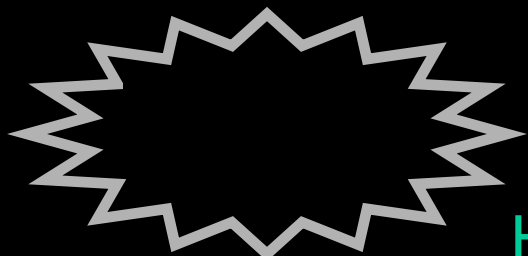


Lecture 4

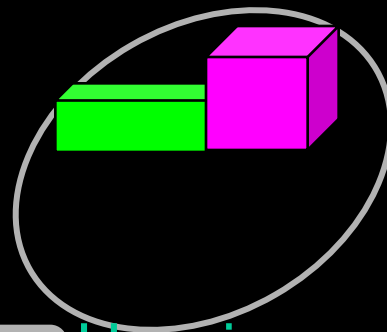
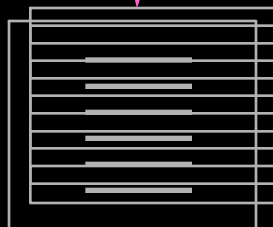
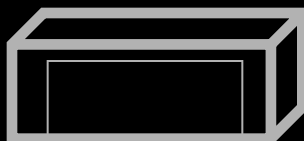
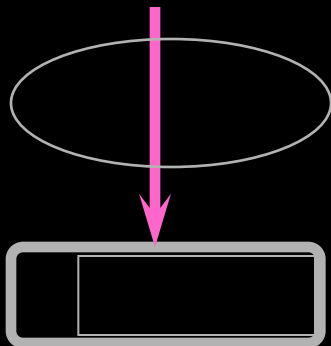
IR Modeling

Traditional IR Models

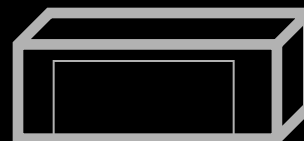
- Content Analysis
 - Statistical Characteristics of Text Collections
 - Zipf distribution
 - Statistical dependence
 - Term Vector Representations
- Inverted Indexes
- Ranking



How is
the query
constructed?



How is
the text
processed?



A Taxonomy of IR Models

- Boolean Model(Set Theory)
 - Extended Boolean model
 - Fuzzy model
- Vector Model(Algebraic Theory)
 - Generalized vector model
 - Latent semantic index
 - Neural networks
- Probabilistic Model
 - Inference network
 - Belief network

Basic Concepts of Classic IR

- index terms (usually nouns): index and summarize
- weight of index terms
- Definition
 - $K = \{k_1, \dots, k_t\}$: a set of all index terms
 - $w_{i,j}$: a weight of an index term k_i of a document d_j
 - $\vec{d}_j = (w_{1,j}, w_{2,j}, \dots, w_{t,j})$: an *index term vector* for the document d_j
 - $g_i(\vec{d}_j) = w_{i,j}$
- assumption
 - index term weights are *mutually independent*

$w_{i,j}$ associated with (k_i, d_j) tells us nothing about $w_{i+1,j}$ associated with (k_{i+1}, d_j)

The terms *computer* and *network* in the area of computer networks

Boolean Model

- The index term weight variables are all binary, i.e., $w_{i,j} \in \{0,1\}$
- A query q is a Boolean expression (and, or, not)
- $\vec{q}_{dnf}$: the *disjunctive normal form* for q
- $\vec{q}_{cc}$: conjunctive components of $\vec{q}_{dnf}$
- $\text{sim}(d_j, q)$: similarity of d_j to q
 - 1: if $\exists \vec{q}_{cc} \mid (\vec{q}_{cc} \in \vec{q}_{dnf} \wedge (\forall k_i, g_i(\vec{d}_j) = g_i(\vec{q}_{cc})))$
 - 0: otherwise

d_j is relevant to q

Boolean Model *(Continued)*

- advantage: simple
- disadvantage
 - binary decision (relevant or non-relevant) without $\vec{}$ grading scale
 - exact match (no partial match)
 - e.g., $d_j=(0,1,0)$ is non-relevant to $q=(k_a \wedge (k_b \vee \neg k_c))$
 - retrieve too few or too many documents

Basic Vector Space Model

- *Term vector* representation of
documents $D_i = (a_{i1}, a_{i2}, \dots, a_{it})$
queries $Q_j = (q_{j1}, q_{j2}, \dots, q_{jt})$
- t distinct terms are used to characterize content.
- Each term is identified with a term vector T .
- t vectors are linearly independent.
- Any vector is represented as a linear combination of the t term vectors.
- The r th document D_r can be represented as a document vector, written as

$$D_r = \sum_{i=1}^t a_{ri} T_i$$

Document Collection

- A collection of n documents can be represented in the vector space model by a term-document matrix.
- An entry in the matrix corresponds to the “weight” of a term in the document; zero means the term has no significance in the document or it simply doesn't exist in the document.

$$\begin{pmatrix} & \mathbf{T}_1 & \mathbf{T}_2 & \dots & \mathbf{T}_t \\ \mathbf{D}_1 & w_{11} & w_{21} & \dots & w_{t1} \\ \mathbf{D}_2 & w_{12} & w_{22} & \dots & w_{t2} \\ \vdots & \vdots & \vdots & & \vdots \\ \vdots & \vdots & \vdots & & \vdots \\ \mathbf{D}_n & w_{1n} & w_{2n} & \dots & w_{tn} \end{pmatrix}$$

Documents as vectors

- So we have a $|V|$ -dimensional vector space
- Terms are axes of the space
- Documents are points **or vectors** in this space
- Very high-dimensional: hundreds of millions of dimensions when you apply this to a web search engine
- This is a very sparse vector - most entries are zero. $d_j = (w_{1j}, w_{2j}, \dots, w_{tj})$

Queries as vectors

- Key idea 1: Do the same for queries: represent them as vectors in the space
- Key idea 2: Rank documents according to their proximity to the query in this space
- proximity = similarity of vectors
- proximity $\approx$ inverse of distance
- Recall: We do this because we want to get away from the you're-either-in-or-out Boolean model.
- Instead: rank **more relevant** documents higher than less relevant documents

