Chương 3

CSDL văn bản (Text/Document database)

Nguyễn Thanh Bình
Đỗ Thanh Nghị

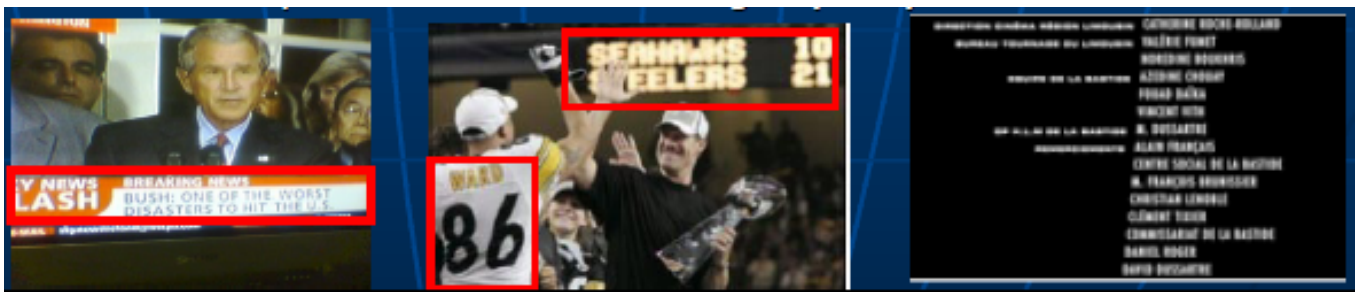
Khoa CNTT&TT - Đại học Cần Thơ

8/2010

Nội dung

CSDL văn bản là gì?

- CSDL văn bản chứa các tài liệu ở dạng văn bản.
- Nguồn dữ liệu văn bản: sách, báo, tạp chí, các meta-data, phụ đề phim, nhận dạng chữ tự động, . . .
- Một tài liệu có thể chỉ là một câu, một đoạn văn, một chương sách hay cả quyển sách.

