2, ..., d_N. Hãy chứng minh nếu biểu diễn vec-tơ của các văn bản đã được chuẩn hóa thành các vec-tơ đơn vị thì kết quả xếp hạng theo thứ tự tăng dần khoảng cách Euclide giống kết quả xếp hạng theo thứ tự giảm dần độ tương đồng cosine

Bài tập 4.2

a) Trọng số idf của từ xuất hiện trong mọi văn bản bằng bao nhiêu? So sánh ảnh hưởng của trọng số idf với thao tác lọc từ dừng?

b) Trọng số tf-idf của một từ có thể vượt quá 1 hay không?

Bài tập 4.3

Cho các dữ liệu thống kê tf và df như sau:

tf(t, d)	Doc1	Doc2	Doc3
xe máy	27	4	24
ô tô	3	33	0
bảo hiểm	0	33	29
tốt nhất	14	0	17

t	df	idf
xe máy	18 165	
ô tô	6 723	
bảo hiểm	19 241	
tốt nhất	25 235	

Cho N = 800 000

a) Hãy tính ma trận trọng số tf.idf

b) Xếp hạng các văn bản theo truy vấn "bảo hiểm ô tô tốt nhất" dựa trên sơ đồ nnn.atc

Bài tập 4.4

Giả sử các đại lượng chất lượng tĩnh xác định được cho các văn bản d1, d2, d3 lần lượt là 0.25, 0.5 và 1. Hãy vẽ chỉ mục ngược với các thẻ định vị được sắp xếp theo $g(d) + tf_{norm}(t, d)$. Biết rằng:

tf ở dạng thô			
	d1	d2	d3
t1	27	4	24
t2	3	33	0
t3	0	33	29
t4	14	0	17

tf đã chuẩn hóa cosine			
	d1	d2	d3
t1	0.88		
t2	0.10		
t3	0		
t4	0.46		

Có thứ tự ổn định trong trường hợp này hay không?

Bài tập 4.5

Trong trường hợp phân cấp chỉ mục, đối với các truy vấn nhiều từ chúng ta cần các thẻ định vị với độ dài tối thiểu là bao nhiêu trong cấp đầu tiên để có thể lấy đủ top K văn bản?

Bài tập 4.6

Vấn đề láng giềng gần nhất trên mặt phẳng có thể phát biểu như sau: Cho N điểm trên mặt phẳng, và một điểm tương ứng với truy vấn q. Chúng ta cần tìm K điểm (ví dụ K = 3) gần với q nhất trong N điểm.

Để tránh tính khoảng cách từ q đến tất cả các điểm còn lại có thể áp dụng phương pháp thu hẹp phạm vi tìm kiếm dựa trên phân cụm.

Hãy lấy một phản ví dụ với hai cụm sao cho phương pháp thu hẹp phạm vi tìm kiếm trả về kết quả sai/khác với kết quả vét cạn (bỏ qua một số điểm gần hơn).

Bài tập 4.7

Cho bộ dữ liệu văn bản gồm các văn bản sau:

d1: a b e g a

d2: b b f h b

d3: f g h h a g h

d4: a g b c c a c c c

d5: b f g g a g

d6: a e f f e e f e a e e

a) Hãy vẽ chỉ mục ngược có chứa tf (mã văn bản tăng dần).

b) Hãy tách chỉ mục thu được ở mục a thành 2 cấp, trong đó chỉ mục ở cấp 1 chỉ chứa các thẻ định vị với tf > 2, chỉ mục ở cấp 2 chứa tất cả các thẻ định vị còn lại.

c) Sử dụng sơ đồ **nnu.ntn** và tính kết quả tìm kiếm theo truy vấn a c e trong 2 trường hợp: 1) chỉ tìm kiếm ở chỉ mục cấp 1; 2) tìm kiếm đầy đủ ở cả 2 cấp; sau đó kiểm tra xem có khác biệt nào trong 2 trường hợp hay không?