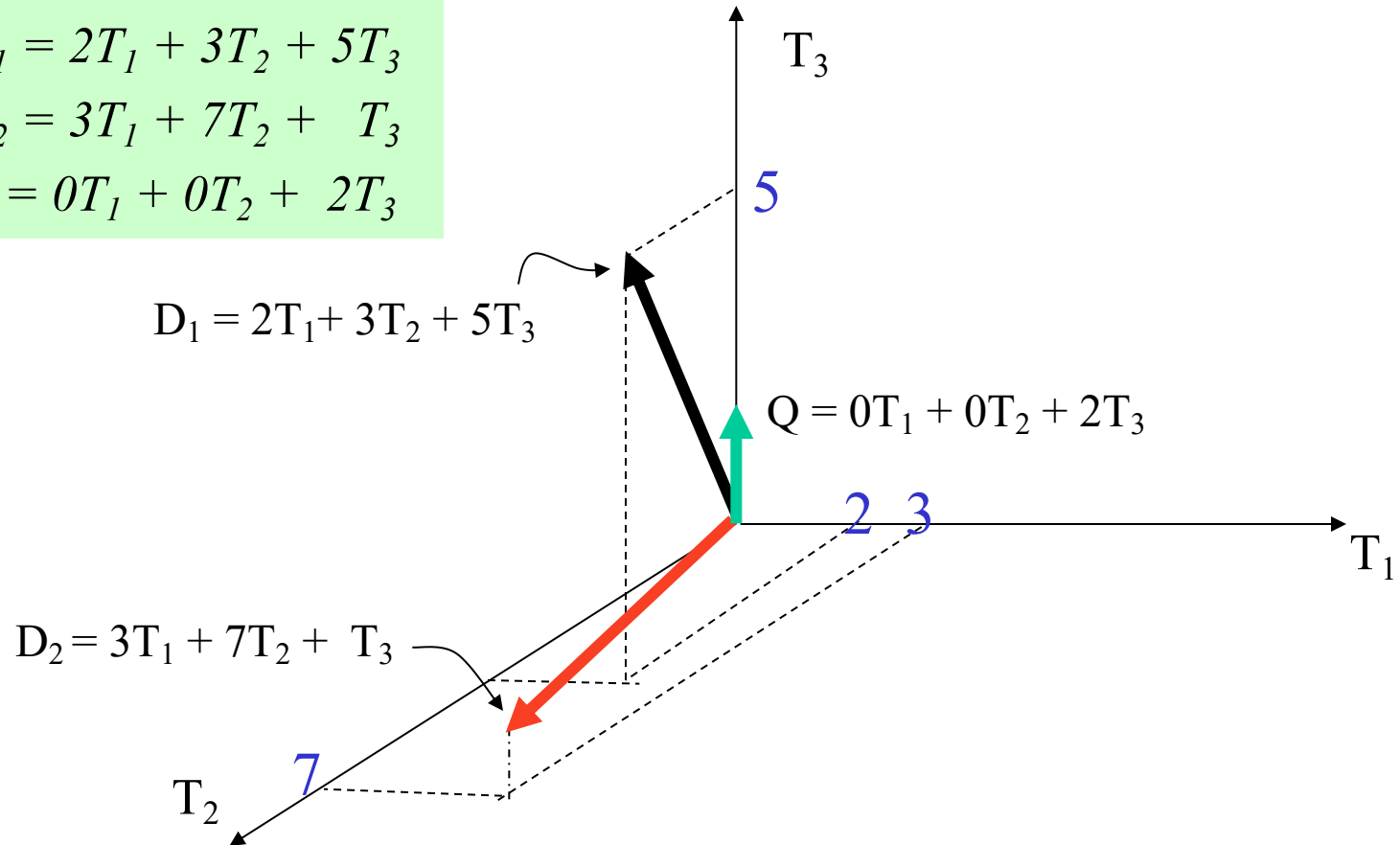
Graphic Representation

Example:

$$D_1 = 2T_1 + 3T_2 + 5T_3$$

$$D_2 = 3T_1 + 7T_2 + T_3$$

$$Q = 0T_1 + 0T_2 + 2T_3$$



Rank Retrieval over Vector Space

- Retrieval based on *similarity* between query and documents.
- Output documents are ranked according to *similarity* to query.

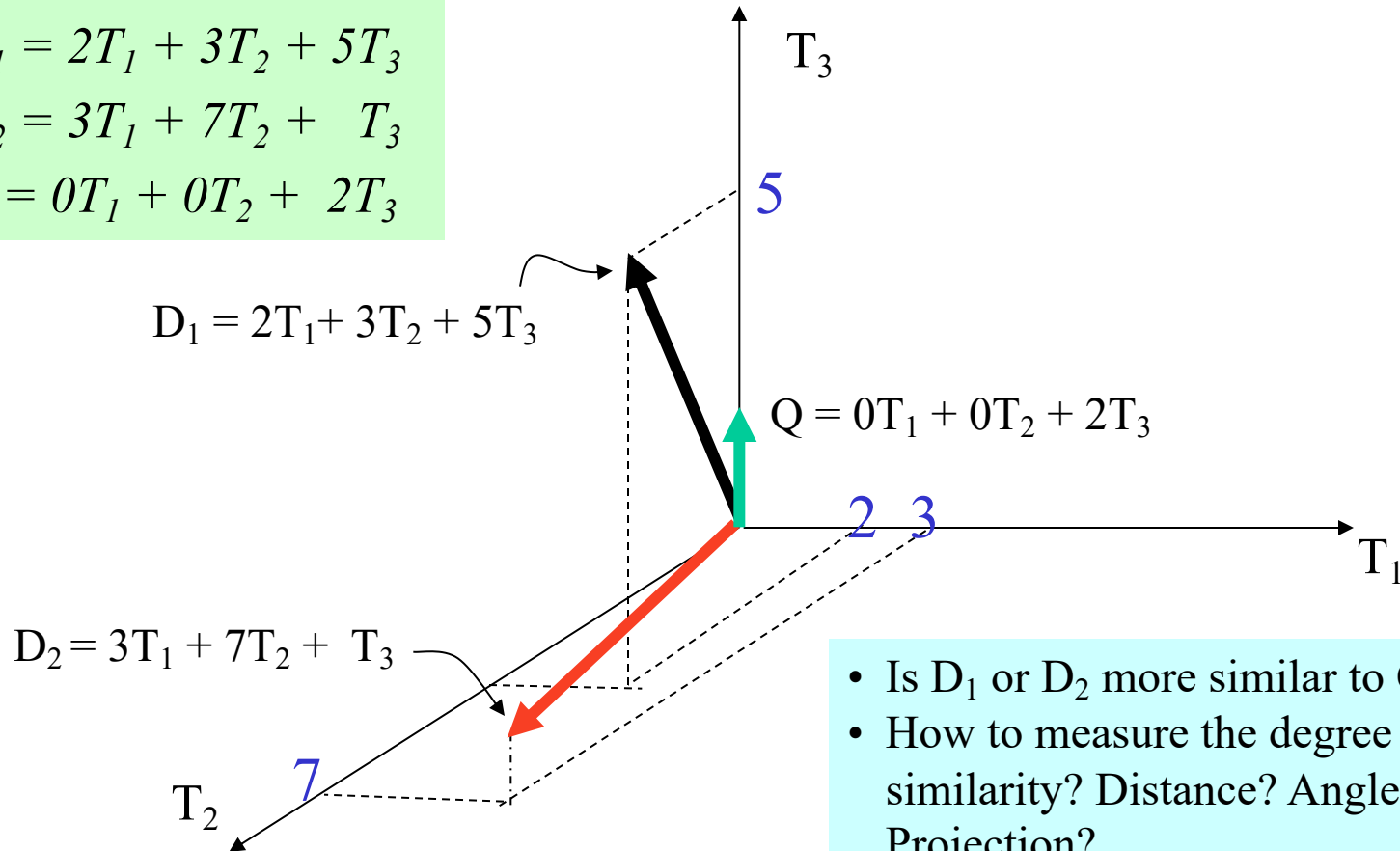
Graphic Representation

Example:

$$D_1 = 2T_1 + 3T_2 + 5T_3$$

$$D_2 = 3T_1 + 7T_2 + T_3$$

$$Q = 0T_1 + 0T_2 + 2T_3$$



- Is D_1 or D_2 more similar to Q ?
- How to measure the degree of similarity? Distance? Angle? Projection?

Similarity Measure

- A **similarity measure** is a function that computes the *degree of similarity* between two vectors.
- Using a similarity measure between the query and each document:
 - It is possible to rank the retrieved documents in the order of presumed relevance.
 - It is possible to enforce a certain threshold so that **the size of the retrieved set can be controlled**.

Similarity Measure

- measure by product of two vectors

$$x \cdot y = |x| |y| \cos \alpha$$

- document-query similarity

document vector:

$$D_r = \sum_{i=1}^t a_{ri} T_i$$

term vector:

$$Q_s = \sum_{j=1}^t q_{sj} T_j$$

$$D_r \text{ 与 } Q_s = \sum_{i,j=1}^t a_{ri} q_{sj} T_i \text{ 与 } T_j$$

- how to determine the vector components and term correlations?