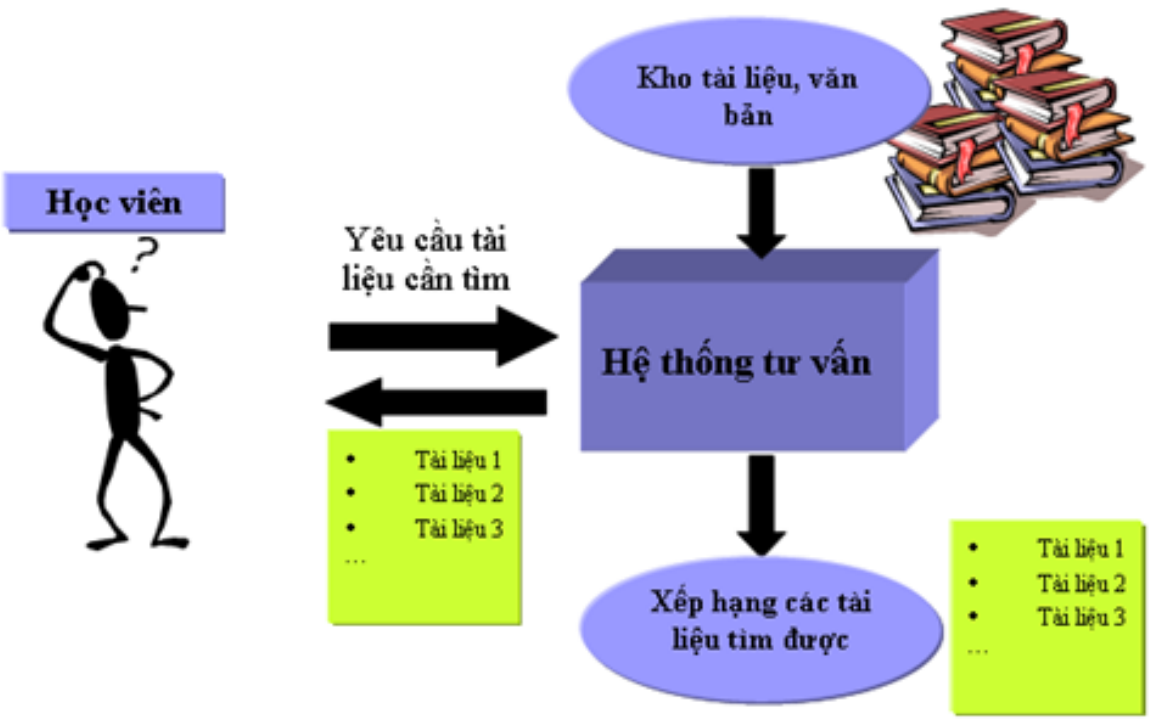


Ví dụ về CSDL văn bản

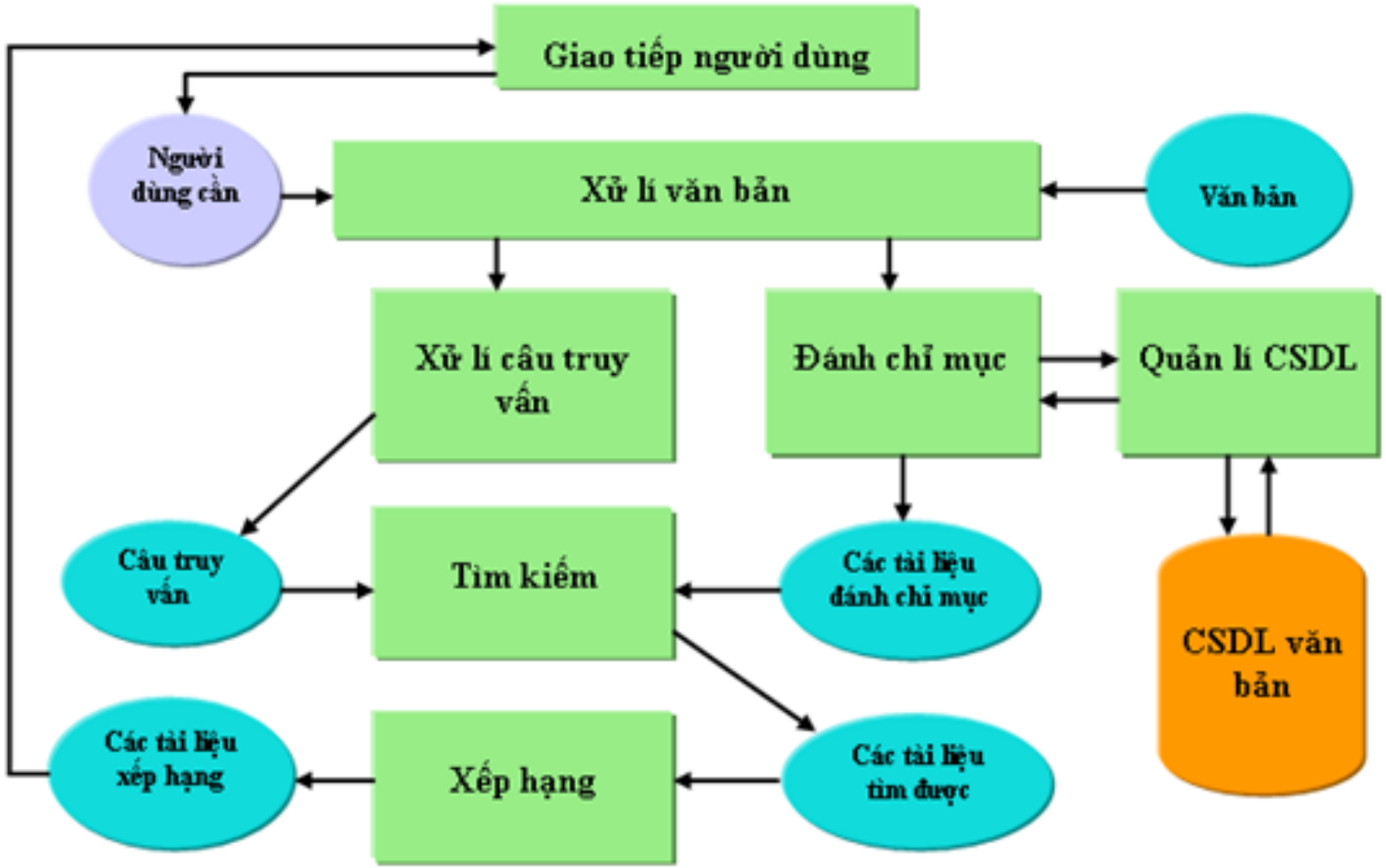
DocID	String
d_1	Jose Orojuelo's Operations in Bosnia.
d_2	The Medelin Cartel's Financial Organization.
d_3	The Cali Cartel's Distribution Network.
d_4	Banking Operations and Money Laundering.
d_5	Profile of Hector Gomez.
d_6	Connections between Terrorism and Asian Dope Operations.
d_7	Hector Gomez's: How he Gave Agents the Slip in Cali.
d_8	Sex, Drugs, and Videotape.
d_9	The Iranian Connection.
d_{10}	Boating and Drugs: Slips owned by the Cali Cartel.

Hệ tìm kiếm tài liệu văn bản

- Người sử dụng cần tìm một văn bản về chủ đề T.
- Hệ tìm kiếm văn bản sẽ tìm trong CSDL các văn bản có liên quan đến chủ đề T, sau đó xếp hạng theo mức độ liên quan đến chủ đề T và trả kết quả cho người dùng.



Sơ đồ hoạt động tìm kiếm văn bản



Chỉ mục tài liệu văn bản

Để việc lưu trữ, tìm kiếm, sắp xếp các tài liệu văn bản có hiệu quả, ta cần phải tạo chỉ mục cho chúng.

- Việc tạo chỉ mục cho văn bản bao gồm việc xác định các ý (chủ đề) chính chứa trong văn bản.
- Việc tạo chỉ mục được thực hiện bằng cách liên kết văn bản với một tập các từ *(word)* gọi là **từ chỉ mục** *(index term)* hay **khóa chỉ mục** *(index key)*.

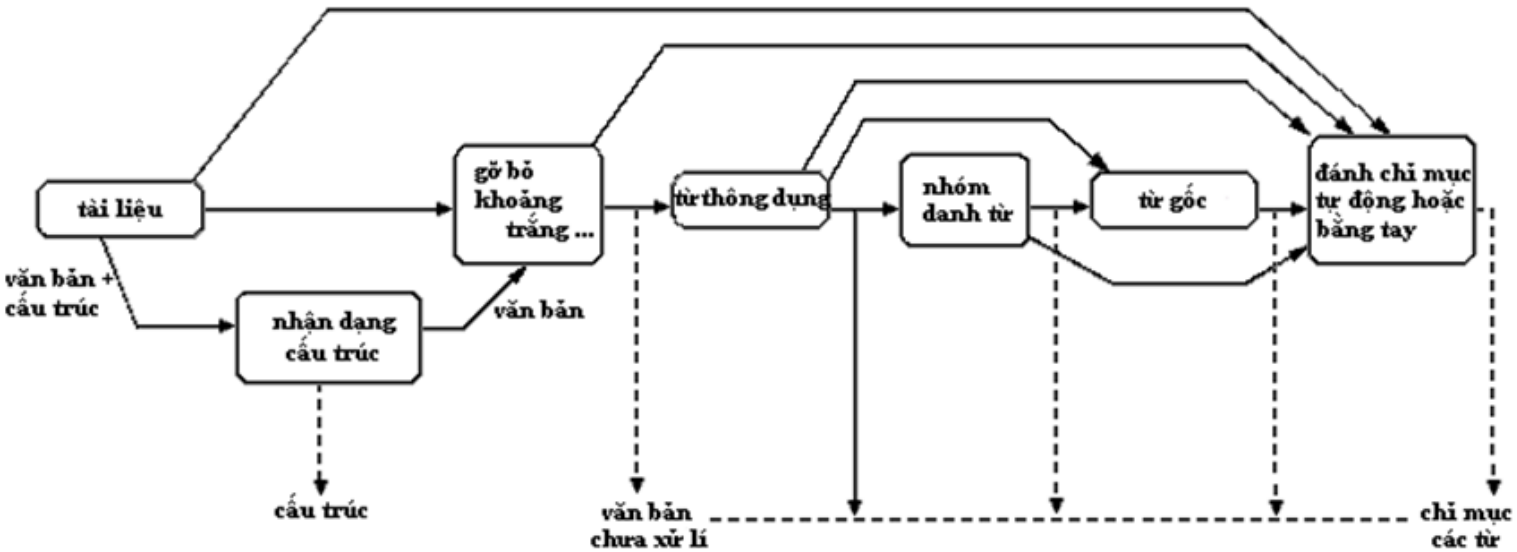
Các bước tạo chỉ mục CSDL văn bản

- Tiền xử lý các tài liệu văn bản nhằm mục đích lọc ra các từ chỉ mục (index terms).
- Tạo cấu trúc cho các từ chỉ mục sử dụng các cấu trúc dữ liệu đa chiều, bảng băm, hay các tập tin nghịch đảo.
- Chọn trọng số gán cho các từ chỉ mục.
- Xây dựng mô hình để so sánh các từ chỉ mục.

Tiền xử lý văn bản

Quá trình tiền xử lý văn bản gồm các bước:

- Cắt văn bản thành những từ, loại bỏ khoảng trắng và các ký hiệu lạ.
- Lọc bỏ các từ thông dụng *(stop words)* có trong văn bản.
- Quy những từ có trong văn bản về những từ gốc.
- Đếm số từ có trong văn bản để lập bảng tần số.



Tách từ và loại bỏ từ thông dụng

Tách từ

- Cắt văn bản thành một dãy các từ, bỏ khoảng trắng và qui đổi chữ hoa thành chữ thường.
- Bỏ hết tất cả ký tự lạ và dấu chấm; chỉ sử dụng các từ cắt được là những từ và kí tự thuộc bảng chữ cái.

Loại bỏ từ thông dụng

- Từ thông dụng các từ, nhóm từ mà nếu bỏ đi sẽ không ảnh hưởng đến nội dung hoặc kết quả tìm kiếm trên văn bản.
- Để hiệu quả hơn trong các thao tác xử lí, các chuỗi trong từ thông dụng được tổ chức ở dạng bảng băm nhằm tối ưu thời gian truy xuất và xử lí.