Similarity Measure *(Continued)*

- vector components

$$A = \begin{matrix} & T_1 & T_2 & T_3 & T_t \\ \begin{matrix} D_1 \\ D_2 \\ \vdots \\ D_n \end{matrix} & \left| \begin{matrix} a_{11} & a_{12} & \cdots & a_{1t} \\ a_{21} & a_{22} & \cdots & a_{2t} \\ \vdots & \vdots & \vdots & \vdots \\ a_{n1} & a_{n2} & \cdots & a_{nt} \end{matrix} \right| \end{matrix}$$

Similarity Measure *(Continued)*

- term correlations $T_i \cdot T_j$ are not available
assumption: term vectors are orthogonal

$$T_i \cdot T_j = 0 \quad (i \neq j) \quad T_i \cdot T_j = 1 \quad (i = j)$$

- Assume that terms are uncorrelated.

$$\text{sim}(D_r, Q_s) = \sum_{i,j=1}^t a_{ri} q_{sj}$$

- Similarity measurement between documents

$$\text{sim}(D_r, D_s) = \sum_{i,j=1}^t a_{ri} a_{sj}$$

Simple query-document similarity computation

- $D_1=2T_1+3T_2+5T_3$ $D_2=3T_1+7T_2+1T_3$
 $Q=0T_1+0T_2+2T_3$
- similarity computations for uncorrelated terms
 $\text{sim}(D_1, Q)=2 \cdot 0 + 3 \cdot 0 + 5 \cdot 2 = 10$
 $\text{sim}(D_2, Q)=3 \cdot 0 + 7 \cdot 0 + 1 \cdot 2 = 2$
- D_1 is preferred

Simple query-document similarity computation (*Continued*)

- | | | | |
|-------|-------|-------|-------|
| | T_1 | T_2 | T_3 |
| T_1 | 1 | 0.5 | 0 |
| T_2 | 0.5 | 1 | -0.2 |
| T_3 | 0 | -0.2 | 1 |
- similarity computations for correlated terms

$$\begin{aligned} \text{sim}(D_1, Q) &= (2T_1 + 3T_2 + 5T_3) \cdot (0T_1 + 0T_2 + 2T_3) \\ &= 4T_1 \cdot T_3 + 6T_2 \cdot T_3 + 10T_3 \cdot T_3 \\ &= -6 * 0.2 + 10 * 1 = 8.8 \end{aligned}$$

$$\begin{aligned} \text{sim}(D_2, Q) &= (3T_1 + 7T_2 + 1T_3) \cdot (0T_1 + 0T_2 + 2T_3) \\ &= 6T_1 \cdot T_3 + 14T_2 \cdot T_3 + 2T_3 \cdot T_3 \\ &= -14 * 0.2 + 2 * 1 = -0.8 \end{aligned}$$
- D_1 is preferred

Vector Model

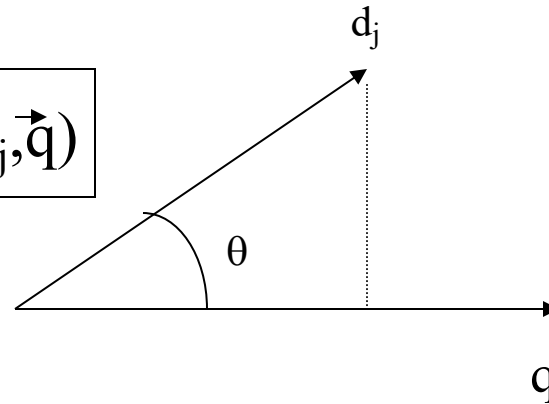
- $w_{i,j}$: a positive, *non-binary weight* for (k_i, d_j)
- $w_{i,q}$: a positive, *non-binary weight* for (k_i, q)
- $\vec{q} = (w_{1,q}, w_{2,q}, \dots, w_{t,q})$: a query vector,
where t is the total number of index terms in
the system
- $\vec{d}_j = (w_{1,j}, w_{2,j}, \dots, w_{t,j})$: a document vector

Similarity of document d_j w.r.t. query q

- The correlation between vectors $\vec{d}_j$ and $\vec{q}$

$$sim(d_j, q) = \frac{\vec{d}_j \bullet \vec{q}}{|\vec{d}_j| \times |\vec{q}|} \quad \boxed{\cos(\vec{d}_j, \vec{q})}$$

$$= \frac{\sum_{i=1}^t w_{i,j} \times w_{i,q}}{\sqrt{\sum_{i=1}^t w_{i,j}^2} \times \sqrt{\sum_{j=1}^t w_{i,q}^2}}$$



- $|\vec{q}|$ does not affect the ranking
- $|\vec{d}_j|$ provides a normalization

document ranking

- Similarity (*i.e.*, $\text{sim}(q, d_j)$) between 0 and 1.
- Retrieve the documents with a degree of similarity above a predefined threshold (allow partial matching)

Term weighting techniques

- IR problem vs Classification problem:
 - user query: a specification of a set A of objects
 - classification problem: determine which documents are in the set A (*relevant*), which ones are not (*non-relevant*)
 - intra-cluster similarity
 - the features better describe the objects in the set A
 - tf factor in vector model
the raw frequency of a term k_i inside a document d_j
 - inter-cluster similarity
 - the features better distinguish the the objects in the set A from the remaining objects in the collection C
 - idf factor (inverse document frequency) in vector model
the inverse of the frequency of a term k_i among the documents in the collection

Definition of tf

- N : total number of documents in the system
- n_i : the number of documents in which the index term k_i appears
- $freq_{i,j}$: the raw frequency of term k_i in the document d_j
- $f_{i,j}$: the *normalized frequency* of term k_i in document d_j

$$f_{i,j} = \frac{freq_{i,j}}{\max_l freq_{l,j}} \quad (0 \sim 1)$$

Term t_l has maximum frequency
in the document d_j

Definition of *idf* and *tf-idf* scheme

- idf_i : inverse document frequency for k_i

$$idf_i = \log \frac{N}{n_i}$$

- $w_{i,j}$: term-weighting by *tf-idf* scheme

$$w_{i,j} = f_{i,j} \times \log \frac{N}{n_i}$$

- *query term* weight (Salton and Buckley)

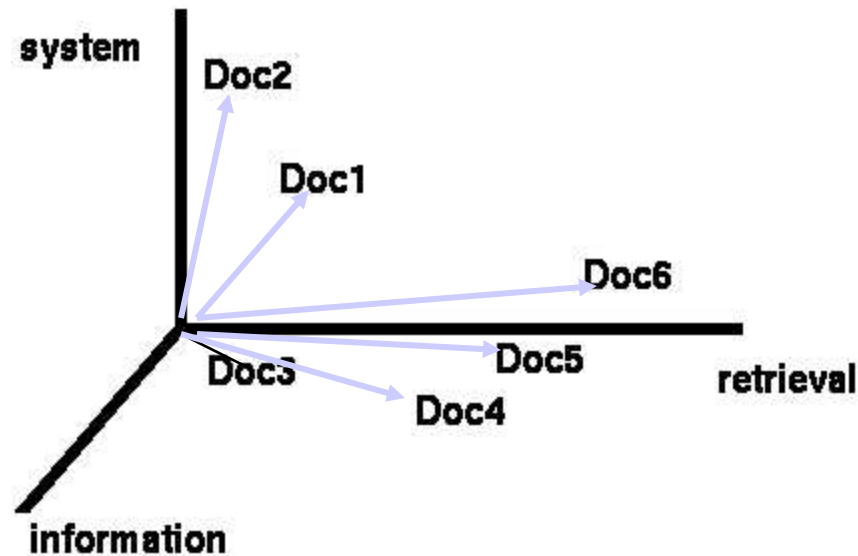
$$w_{i,q} = \left(0.5 + \frac{0.5 \text{freq}_{i,q}}{\max_l \text{freq}_{l,q}}\right) \times \log \frac{N}{n_i}$$