Đưa về từ gốc

- Một từ sẽ có nhiều biến thể khác nhau.
  - Ví dụ: drug, drugged, drugs; stop, stopped, stopping; child, children; . . .
- Để quy các từ về từ gốc *(stem)*, ta có thể sử dụng từ điển hoặc một chương trình xử lý tự động.
- Các chương trình xử lý tự động (ví dụ: Porter Stemmer [a]) sẽ dựa trên các luật để quy một từ về từ gốc.
  - sses $\rightarrow$ ss ; ies $\rightarrow$ i ; ational $\rightarrow$ ate ; tional $\rightarrow$ tion

[a] `http://tartarus.org/~martin/PorterStemmer/`

Ví dụ

Văn bản gốc:

Many large organizations are nothing but data processing engines.

Insurance companies, banks, financial services companies, and the IRS are all organizations that live in a sea of data. Most of what they do is process data.

Văn bản sau khi tách từ:

Many | large | organizations | are | nothing | but | data | processing | engines | .

Insurance | companies | , | banks | , | financial | services | companies | , | and | the | IRS | are | all | organizations | that | live | in | a | sea | of | data | . | Most | of | what | they | do | is | process | data | .

Ví dụ

Bỏ khoảng trắng và ký tự lạ, đổi chữ hoa thành chữ thường:

many large organizations are nothing but data processing engines insurance companies banks financial services companies and the irs are all organizations that live in a sea of data most of what they do is process data

Bỏ các từ thông dụng:

many large organizations are nothing but data processing engines insurance companies banks financial services companies and the irs are all organizations that live in a sea of data most of what they do is process data

organizations data processing engines insurance companies banks financial services companies irs organizations live sea data process data

Ví dụ

Đưa về từ gốc:

organizations	data	processing	engines	insurance	companies	banks	financial
services	companies	irs	organizations	live	sea	data	process	data

organiz	data	process	engin	insur	compan	bank	financ	servic	compan
irs	organiz	live	sea	data	process	data			

Tính tần số xuất hiện các từ:

bank	1	compan	2	data	3	engin	1	financ	1	insur	1
irs	1	live	1	organiz	2	process	2	sea	1	servic	1

Biểu diễn văn bản sử dụng bảng tần số

- Sau quá trình tiền xử lý văn bản, mỗi văn bản còn lại một tập các từ chỉ mục và tần số xuất hiện của từ đó trong văn bản.
- Tập hợp toàn bộ các văn bản trong CSDL sẽ tạo nên bảng tần số.
- Trong bảng tần số, mỗi văn bản được biểu diễn dưới dạng một **vector tần số**.

Bảng tần số

Bảng tần suất (*FreqT*) là bảng dùng để biểu diễn sự xuất hiện của từ trong tài liệu (từ / tài liệu).

- Mỗi dòng trong *FreqT* biểu diễn một từ.
- Mỗi cột trong *FreqT* biểu diễn một tài liệu.
- Mỗi giá trị $Freq(i, j)$ trong bảng cho biết số lần xuất hiện của từ t_i trong tài liệu d_j.

Định nghĩa

Gọi D là một tập gồm n văn bản và T là một tập gồm m từ có trong các văn bản của D. Bảng tần số **FreqT**, gắn với D và T, là một ma trận kích thước $(m \times n)$ thỏa:

$$\textbf{FreqT}(i, j) = \text{số lần xuất hiện từ } t_i \text{ trong văn bản } d_j$$

Ví dụ về bảng tần số

DocID	String
d_8	Sex, Drugs, and Videotape.
d_9	The Iranian Connection.
d_{10}	Boating and Drugs: Slips owned by the Cali Cartel.

Term/Doc	d_8	d_9	d_{10}
sex	1	0	0
drug	1	0	1
videotape	1	0	0
iran	0	1	0
connect	0	1	0
boat	0	0	1
slip	0	0	1
own	0	0	1
cali	0	0	1
cartel	0	0	1

Một ví dụ khác về bảng tần số

Xem xét bảng tần số sau:

Term/Doc	d_1	d_2	d_3	d_4	d_5	d_6
t_1	615	390	10	10	18	65
t_2	15	4	76	217	91	816
t_3	2	8	815	142	765	1
t_4	312	511	677	11	711	2
t_5	45	33	516	64	491	59

- d_1, d_2 là tương tự nhau, bởi vì sự phân phối của các từ trong d_1 tương tự sự phân phối các từ trong d_2.
  - Cả hai đều có nhiều từ t_1, t_4, ít từ t_2, t_3 và một tần số vừa phải từ t_5.
- d_3 và d_5 cũng tương tự nhau.
- d_4 và d_6 khác nhau.