$\text{freq}_{i,q}$: the raw frequency of the term k_i in q

Analysis of vector model

- advantages
 - its *term-weighting* scheme improves *retrieval performance*
 - its *partial matching* strategy allows retrieval of documents that *approximate* the query conditions
 - its *cosine ranking* formula sorts the documents according to their *degree of similarity* to the query
- disadvantages
 - indexed terms are assumed to be *mutually independently*

Documents in 3D Space

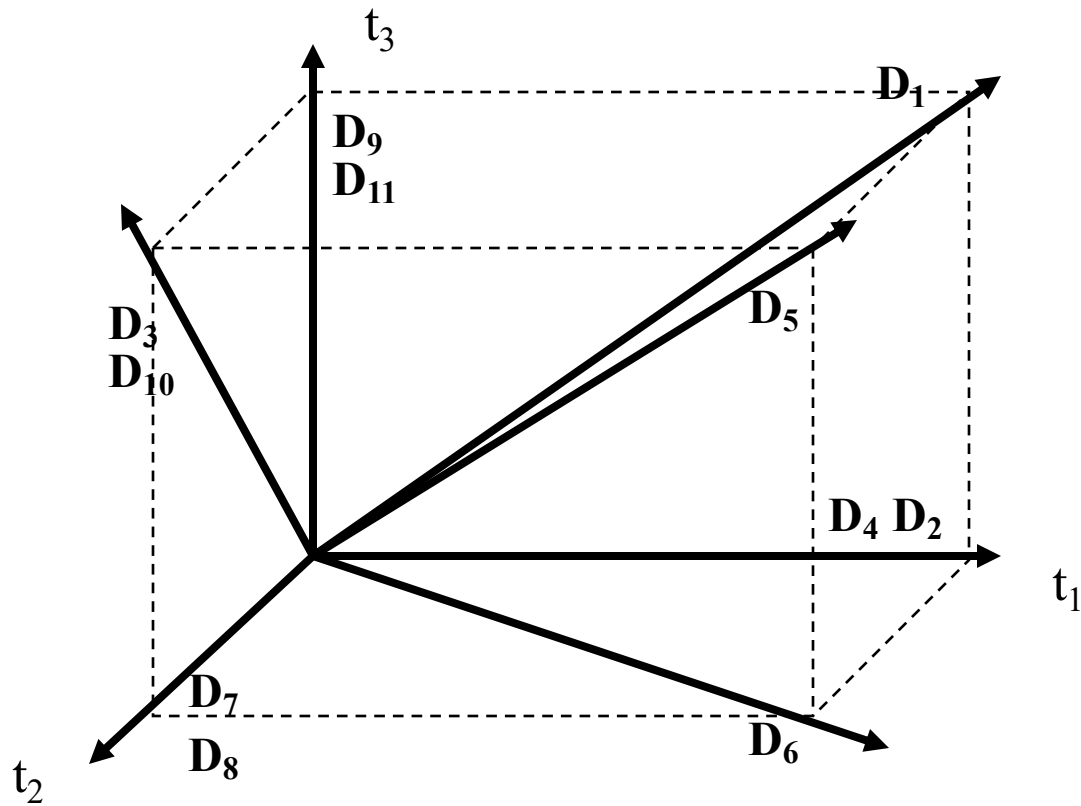


Assumption: Documents that are “close together” in space are similar in meaning.

Vector Space Model

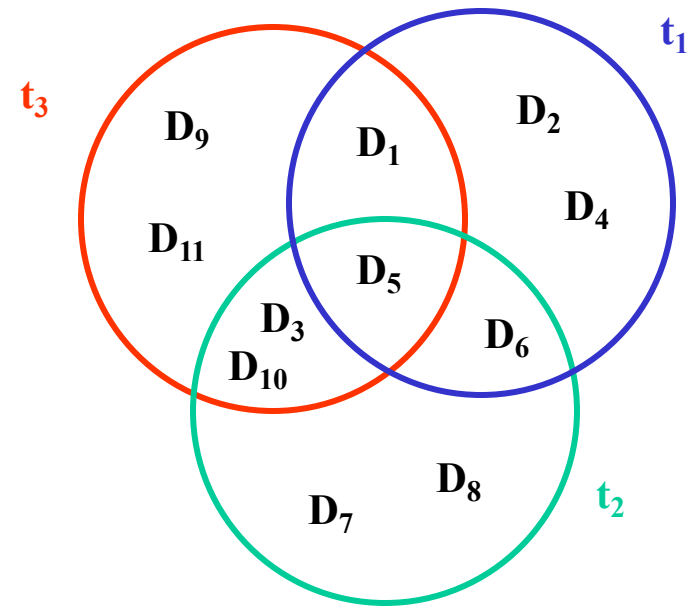
- Documents are represented as vectors in term space
 - Terms are usually stems
 - Documents represented by binary vectors of terms
- Queries represented the same as documents
- Query and Document weights are based on length and direction of their vector
- A vector distance measure between the query and documents is used to rank retrieved documents

Documents in Vector Space



Vector Space Documents and Queries

<i>docs</i>	<i>t1</i>	<i>t2</i>	<i>t3</i>	RSV=Q.Di
D1	1	0	1	4
D2	1	0	0	1
D3	0	1	1	5
D4	1	0	0	1
D5	1	1	1	6
D6	1	1	0	3
D7	0	1	0	2
D8	0	1	0	2
D9	0	0	1	3
D10	0	1	1	5
D11	0	0	1	3
<i>Q</i>	1	2	3	
	<i>q1</i>	<i>q2</i>	<i>q3</i>	



Similarity Measures

$$|Q \cap D|$$

Simple matching (coordination level match)

$$2 \frac{|Q \cap D|}{|Q| + |D|}$$

Dice's Coefficient

$$\frac{|Q \cap D|}{|Q \cup D|}$$

Jaccard's Coefficient

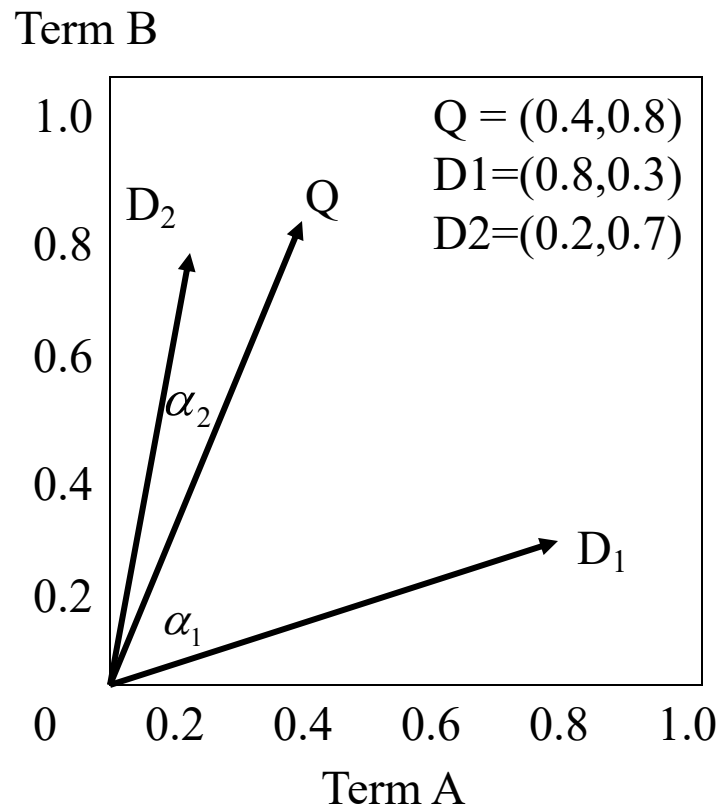
$$\frac{|Q \cap D|}{|Q|^{\frac{1}{2}} \times |D|^{\frac{1}{2}}}$$

Cosine Coefficient

$$\frac{|Q \cap D|}{\min(|Q|, |D|)}$$

Overlap Coefficient

Vector Space with Term Weights and Cosine Matching



$$D_i = (d_{i1}, w_{di1}; d_{i2}, w_{di2}; \dots; d_{it}, w_{dit})$$

$$Q = (q_{i1}, w_{qi1}; q_{i2}, w_{qi2}; \dots; q_{it}, w_{qit})$$

$$\text{sim}(Q, D_i) = \frac{\sum_{j=1}^t w_{q_j} w_{d_{ij}}}{\sqrt{\sum_{j=1}^t (w_{q_j})^2 \sum_{j=1}^t (w_{d_{ij}})^2}}$$

$$\begin{aligned} \text{sim}(Q, D2) &= \frac{(0.4 \cdot 0.2) + (0.8 \cdot 0.7)}{\sqrt{[(0.4)^2 + (0.8)^2] \cdot [(0.2)^2 + (0.7)^2]}} \\ &= \frac{0.64}{\sqrt{0.42}} = 0.98 \end{aligned}$$

$$\text{sim}(Q, D_1) = \frac{.56}{\sqrt{0.58}} = 0.74$$

Probabilistic Model

- Given a query, there is an *ideal answer set*
 - a set of documents which contains exactly the relevant documents and no other
- query process
 - a process of specifying *the properties* of an ideal answer set
- problem: what are the properties?

Probabilistic Principle

- Given a *user query* q and a *document* d_j in the collection, the probabilistic model estimates the probability that user will find d_j relevant
- assumptions
 - The probability of relevance depends on query and document representations only
 - There is a subset of all documents which the user prefers as the answer set for the query q
- Given a query, the probabilistic model assigns to each document d_j a measure of its similarity to the query

$$\frac{P(d_j \text{ relevant} - \text{to } q)}{P(d_j \text{ nonrelevant} - \text{to } q)}$$

Probabilistic Principle

- $w_{i,j} \in \{0,1\}$, $w_{i,q} \in \{0,1\}$: the index term weight variables are all binary non-relevant
- q : a query which is a subset of index terms
- R : the set of documents known to be *relevant*
- $\overline{R}$: the set of documents known to be *non-relevant*
- $P(R|\vec{d_j})$: the probability that the document d_j is *relevant* to the query q
- $P(\overline{R}|\vec{d_j})$: the probability that d_j is *non-relevant* to q

Similarity

- $\text{sim}(d_j, q)$: the similarity of the document d_j to the query q

$$\text{sim}(d_j, q) = \frac{P(R | \vec{d_j})}{P(\bar{R} | \vec{d_j})} \quad (\text{by definition})$$

$$\text{sim}(d_j, q) = \frac{P(\vec{d_j} | R) \times P(R)}{P(\vec{d_j} | \bar{R}) \times P(\bar{R})} \quad (\text{Bayes' rule})$$

$$\text{sim}(d_j, q) \approx \frac{P(\vec{d_j} | R)}{P(\vec{d_j} | \bar{R})} \quad (P(R) \text{ and } P(\bar{R}) \text{ are the same for all documents})$$

$P(\vec{d_j} | R)$: the probability of randomly selecting the document d_j from the set of R of relevant documents

$P(R)$: the probability that a document randomly selected from the entire collection is relevant

$$\text{sim}(d_j, q) \approx \frac{P(\overline{d_j} | R)}{P(\overline{d_j} | \overline{R})}$$

$$= \log \frac{\prod_{i=1}^t (P(k_i | R))^{g_i(\overline{d_j})} \times (P(\overline{k_i} | R))^{1-g_i(\overline{d_j})}}{\prod_{i=1}^t (P(k_i | \overline{R}))^{g_i(\overline{d_j})} \times (P(\overline{k_i} | \overline{R}))^{1-g_i(\overline{d_j})}}$$

$$= \sum_{i=1}^t \log \frac{(P(k_i | R))^{g_i(\overline{d_j})} \times (P(\overline{k_i} | R))^{1-g_i(\overline{d_j})}}{(P(k_i | \overline{R}))^{g_i(\overline{d_j})} \times (P(\overline{k_i} | \overline{R}))^{1-g_i(\overline{d_j})}}$$

$$= \sum_{i=1}^t \log \frac{(P(k_i | R) \times P(\overline{k_i} | \overline{R}))^{g_i(\overline{d_j})} \times (P(\overline{k_i} | R))}{(P(k_i | \overline{R}) \times P(\overline{k_i} | R))^{g_i(\overline{d_j})} \times (P(\overline{k_i} | \overline{R}))}$$

$$= \sum_{i=1}^t g_i(\overline{d_j}) \times \log \frac{P(k_i | R) \times P(\overline{k_i} | \overline{R})}{P(k_i | \overline{R}) \times P(\overline{k_i} | R)} + \sum_{i=1}^t \frac{P(\overline{k_i} | R)}{P(\overline{k_i} | \overline{R})}$$

$$= \sum_{i=1}^t g_i(\overline{d_j}) \times \log \frac{P(k_i | R) \times (1 - P(k_i | \overline{R}))}{P(k_i | \overline{R}) \times (1 - P(k_i | R))} + \sum_{i=1}^t \frac{P(\overline{k_i} | R)}{P(\overline{k_i} | \overline{R})}$$

$P(k_i | R)$: the probability that the index term k_i is present in a document randomly selected from the set R .

$P(\overline{k_i} | R)$: the probability that the index term k_i is **not** present in a document randomly selected from the set R .

independence assumption of index terms

$$\begin{aligned}
sim(d_j, q) &\approx \frac{P(\overrightarrow{d_j} | R)}{P(\overrightarrow{d_j} | \overline{R})} \\
&= \sum_{i=1}^t g_i(\overrightarrow{d_j}) \times \log \frac{P(k_i | R) \times (1 - P(k_i | \overline{R}))}{P(k_i | \overline{R}) \times (1 - P(k_i | R))} + \sum_{i=1}^t \frac{P(\overline{k_i} | R)}{P(\overline{k_i} | \overline{R})} \\
&= \sum_{i=1}^t g_i(\overrightarrow{d_j}) \times \left(\log \frac{P(k_i | R)}{(1 - P(k_i | R))} \right) + \log \frac{(1 - P(k_i | \overline{R}))}{P(k_i | \overline{R})} + \sum_{i=1}^t \frac{P(\overline{k_i} | R)}{P(\overline{k_i} | \overline{R})} \\
&\approx \sum_{i=1}^t g_i(\overrightarrow{d_j}) \times \left(\log \frac{P(k_i | R)}{(1 - P(k_i | R))} \right) + \log \frac{(1 - P(k_i | \overline{R}))}{P(k_i | \overline{R})}
\end{aligned}$$

Problem: where is the set R?