Bảng tần số

- Tuy nhiên, việc đếm một cách đơn giản các từ không chỉ ra được mức độ quan trọng của các từ trong văn bản.
  - **Ví dụ**: một từ có số lần xuất hiện như nhau trong 2 văn bản. Tuy nhiên, trong văn bản có độ dài ngắn hơn thì từ sẽ có ý nghĩa quan trọng hơn $\Rightarrow$ khái niệm **tf** *(term frequency)*.
  - **Ví dụ**: một từ xuất hiện trong rất ít văn bản sẽ có ý nghĩa quan trọng hơn một từ xuất hiện trong rất nhiều văn bản $\Rightarrow$ khái niệm **idf** *(inverse document frequency)*.

$\Rightarrow$ Để đánh giá mức độ quan trọng của các từ trong tài liệu, người ta không chỉ dựa vào số lần xuất hiện của các từ trong văn bản (thông tin cục bộ) mà còn dựa vào cả những thông tin toàn cục của hệ thống (CSDL văn bản).

- phương pháp tính trọng số **tf-idf**

TF-IDF

Việc tính trọng số các phần tử của tài liệu trong FreqT được tính theo công thức:

$$W_{ij} = \left(\frac{F_{ij}}{\sum_k F_{kj}} \right) \times \log \left(\frac{N}{DF_i} \right)$$

Trong đó:

- N là tổng số tài liệu.
- DF_i là số tài liệu có xuất hiện từ i.
- F_{ij} là tần số xuất hiện của từ i trong văn bản j

Ta có:

- $TF = \left(\frac{F_{ij}}{\sum_k F_{kj}} \right)$ *(term frequency)* thể hiện trọng số của từ i trong văn bản j.
- $IDF = \log \left(\frac{N}{DF_i} \right)$ *(inverse document frequency)* thể hiện sự quan trọng của từ i trong CSDL.

Ví dụ về TF-IDF

Ví dụ:

- Trong tài liệu d_1 có 100 từ, có xuất hiện 3 từ *car* $\rightarrow tf_{car,d_1} = 0.03$
- Giả sử CSDL có 10.000.000 tài liệu và từ *car* có xuất hiện trong 1.000 tài liệu $\rightarrow idf_{car} = 4$
- Vậy $\mathbf{tf - idf}_{car,d_1} = 0.03 \times 4 = 0.12$

Ví dụ về TF-IDF

Ví dụ:

- Trong tài liệu d_1 có 100 từ, có xuất hiện 3 từ *car* $\rightarrow tf_{car,d_1} = 0.03$
- Giả sử CSDL có 10.000.000 tài liệu và từ *car* có xuất hiện trong 1.000 tài liệu $\rightarrow idf_{car} = 4$
- Vậy $\mathbf{tf - idf}_{car,d_1} = 0.03 \times 4 = 0.12$

Ví dụ: bạn hãy tính các giá trị **tf-idf** cho bảng tần số sau:

term	d_1	d_2	d_3	df_t
car	27	4	24	18.165
auto	3	33	0	6.723
insurance	0	33	29	19.241
best	14	0	17	25.235

cho biết tổng số tài liệu là 806.791 tài liệu.

Xử lý câu truy vấn

- Giả sử người sử dụng đặt câu truy vấn Q: *Tìm 25 văn bản có liên quan nhiều nhất đến* **giao dịch ngân hàng** *và* **ma túy**.
- Câu truy vấn Q trên thực hiện truy tìm văn bản có liên quan đến 2 từ khóa mà từ gốc là: **drug** và **bank**.
- Nếu ta xem câu truy vấn Q cũng là một văn bản, khi đó ta có thể áp dụng các bước cắt từ, loại bỏ khoảng trắng và ký hiệu lạ, loại bỏ từ thông dụng, quy về từ gốc, ... $\Rightarrow$ Q có thể biểu diễn dưới dạng một vector cột.
- Chúng ta cần tìm các cột trong **FreqT** gần với vector cột gắn với Q.

Độ đo tương đồng

Là độ đo dùng để đánh giá mức độ giống nhau của 2 tài liệu.

Ta có các độ đo tương đồng sau:

1. **Độ đo khoảng cách giữa các từ** *(term distance)* giữa truy vấn Q và một văn bản d_r được cho bởi:

$$\sqrt{\sum_{j=1}^{M}(vec_Q(j) - \mathbf{FreqT}(j,r))^2}$$

2. **Độ đo khoảng cách cosine** *(cosine distance)*: đây là độ đo khoảng cách thường được sử dụng trong CSDL văn bản.

$$\frac{\sum_{j=1}^{M}(vec_Q(j) \times \mathbf{FreqT}(j,r))}{\sqrt{\sum_{j=1}^{M} vec_Q(j)^2} \times \sqrt{\sum_{j=1}^{M} \mathbf{FreqT}(j,r)^2})}$$

Trong đó $vec_Q(i)$ chỉ số lần xuất hiện của từ t_i trong truy vấn Q.

Mô hình tìm kiếm với toán tử Boolean

- Sử dụng các phép toán Boolean giữa các dòng có chứa từ khóa.
  - Google hỗ trợ các phép toán AND (mặc định), XOR (từ khóa OR), NOT (ký hiệu -)
  - **Ví dụ**: tin AND học ; tin OR học ; tin -học
- Sử dụng các trọng số để xếp hạng tài liệu tìm được.
- Tuy nhiên, việc tìm kiếm chính xác trên các từ khóa thường không mang lại kết quả tốt.
- Người ta thường phải mở rộng câu truy vấn để đáp ứng yêu cầu của người dùng.
- Ví dụ:

$$Q_i = \left(0.5 + 0.5 \times \frac{F_i}{\sum F_i}\right) \times \log\left(\frac{N}{DF_i}\right)$$

Trở ngại của việc tìm kiếm chính xác

1. Sự đồng nghĩa: tìm kiếm trên từ khóa **car** sẽ không thu được những tài liệu nói về **auto**.
2. Sự đa nghĩa: tìm kiếm trên từ khóa **Apple** sẽ trả ra cả các tài liệu nói về cây táo và nói về máy tính.

$\Rightarrow$ có thể sử dụng kỹ thuật *chỉ mục ngữ nghĩa tiềm ẩn* (LSI) để khắc phục.

Latent Semantic Indexing

- LSI là một kỹ thuật chỉ mục và truy xuất dữ liệu sử dụng kỹ thuật *phân tích giá trị riêng* (SVD) để xác định các mối quan hệ giữa các **từ** (terms) và **khái niệm** (concepts) có trong một tập văn bản phi cấu trúc. [1]
- LSI dựa trên nguyên lý là các từ được sử dụng trong cùng ngữ cảnh thường có xu hướng có nghĩa giống nhau.
- Đặc điểm nổi bật của LSI là khả năng rút ra các nội dung khái niệm từ văn bản thông qua việc thiết lập mối kết hợp giữa các từ xuất hiện trong các ngữ cảnh giống nhau $\Rightarrow$ LSI làm lộ ra các cấu trúc ngữ nghĩa tiềm ẩn nằm bên dưới của việc sử dụng từ trong văn bản.

[1] http://en.wikipedia.org/wiki/Latent_semantic_indexing

Kỹ thuật SVD

Kỹ thuật SVD là nền tảng của LSI.

SVD biến đổi ma trận tần suất *FreqT* (kích thước $m \times n$, hạng ma trận là r) thành tích của 3 ma trận:

- ma trận từ - khái niệm (term - concept) **T**, là ma trận trực giao kích thước $m \times r$.
- ma trận giá trị riêng **S**, là ma trận chéo không tăng kích thước $r \times r$.
- ma trận khái niệm - tài liệu (concept - document) **D**, là ma trận trực giao kích thước $n \times r$.

$$FreqT = T \times S \times D^T$$

Nhắc lại về ma trận

- Một ma trận **M** được gọi là *trực giao* nếu

$$\mathbf{M}^T \times \mathbf{M} = \mathbf{I}$$

  - **Ví dụ**: $C = \begin{bmatrix} 1 & 1 \\ 0 & 0 \end{bmatrix}$

- Ma trận **M** có kích thước $(m \times m)$ được gọi là *ma trận chéo* nếu với mọi $1 \leq i, j \leq m$:

$$i \neq j \rightarrow \mathbf{M}(i, j) = 0$$

  - **Ví dụ**: $A = \begin{bmatrix} 5 & 0 & 0 \\ 0 & 4 & 0 \\ 0 & 0 & 1 \end{bmatrix} \qquad B = \begin{bmatrix} 0 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 0 \end{bmatrix}$

- Ma trận **M** có kích thước $(m \times m)$ được gọi là *ma trận không tăng (non-increasing)* nếu với mọi $1 \leq i, j \leq m$:

$$i \leq j \rightarrow \mathbf{M}(i, i) \geq \mathbf{M}(j, j)$$

  - **Ví dụ**: A là ma trận không tăng.

Ví dụ về SVD

Cho ma trận:

$$FreqT = \begin{bmatrix} 1.44 & .52 \\ .92 & 1.44 \end{bmatrix}$$

Hãy phân tích SVD ma trận trên.