Initial guess

- $P(k_i|R)$ is constant for all index terms k_i .

$$p(k_i | R) = 0.5$$

- The distribution of index terms among the non-relevant documents can be approximated by the distribution of index terms among all the documents in the collection.

$$P(k_i | \bar{R}) = \frac{n_i}{N}$$

(assume $N \gg |R|$, $N \approx |\bar{R}|$)

Initial ranking

- V : a subset of the documents initially retrieved and ranked by the probabilistic model (*top r documents*)
- V_i : subset of V composed of documents which contain the index term k_i
- Approximate $P(k_i|R)$ by the distribution of the index term k_i among the documents retrieved so far.
- Approximate $P(k_i|\bar{R})$ by considering that all the non-retrieved documents are not relevant.

$$P(k_i | R) = \frac{V_i}{V}$$

$$P(k_i | \bar{R}) = \frac{n_i - V_i}{N - V}$$

Small values of V and V_i

$$P(k_i | R) = \frac{V_i}{V}$$

a problem when $V=1$ and $V_i=0$

$$P(k_i | \bar{R}) = \frac{n_i - V_i}{N - V}$$

- alternative 1

$$P(k_i | R) = \frac{V_i + 0.5}{V + 1}$$

$$P(k_i | \bar{R}) = \frac{n_i - V_i + 0.5}{N - V + 1}$$

- alternative 2

$$P(k_i | R) = \frac{V_i + \frac{n_i}{N}}{V + 1}$$

$$P(k_i | \bar{R}) = \frac{n_i - V_i + \frac{n_i}{N}}{N - V + 1}$$

Analysis of Probabilistic Model

- advantage
 - documents are ranked in decreasing order of their probability of being relevant
- disadvantages
 - the need to guess the initial separation of documents into relevant and non-relevant sets
 - do not consider the frequency with which an index terms occurs inside a document
 - the independence assumption for index terms

Comparison of classic models

- Boolean model: the weakest classic model
- Vector model is expected to outperform the probabilistic model with general collections (Salton and Buckley)

補充資料

- 三大模型的各種變形方案及應用

Alternative Set Theoretic Models

-Fuzzy Set Model

- Model
 - a query term: a fuzzy set
 - a document: degree of membership in this set
 - membership function
 - Associate membership function with the elements of the class
 - 0: no membership in the set
 - 1: fully membership
 - 0~1: marginal elements of the set

↑
documents

Fuzzy Set Theory

a class



- A fuzzy subset A of a universe of discourse U is characterized by a membership function $\mu_A: U \rightarrow [0,1]$ which associates with each element u of U a number $\mu_A(u)$ in the interval $[0,1]$

a document

- complement: $\mu_{\bar{A}}(u) = 1 - \mu_A(u)$
- union: $\mu_{A \cup B}(u) = \max(\mu_A(u), \mu_B(u))$
- intersection: $\mu_{A \cap B}(u) = \min(\mu_A(u), \mu_B(u))$

Examples

- Assume $U = \{k_1, k_2, k_3, k_4, k_5, k_6\}$
- Let A and B be $\{k_1, k_2, k_3\}$ and $\{k_2, k_3, k_4\}$, respectively.
- Assume $\mu_A = \{k_1/0.8, k_2/0.7, k_3/0.6, k_4/0, k_5/0, k_6/0\}$ and $\mu_B = \{k_1/0, k_2/0.6, k_3/0.8, k_4/0.9, k_5/0, k_6/0\}$
- $\mu_A^-(u) = 1 - \mu_A(u) = \{k_1:0.2, k_2:0.3, k_3:0.4, k_4:1, k_5:1, k_6:1\}$
- $\mu_{A \cup B}(u) = \max(\mu_A(u), \mu_B(u)) = \{k_1:0.8, k_2:0.7, k_3:0.8, k_4:0.9, k_5:0, k_6:0\}$
- $\mu_{A \cap B}(u) = \min(\mu_A(u), \mu_B(u)) = \{k_1:0, k_2:0.6, k_3:0.6, k_4:0, k_5:0, k_6:0\}$

Fuzzy Information Retrieval

- basic idea
 - Expand the set of index terms in the query with related terms (from the thesaurus) such that additional relevant documents can be retrieved
 - A thesaurus can be constructed by defining a term-term correlation matrix $\vec{c}$ whose rows and columns are associated to the index terms in the document collection
- keyword correlation matrix*

Fuzzy Information Retrieval

(Continued)

- normalized correlation factor $c_{i,l}$ between two terms k_i and k_l ($0 \sim 1$)

$$c_{i,l} = \frac{n_{i,l}}{n_i + n_l - n_{i,l}} \quad \text{where} \quad \begin{cases} n_i \text{ is \# of documents containing term } k_i \\ n_l \text{ is \# of documents containing term } k_l \\ n_{i,l} \text{ is \# of documents containing } k_i \text{ and } k_l \end{cases}$$

- In the fuzzy set associated to each index term k_i , a document d_j has a degree of membership $\mu_{i,j}$

$$\mu_{i,j} = 1 - \prod_{k_l \in d_j} (1 - c_{i,l})$$

Example

Query $q = k_a \wedge (k_b \vee \neg k_c)$

disjunctive normal form $\xrightarrow{\quad} q_{\text{dnf}} = (1, 1, 1) \vee (1, 1, 0) \vee (1, 0, 0)$

- (1) the degree of membership in a disjunctive fuzzy set is computed using an algebraic sum (*instead of max function*) *more smoothly*
- (2) the degree of membership in a conjunctive fuzzy set is computed using an algebraic product (*instead of min function*)

$$\mu_{q,j} = \mu_{cc1+cc2+cc3,j}$$

$$= 1 - \prod_{i=1}^3 (1 - \mu_{cc_i,j})$$

$$= 1 - (1 - \mu_{a,j} \mu_{b,j} \mu_{c,j}) \times (1 - \mu_{a,j} \mu_{b,j} (1 - \mu_{c,j})) \times (1 - \mu_{a,j} (1 - \mu_{b,j}) (1 - \mu_{c,j}))$$

Recall $\mu_{\bar{A}}(u) = 1 - \mu_A(u)$



Fuzzy Set Model Summary

- Matching degree (the matching of a “document” to the “query”)
- Membership μ_A

$$\text{OR: } \mu_{A \cup B} = \max (\mu_A , \mu_B)$$

$$\text{AND: } \mu_{A \cap B} = \min (\mu_A , \mu_B)$$

$$\text{NOT: } \mu_{\bar{A}} = 1 - \mu_A$$

- Binary decision → grade membership
- term-term correlation
- Fuzzy Set operators (min/max → fuzzy operator)
- “index-term” set $\longleftrightarrow$ “related-term”
(query)

↑
“thesaurus” 同義字字典

Extended Boolean Model

- (different from Boolean) Partial matching & terms weighting
- Boolean query e.g. $q = K_1 \wedge K_2$
 $D_1 \supset K_1$, $D_2 \supset K_2$, $D_3 \supset K_3$, K_4 query results are the same
- Using “Similarity” to represent “matching degree”
 $\Rightarrow Sim(q, dj)$
example: distance, algebraic mean, ...

Alternative Algebraic Model: Generalized Vector Space Model

- independence of index terms
 - $\vec{k}_i$: a vector associated with the index term k_i
 - the set of vectors $\{k_1, k_2, \dots, k_t\}$ is linearly independent
 - orthogonal: $\vec{k}_i \bullet \vec{k}_j = 0$ for $i \neq j$
 - The index term vectors are assumed linearly independent but are not pairwise orthogonal in generalized vector space model
 - The index term vectors, which are not seen as the basis of the space, are composed of *smaller components* derived from the particular collection.

Generalized Vector Space Model

- $\{k_1, k_2, \dots, k_t\}$: index terms in a collection
- $w_{i,j}$: binary weights associated with the term-document pair $\{k_i, d_j\}$
- The patterns of term *co-occurrence* (inside documents) can be represented by a set of 2^t *minterms*

$m_1=(0, 0, \dots, 0)$: point to documents containing none of index terms

$m_2=(1, 0, \dots, 0)$: point to documents containing the index term k_1 only

$m_3=(0,1,\dots,0)$: point to documents containing the index term k_2 only

$m_4=(1,1,\dots,0)$: point to documents containing the index terms k_1 and k_2

...