Ví dụ về SVD

Cho ma trận:

$$FreqT = \begin{bmatrix} 1.44 & .52 \\ .92 & 1.44 \end{bmatrix}$$

Hãy phân tích SVD ma trận trên.

$$FreqT = \begin{bmatrix} -.66 & -.75 \\ -.75 & .66 \end{bmatrix} \times \begin{bmatrix} 2.17 & 0 \\ 0 & .73 \end{bmatrix} \times \begin{bmatrix} -.75 & -.66 \\ -.66 & .75 \end{bmatrix}$$

Rút gọn ma trận

- Sau khi áp dụng SVD lên ma trận *FreqT*, ta có thể chỉ cần giữ lại k giá trị riêng lớn nhất trong ma trận **S**. Thông thường $k = 100..300$ và $k \ll r$.
- Việc giữ lại k giá trị riêng lớn nhất trong **S** sẽ bảo toàn các thông tin ngữ nghĩa quan trọng nhất trong văn bản, đồng thời loại bỏ nhiễu và các hiệu ứng không mong muốn khác trong ma trận gốc *FreqT*.
- Việc chỉ giữ lại k giá trị riêng lớn nhất trong **S** cũng đồng thời giúp làm giảm rất đáng kể kích thước của 2 ma trận **T** và **D** ($T_{m\times k}$ và $D_{n\times k}$).
- Ta tính lại ma trận tần suất mới:

$$FreqT_k = T_{m\times k} \times S_{k\times k} \times D^T_{n\times k}$$

Ví dụ về rút gọn ma trận

Giả sử ma trận **FreqT** có SVD:

$$\begin{pmatrix} a_1^1 & a_2^1 & a_3^1 & a_4^1 & a_5^1 \\ a_1^2 & a_2^2 & a_3^2 & a_4^2 & a_5^2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ a_1^M & a_2^M & a_3^M & a_4^M & a_5^M \end{pmatrix} \begin{pmatrix} 20 & 0 & 0 & 0 & 0 \\ 0 & 16 & 0 & 0 & 0 \\ 0 & 0 & 12 & 0 & 0 \\ 0 & 0 & 0 & 0.08 & 0 \\ 0 & 0 & 0 & 0 & 0.004 \end{pmatrix} \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 & \cdots & b_N^1 \\ b_1^2 & b_2^2 & b_3^2 & \cdots & b_N^2 \\ \cdots & \cdots & \cdots & \cdots & \cdots \\ b_1^5 & b_2^5 & b_3^5 & \cdots & b_N^5 \end{pmatrix}$$

Chúng ta giữ lại 3 giá trị riêng lớn nhất:

$$\begin{pmatrix} a_1^1 & a_2^1 & a_3^1 \\ a_1^2 & a_2^2 & a_3^2 \\ \cdots & \cdots & \cdots \\ a_1^M & a_2^M & a_3^M \end{pmatrix} \begin{pmatrix} 20 & 0 & 0 \\ 0 & 16 & 0 \\ 0 & 0 & 12 \end{pmatrix} \begin{pmatrix} b_1^1 & b_2^1 & b_3^1 & \cdots & b_N^1 \\ b_1^2 & b_2^2 & b_3^2 & \cdots & b_N^2 \\ b_1^3 & b_2^3 & b_3^3 & \cdots & b_N^3 \end{pmatrix}$$

Phân tích

- Thông thường, trong một CSDL văn bản, số lượng từ m và số lượng văn bản n là rất lớn.
  - trong tiếng Anh, số lượng từ m có thể lên đến trên 100.000 từ (bao gồm cả tên riêng).
  - trong một CSDL văn bản, số lượng văn bản n có thể có trên 1.000.000 văn bản.
- Với kỹ thuật SVD trên, LSI cho phép tìm một tập con tương đối nhỏ k *khái niệm* sao cho vẫn giúp phân biệt được n văn bản tốt nhất.
- Trên thực tế, người ta nhận thấy LSI hoạt động hiệu quả với $k \approx 200$.
- **Ưu điểm**: mỗi văn bản lúc này được biểu diễn bằng một vector cột có kích thước 200, thay vì kích thước m.