$m_{2^t}=(1, 1, \dots, 1)$: point to documents containing all the index terms

- $g_i(m_j)$: return the weight $\{0,1\}$ of the index term k_i in the minterm m_j ($1 \leq i \leq t$)

Generalized Vector Space Model

(Continued)

$$\vec{m}_1 = (1, 0, \dots, 0, 0)$$

$$\vec{m}_2 = (0, 1, \dots, 0, 0)$$

...

$$\vec{m}_i \bullet \vec{m}_j = 0 \text{ for } i \neq j$$

$$\vec{m}_t = (0, 0, \dots, 0, 1)$$

(the set of $\vec{m}_i$ are pairwise orthogonal)

- $\vec{m}_i$ (2^t -tuple vector) is associated with minterm m_i (t-tuple vector)
- e.g., $\vec{m}_4$ is associated with m_4 containing k_1 and k_2 , and no others
- co-occurrence of index terms inside documents:
dependencies among index terms

Generalized Vector Space Model

(Continued)

- Determine the index vector $\vec{k}_i$ associated with the index term k_i

$$\vec{k}_i = \frac{\sum_{\forall r, g_l(m_r)=1} c_{i,r} \vec{m}_r}{\sqrt{\sum_{\forall r, g_l(m_r)=1} c_{i,r}^2}}$$

Collect all the vectors $\vec{m}_r$ in which the index term k_i is in state 1.

$$c_{i,r} = \sum_{d_j | g_l(\vec{d}_j) = g_l(m_r) \text{ for all } l} w_{i,j}$$

Sum up $w_{i,j}$ associated with the index term k_i and document d_j whose term occurrence pattern coincides with minterm m_r

Generalized Vector Space Model

(Continued)

- $\vec{k}_i \bullet \vec{k}_j$ quantifies a degree of correlation between k_i and k_j

$$\vec{k}_i \bullet \vec{k}_j = \sum_{\forall r | g_i(m\mathcal{R})=1 \wedge g_j(m\mathcal{R})=1} c_{i,r} \times c_{j,r}$$

- standard cosine similarity is adopted

$$\begin{aligned} \vec{d}_j &= \sum_{\forall i} w_{i,j} \vec{k}_i & \vec{q} &= \sum_{\forall i} w_{i,q} \vec{k}_i \\ \vec{k}_i &= \frac{\sum_{\forall r, g_i(m\mathcal{R})=1} c_{i,r} \vec{m}_r}{\sqrt{\sum_{\forall r, g_i(m\mathcal{R})=1} c_{i,r}^2}} \end{aligned}$$

Latent Semantic Indexing Model

- representation of documents and queries by index terms
 - problem 1: many unrelated documents might be included in the answer set
 - problem 2: relevant documents which are not indexed by any of the query keywords are not retrieved
- possible solution: concept matching instead of index term matching
 - application in cross-language information retrieval

Basic idea

- Map each document and query vector into a lower dimensional space which is associated with concepts
- Retrieval in the reduced space may be superior to retrieval in the space of index terms

Definition

- t : the number of index terms in the collection
- N : the total number of documents
- $\vec{M}=(M_{ij})$: a term-document association matrix with t rows and N columns
- M_{ij} : a weight $w_{i,j}$ associated with the term-document pair $[k_i, d_j]$ (e.g., using tf-idf)

Singular Value Decomposition

$$A \in R^{n \times n}$$

$$(1) A = A^T$$

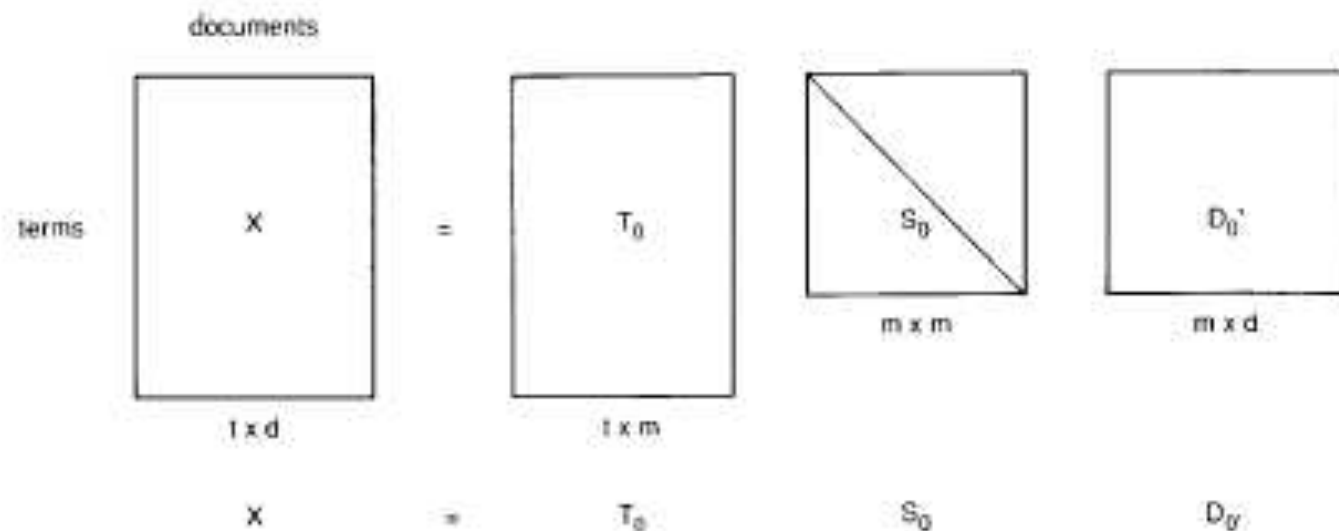
$$\exists Q \in R^{n \times n} \quad st \quad QQ^T = I \quad \{Q^T Q = I\} \quad \text{orthogonal}$$

singular value decomposition :

$$A = QDQ^T \quad \{A^T = (QDQ^T)^T = (Q^T)^T D^T Q^T = QDQ^T = A\}$$

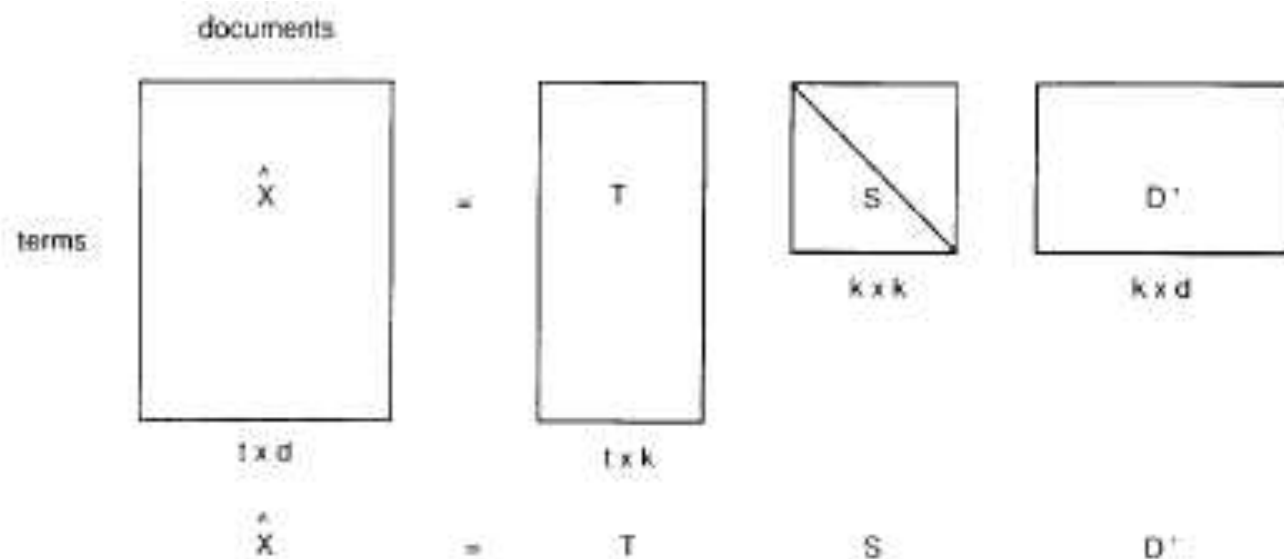
$$\text{where } D = \begin{pmatrix} \lambda_1 & & & & & \\ & \lambda_2 & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ 0 & & & & \ddots & \\ & & & & & \ddots \\ & & & & & & \lambda_n \end{pmatrix} \quad \text{diagonal matrix}$$