Phân tích

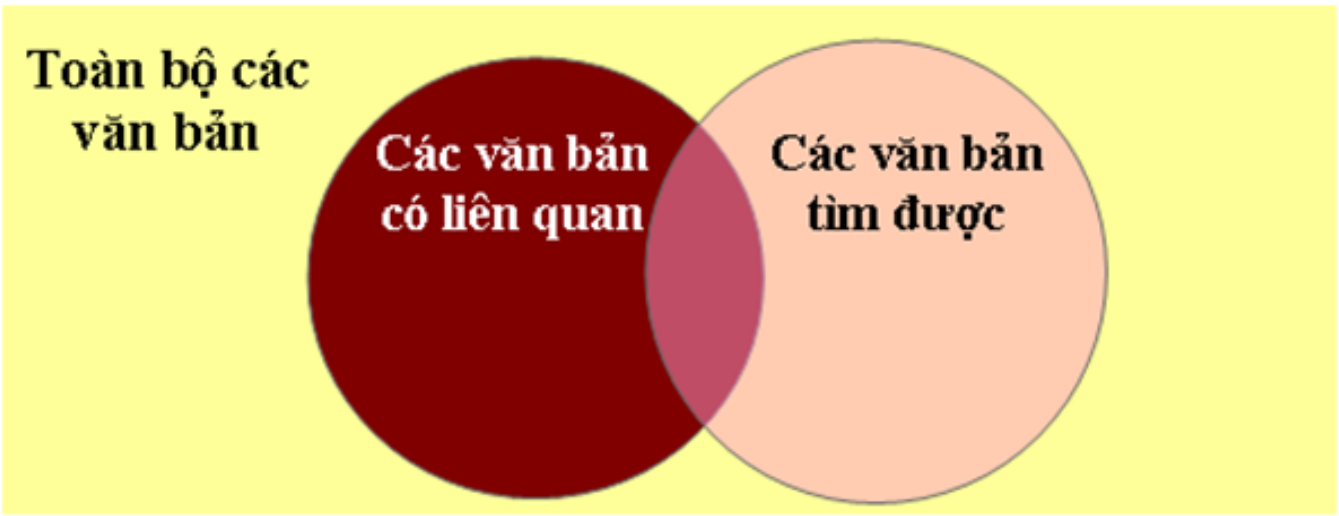
- Thông thường, k được chọn là 200.
- Kích thước của bảng tần suất gốc thường rất lớn. Cho dù chỉ xử lý trên một CSDL văn bản nhỏ, chúng ta vẫn dễ dàng đạt được $m = 10.000$ từ và $n = 1.000.000$ văn bản.
- Kích thước của 3 ma trận sau khi rút gọn số chiều là:
  - Kích thước ma trận đầu tiên là $(m \times k) \approx (10.000 \times 200)$
  - Kích thước ma trận suy biến là $(k \times k) \approx (200 \times 200)$
  - Kích thước ma trận cuối là $(n \times k) \approx (1.000.000 \times 200)$
- Như vậy, chúng ta chỉ xử lý và tính toán trên khoảng 200 triệu ô ma trận, thay vì xử lý trên một ma trận có kích thước $(m \times n) \approx (10.000 \times 1.000.000)$ hay 10 tỉ ô.

4 bước của LSI

1. **Tạo bảng**: tạo ma trận tần suất **FreqT**.
2. **Phân tích SVD**: tính các phân tích giá trị riêng (T, S, D) từ ma trận **FreqT**.
3. **Xác định tập vector**: với mỗi văn bản d, ta có $vec(d)$ là vector tương ứng của văn bản d trong ma trận D.
4. **Tạo chỉ mục**: lưu trữ tập các $vec(d)$ và lập chỉ mục bởi một trong nhiều kỹ thuật chỉ mục.

Hiệu quả tìm kiếm thông tin

- Khi tìm kiếm một câu truy vấn do người dùng gõ vào, kết quả trả về không hẳn lúc nào cũng như ý người dùng mong muốn.
- Các kết quả có thể rơi vào những trường hợp đặc biệt như: văn bản có liên quan nhưng không tìm được, văn bản tìm được nhưng không liên quan.



Precision và Recall

Giả sử:

- *D* là tập hữu hạn các văn bản.
- *A* là một thuật toán có tham số đầu vào là chuỗi chủ đề *T* và trả về một tập văn bản $\mathbb{T}$.

$$\begin{aligned} \textit{Precision} &= \frac{\text{số tài liệu tìm đúng}}{\text{số tài liệu tìm được}} \\ &= \frac{\text{số tài liệu tìm được có liên quan đến chủ đề T trong } \mathbb{T}}{\text{số tài liệu tìm được trong } \mathbb{T}} \end{aligned}$$

$$\begin{aligned} \textit{Recall} &= \frac{\text{số tài liệu tìm đúng}}{\text{số tài liệu đúng trong CSDL}} \\ &= \frac{\text{số tài liệu tìm được có liên quan đến chủ đề T trong } \mathbb{T}}{\text{số tài liệu có liên quan đến chủ đề T trong CSDL}} \end{aligned}$$

Ví dụ về Precision và Recall

Trong một CSDL văn bản có:

- 1000 văn bản
- 600 văn bản liên quan đến chủ đề T

Áp dụng một giải thuật truy vấn văn bản, ta thu được:

- 750 văn bản
- 450 văn bản liên quan đến chủ đề T

Ta có:

$$P_t = \frac{450}{750} = 60\%$$

$$R_t = \frac{450}{600} = 75\%$$

Tài liệu tham khảo

V.S.Subrahmanian.
Chapter 6 - Principles of Multimedia Database Systems
Morgan Kaufman Prbbess, 1998.

Nabil R. Adam.
Bài giảng Multimedia Information Systems
Rutgers University, 2003.