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq 0$$



Singular value decomposition of the term x document matrix, X . Where:

- T_0 has orthogonal, unit-length columns ($T_0' T_0 = I$)
- D_0' has orthogonal, unit-length columns ($D_0' D_0 = I$)
- S_0 is the diagonal matrix of singular values
- t is the number of rows of X
- d is the number of columns of X
- m is the rank of X ($\leq \min(t, d)$)



Reduced singular value decomposition of the term x document matrix, X . Where:

T has orthogonal, unit-length columns ($T^T T = I$)

D has orthogonal, unit-length columns ($D^T D = I$)

S is the diagonal matrix of singular values

t is the number of rows of X

d is the number of columns of X

m is the rank of X ($\leq \min(t, d)$)

k is the chosen number of dimensions in the reduced model ($k \leq m$)

Technical Memo Example

Titles

- | | |
|-----|--|
| c1: | <i>Human machine interface for Lab ABC computer applications</i> |
| c2: | <i>A survey of user opinion of computer system response time</i> |
| c3: | <i>The EPS user interface management system</i> |
| c4: | <i>System and human system engineering testing of EPS</i> |
| c5: | <i>Relation of user-perceived response time to error measurement</i> |
| m1: | <i>The generation of random, binary, unordered trees</i> |
| m2: | <i>The intersection graph of paths in trees</i> |
| m3: | <i>Graph minors IV: Widths of trees and well-quasi-ordering</i> |
| m4: | <i>Graph minors: A survey</i> |

Terms

Documents

[illegible]

$$A \in R^{n \times n}$$

$$(2) A \neq A^T$$

$$\exists U, V \in R^{n \times n} \quad st \quad U^T U = I, V^T V = I \quad \text{orthogonal}$$

singular value decomposition :

$$(AB)^T = B^T A^T$$

$$A = UDV^T$$

$$AA^T = (UDV^T)(UDV^T)^T = (UDV^T)(VDU^T) = UD^2U^T$$

where D =

$$\begin{pmatrix} \lambda_1 & & & & & \\ & \lambda_2 & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & 0 & & & \ddots & \\ & & & & & \ddots \\ & & & & & & \lambda_n \end{pmatrix}$$

diagonal matrix

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq 0$$

$$A = QDQ^T$$

$$AQ = QDQ^T Q = QD$$

where $Q = [q_1 \quad q_2 \quad \dots \quad q_n]$, q_i : a column vector

$$A[q_1 \quad q_2 \quad \dots \quad q_n] = [q_1 \quad q_2 \quad \dots \quad q_n] \begin{pmatrix} \lambda_1 & & & 0 \\ & \lambda_2 & & \\ & & \ddots & \\ & & & \ddots \\ & & & & \lambda_n \end{pmatrix}$$

$$[Aq_1 \quad Aq_2 \quad \dots \quad Aq_n] = [\lambda_1 q_1 \quad \lambda_2 q_2 \quad \dots \quad \lambda_n q_n]$$

$$Aq_1 = \lambda_1 q_1 \quad Aq_2 = \lambda_2 q_2 \quad \dots \quad Aq_n = \lambda_n q_n$$

$\lambda_1, \lambda_2, \dots, \lambda_n$ are eigenvalues of A ,

Q_k is λ_k 's corresponding eigenvector

Singular Value Decomposition

$\overrightarrow{M}$: a term – document matrix with t rows and N columns

$$\overrightarrow{M} = \overrightarrow{K} \overrightarrow{S} \overrightarrow{D}^t$$

$\overrightarrow{M}^t \overrightarrow{M}$: a $N \times N$ document – to – document matrix

$\overrightarrow{M} \overrightarrow{M}^t$: a $t \times t$ term – to – term matrix

According to

$$\overrightarrow{M} \in R^{t \times N}$$

$\exists \overrightarrow{K}$: the matrix of eigenvectors derived from $\overrightarrow{M} \overrightarrow{M}^t$ $\overrightarrow{K}^t \overrightarrow{K} = I$

$\overrightarrow{D}$: the matrix of eigenvectors derived from $\overrightarrow{M}^t \overrightarrow{M}$ $\overrightarrow{D}^t \overrightarrow{D} = I$

$$\overrightarrow{M} = \overrightarrow{K} \overrightarrow{S} \overrightarrow{D}^t$$

$\overrightarrow{M}^t \overrightarrow{M}$: *document – to – document matrix*

$$= (\overrightarrow{K} \overrightarrow{S} \overrightarrow{D}^t)^t (\overrightarrow{K} \overrightarrow{S} \overrightarrow{D}^t)$$

$$= (\overrightarrow{D} \overrightarrow{S}^t \overrightarrow{K}^t) (\overrightarrow{K} \overrightarrow{S} \overrightarrow{D}^t)$$

$$= \overrightarrow{D} \overrightarrow{S}^2 \overrightarrow{D}^t$$

$\overrightarrow{M} \overrightarrow{M}^t$: *term – to – term matrix*

$$= (\overrightarrow{K} \overrightarrow{S} \overrightarrow{D}^t) (\overrightarrow{K} \overrightarrow{S} \overrightarrow{D}^t)^t$$

$$= (\overrightarrow{K} \overrightarrow{S} \overrightarrow{D}^t) (\overrightarrow{D} \overrightarrow{S}^t \overrightarrow{K}^t)$$

$$= \overrightarrow{K} \overrightarrow{S}^2 \overrightarrow{K}^t$$

對照 $A = QDQ^T$

Q is matrix of eigenvectors of A

D is diagonal matrix of singular values
得到

$\overrightarrow{D}$: *the matrix of eigenvectors*

derived from $\overrightarrow{M}^t \overrightarrow{M}$

$\overrightarrow{K}$: *the matrix of eigenvectors*

derived from $\overrightarrow{M} \overrightarrow{M}^t$

$\overrightarrow{S}$: $r \times r$ diagonal matrix of singular values, where $r = \min(t, N)$

$s < r$ (Concept space is reduced)

Consider only the s largest singular values of $\vec{S}$

$$\begin{pmatrix} \lambda_1 & & & & & \\ & \lambda_2 & & & & \\ & & \ddots & & & \\ & & & \ddots & & \\ & 0 & & & \ddots & \\ & & & & & \ddots & \\ & & & & & & \lambda_n \end{pmatrix}$$

$$\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_n \geq 0$$

The resultant $\vec{M}_s$ matrix is the matrix of rank s which is closest to the original matrix M in the least square sense.

$$\vec{M}_s = \vec{K}_s \vec{S}_s \vec{D}_s^t$$

$$(s \ll t, s \ll N)$$

Ranking in LSI

- query: a pseudo-document in the original $\vec{M}$ term-document
 - query is modeled as the document with number 0
 - $\vec{M}_s^t \vec{M}_s$: the ranks of all documents w.r.t this query

Research Issues

- Library systems
 - Cognitive and behavioral issues oriented particularly at a better understanding of which criteria the users adopt to judge relevance
- Specialized retrieval systems
 - e.g., legal and business documents
 - how to retrieve all relevant documents without retrieving a large number of unrelated documents
- The Web
 - User does not know what he wants or has great difficulty in formulating his request
 - How the paradigm adopted for the user interface affects the ranking
 - The indexes maintained by various Web search engine are almost disjoint